COURSE NAME: **BLOCKCHAIN**

GROUP NUMBER:**06**

PROJECT TITLE:**FOOD AUTHENGITATION AND TRACKING WITH ETHEREUM SMART CONTRACTS**

PROJECT SUBMITTED TO: **ANNA UNIVERSITY/NAAN MUDHALVAN**

YEAR: II Year **(2022-2023)**

DEPARTMENT: **ARTIFICIAL INTELLIGENCE AND DATA SCIENCE AND MECHANICAL**

SEMESTER:**04**

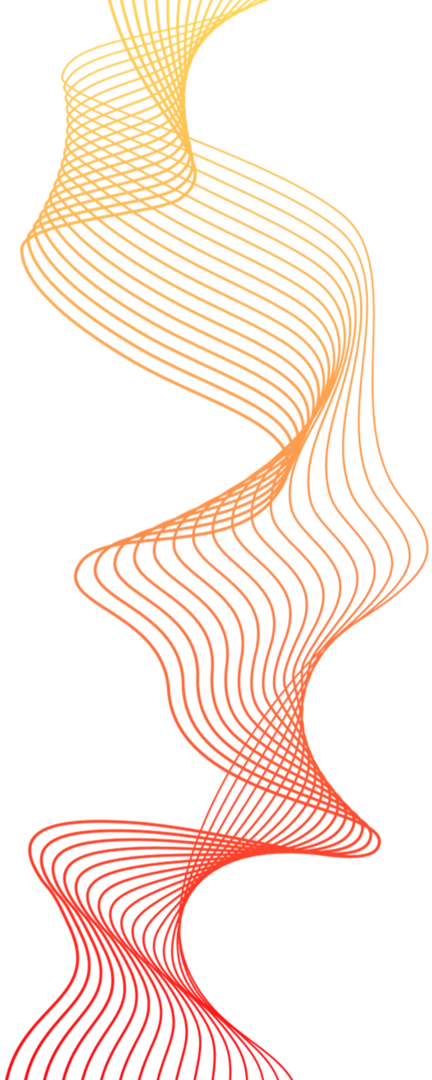
GROUP MEMBERS: S **AKASH , NAVEEN S , MUKILAN M , MUKESHWARAN M, MOORTHI G**

GUIDED BY:**Ms. S.H. ANNIE SILVIYA**

SPOC NAME: **Dr. SRIDHAR S**

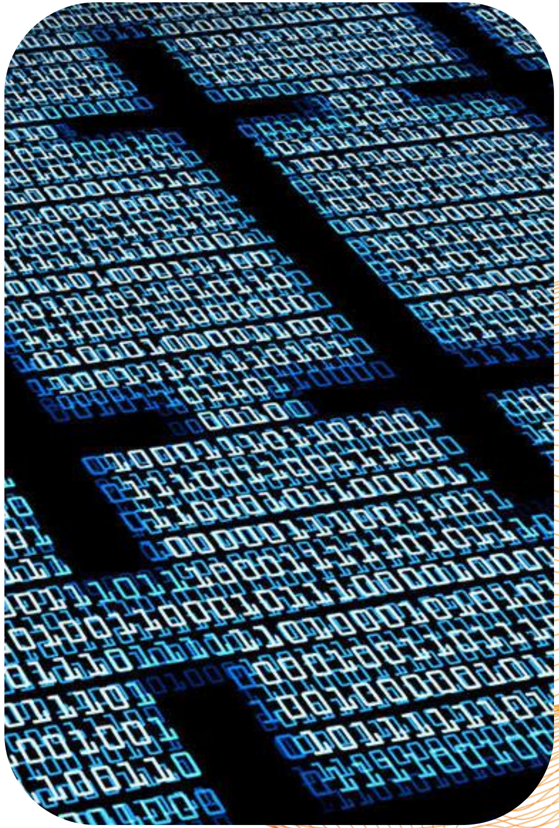
Using Ethereum Blockchain for Food Authentication

This presentation discusses the use of Ethereum blockchain and smart contracts for authenticating food items and ensuring their safety for consumption.



