**FOOD TRACKING SYSTEM**

**NAAN MUDHALVAN**

(2023 – 2024)

**PROJECT REPORT**

Submitted By

TEAM ID : NM2023TMID02816

Aravind R (961820105006)

Arun S (961820105007)

Arun Chripssan A (961820105008)

Beauttus BS (961820105009)

In Partial Fulfilment for the Award of the Degree

Of

BACHELOR OF ENGINEERING

In

ELECTRICAL AND ELECTRONICS ENGINEERING

**PONJESLY COLLEGE OF ENGINEERING,**

**NAGERCOIL - 629004**

**TABLE OF CONTENT**

| **CHAPTER NO.** | **TITLE** | **PAGE NO.** |
| --- | --- | --- |
| **1** | **INTRODUCTION** | 3 |
|  | 1.1 Project Overview | 3 |
|  | 1.2 Purpose | 3 |
| **2** | **LITERATURE SURVEY** | 4 |
|  | 2.1 Existing problem | 4 |
|  | 2.2 References | 4 |
|  | 2.3 Problem Statement Definition | 5 |
| **3** | **IDEATION & PROPOSED SOLUTION** | 6 |
|  | 3.1 Empathy Map Canvas | 6 |
|  | 3.2 Ideation & Brainstorming | 7 |
| **4** | **REQUIREMENT ANALYSIS** | 10 |
|  | 4.1 Functional requirement | 10 |
|  | 4.2 Non-Functional requirements | 11 |
| **5** | **PROJECT DESIGN** | 13 |
|  | 5.1 Data Flow Diagrams & User Stories | 13 |
|  | 5.2 Solution Architecture | 15 |
| **6** | **PROJECT PLANNING & SCHEDULING** | 16 |
|  | 6.1 Technical Architecture | 16 |
|  | 6.2 Sprint Planning & Estimation | 16 |
|  | 6.3 Sprint Delivery Schedule | 18 |
| **7** | **CODING & SOLUTIONS** | 19 |
|  | 7.1 Feature 1 | 22 |
|  | 7.2 Feature 2 | 23 |
|  | 7.3 Database Schema | 23 |
| **8** | **PERFORMANCE TESTING** | 24 |
|  | 8.1 Performance Metrics | 24 |
| **9** | **RESULTS** | 25 |
|  | 9.1 Output Screenshots | 25 |
| **10** | **ADVANTAGES & DISADVANTAGES** | 28 |
| **11** | **CONCLUSION** | 29 |
| **12** | **FUTURE SCOPE** | 30 |
| **13** | **APPENDIX** | 31 |
|  | Source Code | 31 |
|  | GitHub & Project Demo Link | 50 |

**CHAPTER 1**

**INTRODUCTION**

* 1. **PROJECT OVERVIEW:**

A food tracking system is a comprehensive digital solution designed to empower individuals to make healthier dietary choices and manage their nutrition effectively. This project aims to create a user-friendly platform where users can record and analyze their food consumption, monitor their dietary goals, and receive personalized recommendations. The system will feature user registration, food logging, nutritional data retrieval, recommendation algorithms, meal planning, progress tracking, reminders, and reporting functionalities. The project's key objectives include improving nutrition awareness, promoting better eating habits, and providing insights into dietary patterns. Additionally, it will have a business component by offering monetization strategies such as subscription plans or partnerships. The ultimate goal is to make a positive impact on both individual health and the broader food and wellness industry by leveraging technology for enhanced nutritional tracking and healthier living.

* 1. **PURPOSE:**

The purpose of a food tracking system is multifaceted and encompasses several key objectives:

**Health and Nutrition Awareness:** It serves as a tool to help individuals become more aware of their dietary habits and the nutritional content of the foods they consume. This awareness can lead to improved food choices and a better understanding of the impact of nutrition on health.

**Dietary Goal Management:** The system assists users in setting and managing dietary goals, such as weight loss, calorie intake, macronutrient targets, or adherence to specific diets (e.g., vegan, gluten-free). It provides a structured way to work towards these objectives.

**Personalized Recommendations:** By analyzing users' food consumption data and preferences, the system can offer personalized recommendations for meals, snacks, and dietary improvements, making it easier for individuals to achieve their health and wellness goals.

**Convenience and Efficiency:** It provides a convenient and efficient method for users to track their food intake, eliminating the need for manual calculations and enabling real-time updates of nutritional information.

**Data-Driven Decision-Making:** Users can make informed food choices by accessing detailed nutritional information about a wide range of foods, helping them tailor their diets to their specific needs and preferences.

**Meal Planning:** The system facilitates meal planning, recipe creation, and the calculation of the nutritional content of entire meals, making it easier to prepare balanced and healthy dishes.

**Progress Tracking:** Users can track their progress over time, allowing them to measure the effectiveness of their dietary changes and make adjustments as needed.

**Behavior Modification**: It can serve as a behavior modification tool, encouraging individuals to make better food choices, develop healthier eating habits, and adhere to their dietary objectives.

**User Engagement:** Through reminders, notifications, and gamification features, the system can engage users and motivate them to stay on track with their dietary goals.

**Data Insights and Reporting:** The system generates insights and reports on users' food consumption patterns, helping them understand their dietary behaviors and make improvements.

**Privacy and Security:** It ensures the security and privacy of user data, adhering to relevant regulations to protect sensitive health information.

**Business Opportunities:** The system can present business opportunities for monetization through subscription plans, partnerships, or data analysis and insights.

In summary, a food tracking system's primary purpose is to empower individuals to take control of their nutrition, make healthier food choices, and work towards their dietary goals while providing valuable insights and opportunities for businesses in the health and wellness industry.

**CHAPTER 2**

**LITERATURE SURVEY**

**2.1 EXISTING PROBLEM:**

While food tracking systems offer numerous benefits, they also face various challenges and existing problems:

**User Engagement:** Sustaining user engagement over time can be difficult. Many users start tracking their food but abandon the system due to lack of motivation or convenience issues.

**Data Accuracy:** Users may struggle to accurately input food details and portion sizes, leading to inaccurate nutritional data and incomplete food logs.

**Food Variability:** Food databases may not cover every possible food item or variation, making it challenging to find and log certain meals accurately.

**Privacy Concerns:** Users may be hesitant to share personal dietary information due to privacy concerns, especially when using third-party apps or websites.

**Complexity:** Some food tracking systems can be overly complex, leading to user frustration and discouragement.

**Lack of Standardization:** There's no universal standard for food data, leading to inconsistencies and variations in nutritional information between different sources and databases.

**Nutrition Labeling:** Not all food products come with standardized nutrition labels, making it difficult to track their nutritional content accurately.

**Integration with Dining Out:** Tracking meals from restaurants or takeout can be challenging, as nutritional information is often not readily available or may not be entirely accurate.

**Sustainability and Lifestyle Tracking:** Many food tracking systems focus on calorie counting and macronutrients but may not fully address dietary preferences like vegetarianism, veganism, or sustainability concerns.

**User Reliability:** The accuracy of the tracked data relies on the user's commitment and honesty, which may vary.

**Compliance with Special Diets:** Special dietary requirements, allergies, or medical conditions might not be adequately accommodated in standard food tracking systems.

**Platform Fragmentation:** Different food tracking apps and websites may not be interoperable, leading to user data fragmentation and challenges in managing data across platforms.

**Accessibility:** Access to food tracking systems might be limited for individuals with disabilities, posing inclusivity challenges.

**Device Dependency:** Users may need specific devices, such as smartphones or wearable tech, to effectively track their food, excluding those without access to such technology.

**Behavioral Change:** While food tracking can create awareness, it may not always lead to sustained behavioral change or healthier eating habits.

**Cost:** Some advanced food tracking systems or apps may come with a subscription cost, limiting access for some users.

Addressing these existing problems often requires a combination of improved technology, user experience design, data accuracy, and consideration of user motivations and preferences.

**2.2 REFERENCES:**

MyHealthyLivingApp Developers. (2022).

MyHealthyLivingApp: Your Personal Food and Nutrition Tracker (Version 3.0) [Mobile application].

