FACULTY OF COMPUTER SCIENCE AND INFORMATION TECHNOLOGY

DEPARTMENT OF COMMUNICATION TECHNOLOGY AND NETWORK

SEMESTER 6 2024/2025

COURSE: BACHELOR OF COMPUTER SCIENCE (COMPUTER NETWORK) WITH HONORS

COURSE CODE: CNS4504-1: (BLOCKCHAIN)

LECTURER: DR. NORMALIA BINTI SAMIAN

TITLE: BLOCKCHAIN IN THE FOOD SUPPLY CHAIN

NAME	MATRIC NUMBER
NURUL FIRZANA ASYIQIN BINTI ABDULLAH	216663
SITI NAJMA IZZATY BINTI MUHD ASYRAF	216922
AYA SOFEA BINTI ROSLI	214973
FATIN NASUHA BINTI ZURAIDI	215190
AZYAN SYAZWANI BINTI SETIA	215014

TABLE OF CONTENT

1.0 Introduction

The agricultural food supply chain is a vital system that ensures the delivery of food products from farms to consumers in a safe, efficient, and trustworthy manner. These products include raw agricultural goods such as vegetables, fruits, grains, and livestock-based items that undergo several stages of processing and distribution before reaching the market. As the agricultural supply chain becomes more complex due to modern trade and multiple intermediaries, challenges such as contamination, fraud, and the loss of traceability continue to emerge.

Traditional systems for managing agricultural food supply chains often rely on disconnected methods of data recording. These include paper-based records or outdated software tools that lack real-time connectivity and transparency. As a result, information about where food comes from, how it was handled, and whether it meets safety standards is often missing or difficult to verify. This creates risks for consumers, producers, and regulatory agencies, particularly during food recalls or contamination incidents.

Blockchain technology offers a promising solution to overcome these challenges. As a secure, decentralized, and tamper-resistant digital ledger, blockchain enables each transaction in the supply chain to be recorded and verified in a way that cannot be altered. This enhances traceability, ensures data integrity, and promotes accountability among all participants. In the context of agriculture, blockchain can support farm-to-table tracking by capturing important data from planting and harvesting to processing, packaging, and retail distribution.

This project proposes a blockchain-based traceability system designed specifically for the agricultural food supply chain. By using smart contracts and a web-based platform, the system allows key actors such as farmers, manufacturers, distributors, and retailers to input and track product data at every stage. Consumers can access verified information about agricultural products by scanning QR codes linked to the blockchain. The goal is to improve transparency, strengthen food safety, and build greater trust in the agricultural supply process.

2.0 Problem Statement

The agricultural food supply chain is currently affected by several significant challenges that reduce its efficiency and reliability. One of the most critical issues is the lack of transparency across different stages of the supply network. Food products originating from farms pass through various intermediaries, and each stage introduces a risk of miscommunication or data loss. Without a clear and verifiable record, it is difficult to determine the origin of agricultural food products or confirm whether they have been handled according to safety and quality standards.

Another ongoing problem is the use of outdated and disconnected systems for tracking food movement. Many agricultural supply chain participants still rely on paper documents or basic computer systems that are not integrated or secure. This makes the data vulnerable to errors, tampering, and delays. In the event of contamination, tracing the affected products back to the source becomes time-consuming and unreliable, increasing the risk to public health.

Consumers today demand greater transparency about their food, especially when it comes to agricultural origin and handling processes. However, most supply chains fail to provide this level of visibility, which results in reduced trust and difficulty in verifying authenticity. Regulatory bodies also face challenges in monitoring and enforcing standards due to inconsistent and incomplete data.

There is a clear need for a robust and transparent solution that supports real-time tracking and verification of agricultural food products. By implementing a blockchain-based system, the agricultural supply chain can be transformed into a secure and transparent ecosystem where all data is traceable, verifiable, and accessible. This project aims to develop such a system to address these critical gaps and enhance the overall integrity of agricultural food distribution.

3.0 Objectives

The primary objectives of implementing blockchain in food supply chain include:

I. To enhance transparency and data accessibility by eliminating information silos by providing a shared, decentralized ledger where all stakeholders such as farmer, processor, retailer, consumer and raw material, can record and access verified data and enable real-time visibility into the product journey for all authorized participants.

II. To ensure immutable and tamper-proof records by replacing paper-based and manual logs with digitally signed, blockchain-stored records to prevent fraud, mislabeling, and human error and prevent unauthorized modifications using cryptographic hashing and consensus mechanisms.

III. To accelerate traceability for food safety and recalls by reducing time required to track products during food safety incidents or recalls by enabling instant product history retrieval via QR codes or batch IDs and automate contamination alerts using smart contracts to trigger recalls when safety thresholds such as temperature breach are violated.

IV. To strengthen consumer trust and uphold brand integrity by providing consumers with verifiable product information like origin, certification, and expiry dates via mobile apps or QR scans.

V. To ensure regulatory compliance and simplify audits by streamlining the compliance process by embedding safety and quality standards into blockchain records, enabling automated reporting and easier verification by regulatory authorities.

4.0 System Design and Architecture

4.1 System Overview

The system consists of four core components

1. Truffle Framework: compiles and deploys smart contracts
2. Deployed Smart Contract: executes business logic on the blockchain
3. Contract API Call: facilitates interaction between the client and blockchain
4. Client Web Browser: provides a user interface for interactions

The workflow follows a frontend-backend model where the client interacts with the blockchain via smart contract calls.

4.2 Component Breakdown

1. Truffle Framework
   Purpose: a development environment for Ethereum smart contracts.
   Functions:
   - Compiles Solidity code into bytecode.
   - Manages deployments using migration scripts.
   - Support testing

   Output: generates ABI (Application Binary Interface) and contract address for interaction.
2. Deployed Smart Contract
   Purpose: immutable, self-executing code on the blockchain
   Functions:
   - Stores critical data
   - Enforces business rules

   Example: a food supply chain contract storing batch origins and transit history

3. Contract API Call
   Purpose: bridges the frontend and blockchain
   Tools:
   - web3.js / ethers.js (JavaScript libraries)
   - MetaMask (for wallet integration).

Process:

1. Client initiates a request (e.g., getProductDetails).
2. The request is signed via MetaMask.
3. Blockchain processes the transaction (read/write).

4. Client Web Browser

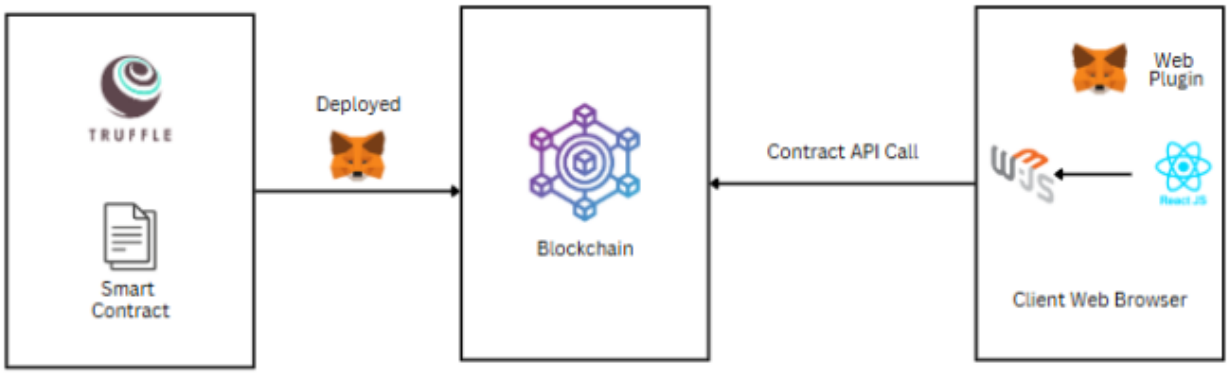
   Purpose: user-facing interface

   Functions:

   - Displays blockchain data (e.g., product history).
   - Submits transactions (e.g., adding new product).

   Technologies: React.js

4.3 Workflow



1. Development Phase:
   - Smart contract is written in Solidity.
   - Truffle compiles and deploys it to the blockchain (testnet/mainnet).

2. Deployment Phase (MetaMask + Blockchain)
   - The smart contract is deployed to the blockchain after development.
   - Metamask acts as a crypto wallet and bridge between the browser and the blockchain. It signs and sends the deployment transaction.
   - The blockchain stores the deployed smart contracts on-chain, making it immutable and accessible to anyone.

3. User Interaction Phase:
    - Client accesses the web app, which connects to the contract via web3.js.
    - For state-modifying actions (e.g., addRecord), MetaMask prompts for a signature and gas payment.

4. Data Retrieval Phase:
    - Read-only calls (e.g., verifyProduct) fetch data without gas fees.
    - Results are displayed on the UI.

In summary:

1. Smart contracts are built and deployed using Truffle + MetaMask.
2. They are hosted on the blockchain (e.g., Ethereum or Ganache).
3. React.js front end interacts with the blockchain using web3.js and MetaMask.
4. Users on the browser send interactions (e.g., add records, view profile) via API calls to the contract.

5.0 Smart Contract Development

The FoodSupplyChain smart contract is developed in Solidity (^0.8.19) to facilitate a transparent and traceable food supply chain system. It will track the entire journey of the food item from harvest by a farmer to the processor and retailer and lastly, delivery to the consumer. This process will ensure accountability for each stage. This contract will enforce strict access control, lifecycle state transitions, and event logging to maintain the integrity of operations.

5.1 Functionalities

1. **Decentralized Item Registry:**

   Each food item is represented using a structured Item that stores relevant metadata including product name, creation timestamp, state, and involved actors.

2. **Lifecycle Management:**

   A custom enum State is used to show each stage of a food item. The Stage in enum State are Harvested, Processed, Packaged, ForSale, Sold and Delivered. These transitions are strictly enforced to ensure logical workflow progression.

3. **Access Control via Modifiers**

   The contract applies access restrictions using modifiers like verifyCaller, inState, paidEnough, and checkValue to enforce business logic and protect against unauthorized actions.

4. **Automatic Payment Handling**

   When a consumer buys a product, the smart contract ensures the correct amount is sent to the retailer, and any overpayment is refunded automatically.

5. **Event Logging:**

   Each state change emits a corresponding event, which are Harvested, Processed, Packaged, ForSale, Sold, Delivered for transparency and integration with off-chain applications.

5.2 Smart Contract Functions

Function	Description	Actor
harvestItem()	Registered a new product by name	Farmer
processItem()	Marks the product as processed	Same Farmer
packageItem()	Marks the product as packaged	Processor
sellItem()	Puts item up for sale at a set price	Retailer
buyItem()	Allows consumer to purchase item with payment	Consumer
confirmDelivery()	Confirm item was received	Same consumer
fetchItem()	Return item details	Any user

5.3 Modifiers Used

Modifiers	Description
onlyOwner	Restricts access to the contract deployer, which is currently only used for ownership info.
verifyCaller(address)	Ensures the caller matches the expected address such as farmer, processor, retailer, and consumer.
inState(uint256, State)	Verifies the item is in the expected lifecycle state.
paidEnough(uint256)	Checks if the buyer sent enough Ether.
checkValue(uint256)	Refunds any excess payment to the buyer after purchase.