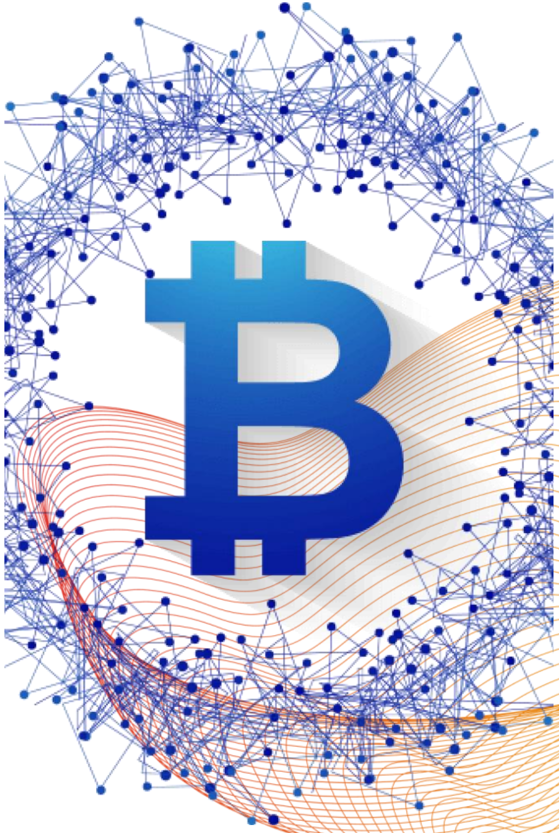
The Problem

- Fruits and vegetables often do not have expiry dates mentioned.
- It is important to know the origin of the food item and when it was sent from the farmer to the distributor.
- Consumers need to trust the authenticity of the food they are consuming.



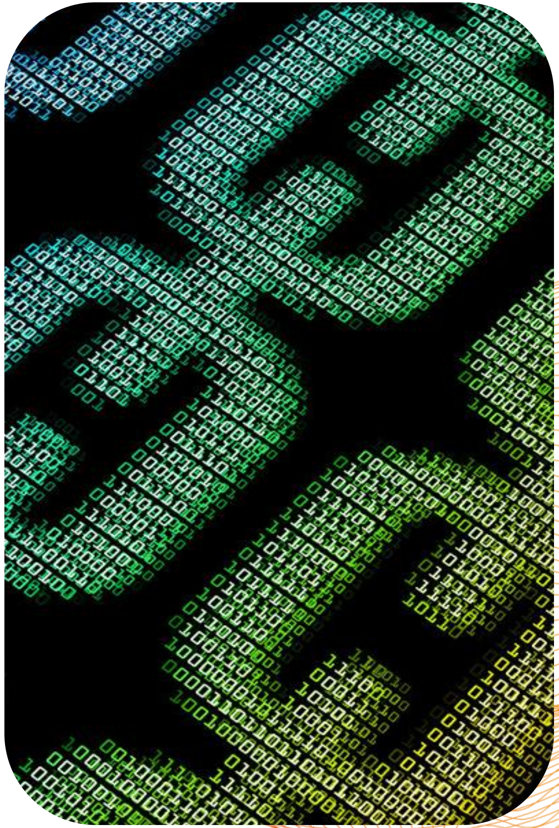
Introduction to Ethereum Blockchain

- A decentralized, distributed ledger technology that allows for secure and transparent transactions.
- Uses smart contracts to automate processes and functions.
- Built on a peer-to-peer network of nodes.



What is a Smart Contract?

- Self-executing contracts with the terms of the agreement between buyer and seller being directly written into lines of code.
- Cannot be modified once deployed on the blockchain.
- Automates processes and reduces the need for intermediaries.



Designing the Smart Contract

- Using Solidity programming language to write the smart contract.
- Defining variables like food type, origin, and date of dispatch.
- Creating functions for inserting and retrieving data.

Interacting with the Ethereum Blockchain

- Using MetaMask wallet to interact with the Ethereum blockchain.
- Connecting to the network and importing the smart contract.
- Sending transactions to insert data onto the blockchain.