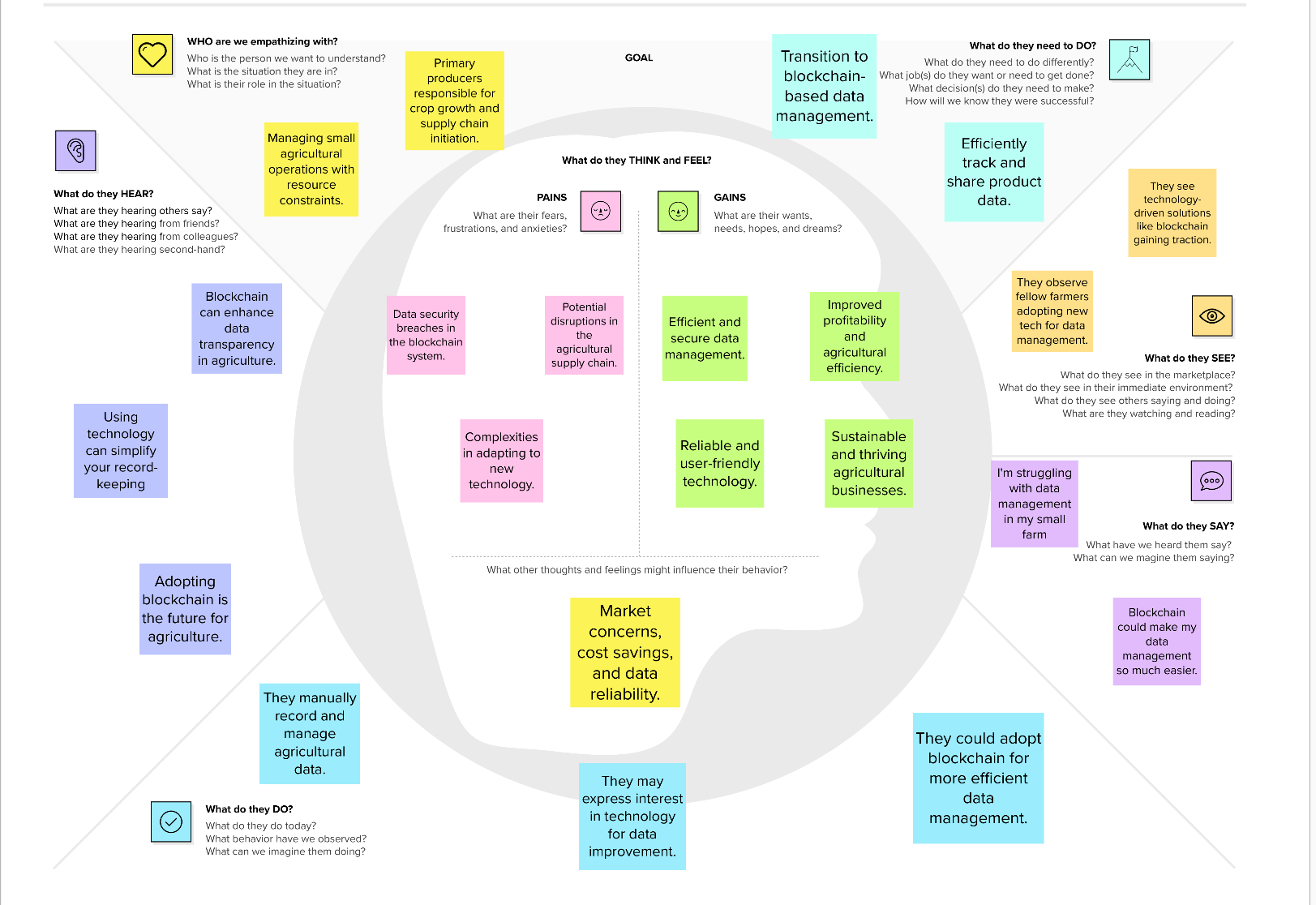
HealthTech Co.<https://www.myhealthylivingapp.com/>

* 1. **PROBLEM STATEMENT DEFINITION:**

The business problem for a food tracking system is the need to help individuals monitor their dietary intake, make informed food choices, and achieve specific health or nutrition goals. This system should provide a convenient and effective way for users to track and analyze their food consumption, manage dietary restrictions, and receive personalized recommendations, ultimately promoting healthier eating habits and improved well-being.

**CHAPTER 3**

**IDEATION & PROPOSED SOLUTION**

**3.1 EMPATHY MAP CANVAS**

**3.2 IDEATION & BRAINSTORMING**

Ideation and brainstorming are essential steps in the development of a food tracking system. To generate creative and innovative ideas, consider the following approaches:

**User Personas:** Create user personas that represent the target audience. Think about the needs, preferences, and pain points of different types of users, such as health-conscious individuals, fitness enthusiasts, parents, or people with dietary restrictions.

**Problem Statement:** Clearly define the problem you aim to solve with the food tracking system. This might be related to health improvement, weight management, dietary awareness, or convenience in meal planning.

**Feature Prioritization:** List the core features that the system must have. Prioritize these based on user needs and the system's primary goals. Some essential features might include food logging, nutritional data, and personalized recommendations.

**Innovation Workshop:** Conduct innovation workshops with your development team and stakeholders to brainstorm new and unique features. Explore cutting-edge technologies like machine learning for automatic food recognition or integration with IoT devices for real-time data collection.

**Market Research:** Study existing food tracking systems and applications to identify gaps and areas for improvement. Analyze user reviews and feedback to understand what users like and dislike about current solutions.

**Collaboration:** Collaborate with nutritionists, dietitians, fitness experts, and other professionals in the health and wellness field to gain insights and ideas for the system's features and functionality.

**Gamification:** Explore the use of gamification elements to keep users engaged and motivated. Consider adding rewards, challenges, and achievements for users who consistently track their food and meet their dietary goals.

**Social Interaction:** Think about incorporating social features that allow users to share their progress, recipes, and achievements with a community of like-minded individuals. Social interaction can enhance user engagement.

**Integration with Wearables:** Consider integrating the system with wearable devices and fitness trackers to automatically capture data on physical activity, heart rate, and sleep patterns for a holistic view of health and wellness.

**AI and Machine Learning:** Leverage AI and machine learning for advanced features such as predicting dietary trends, identifying nutritional gaps, and offering even more personalized recommendations.

**Environmental Impact:** Address sustainability concerns by including features that help users make eco-friendly food choices by providing information about the environmental impact of different foods.

**Accessibility:** Ensure that the system is accessible to individuals with disabilities, complying with accessibility standards.

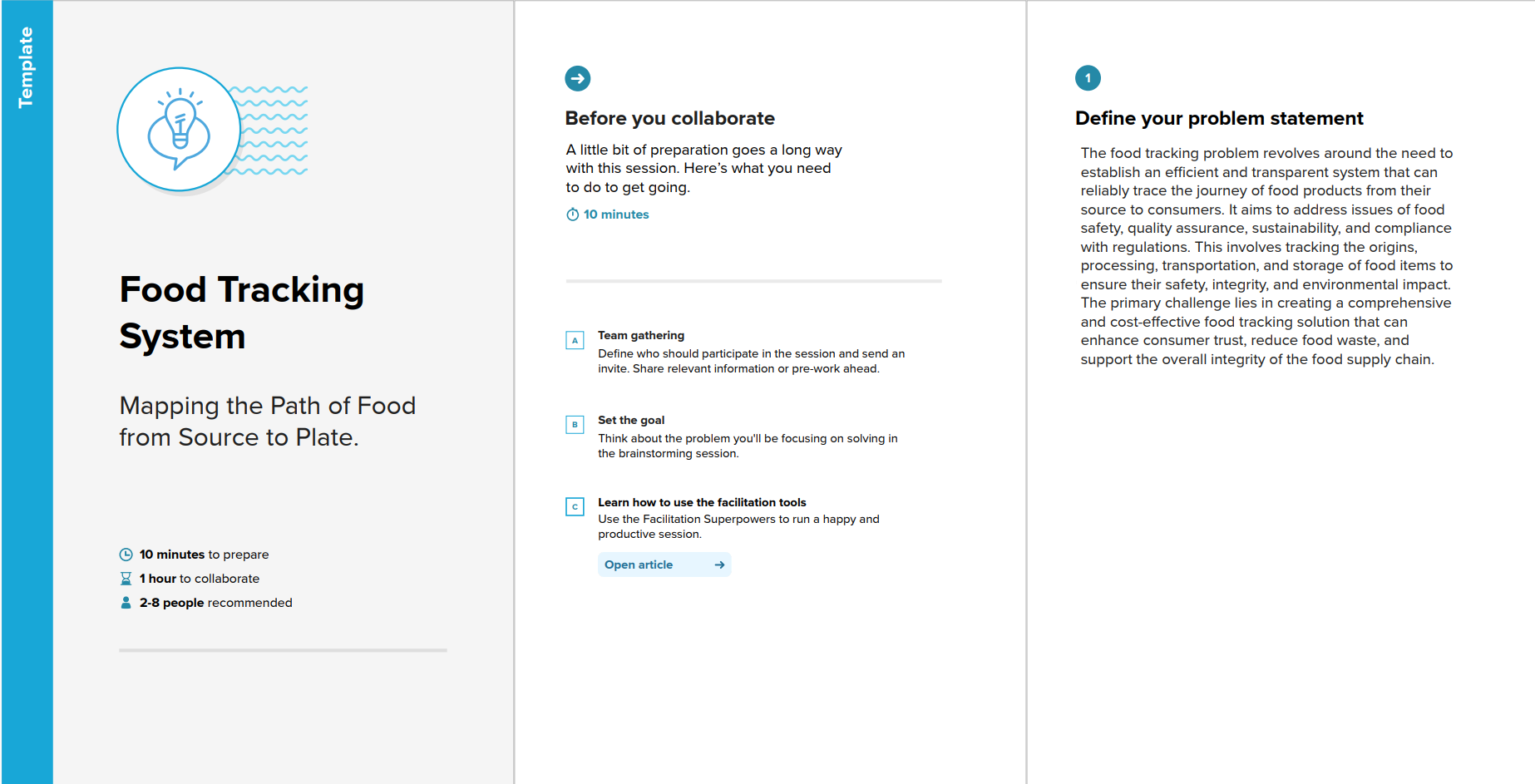
**Business Model:** Develop a clear monetization strategy, whether through premium subscriptions, partnerships with food brands, or data monetization.