5.4 Development and Deployment Environment

The smart contract was developed using Solidity V0.8.19. This deployment environment includes Truffle, Ganache, Metamask, and Ethers.js for implementation and testing. The contract was tested on both Ganache local blockchain and the Sepolia testnet. This will ensure functionality across different network environments. This smart contract was compiled using solc and the deployment process was completed successfully. All key event logs were verified, confirming that the contract transitions and interactions are functioning as intended.

The contract delivers an efficient and traceable system for managing agricultural food products. It clearly models the product journey using immutable blockchain logic and ensures fair payments, secure and safe transitions, and real-time auditability through events. It is well-suited for real-world use in farm-to-table or agri-retail systems that need lightweight yet accountable traceability.

6.0 Web3 Interface Design

6.1 Role-Based UI Functionalities

By acting as a conduit between users and the Ethereum blockchain, the Web3 interface enables safe and easy interaction with smart contracts. The system makes use of Web3.js for blockchain interactions, React.js for the UI, and MetaMask for transaction signature and wallet integration. The role-based user interface features offered to each stakeholder are listed below:

1. Farmers: Register new products

   Through the registration of fresh products, farmers play a fundamental role in the supply chain. They are shown a simple and easy-to-use interface where they may enter important details such as the product name, kind (fruits, veggies), place of origin, and harvest date. After being submitted, this data is associated with a distinct batch ID and saved on the blockchain. This guarantees that every product has an identifiable digital identity that can be tracked over the course of its life. Farmers may also browse batches that have already been registered and track their progress as the product passes through the supply chain thanks to the user interface.

2. Transporters: Log delivery status

   Moving commodities between various stakeholders, such as from a farmer to a processor or from a processor to a retailer, is the responsibility of transporters. Transporters can use the interface to update the system with real-time delivery data, including the route travelled, the present state of the items, and the timestamps for pickup and drop-off. An unchangeable delivery history is ensured by the instantaneous recording of every log entry on the blockchain. This logging capability is essential for confirming if goods such as those that are sensitive to temperature were carried in a safe and appropriate manner.

3. Retailers: Confirm receipt and quality

   Before goods are delivered to customers, retailers serve as the last expert inspection point. Retailers may examine their delivery history, verify product receipt, and check incoming product batches in their own dashboard. Additionally, they can include comments on the quality of the received goods, including if they are acceptable, broken, expired, etc. After verification, this data is updated on the blockchain to indicate the batch's ownership and current status. This validation process guarantees that any problems can be linked to their origin and helps maintain responsibility.

4. Consumers: Scan QR to trace history

   Simplicity and transparency are key design elements of the consumer-facing interface. A printed QR code produced by the system is included with every product batch. Customers use a mobile device to scan this code, which takes them to a webpage that details the product's whole journey from farm to processor, transporter, retailer, and finally into their hands. Verified information is shown on the interface, including the actors participating, the origin, the timestamps of each transaction, and any certificates that are connected. This encourages educated buying decisions and increases customer trust.

6.2 UI Screenshots and Mockups

To support the above functionalities, various UI mockups and real screenshots are included in the Appendix section of the report. These visuals cover:

1. Landing Page – Connect Wallet to Get Started

   Before users can use the platform, they must first link their MetaMask wallet, which is shown on this screen. This permits safe, blockchain-based transactions and guarantees identity authentication. For all user roles, it serves as the initial checkpoint in creating a trusted session.

2. Landing Page – Login Dashboard and Unregistered Account Notice

   Users are sent to the login dashboard after connecting their wallet. The user is notified that they need to register if the wallet address is not linked to any existing accounts in the system. This makes sure that only verified users can carry out role-specific tasks on the blockchain and helps stop unwanted access.

3. Register Page – Role Selection and Registration (Example: Farmer)

   Selecting a job (Farmer, Processor, Retailer, or Consumer) and entering other information like their name and contact details is how new users must register. This article shows how role-based access control is started at the interface level and documents the user onboarding procedure. A farmer is registered in the example, and the information is then saved and validated on the blockchain.

4. Login Page – Through Wallet and Email

   An alternate login procedure that combines email verification and MetaMask wallet authentication is displayed on this page. By ensuring that individuals are individually recognised and permitted access to role-specific dashboards and smart contract operations, dual authentication provides an additional degree of protection.

5. Farmer Page – Product Registration Form and Batch List

   For farmers, this is the primary interface. It consists of:

   - A form for entering product information, including name, kind, and harvest site.
   - A table providing the current status of every batch that has been registered.

     The smart contract processes each submission, and the batch that is produced is permanently saved on the blockchain. This illustration shows how the Web3 interface makes it simple for farmers to manage their product data.

6. Processor Page – Batch Details and Action Options

   Through the processor interface, users can:

   - View batches of inbound products.
   - Update packaging and other processing steps.
   - Give merchants ownership.

     In order to guarantee the traceability of food products during transformation, the screenshot illustrates how processors monitor and manage raw materials. This is a direct reflection of the smart contract's processItem() and packageItem() operations.

7. Processor (continued) and Retailer Page – Confirm Delivery

   Access to retailer data, which indicates the future destination of batches, is available through the continuing processor interface. The retailer dashboard, however, enables users to:

   - View and verify that the processed batches were received.
   - Comment about the state or quality of the product.

     The product's ownership and condition are updated on-chain after confirmation.

8. Retailer (continued) Interface – Additional Product Details

   Retailers may access more product details and engagement options on this page. It confirms that quality comments, product tracking, and handover events are managed and safely recorded on the blockchain.

9. Consumer Page – QR Code Scanner Interface

   Customers use a mobile device to scan QR codes that are affixed to product packaging. A traceability page that results from this displays:

   - The path taken by the product from farmer to store.
   - Timestamps for every phase.
   - Any certifications (such organic or halal) that have been submitted.

     This image demonstrates how consumers may use transparency tools and confirms that getProductDetails() works as intended.

10. QR Code Detail View – Complete Product History

    This interface shows the product's complete transaction history when a QR code has been scanned. This contains timestamps, process steps, all prior owners, and any documents connected by IPFS. This demonstrates how blockchain technology makes product history accessible and impenetrable while also bolstering consumer trust.

11. Error Handling – Console Messages and Role Validation

    This page demonstrates the system's error-handling capabilities. Using the console logs, one can:

    - When a transporter attempts to register a product, for example, display access violations.
    - Display validation issues in the form (such as missing input fields).

      By ensuring that unauthorised activities are denied with unambiguous feedback, these logs preserve data integrity and uphold the security of smart contracts.

7.0 Implementation and Features

Throughout the food supply chain, the system's numerous state-of-the-art features guarantee transparency, data security, and real-time traceability. The Web3 interface and smart contract are closely linked with these capabilities.

1. Transaction history with timestamps

   On the blockchain, a timestamp is appended to each state change that occurs during the product lifetime, including harvesting, processing, packaging, shipping, and receiving. The Ethereum network automatically generates these timestamps, which are unchangeable. To create a chronological timeline, the interface obtains and shows these timestamps in the user interface (for example, for both customers and shops). This makes it simpler to audit or look into food recalls by confirming the path taken by each product.

2. QR code generation for each product batch

   The system immediately creates a QR code that contains the product's batch ID once the farmer registers the product. The merchandise has this QR code written on it. A specific user interface page displaying the product's origin, travel, and ownership history is shown when it is scanned. As soon as the product is created, a QR code is generated that points straight to the getProductDetails() method.

3. IPFS integration for uploading certificates

   The system incorporates IPFS (InterPlanetary File System) to enable documents such as health and safety certifications. A file uploaded by a user (such as a farmer or processor) is saved on IPFS, and the content hash that is returned is noted on-chain. This guarantees that the certificate is publicly verifiable and impenetrable. An option to upload files is included in the interface, and the uploaded assets are linked to the product description page.

4. Real-time updates using Ethereum events

   Ethereum events like ItemHarvested, OwnershipTransferred, and ItemReceived are released by the smart contracts. Web3.js listeners in the frontend record these events and automatically update the UI, saving users from having to reload the page. For instance, the batch status on the dashboard instantly changes to "Received" when a store receives a delivery, and customers who scan the QR code will likewise see the updated information instantly.

5. Role-Based Access Control and Input Validation

   There are stringent role-based restrictions in the system. Only customers may access traceability data, only retailers can verify delivery, and only registered farmers are able to register items. The system registers the error and prohibits any effort by a user to carry out an unauthorised activity (such as a transporter attempting to add a new product). Before being submitted, all forms are further verified to make sure the data is correct and full.

6. Automatic Payment and Refund Handling

   Despite not being a major component of the UI mockups, the smart contract has functionality to control Ether transfers. Funds are automatically sent to the vendor when a customer purchases a goods. The contract refunds the excess money if it is overpaid using the checkValue and paidEnough modifiers. Without the involvement of third parties, this guarantees equity and streamlines financial management.

The implementation closely conforms to the interface design. Smart contract operations and frontend components are directly mapped to every user interaction, including product registration, delivery confirmation, and QR code scanning. These characteristics work together to provide a safe, open, and user-friendly way to track agricultural products from farm to table.

8.0 Testing and Results

The accuracy, usability, and performance of the blockchain-based food supply chain traceability system were assessed through extensive testing. Testing focused on validating smart contract functionality, role-based user interaction, and frontend integration, using a simulated local blockchain environment with **Truffle**, **Ganache**, **MetaMask**, and **React.js.**

8.1 Smart Contract Validation

Truffle's integrated testing framework was used to thoroughly test the smart contracts which control the blockchain-based food supply chain system. The platform's basic functionality consists of these contracts, which automate product registration, track product movement, maintain access control, and preserve data integrity for all supply chain participants. In order to enable real-time contract execution, event monitoring, and state validation without network delays, the testing environment used Ganache to simulate interactions on a local Ethereum blockchain.

The main functions tested were

1. registerProduct()
2. transferOwnership()
3. getProductDetails()
4. addParticipant().

The **registerProduct()** function allowed suppliers to register new raw materials by providing details such as product name and origin. After registration, the system created an individual ID for every batch, permanently saved the data on the blockchain, and sent out an event to indicate that the registration was successful. This verified that the event log could be retrieved for frontend changes and that new product data was accurately logged.

The **transferOwnership()** function was tested next. As a raw material moved through the distribution phases, this feature allowed supply chain actors such as manufacturers and distributors to change its ownership and status. In order to provide traceability, the test verified that the function correctly changed the raw material's current owner and kept a historical log of all transfers.

The **getProductDetails()** function allowed retrieval of a complete traceability record for any registered raw material using its batch ID. When a consumer scanned the product's QR code,

this feature gave back all relevant data, including timestamps, origin, and transfer history. Testing revealed that the returned data was correct, comprehensive, and consistent with the blockchain ledger's entries.

