BLOCKCHAIN-BASED ORGANIC FOOD TRACEABILITY SYSTEM

(A Supply Chain Management System using Ethereum Smart Contracts)

Project Summary:

Our Blockchain-based Organic Food Traceability System revolutionizes supply chain management by providing transparent tracking of organic food products from producer to consumer. Using Ethereum smart contracts and a web-based interface, this system ensures authenticity and builds trust among all stakeholders.

Problem Statement:

Traditional supply chains often lack transparency, making it difficult to:

- Verify product authenticity
- Track product journey
- Ensure proper handling
- Maintain accountable records

Solution:

Our system addresses these challenges by implementing:

- Blockchain-based record keeping
- QR code product tracking
- Role-based access control
- Real-time status update.

System Architecture

Core Components

1. **Blockchain Layer**
    - Ethereum Smart Contract for product tracking
    - Ganache for local blockchain development
    - MetaMask for wallet interaction
2. **Backend Layer**
    - Flask web server
    - Web3.py for blockchain interaction
    - QR code generation system
3. **Frontend Layer**
    - HTML/CSS interface using Tailwind CSS
    - Role-specific dashboards
    - QR code scanner integration

User Roles

1. **Producer**
    - Register new organic products
    - Generate unique QR codes
    - Initiate product tracking
2. **Shipper**
    - Update shipping status
    - Track in-transit products
    - Manage shipment records
3. **Deliverer**
    - Confirm product deliveries
    - Update final status
    - Verify product authenticity

Development Environment

1. **Required Software**
    - Visual Studio Code
    - Node.js
    - Python 3.x

- Ganache
- MetaMask
- Remix IDE

Step-by-Step Setup Guide

1. Development Environment Setup

A. Basic Software Installation

1. **Visual Studio Code**
   - Download from: https://code.visualstudio.com/
   - Install with default settings
2. **Node.js**
   - Download from: https://nodejs.org/
   - Choose LTS version
   - Install with default settings

   Verify installation:

   - `npm -version`

3. **Python Setup:**
   - `Download Python 3.x`

Install required packages:

```
pip install flask
pip install web3
pip install qrcode
pip install Pillow
```

B. Blockchain Tools Setup:

1. **Ganache Installation:**

    - Download from: https://trufflesuite.com/ganache/
    - Install with default settings
    - Launch and click "QUICKSTART ETHEREUM"
    - Note down:
        - RPC Server URL: http://127.0.0.1:7545
        - Network ID: 1337

2. **MetaMask Setup:**

    - Install MetaMask browser extension
    - Create new network:
        - Network Name: Ganache
        - RPC URL: http://127.0.0.1:7545
        - Chain ID: 1337
        - Currency Symbol: ETH
    - Import Ganache account:
        - Copy private key from Ganache
        - Import into MetaMask

Project Implementation

1. Project Structure Setup

Create a new directory with the following structure:

Copy

```
supply-chain-app/
├── app.py
├── contract_abi.json
├── templates/
│   ├── login.html
│   ├── producer_dashboard.html
│   ├── shipper_dashboard.html
│   ├── deliverer_dashboard.html
│   └── scan_qr.html
└── Supply_Chain.sol
```

2. Smart Contract Development:

A. Contract Creation

1. Open Remix IDE (https://remix.ethereum.org/)
2. Create new file: Supply_Chain.sol
3. Set compiler version to 0.5.16
4. Copy and paste the smart contract code

B. Contract Deployment

1. In Remix IDE:
    - Compile Supply_Chain.sol
    - Select "Injected Web3" environment
    - Ensure MetaMask is connected to Ganache
    - Deploy contract
    - Save contract address
    - Copy ABI to contract_abi.json

C. Key Contract Features

- Product registration
- Role management
- Status tracking
- QR code integration
- Access control

3. Flask Application Setup

A. Initial Configuration

- Create app.py
- Set up Web3 connection
- Configure contract interaction
- Implement user authentication

Project Implementation Explanation

1. Smart Contract Development:

The Smart Contract is the foundation of our system that:

- Creates digital representation of products
- Manages user roles (Producer, Shipper, Deliverer)
- Tracks product status changes
- Stores product information and QR codes
- Controls who can perform which actions

2. Web Application (Flask)

Our Flask backend:

- Connects web interface with blockchain
- Handles user login and authentication
- Manages product creation and updates
- Generates QR codes for products
- Processes product status changes

3. User Interface

Created three main dashboards:

1. Producer Dashboard
    - Form to register new products
    - Generate QR codes
    - View product status
2. Shipper Dashboard
    - List of available products
    - Update shipping status
    - Track product movements

3. Deliverer Dashboard
    - View products in transit
    - Mark deliveries complete
    - Update final status