**Privacy and Data Security:** Consider innovative ways to enhance user data privacy and security, such as blockchain technology for secure and transparent data storage.

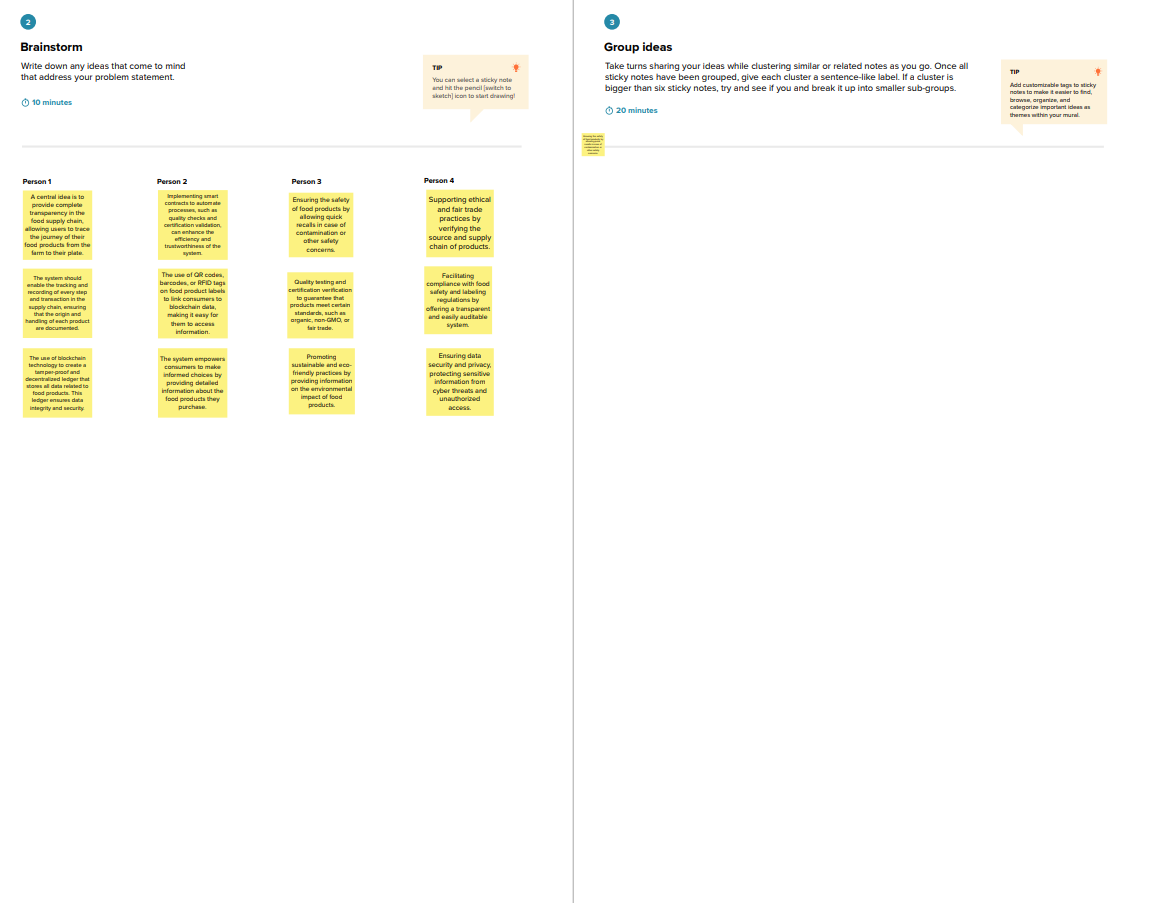
**Feedback Mechanisms:** Implement feedback mechanisms, allowing users to suggest new features and improvements. Actively collect and analyze user feedback to guide future enhancements.

Remember that ideation and brainstorming are iterative processes. Encourage open and creative thinking, involve your development team, and be open to adapting your ideas as you gather more insights and feedback from potential users.

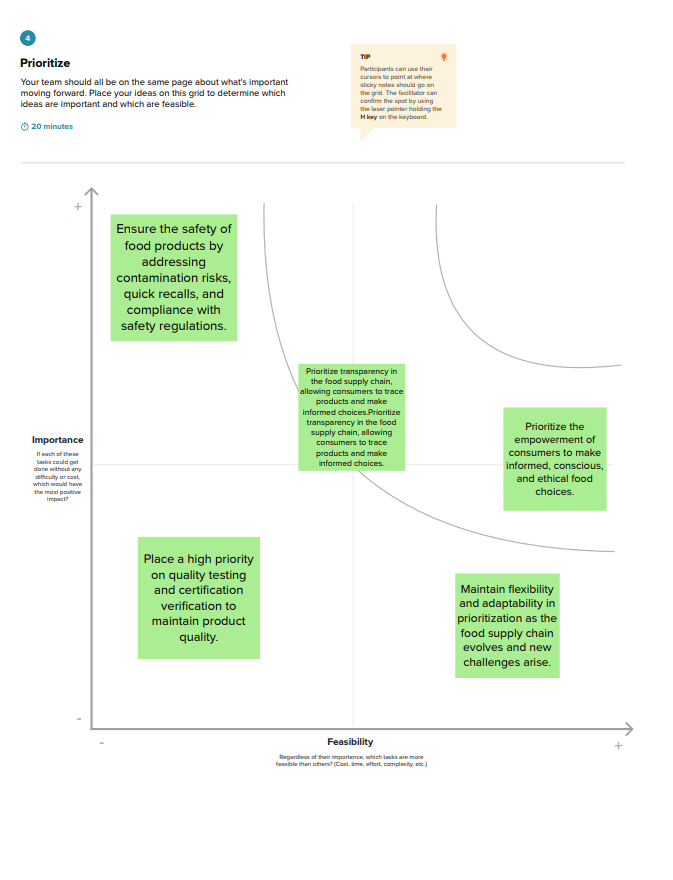
**Step-1: Team Gathering, Collaboration and Select the Problem Statement**

****

**Step-2: Brainstorm, Idea Listing and Grouping**

****

**Step-3: Idea Prioritization**

****

**CHAPTER 4**

**REQUIREMENT ANALYSIS**

**4.1 FUNCTIONAL REQUIREMENTS:**

| **FR NO** | **FUNCTIONAL REQUIREMENT** | **SUB REQUIREMENT** |
| --- | --- | --- |
| FR – 1 | User Registration and Authentication | User accounts with secure authentication methods to protect user data. |
| FR – 2 | Food Database | * A comprehensive database of food items with nutritional information. * The ability to add custom foods and recipes. |
| FR – 3 | Food Logging | * User-friendly interfaces for logging consumed foods, including portion sizes. * The option to log meals, snacks, and beverages. |
| FR – 4 | Nutritional Data | Accurate nutritional information for each food item, including calories, macronutrients, micronutrients, and serving sizes. |
| FR – 5 | Recommendation Engine | Algorithms to provide personalized dietary recommendations based on user goals and preferences. |
| FR – 6 | Meal Planning | * Tools for planning and scheduling meals. * The ability to create and store meal plans. |
| FR - 7 | Progress Tracking | Features for users to track their progress toward health and nutrition goals. |
| R - 8 | Reminders and Notifications | Reminders to log meals or reach daily nutritional targets. |

These functional requirements are designed to ensure that the "Food Tracking System" is capable of recording, querying, and securely managing data on the Ethereum blockchain.