Another important function, **addParticipant()**, was also validated. This feature made it possible for system administrators to add new users and give them a role, such as supplier, manufacturer, or distributor. In order to confirm that role-based logic and access control were operating as anticipated, tests made sure that each participant could only access features associated with their assigned role.

A crucial component of the testing procedure was the validation and verification of security. Role modifiers like onlySupplier, onlyManufacturer, and onlyDistributor were utilized in the contract to limit critical functions to the appropriate user roles. The smart contract appropriately denied requests and provided clear error messages when unauthorized roles tried to carry out activities. To make sure that duplicate entries, improper formats, and empty fields were not allowed, input validation was also verified. For instance, the system prohibited enrolling a participant with an existing address or providing a blank product name.

Finally, throughout each exchange, Ethereum occurrences were tracked. Web3 listeners were used to record and show the custom events that were broadcast by functions like **RegisterProduct()** and **transferOwnership()** in the user interface. This added another layer of verification for successful operations and enabled the system to reflect changes in real time.

Overall, smart contract testing confirmed that the logic was solid, secure, and well-aligned with the needs of a real-world agricultural supply chain. The system handled role-based actions accurately, maintained reliable trace data, and provided a strong foundation for the Web3 interface.

8.2 User Interface Testing

The blockchain-based food supply chain system's user interface was extensively tested to make sure it interacts with the Ethereum blockchain's smart contracts. React.js was used to build the frontend, Web3.js and MetaMask were used for integration, and Ganache was utilized as the local blockchain environment to simulate transactions. Every user position, including distributor, producer, supplier, and consumer, was evaluated according to how it actually handles raw materials during the supply chain process.

For the supplier role, testing focused on the ability to register new raw material batches into the blockchain system. Suppliers could fill out a form with product names, provenance, and other relevant data. Following submission and approval via MetaMask, a distinct batch ID was established and the data was permanently stored on the blockchain. The successful data registration and UI-blockchain synchronization were subsequently confirmed when the new raw material entry showed up in the product list. As their raw materials progressed through the supply chain, suppliers could likewise monitor their progress.

The manufacturer (processor) role was tested next. Receiving raw materials from suppliers and turning them into completed or semi-finished goods fell within the authority of manufacturers. Through the interface, they were able to update the blockchain records, transfer ownership, and track product handling details. This stage verified that the raw material to processed product transition was transparent and traceable. Incoming batches were also shown on the manufacturer's portal, which also enabled status confirmation prior to processing.

The distributor (retailer) role was tested by confirming the receipt of finished goods from the manufacturer. Through the system, the distributor might label things as received, examine the history, and access the batch details. The blockchain modified the ownership and timestamp according to the distributor's confirmation of receipt. This feature tracked each transaction between parties, confirming the smart contract's capacity to maintain supply chain integrity.

The consumer role was tested through the QR code scanning functionality. Consumers could use their smartphone to scan a unique QR code that was affixed to the product package. The complete lifespan of the product from its supplier-sourced raw material origin through the production and distribution phases to its ultimate retail form was retrieved by calling the **getProductDetails()** function. Transparency was promoted and customers were able to confirm the product's safety and authenticity thanks to the clear display of the traceability data.

Lastly, the admin role provided access to key management features. Administrators were allowed to designate roles like distributor, manufacturer, or supplier and register participants in the system. Error logs, contract data including smart contract address and deployment progress, and system monitoring tools were also incorporated in the admin dashboard. This made it possible to effectively monitor the system's functioning and guarantee safe and responsible user administration.

In conclusion, the user interface testing verified that every supply chain member could accurately interact with the blockchain in accordance with their designated role. From registering raw materials to delivering the final product to consumers, every action was recorded, reflected in the UI, and validated on the blockchain to ensure transparency, traceability, and trust across the entire food supply chain.

8.3 Test Case Table Based on Smart Contract Flow

Test Case	Feature	Role	Action	Expected Result	Status
1	Register Raw Material Batch	Supplier	Submit new raw material such as Chili	Raw material registered with ID and data	✅ Pass
2	View Registered Products	Supplier/Admin	View product list in system	Raw material batch details are displayed	✅ Pass
3	Add Supply Chain Participants	Admin	Register Manufacturer, Distributor, etc.	Participants stored with assigned roles	✅ Pass
4	Transfer Ownership (Deliver)	Manufacturer	Log material as processed/delivered	Ownership updated to manufacturer	✅ Pass
5	Confirm Product Receipt	Distributor	Mark item as received	Ownership updated to distributor	✅ Pass
6	Scan Product Traceability	Consumer	Scan QR code on packaging	Full product history is shown	✅ Pass
7	Error Handling (Wrong Role)	Unauthorized User	Try action without permission	Action rejected with error	✅ Pass

8	Input Validation	All Roles	Submit form with empty fields	Error message shown	✅ Pass

This test verified whether a supplier could successfully register a new batch of raw materials such as Chili in the system. The data was successfully saved on the blockchain with a distinct batch ID after the necessary information was entered and the form was submitted. This verified that suppliers may use the **registerProduct()** function as intended.

The system was tested to ensure that both suppliers and admins could view the list of registered raw materials. The name, origin, and timestamp of each product were accurately presented by the interface demonstrating that the data retrieval and presentation from the blockchain were operating as planned.

In this test, the admin role was used to add new users such as manufacturers and distributors to the system. The feature, which stored participant data and assigned roles, operated as planned. By doing this, it was made sure that only authorized users could carry out actions.

Once raw materials were registered, the manufacturer tested the ability to receive and log the delivery. The blockchain was updated to reflect the successful transfer of product ownership from the supplier to the manufacturer. This illustrates how useful the **transferOwnership()** function is.

After production, the item was marked as received in order to test the distributor's function. The ownership status was modified to reflect the new holder after the action was noted on the blockchain. At the distribution stage, this guaranteed accountability and traceability.

Customers used a QR code linked to the products to test the system. The item's complete history, from raw material registration to the last delivery, was shown when it was scanned. This demonstrated that traceability information was transparent, comprehensive, and available through the **getProductDetails()** function.

This test aimed to imitate a user carrying out an unauthorized action, such a distributor attempting to register a raw material. The action was properly blocked by the system, which also showed the relevant error notice. This demonstrated that role-based limitations were operating as intended.

In order to verify input validation, forms with blank or missing fields were finally submitted for a variety of positions. The system effectively blocked submission and showed error messages, demonstrating that appropriate input verification was used to preserve the integrity and quality of the data.

9.0 Conclusion

The implementation of blockchain technology in the food supply chain addresses a range of long-standing issues, particularly in relation to transparency, traceability, and food safety. Traditional systems often lack a centralized source of truth, making it difficult to trace the origin of food products, verify authenticity, or respond swiftly to contamination events. By adopting blockchain, a decentralized, immutable ledger for every transaction and movement of goods is securely recorded, creating a verifiable audit trail accessible to all stakeholders.

This project successfully developed a smart contract using Solidity, deployed through Truffle and Ganache, and integrated it with a user-friendly frontend built with React and Web3.js. The platform enables role-based interactions for farmers, manufacturers, distributors, retailers, and consumers, allowing them to register, supply, order, and trace products efficiently.

Beyond technical implementation, the system aligns with global trends toward digital transformation in agriculture and food industries. It empowers consumers with transparent access to product history, boosts trust in food sources, and assists regulatory bodies in enforcing compliance. This fosters a safer, more accountable, and sustainable food ecosystem, setting a strong foundation for future innovation in supply chain management.

10.0 Recommendations

I. Integrate IoT for Real-Time Data Collection

Integrating IoT sensors can significantly enhance the quality of data stored on the blockchain. For example, temperature and humidity sensors used during transportation can automatically record environmental conditions to ensure that food safety standards are maintained. These real-time readings can also activate specific smart contract rules. For instance, the system can prevent the distribution of products if the recorded conditions do not meet the required safety thresholds.

II. Pilot Test in a Real Supply Chain Environment

Before full-scale deployment, the system should be tested in a controlled, real-world setting. Collaborate with local farms, food manufacturers, and retailers to gather user feedback and identify system limitations, such as performance bottlenecks or data entry errors.

III. Deploy on a Public Blockchain or Layer 2 Solution

While development on a local blockchain using Ganache is sufficient for testing, deploying on a public blockchain (like Ethereum mainnet or a Layer 2 solution such as Polygon) would ensure greater transparency, immutability, and scalability. A public deployment also allows for easier integration with other blockchain applications.

IV. Improve UI/UX for Broader Adoption

A more intuitive user interface would encourage adoption among non-technical users, particularly small-scale farmers and traditional retailers. Features such as multilingual support, tooltips, and interactive tutorials can significantly improve the onboarding process.

11.0 References

Duan, J., Zhang, C., Gong, Y., Brown, S., & Li, Z. (2020, March 1). A content-analysis based literature review in blockchain adoption within food supply chain. *International Journal of Environmental Research and Public Health*. MDPI AG. https://doi.org/10.3390/ijerph17051784

Rana, R. L., Tricase, C., & De Cesare, L. (2021, October 22). Blockchain technology for a sustainable agri-food supply chain. *British Food Journal*. Emerald Group Holdings Ltd. https://doi.org/10.1108/BFJ-09-2020-0832