This Photo by Unknown Author is licensed under CC BY-NC

Benefits of Using Ethereum Blockchain for Food Authentication

- Authenticity and origin of food items can be verified.
- Transactions are secure and transparent, ensuring trust between buyers and sellers.
- Reduction in the need for intermediaries, leading to cost savings.

Challenges and Limitations

- Smart contract code must be error-free to avoid vulnerabilities.
- High fees and slow transaction times during times of high network congestion.
- Need for technological expertise to interact with the blockchain.

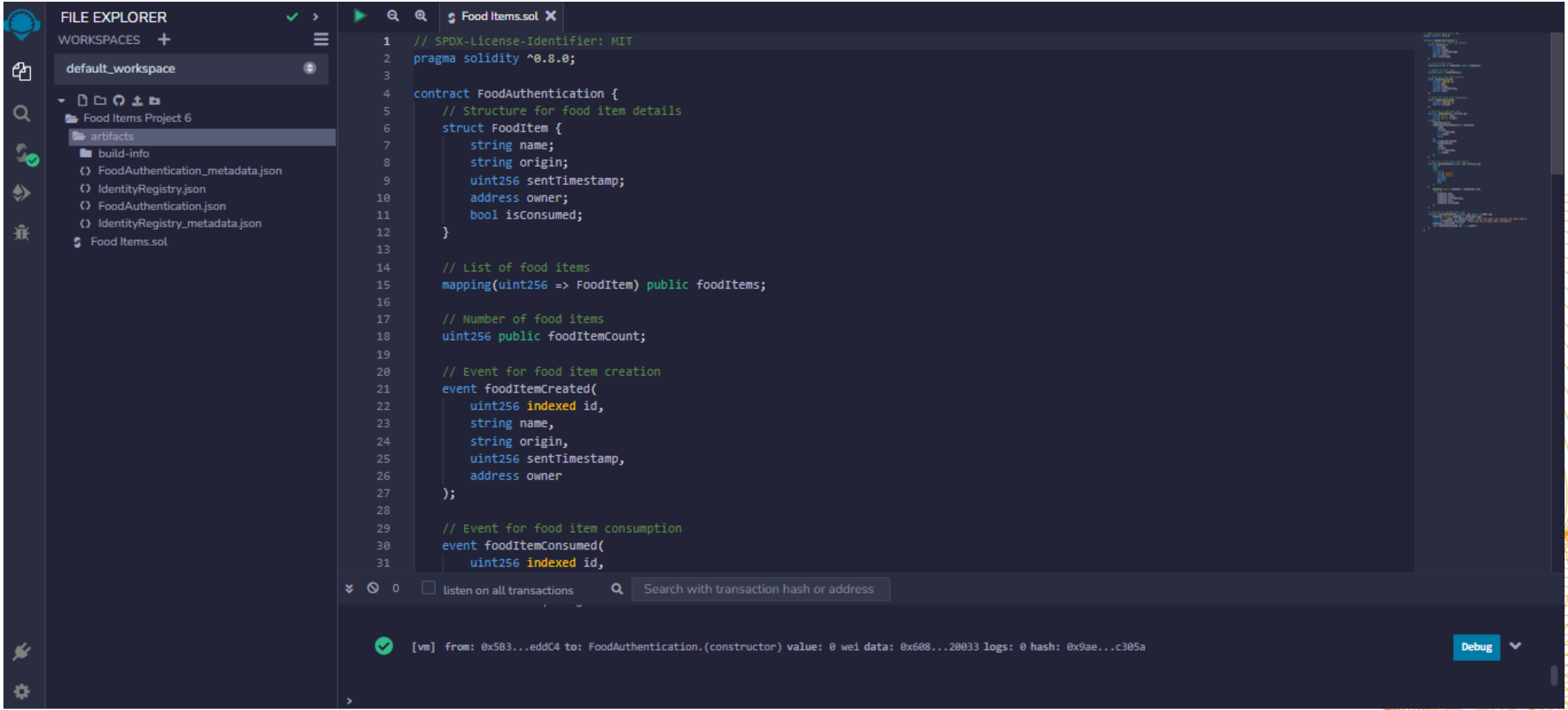
Future Possibilities

- Integration with IoT devices for real-time monitoring of food items.
- Adoption by food certification agencies for regulatory compliance.
- Use of blockchain for tracking supply chain information.