**4.2 NON-FUNCTIONAL REQUIREMENTS:**

| **NFR NO** | **NON-FUNCTIONAL REQUIREMENT** | **DESCRIPTION** |
| --- | --- | --- |
| NFR – 1 | Scalability | The system should be designed to easily scale to accommodate an increasing user base and growing data. |
| NFR – 2 | Reliability | The system should be available and reliable, with minimal downtime or service interruptions. |
| NFR – 3 | Security | Data should be stored securely, with encryption to protect sensitive user information. |
| NFR – 4 | Data Backup and Recovery | Regular data backups should be performed to prevent data loss in the event of system failures. |
| NFR – 5 | Usability | The system should be user-friendly, with an intuitive and easy-to-navigate interface. |
| NFR – 6 | Accessibility | The system should comply with accessibility standards (e.g., WCAG) to make it usable by individuals with disabilities. |
| NFR – 7 | Response Time | Define acceptable response times for various system operations (e.g., loading a page, submitting data). |

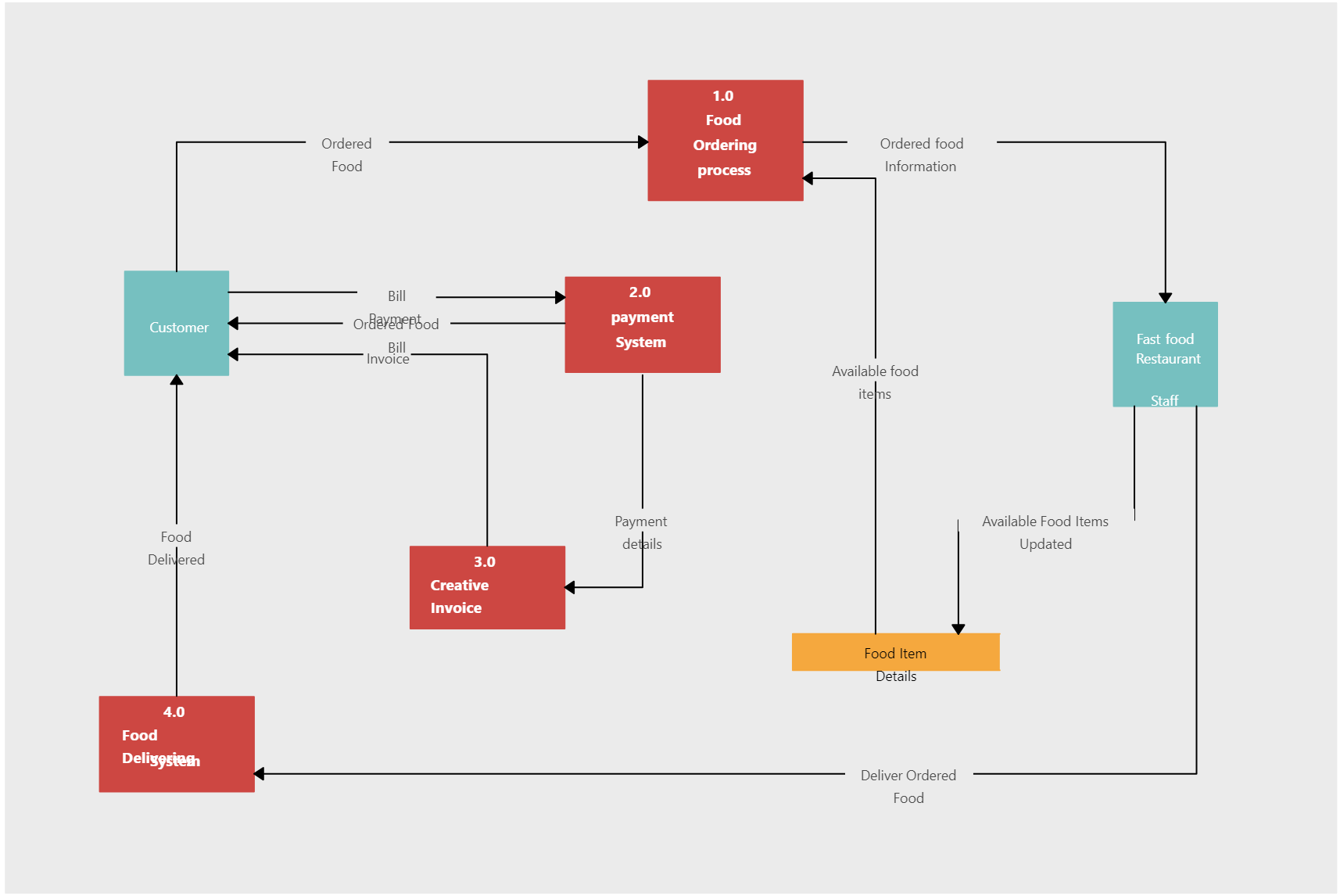
These non-functional requirements outline the performance, reliability, security, and usability standards that the "Food Tracking System" should adhere to in order to provide a robust and user-centric solution.

**CHAPTER 5**

**PROJECT DESIGN**

**5.1 DATA FLOW DIAGRAMS & USER STORIES**

**DATA FLOW DIAGRAM:**

****

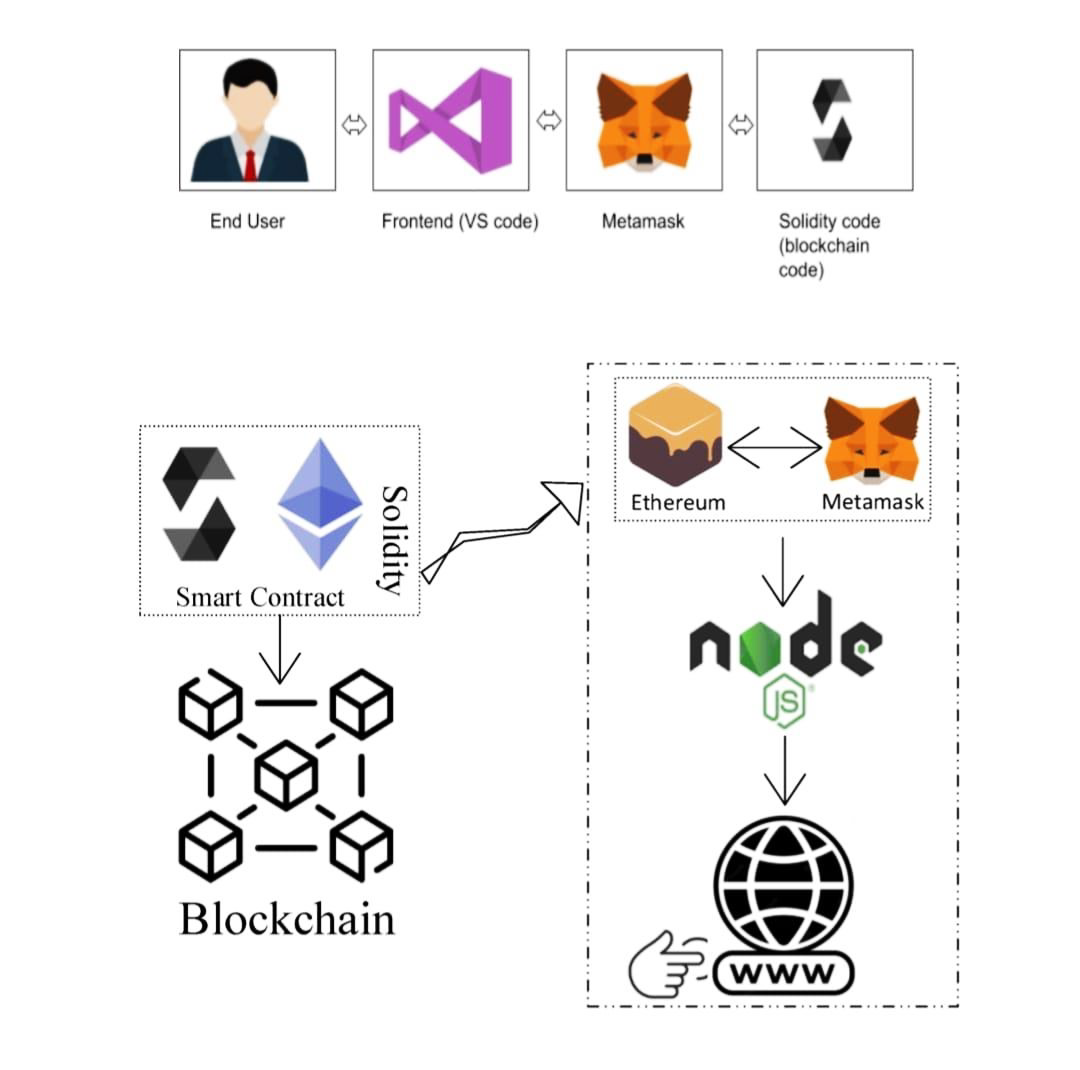
**FOOD TRACKING SYSTEM**

**USER STORIES:**

**User Story 1**: As a health-conscious user, I want to be able to create a personalized profile with my dietary preferences and nutritional goals so that the system can provide tailored recommendations for my meals.

**Requirements:**

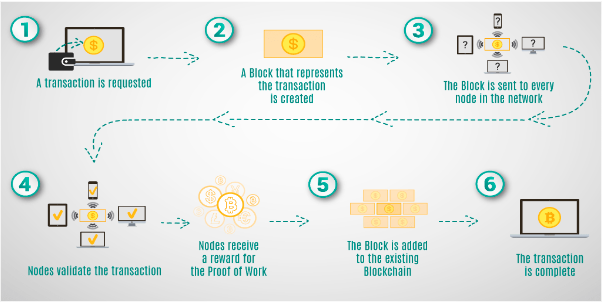
* The system should allow me to upload details of my food items, including food information, livestock data, and supply chain information.
* I should be able to easily update this information when there are changes, such as change food items, new livestock data, or supply chain updates.
* The system should maintain a secure and immutable record of the information I upload, ensuring data accuracy and traceability.
  1. **SOLUTION ARCHITECTURE**



**Interaction between the Web and the Contract**

**CHAPTER 6**

**PROJECT PLANNING & SCHEDULING**

**6.1 TECHNICAL ARCHITECTURE**

**6.2 SPRINT PLANNING & ESTIMATION**

Sprint planning and estimation are crucial components of Agile software development. When planning sprints for a food tracking system, it's important to have a well-defined backlog of user stories and tasks. Here's a general guideline for sprint planning and estimation:

Backlog Grooming (Before Sprint Planning):

- Ensure that your product backlog is up to date with user stories and tasks. The product owner should prioritize these items.