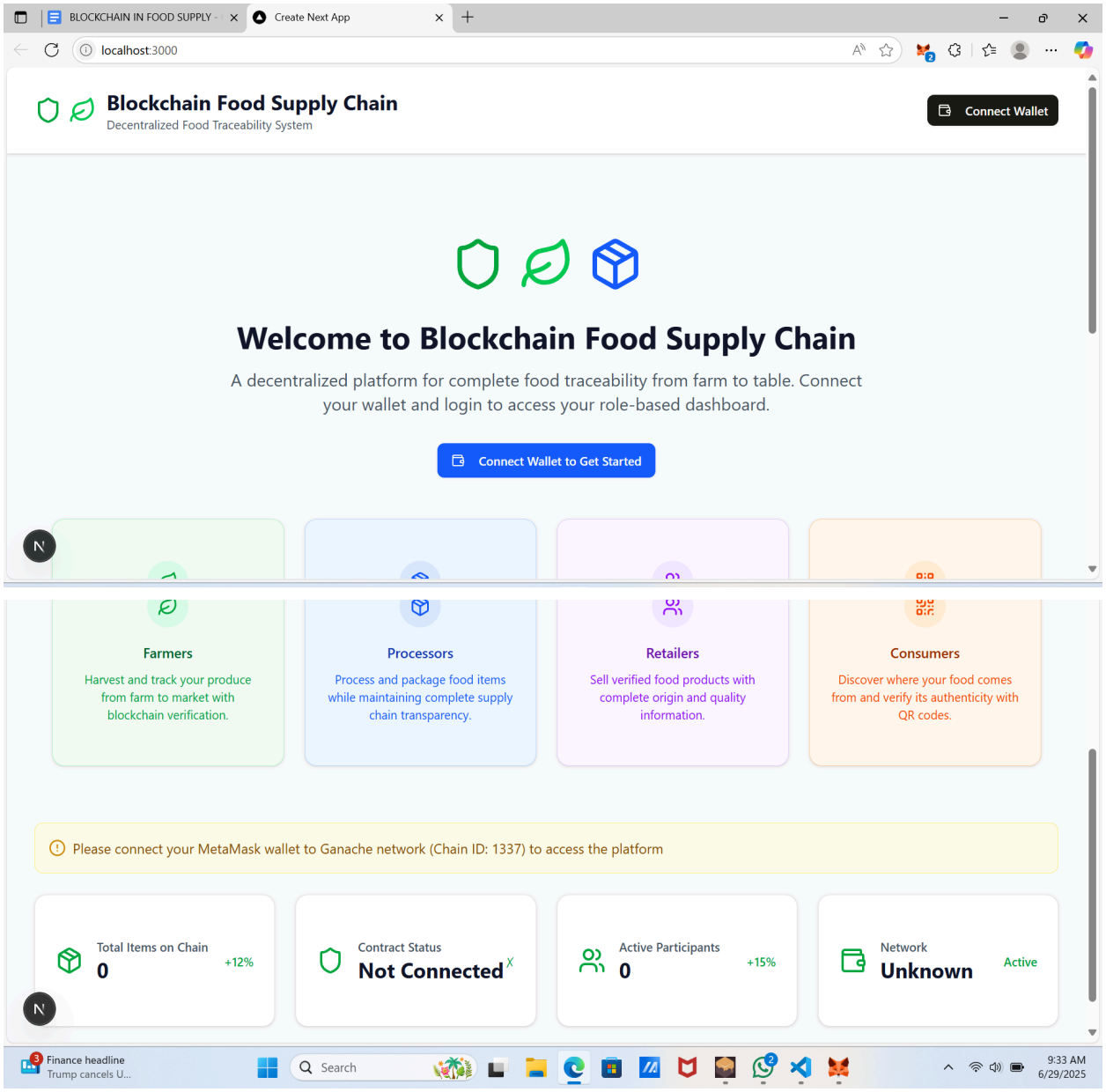
Caro, M. P., Ali, M. S., Vecchio, M., & Giaffreda, R. (2018). Blockchain-based traceability in Agri-Food supply chain management: A practical implementation. In *2018 IoT Vertical and Topical Summit on Agriculture - Tuscany, IOT Tuscany 2018* (pp. 1–4). Institute of Electrical and Electronics Engineers Inc. https://doi.org/10.1109/IOT-TUSCANY.2018.8373021

Shahid, A., Almogren, A., Javaid, N., Al-Zahrani, F. A., Zuair, M., & Alam, M. (2020). Blockchain-Based Agri-Food Supply Chain: A Complete Solution. *IEEE Access*, *8*, 69230–69243. https://doi.org/10.1109/ACCESS.2020.2986257

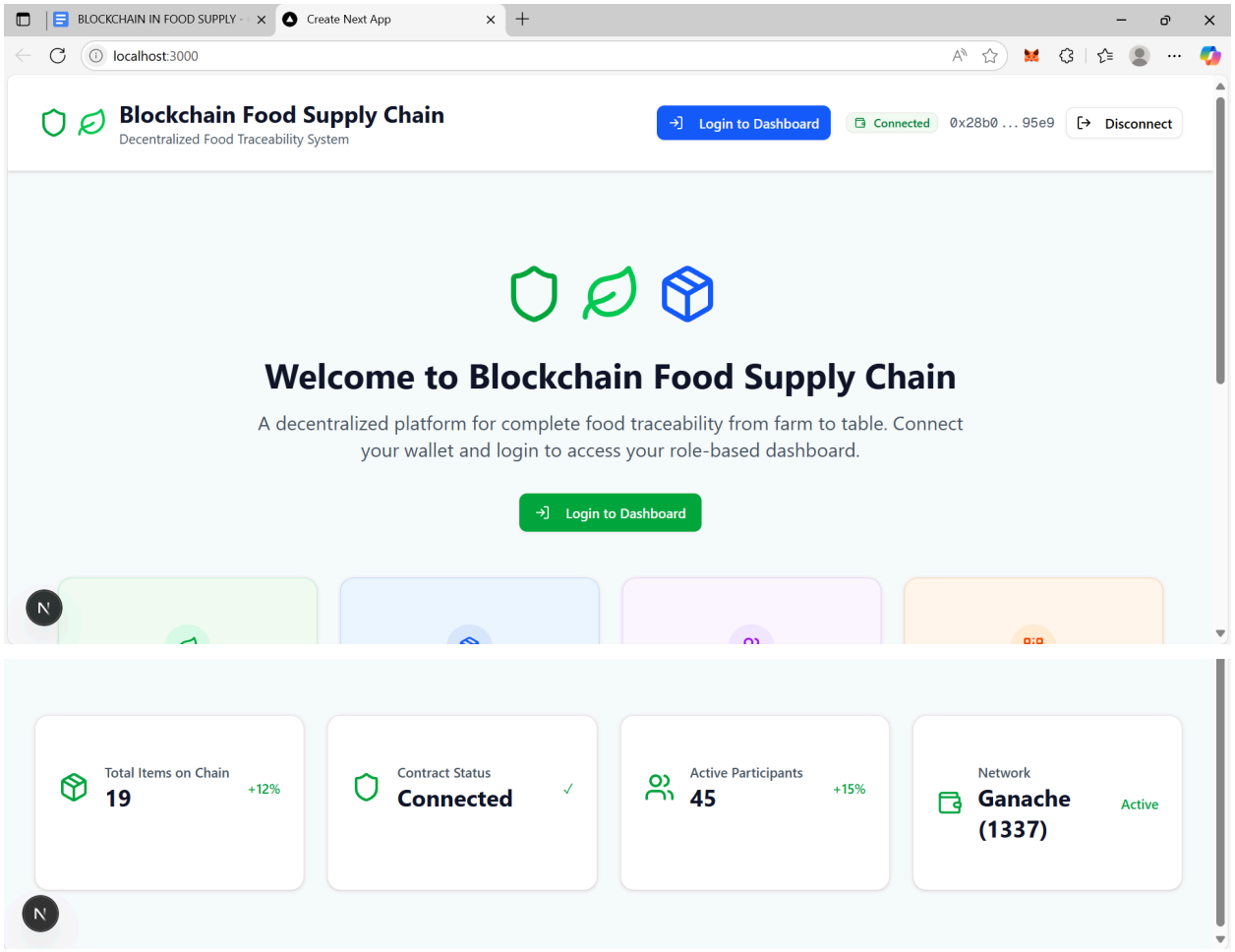
Vu, N., Ghadge, A., & Bourlakis, M. (2023). Blockchain adoption in food supply chains: a review and implementation framework. *Production Planning and Control*, *34*(6), 506–523. https://doi.org/10.1080/09537287.2021.1939902

Appendix

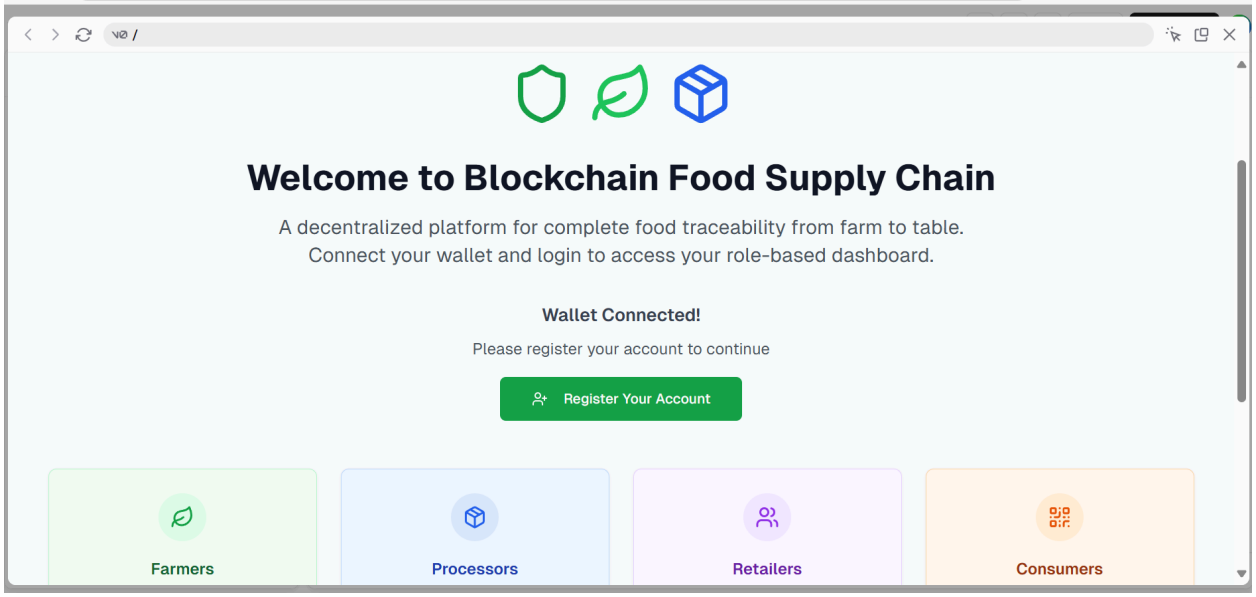
Landing Page - connect wallet to get started



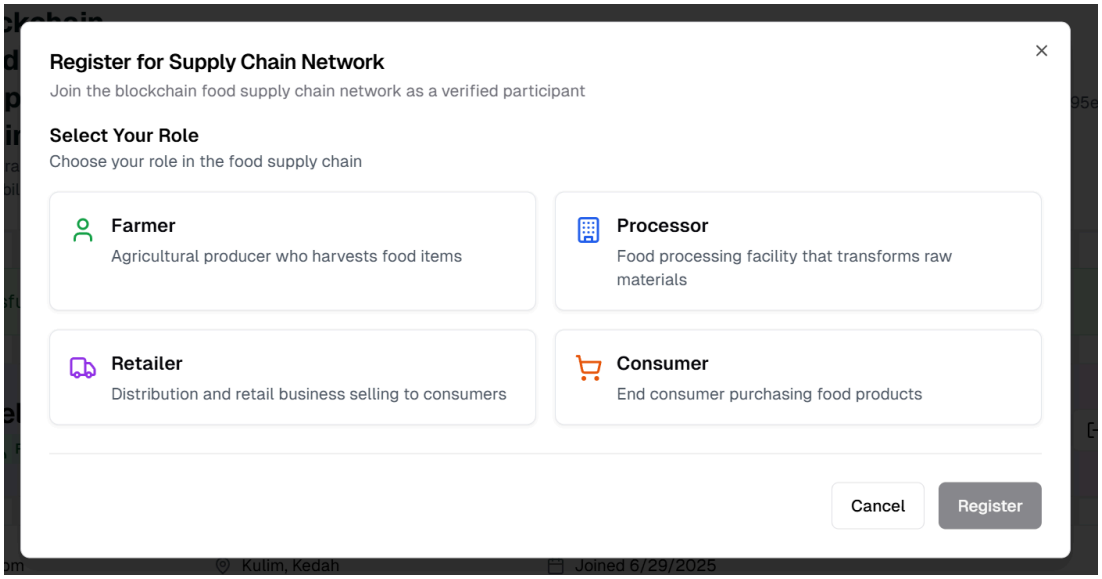
Landing Page - login to dashboard after successful connect to wallet