Examples of Ethereum Blockchain in Food Industry

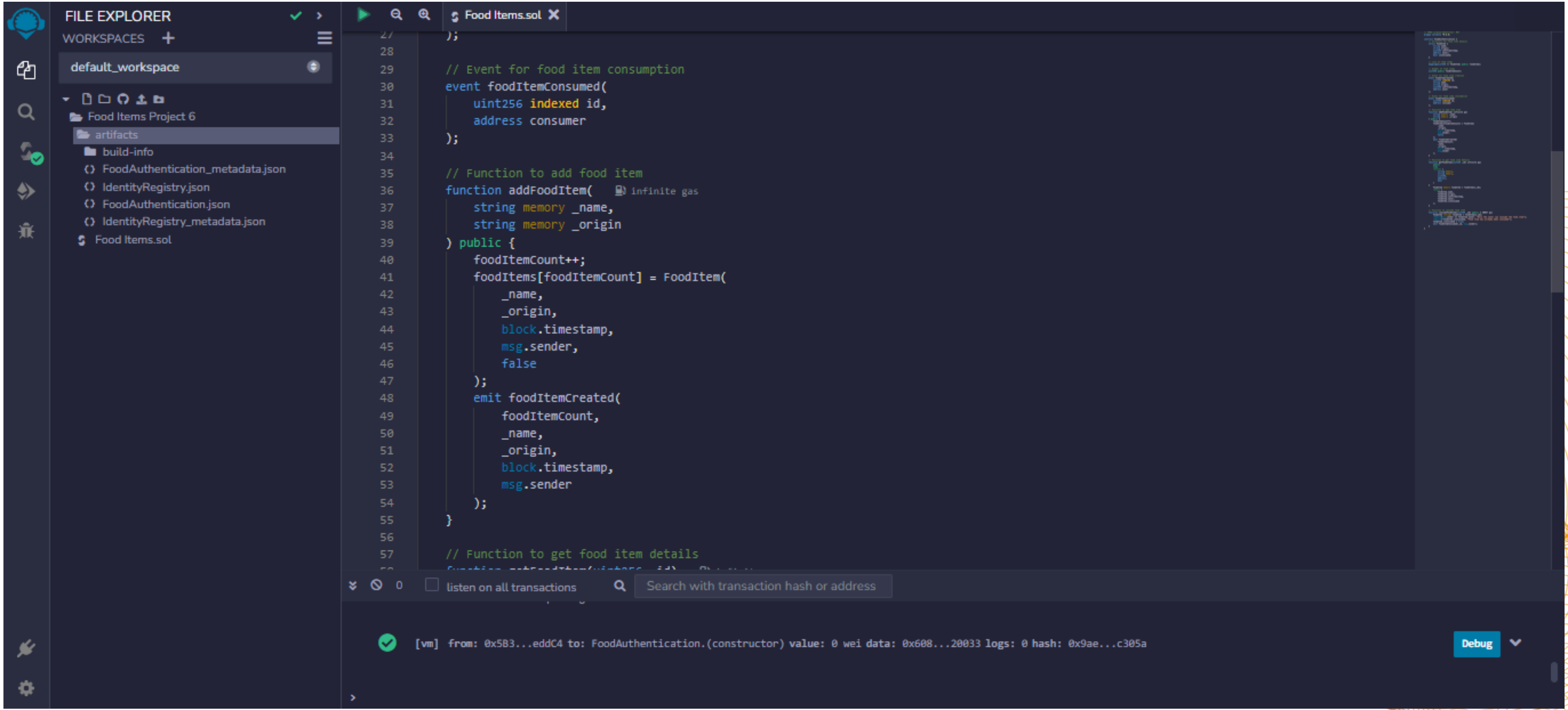
- TE-FOOD in Europe for tracking food items from farm to table.
- Ambrosus in Switzerland for ensuring food safety and origin.
- Ripe.io in the United States for tracking supply chain data.