- Refine and clarify user stories to ensure they are well-understood by the development team.

- Break down larger user stories into smaller, manageable tasks.

Sprint Planning Meeting:

a. Setting Sprint Goals: - Begin by discussing the overall goals and objectives for the upcoming sprint. What do you aim to achieve in this sprint?

b. Selection of User Stories: - Based on the sprint goals and priorities, select user stories from the backlog to work on during this sprint. Ensure that the development team has a clear understanding of these stories.

c. Task Breakdown: - For each user story, break down the work into specific tasks. Tasks should be small enough to complete within the sprint's time frame.

- Assign these tasks to team members based on their skills and availability.

d. Estimation: - Estimate the effort required for each task using a common estimation technique such as story points, planning poker, or hours.

- Be sure to involve the entire development team in the estimation process to benefit from their collective expertise.

3. Capacity Planning: - Determine the team's capacity for the sprint based on their availability, factoring in any potential disruptions (e.g., meetings, holidays).

4. Commitment: - The team commits to completing a certain amount of work (user stories and tasks) during the sprint based on the sprint's capacity.

5. Daily Stand-up Meetings: - Hold daily stand-up meetings during the sprint to track progress, discuss any challenges, and adjust plans as necessary.

6. Sprint Review: - At the end of the sprint, conduct a sprint review to demonstrate the completed work to stakeholders, including the product owner.

7. Sprint Retrospective: - After the sprint review, hold a retrospective meeting to discuss what went well, what could be improved, and what actions can be taken in the next sprint to enhance the development process.

Remember that sprint planning and estimation are iterative processes, and you should refine your approach over time based on the team's experience and the project's needs. Agile principles emphasize flexibility and adaptability, so it's important to be open to adjustments as you progress through each sprint.

**6.3 SPRINT DELIVERY SCHEDULE**

Sprint Delivery Schedule for Food Tracking System Project

| **Sprint Delivery Schedule and its Objectives** | | |
| --- | --- | --- |
| **Sprint 1** | Project Initiation | Set project scope and team roles. |
| **Sprint 2** | Data Modeling | Define data models and prioritize user stories. |
| **Sprint 3** | Blockchain Integration | Begin integrating blockchain technology. |
| **Sprint 4** | User Interfaces and Mobile Apps | Develop user interfaces and mobile apps. |
| **Sprint 5** | Quality Control and Testing | Implement quality control and initiate testing. |
| **Sprint 6** | Compliance and Security | Ensure regulatory compliance and enhance security. |
| **Sprint 7** | Deployment Preparation | Prepare for system deployment. |
| **Sprint 8** | Pilot Testing and Optimization | Conduct pilot tests and optimize the system. |
| **Sprint 9** | Full System Deployment | Roll out the fully operational system to a wider audience. |

This sprint delivery schedule outlines the specific objectives and focus areas for each sprint of the "Blockchain-Enhanced Agriculture Data Management System" project. It ensures that the project progresses systematically, with clear goals for each sprint.

**CHAPTER 7**

**CODING & SOLUTIONING**

**Smart Contract (Solidity)**

// 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);

}

}

The key features related to food tracking in this contract are:

**7.1 FEATURE 1**

* **Food Product Registration and Update:**

The smart contract allows for the registration and updating of food products. Users can add information about an food product, including its name, description, and quantity. This feature is essential for tracking and recording data about various food products within the blockchain. Users, such as farmers or suppliers, can update the product information as needed, ensuring that the data remains accurate and up to date.

**7.2 FEATURE 2**

* **Ownership Control:**

The contract includes an ownership control mechanism. Only the owner of a particular product (as specified by the ‘**owner’** field) can perform actions such as updating the product's information. This feature enhances data security and ensures that only authorized users can modify the information associated with specific products.

**7.3 DATABASE SCHEMA**

In the context of the "Food tracking System," the database schema is primarily defined by the structure of the smart contract deployed on the Ethereum blockchain. The Ethereum blockchain, being a decentralized and distributed ledger, serves as the underlying data storage mechanism for the project, eliminating the need for a traditional relational database schema.

The core data structure within the smart contract is the ‘**foodProduct’** struct, which is used to organize and store information about food products.

Here is the structure of the ‘**foodProduct’** and how data is stored within the contract:

struct foodProduct {

string name;

string description;

uint256 quantity;

address owner;

}

* **name**: Represents the name or title of the agricultural product.
* **description**: Contains a description or additional details about the product.
* **quantity**: Indicates the quantity or amount of the product.
* **owner**: Records the Ethereum address of the user who owns the product.

The data for each food product is stored within the **products** mapping using a unique product identifier as the key. The Ethereum blockchain provides the underlying storage for this data, and it is inherently decentralized and immutable. There is no need for traditional relational database tables, as the blockchain itself acts as the distributed ledger for maintaining the integrity and security of the agricultural data.

This decentralized and blockchain-based database schema ensures that agricultural data is transparent, tamper-proof, and securely managed within the system. Users can access and update data with confidence, and the blockchain's architecture guarantees data immutability and traceability.

**CHAPTER 8**

**PERFORMANCE TESTING**

**8.1 PERFORMANCE METRICS**

Performance testing is crucial for ensuring that a food tracking system operates smoothly and efficiently, especially when handling a large number of users and data. Here are the key aspects and steps to consider when conducting performance testing for a food tracking system:

**Define Performance Metrics:**  - Determine the key performance metrics to measure, which may include response time, throughput, resource utilization, and scalability. Define acceptable thresholds for these metrics.

**Identify Test Scenarios:**  - Create test scenarios that simulate real-world usage of the food tracking system. Consider scenarios for various user activities, such as food logging, meal planning, data retrieval, and reporting.

**Load Testing:**  - Conduct load testing to evaluate the system's performance under expected load conditions. This involves gradually increasing the number of concurrent users and measuring response times and resource consumption.

**Stress Testing**: - Perform stress testing to determine the system's breaking point and its behavior under extreme conditions. This may involve exceeding the system's expected load to identify potential bottlenecks and vulnerabilities.

**Scalability Testing:** - Test the system's ability to scale by adding more resources (e.g., servers, database nodes) and measuring how it performs as the load increases.

**Endurance Testing:** - Run endurance tests to assess system stability over an extended period, simulating continuous use to identify memory leaks or other issues that may occur over time.

**Peak Load Testing:**  - Conduct peak load testing to assess how the system performs at its peak usage times, such as during mealtime or when users log in en masse.

**Data Volume Testing:**  - Test the system's performance with a significant volume of data, including food items, user records, and logs.

**Mobile App Testing:** - If the system includes mobile apps, conduct testing on various mobile devices to ensure the application's performance and responsiveness.

**Network Testing:** - Evaluate how the system performs under different network conditions, including variations in network speed, stability, and latency.

**Database Performance Testing:**

- Focus on the database's performance, including queries and data retrieval times. Optimize database queries and indexing as needed.

**Reporting and Analytics Testing:**  - Test the generation and delivery of reports and analytics to ensure they are produced within acceptable timeframes.

**Third-Party Integrations:** - Verify the performance of third-party integrations, such as wearable device connections or external data sources.

**Security and Data Privacy:**  - Ensure that performance testing considers security aspects, such as the system's ability to handle secure user authentication and data encryption without significant performance degradation.

**Monitor System Resources:**  - Continuously monitor system resources during testing, including CPU and memory usage, database connections, and network bandwidth.

**Analysis and Optimization:**  - Analyze the performance test results and identify bottlenecks, scalability issues, and resource constraints. Optimize the system based on findings.