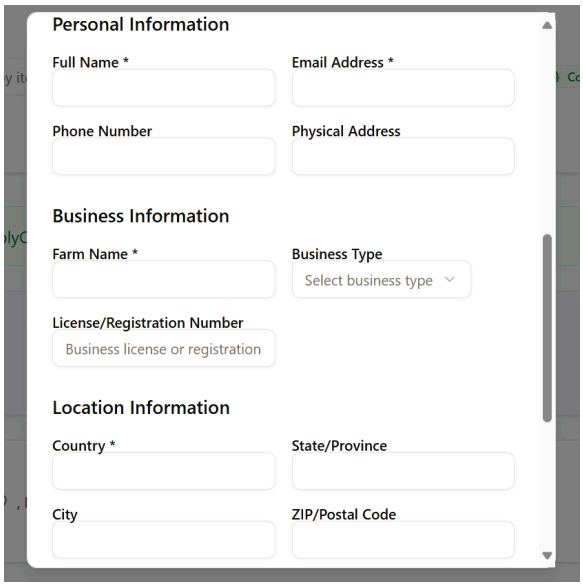


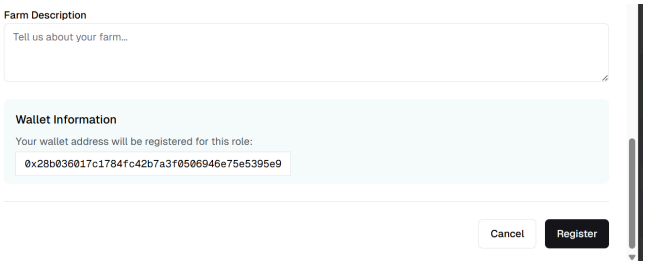
Landing Page- happen when wallet connected but the account use not register

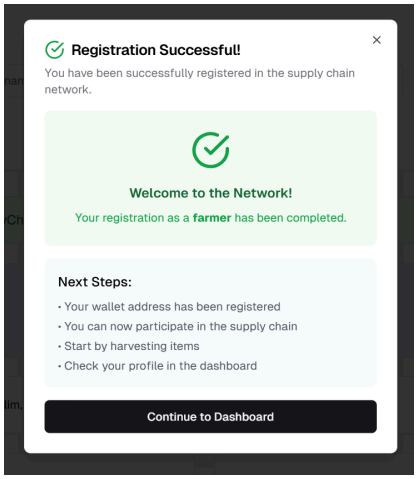


Register Page - choose role and fill in information detail (Chosen Farmer)