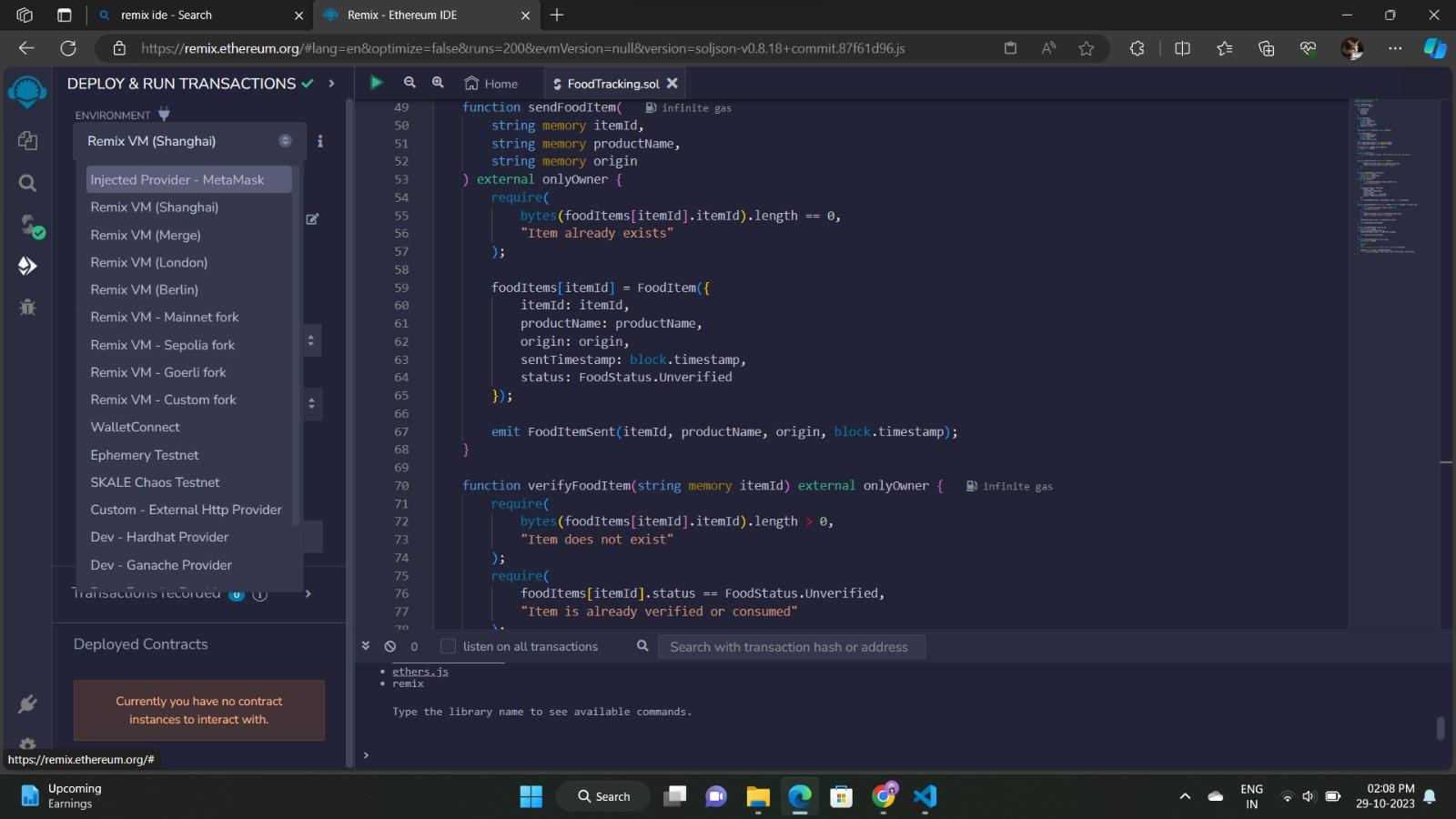
**Repeat Testing Iteratively:** - Repeat performance testing iteratively as changes or enhancements are made to the system to ensure ongoing optimal performance.

**Documentation:**  - Document the results, findings, and any changes made to the system to maintain a record of performance improvements.

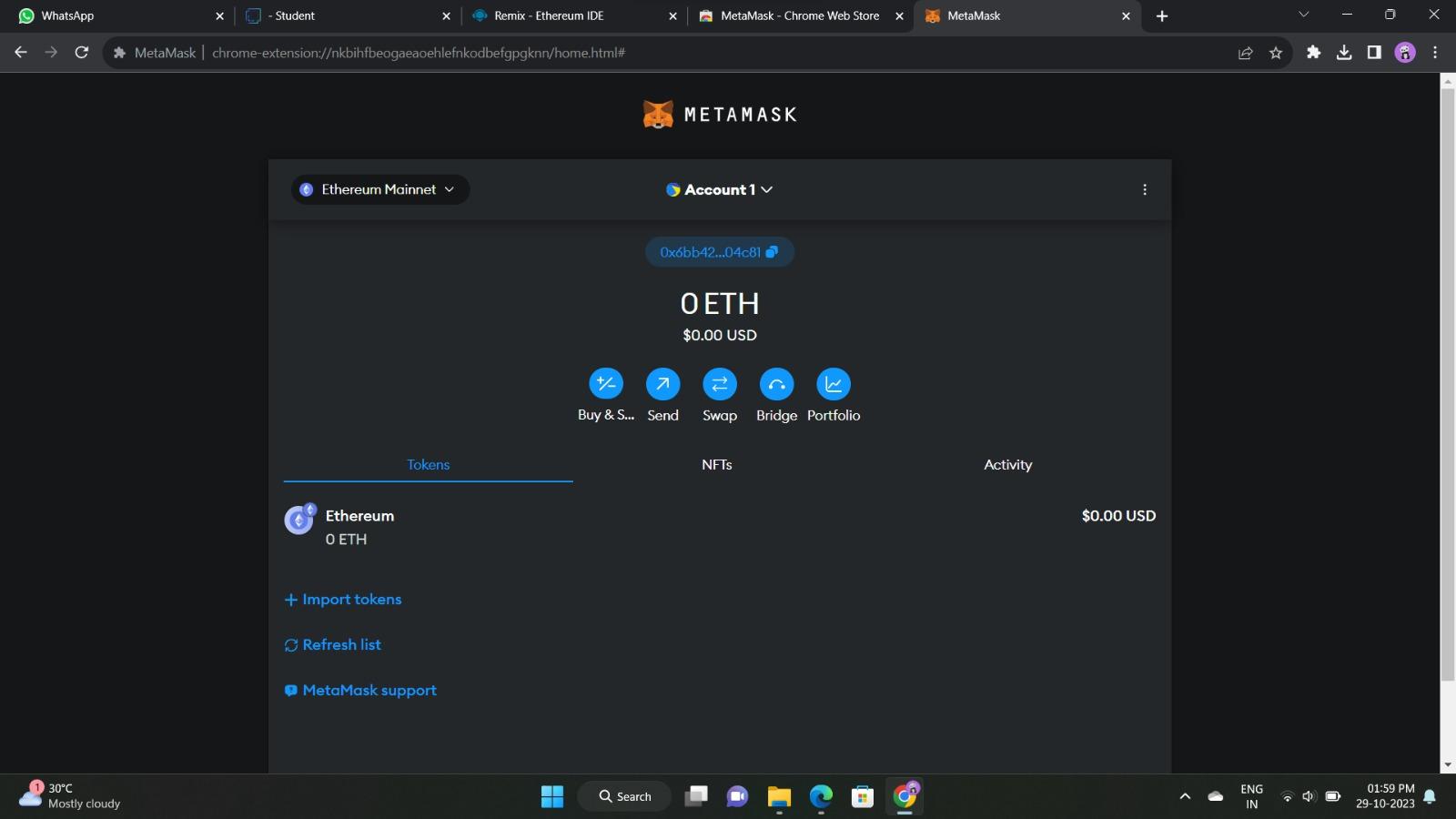
Effective performance testing helps identify and mitigate performance bottlenecks and ensures that the food tracking system can handle expected loads while providing a smooth and responsive user experience.

**CHAPTER 9**

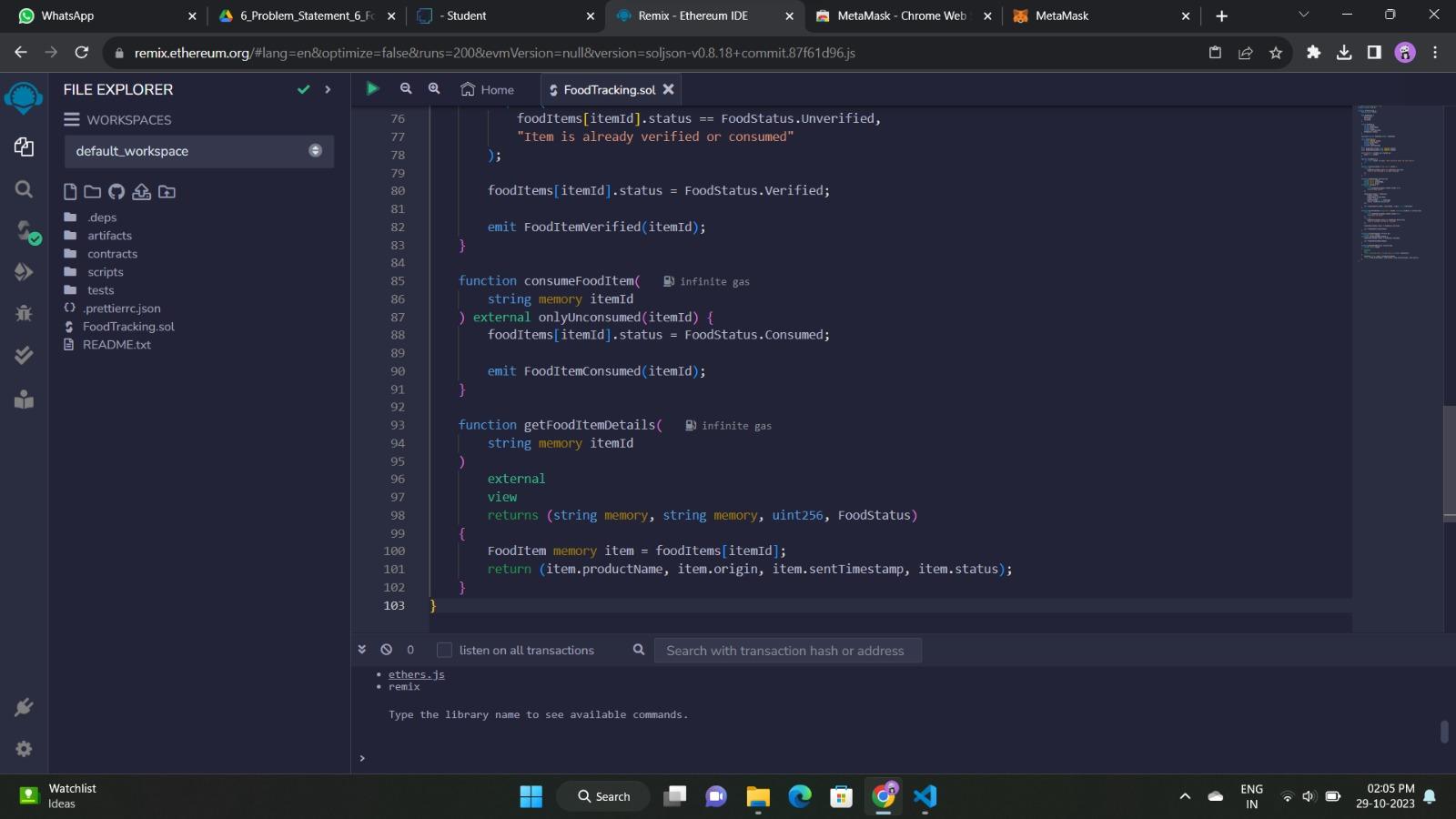
**RESULTS**

**9.1 OUTPUT SCREENSHOTS**

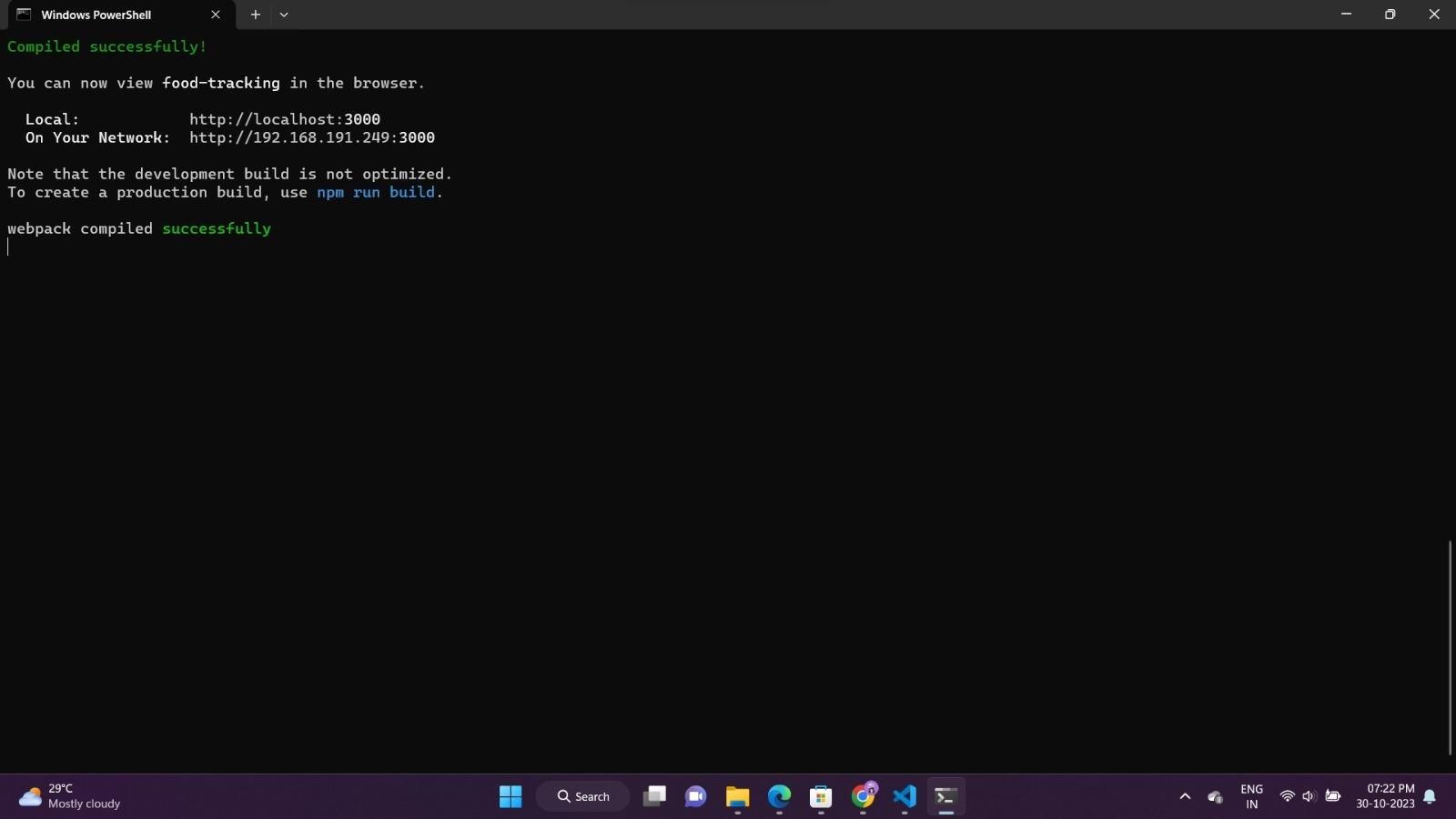
**REMIX IDE – CREATING A SMART CONTRACT**

****

**METAMASK ACCOUNT**

****

**REMIX IDE – DEPLOYED CONTRACT**

****

**TERMINAL OUTPUT SCREEN**

****

**FINAL FRONTEND OUTPUT SCREEN**

**CHAPTER 10**

**ADVANTAGES & DISADVANTAGES**

**Advantages:**

**Improved Nutrition Awareness:** Users become more conscious of their dietary choices and nutritional intake, leading to healthier eating habits.

**Personalized Dietary Guidance:** Systems can provide tailored dietary recommendations and meal plans based on users' goals and preferences.

**Goal Achievement:** Helps users work towards and achieve dietary goals, whether related to weight management, fitness, or specific health needs.

**Efficient Food Logging:** Simplifies the process of logging food intake, offering convenience and time-saving benefits.

**Data-Driven Decisions:** Empowers users to make informed food choices by providing detailed nutritional information for a wide range of foods.

**Progress Tracking:** Allows users to monitor their dietary progress over time, making it easier to measure the effectiveness of dietary changes.

**Customization:** Systems can be customized to accommodate dietary restrictions, allergies, and individual preferences.

**Nutritional Education:** Provides educational resources and insights into the nutritional content of different foods.

**Healthcare Support:** Can be used to support healthcare professionals in monitoring and providing guidance to patients with specific dietary needs.

**Accessibility:** Most food tracking systems are available as mobile apps, making it easy to track meals on the go.

**Disadvantages:**

**User Engagement:** Many users start tracking their food but may not sustain their engagement, leading to incomplete data and limited benefits.

**Data Accuracy:** Users may struggle to accurately input food details and portion sizes, leading to inaccurate nutritional data.

**Food Variability:** Databases may not cover all possible food items or variations, making it challenging to find and log certain meals accurately.

**Privacy Concerns:** Users may have concerns about sharing personal dietary information, especially when using third-party apps or websites.

**Complexity:** Some food tracking systems can be overly complex and challenging to use, which can lead to user frustration.

**Behavior Modification:** While tracking can create awareness, it may not always lead to sustained behavioral change or healthier eating habits.

**Cost:** Some advanced food tracking systems or apps may come with a subscription cost, limiting access for some users.

**User Reliability:** The accuracy of the tracked data relies on the user's commitment and honesty, which may vary.

**Compliance with Special Diets:** Special dietary requirements, allergies, or medical conditions might not be adequately accommodated in standard food tracking systems.

**Device Dependency:** Users may need specific devices, such as smartphones or wearable tech, to effectively track their food, excluding those without access to such technology.

**Integration with Dining Out:** Tracking meals from restaurants or takeout can be challenging, as nutritional information is often not readily available or may not be entirely accurate.