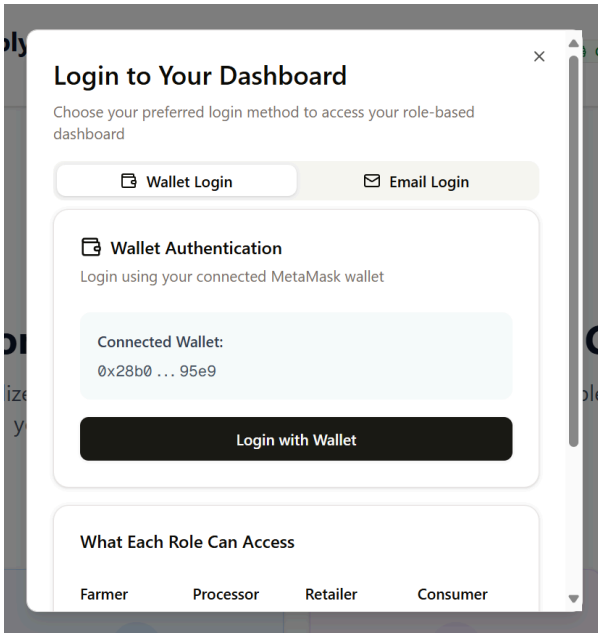


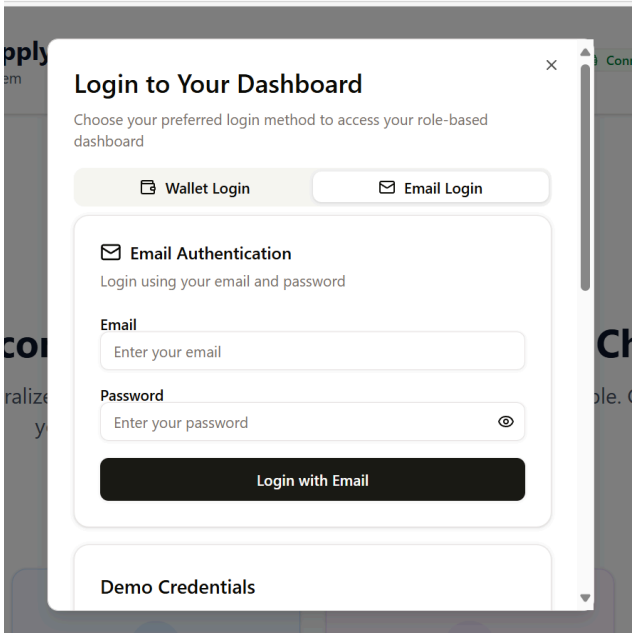


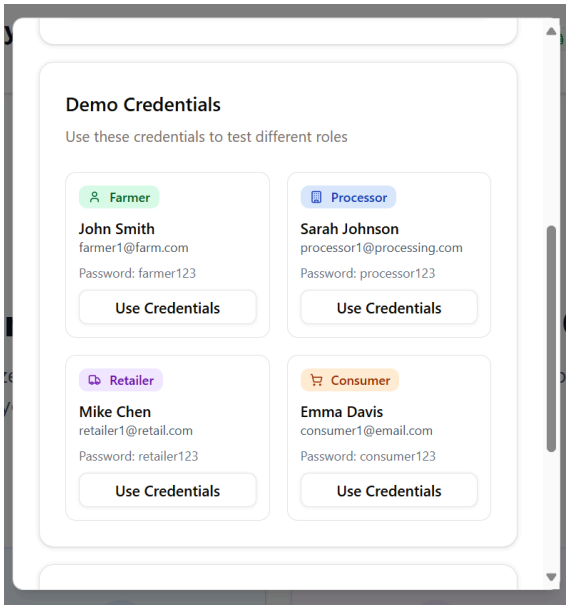


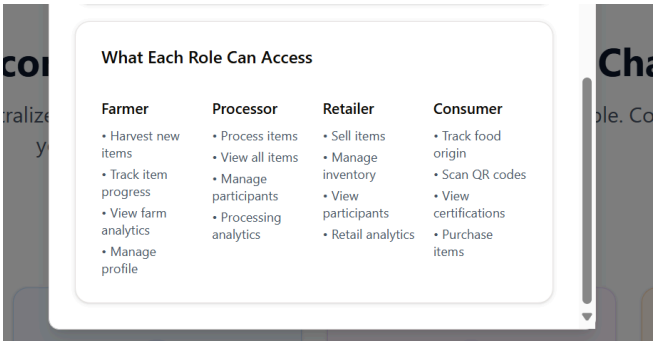


Login page - through wallet and email

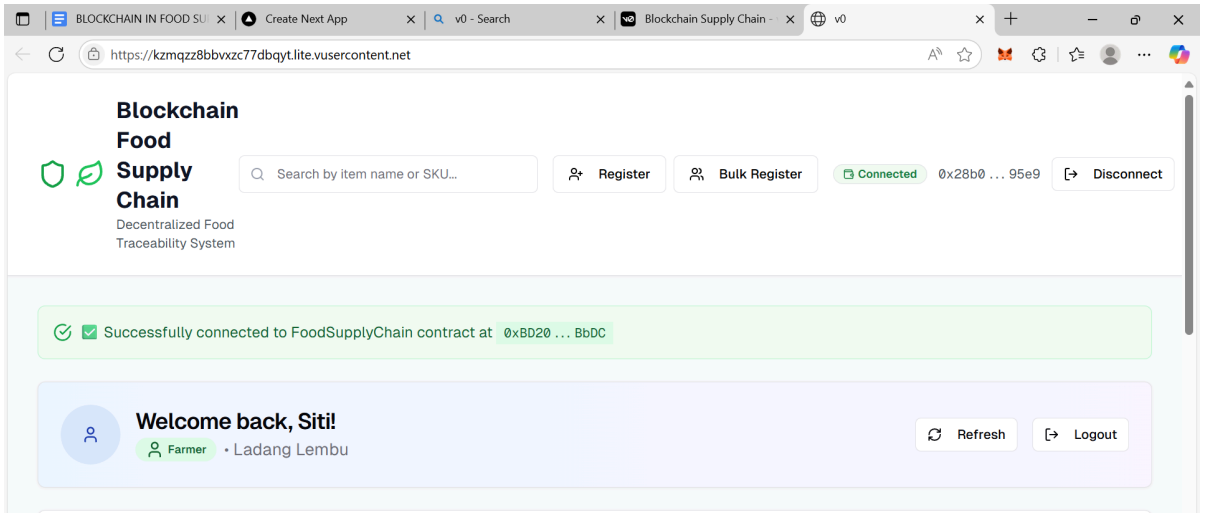


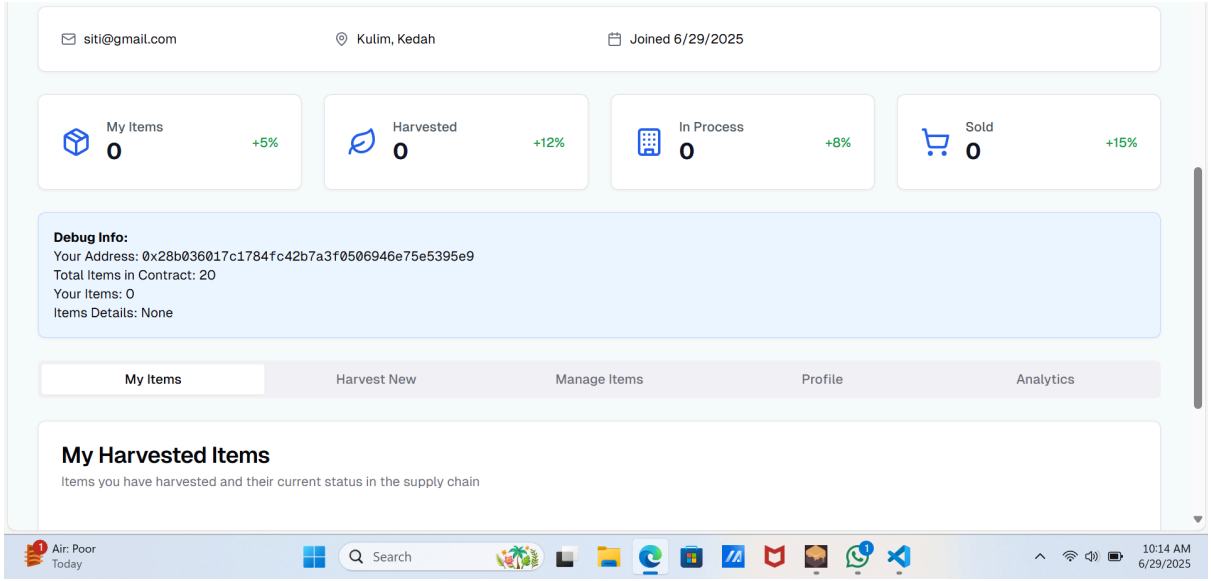


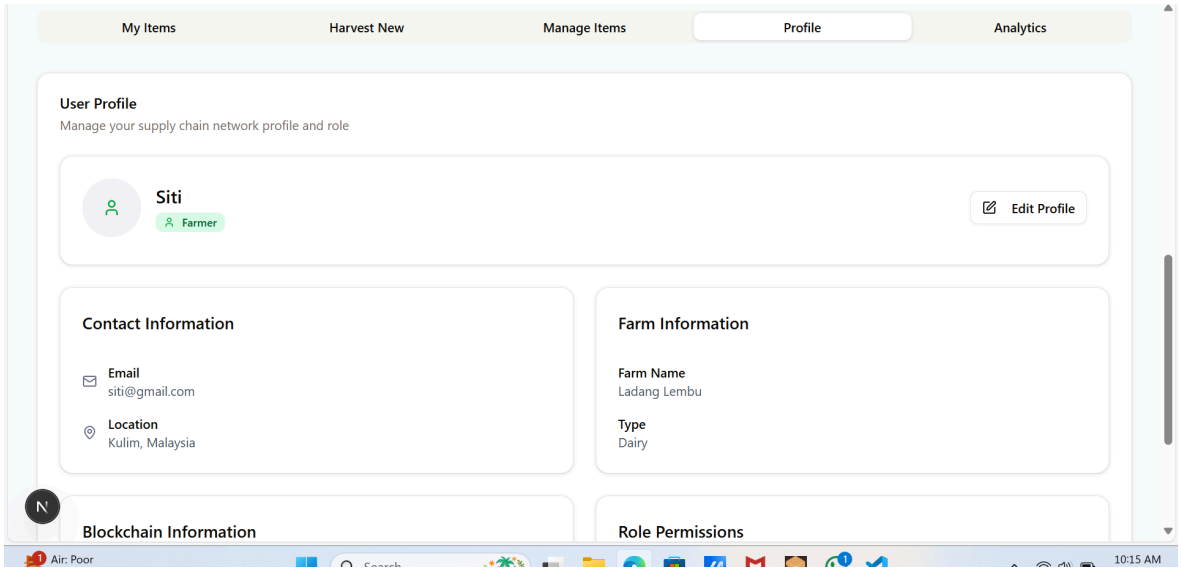




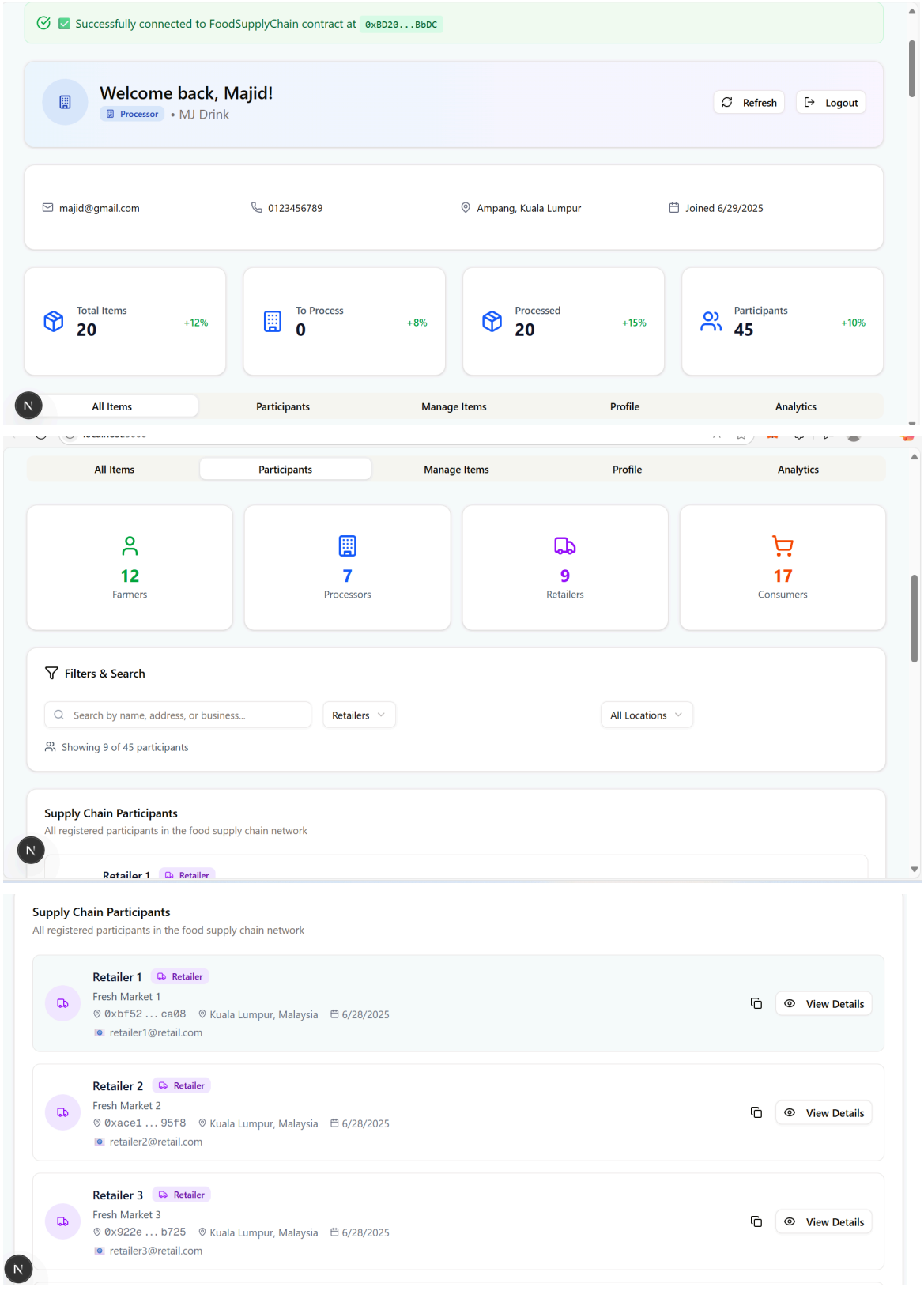
Farmer Page


Blockchain Food Supply Chain
Decentralized Food Traceability System
Search by item name or SKU...
Register
Bulk Register
Connected
0x28b0 ... 95e9
Disconnect
Successfully connected to FoodSupplyChain contract at 0xBD20 ... BbDC
Welcome back, Siti!
Farmer
• Ladang Lembu
Refresh
Logout

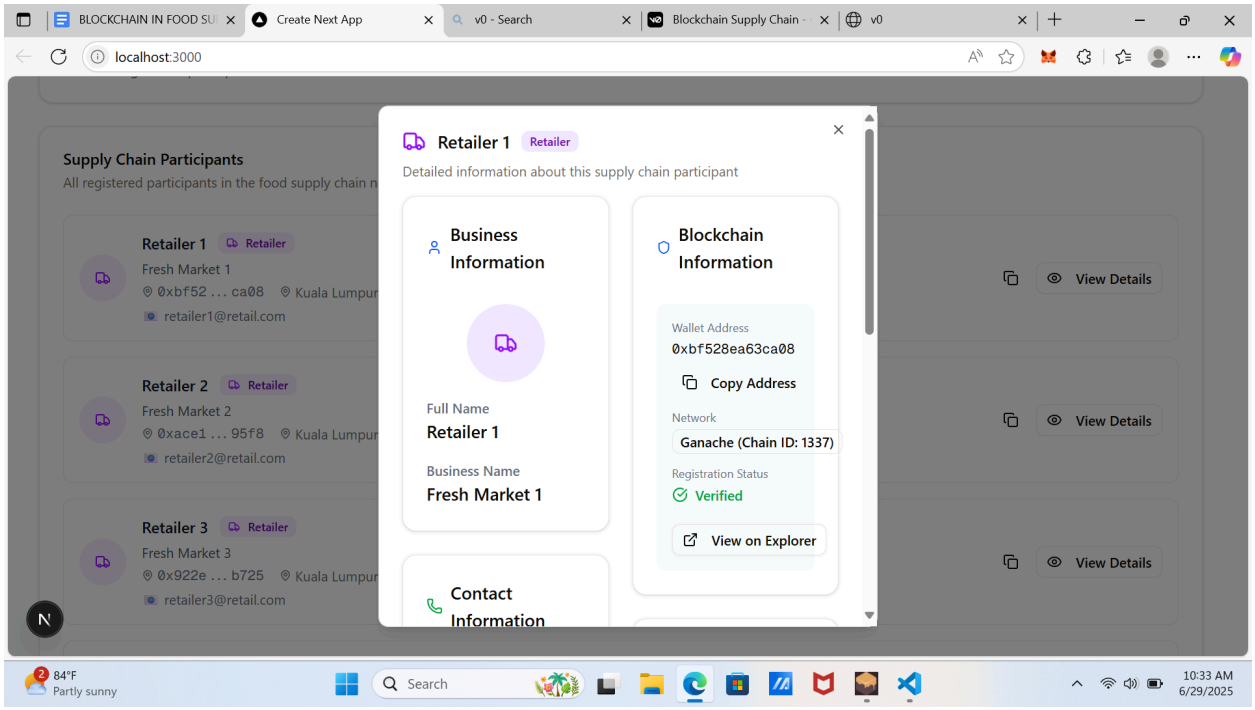

siti@gmail.com
Kulim, Kedah
Joined 6/29/2025
My Items
0
+5%
Harvested
0
+12%
In Process
0
+8%
Sold
0
+15%
Debug Info:
Your Address: 0x28b036017c1784fc42b7a3f0506946e75e5395e9
Total Items in Contract: 20
Your Items: 0
Items Details: None
My Items
Harvest New
Manage Items
Profile
Analytics
My Harvested Items
Items you have harvested and their current status in the supply chain
Air: Poor
Today
Search
10:14 AM
6/29/2025


My Items
Harvest New
Manage Items
Profile
Analytics
User Profile
Manage your supply chain network profile and role
Siti
Farmer
Edit Profile
Contact Information
Email
siti@gmail.com
Location
Kulim, Malaysia
Farm Information
Farm Name
Ladang Lembu
Type
Dairy
Blockchain Information
Role Permissions
Air: Poor
10:15 AM

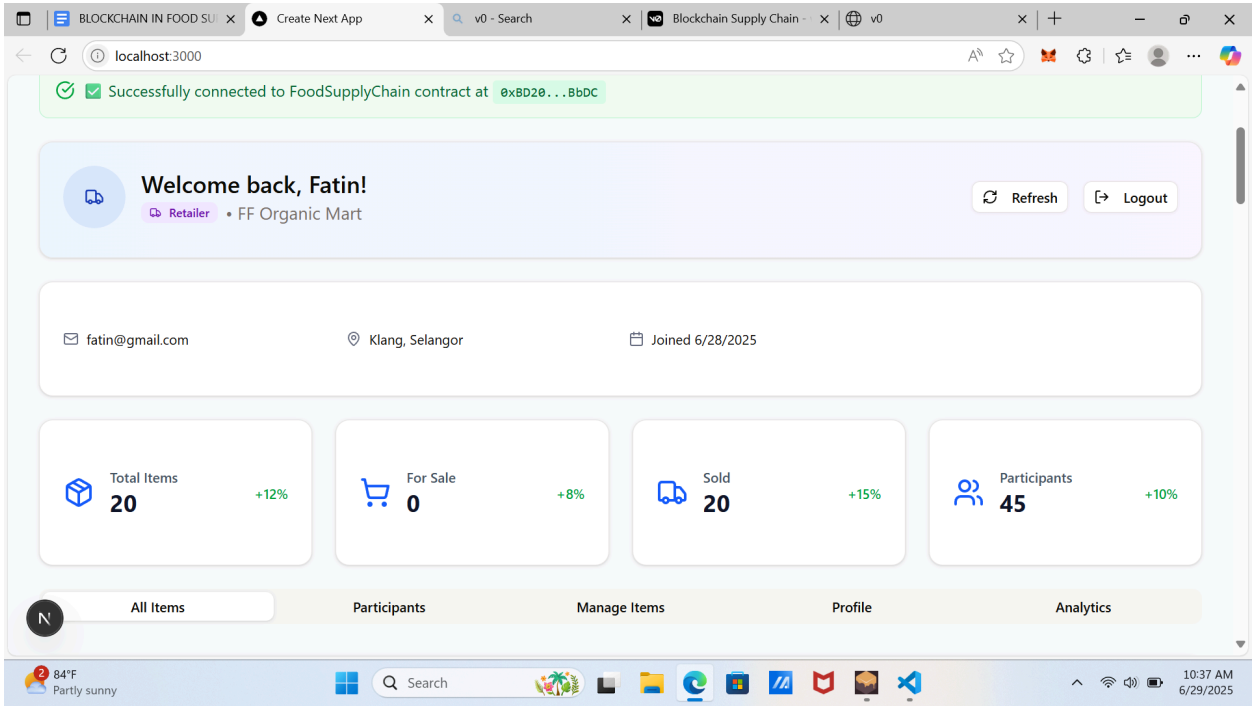
Processor Page

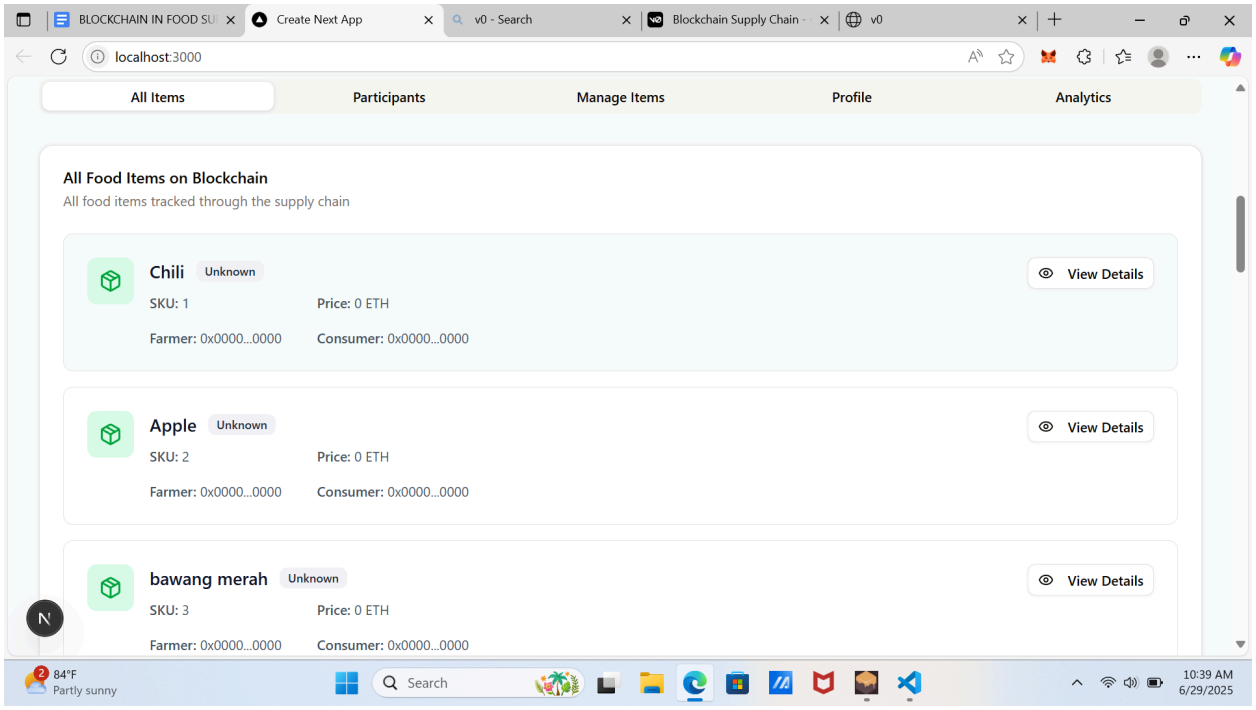


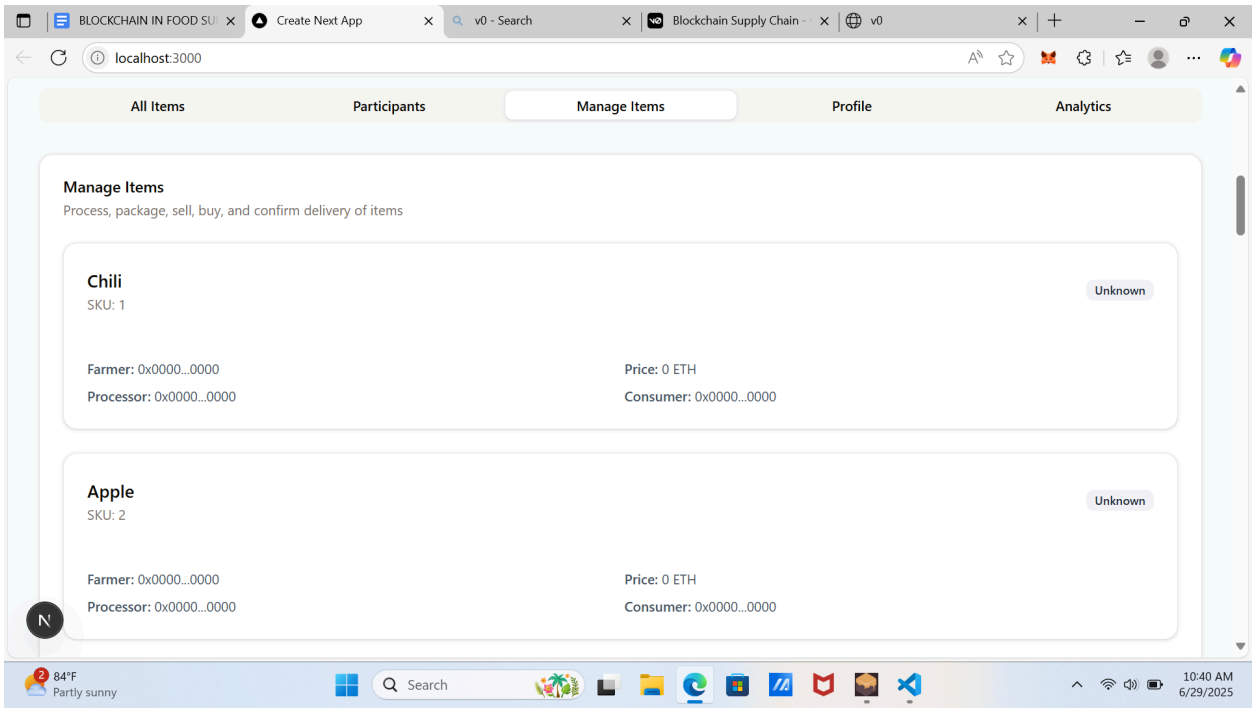
Processor Page (cont) - view detail of retailer