**CHAPTER 11**

**CONCLUSION**

**CONCLUSION:**

In conclusion, a food tracking system serves as a valuable tool in promoting healthier eating habits, enhancing nutrition awareness, and empowering individuals to take control of their dietary choices. By allowing users to log and analyze their food consumption, set dietary goals, and receive personalized recommendations, these systems provide a powerful means to make informed decisions about their nutrition.

However, it's essential to acknowledge that while food tracking systems offer numerous advantages, they are not without their challenges. Issues such as user engagement, data accuracy, privacy concerns, and the potential for complexity can impact the effectiveness and usability of these systems. Furthermore, the sustainability of behavior change remains a challenge, as tracking alone may not guarantee long-term dietary improvements.

Nonetheless, the continuous evolution of technology, the integration of artificial intelligence and machine learning, and a growing focus on user experience are driving innovation in food tracking systems. As these systems become more user-friendly, accurate, and adaptive, they have the potential to play a significant role in improving public health and nutrition.

In the end, the success of a food tracking system lies in its ability to strike a balance between user empowerment, privacy protection, and user-friendly design. It is a promising tool for those seeking a healthier lifestyle, and its continued development and improvement hold the potential to make a positive impact on the well-being of individuals and communities.

**CHAPTER 12**

**FUTURE SCOPE**

**FUTURE SCOPE:**

The future scope for food tracking systems is promising, as these systems continue to evolve and adapt to meet the changing needs of users and leverage advancements in technology. Here are some areas of future growth and development:

**Enhanced Personalization:** Food tracking systems will become even more tailored to individual preferences and dietary requirements. Artificial intelligence and machine learning will play a significant role in providing highly personalized recommendations and meal plans.

**Integration with Wearable Devices:** The integration of food tracking with wearable devices will become more seamless. Wearables will automatically capture data related to physical activity, sleep patterns, and vital signs, providing a more comprehensive view of health and wellness.

**Environmental Impact Tracking:** There will be an increased focus on sustainability. Food tracking systems may incorporate data on the environmental impact of food choices, helping users make eco-friendly dietary decisions.

**Healthcare Integration:** Greater collaboration with healthcare providers will allow for the integration of medical data into food tracking systems. This can provide more holistic support for individuals with specific health conditions or nutritional needs.

**Genetic and Nutrigenomic Insights:** As genetic testing becomes more accessible, food tracking systems may incorporate genetic data to offer personalized dietary recommendations based on an individual's genetic makeup.

**Voice and Conversational Interfaces:** Voice-activated and conversational interfaces will make food tracking even more accessible and convenient. Users can simply speak to their devices to log meals, get recommendations, and receive nutritional information.

**Data Privacy and Security:** Ongoing advancements in data privacy and security measures will be crucial to maintain user trust. Technologies like blockchain may be used to enhance the security and transparency of user data.

**Social Features:** Enhanced social interaction within food tracking systems will encourage users to connect with peers, share recipes, and collaborate on achieving dietary goals. Communities within these systems will provide motivation and support.

**Gamification and Rewards:** Gamification elements will be integrated to motivate and engage users. Challenges, rewards, and achievements will make food tracking more enjoyable and addictive.

**Integration with Ecosystems:** Food tracking systems will integrate more seamlessly with larger health and wellness ecosystems, connecting with fitness apps, healthcare providers, and even grocery shopping services.

**Artificial Intelligence for Food Recognition:** Food recognition technology using AI will allow users to simply take a picture of their meals, and the system will automatically log and analyze the food, making tracking even more effortless.

**Global Food Databases:** Expanding and improving global food databases will ensure the accuracy and availability of nutritional information for diverse cultural and regional cuisines.

**Nutritional Research and Insights:** Food tracking systems will contribute to ongoing nutritional research, providing valuable insights into dietary patterns and their impact on health.

**Government and Corporate Initiatives:** There may be increased collaboration between governments, corporations, and food tracking systems to promote public health and wellness initiatives.

**AI-Powered Health Coaches:** The integration of AI-driven virtual health coaches within food tracking systems will provide users with real-time guidance and support for making healthier choices.

The future of food tracking systems holds significant potential to further empower individuals to make informed dietary decisions, contribute to public health initiatives, and play a central role in personalized health and wellness. As technology continues to advance, these systems will become more intuitive, effective, and integrated into our daily lives.

**CHAPTER 13**

**APPENDIX**

**SOURCE CODE**

**agricultureOnBlockchain.sol**

// 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);

}

}

**Connector.js**

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

const abi = [

{

"inputs": [

{

"internalType": "string",

"name": "itemId",

"type": "string"

}

],

"name": "consumeFoodItem",

"outputs": [],

"stateMutability": "nonpayable",

"type": "function"

},

{

"inputs": [],

"stateMutability": "nonpayable",

"type": "constructor"

},

{

"anonymous": false,

"inputs": [

{

"indexed": true,

"internalType": "string",

"name": "itemId",

"type": "string"

}

],

"name": "FoodItemConsumed",

"type": "event"

},

{

"anonymous": false,

"inputs": [

{

"indexed": true,

"internalType": "string",

"name": "itemId",

"type": "string"

},

{

"indexed": false,

"internalType": "string",

"name": "productName",

"type": "string"

},

{

"indexed": false,

"internalType": "string",

"name": "origin",

"type": "string"

},

{

"indexed": false,

"internalType": "uint256",

"name": "sentTimestamp",

"type": "uint256"

}

],

"name": "FoodItemSent",

"type": "event"

},

{

"anonymous": false,

"inputs": [

{

"indexed": true,

"internalType": "string",

"name": "itemId",

"type": "string"

}

],

"name": "FoodItemVerified",

"type": "event"

},

{

"inputs": [

{

"internalType": "string",

"name": "itemId",

"type": "string"

},

{

"internalType": "string",

"name": "productName",

"type": "string"

},

{

"internalType": "string",

"name": "origin",

"type": "string"

}

],

"name": "sendFoodItem",

"outputs": [],

"stateMutability": "nonpayable",

"type": "function"

},

{

"inputs": [

{

"internalType": "string",

"name": "itemId",

"type": "string"

}

],

"name": "verifyFoodItem",

"outputs": [],

"stateMutability": "nonpayable",

"type": "function"

},

{

"inputs": [

{

"internalType": "string",

"name": "",

"type": "string"

}

],

"name": "foodItems",

"outputs": [

{

"internalType": "string",

"name": "itemId",

"type": "string"

},

{

"internalType": "string",

"name": "productName",

"type": "string"

},

{

"internalType": "string",

"name": "origin",

"type": "string"

},

{

"internalType": "uint256",

"name": "sentTimestamp",

"type": "uint256"

},

{

"internalType": "enum FoodTracking.FoodStatus",

"name": "status",

"type": "uint8"

}

],

"stateMutability": "view",

"type": "function"

},

{

"inputs": [

{

"internalType": "string",

"name": "itemId",

"type": "string"

}

],

"name": "getFoodItemDetails",

"outputs": [

{

"internalType": "string",

"name": "",

"type": "string"

},

{

"internalType": "string",

"name": "",

"type": "string"

},

{

"internalType": "uint256",

"name": "",

"type": "uint256"

},

{

"internalType": "enum FoodTracking.FoodStatus",

"name": "",

"type": "uint8"

}

],

"stateMutability": "view",

"type": "function"

},

{

"inputs": [],

"name": "owner",

"outputs": [

{

"internalType": "address",

"name": "",

"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 = "0x3F356E409d16E85554bF66Ace8227d423eEFf023"

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

**App.js**

import logo from './logo.svg';

import './App.css';

import Home from './Page/Home'

function App() {

return (

<div className="App">

<header className="App-header">

<Home />

</header>

</div>

);

}

export default App;

**index.js**

import React from 'react';

import ReactDOM from 'react-dom/client';

import './index.css';

import App from './App';

import reportWebVitals from './reportWebVitals';

const root = ReactDOM.createRoot(document.getElementById('root'));

root.render(

<React.StrictMode>

<App />

</React.StrictMode>

);

// If you want to start measuring performance in your app, pass a function

// to log results (for example: reportWebVitals(console.log))

// or send to an analytics endpoint. Learn more: https://bit.ly/CRA-vitals

reportWebVitals();

**reportWebVitals.js**

const reportWebVitals = onPerfEntry => {

if (onPerfEntry && onPerfEntry instanceof Function) {

import('web-vitals').then(({ getCLS, getFID, getFCP, getLCP, getTTFB }) => {

getCLS(onPerfEntry);

getFID(onPerfEntry);

getFCP(onPerfEntry);

getLCP(onPerfEntry);

getTTFB(onPerfEntry);

});

}

};

export default reportWebVitals;

**setupTests.js**

import '@testing-library/jest-dom';

**GITHUB & PROJECT DEMO LINK**

* GitHub Link:
* Demo Link: https://drive.google.com/drive/folders/1rhvUr3bXkJmQMxKVURtL9MUahiA2i58u?usp=drive\_link