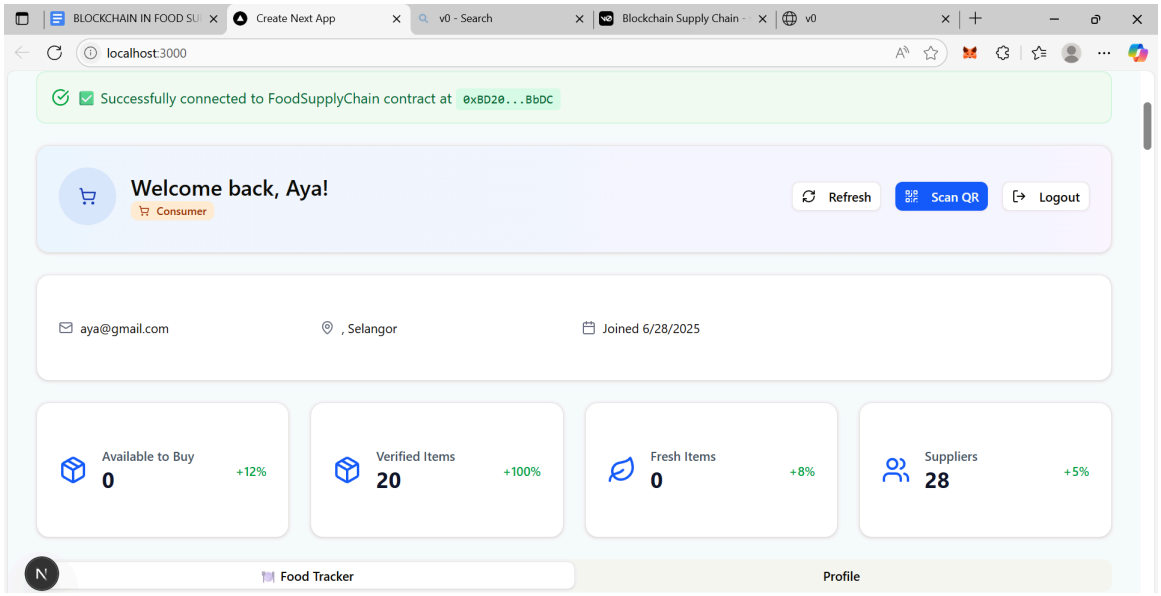
Retailer Page

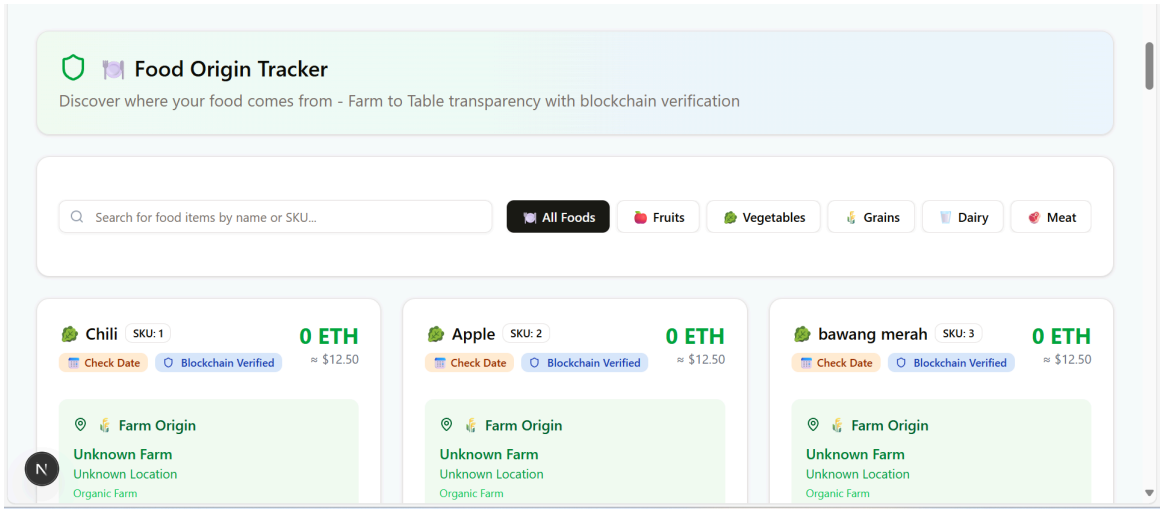


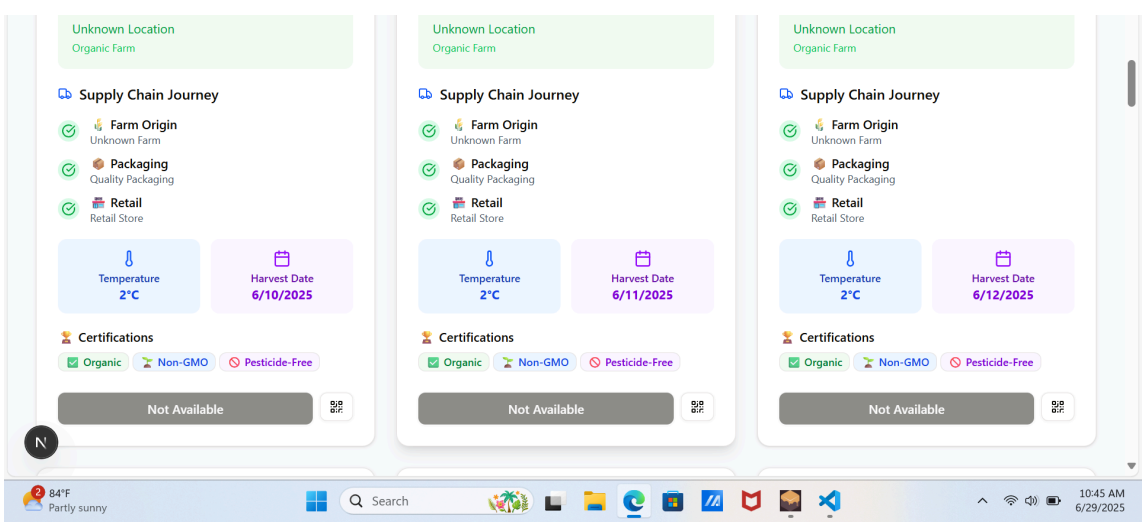

BLOCKCHAIN IN FOOD SU
Create Next App
v0 - Search
Blockchain Supply Chain -
v0
localhost:3000
All Items
Participants
Manage Items
Profile
Analytics
All Food Items on Blockchain
All food items tracked through the supply chain
Chili
Unknown
View Details
SKU: 1
Price: 0 ETH
Farmer: 0x0000...0000
Consumer: 0x0000...0000
Apple
Unknown
View Details
SKU: 2
Price: 0 ETH
Farmer: 0x0000...0000
Consumer: 0x0000...0000
bawang merah
Unknown
View Details
SKU: 3
Price: 0 ETH
Farmer: 0x0000...0000
Consumer: 0x0000...0000
84°F
Partly sunny
Search
10:39 AM
6/29/2025

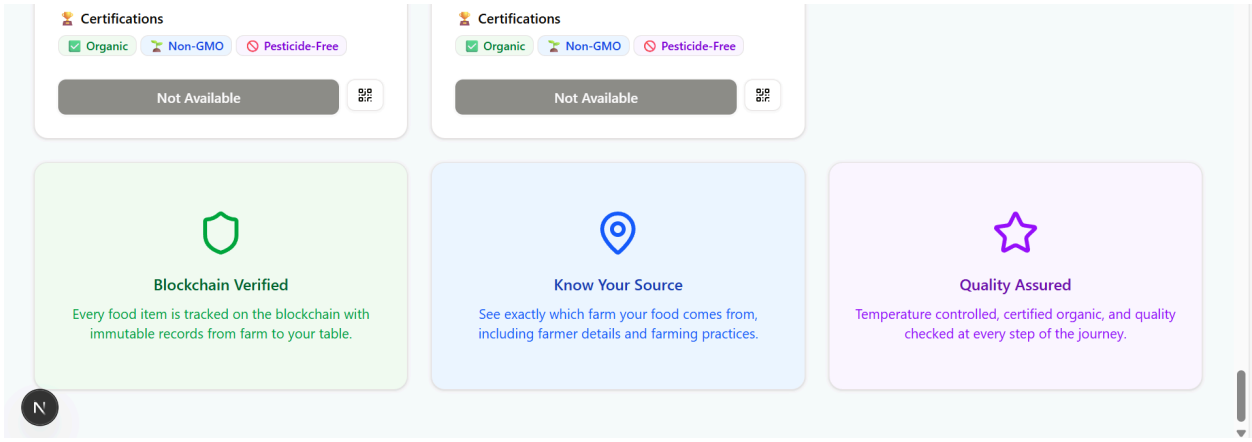

BLOCKCHAIN IN FOOD SU
Create Next App
v0 - Search
Blockchain Supply Chain -
v0
localhost:3000
All Items
Participants
Manage Items
Profile
Analytics
Manage Items
Process, package, sell, buy, and confirm delivery of items
Chili
SKU: 1
Unknown
Farmer: 0x0000...0000
Price: 0 ETH
Processor: 0x0000...0000
Consumer: 0x0000...0000
Apple
SKU: 2
Unknown
Farmer: 0x0000...0000
Price: 0 ETH
Processor: 0x0000...0000
Consumer: 0x0000...0000
84°F
Partly sunny
Search
10:40 AM
6/29/2025

Consumer Page

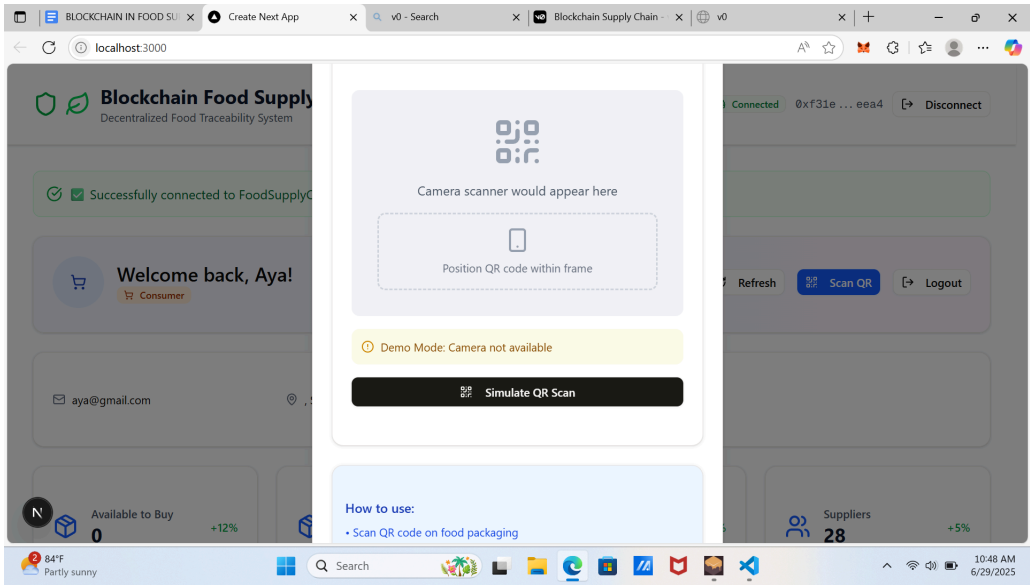


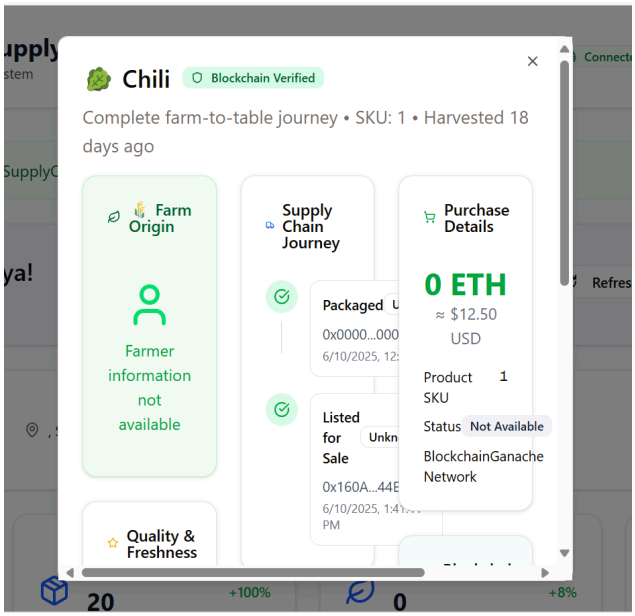






Qr Code Scanner and detail item





Error Handling- anything error will appear at console