CODE:



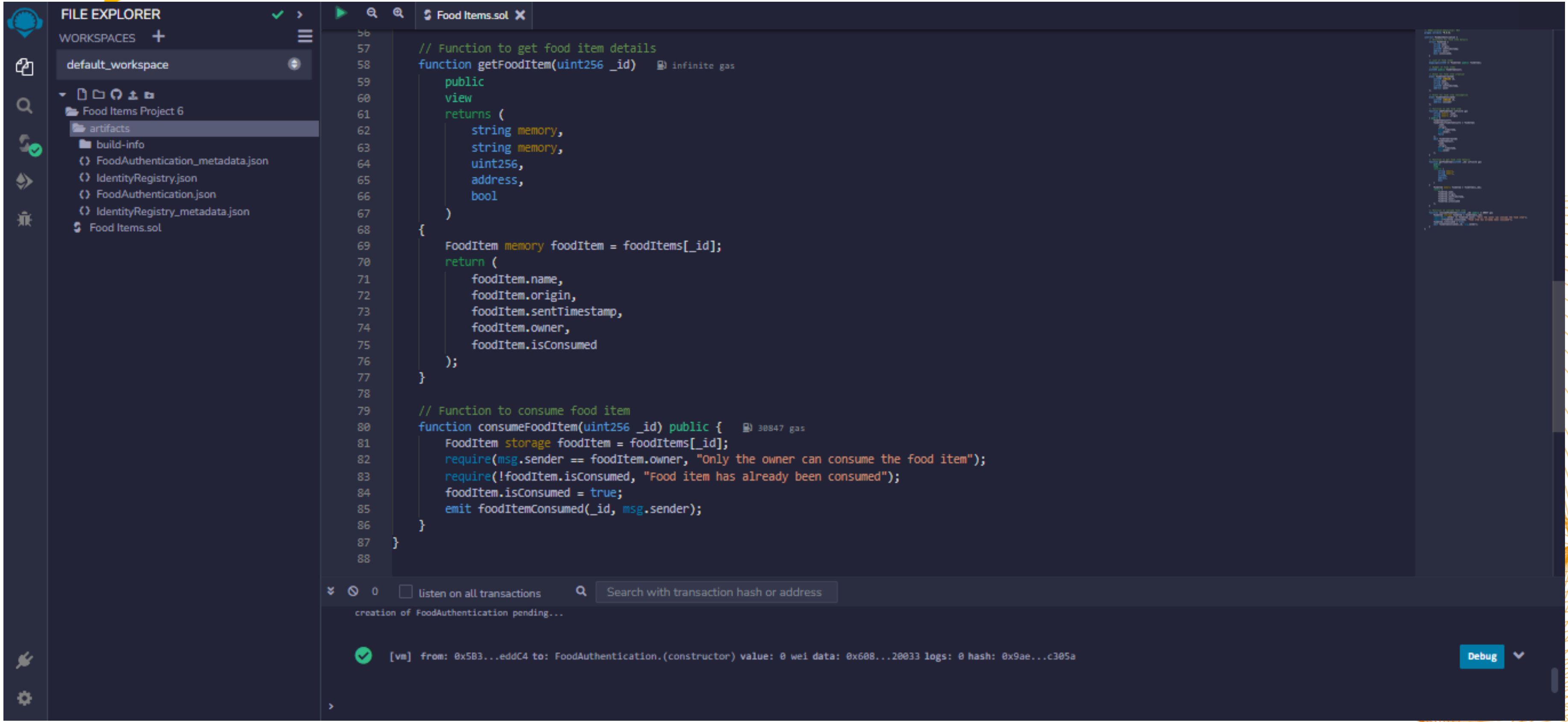
```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.0;

contract FoodAuthentication {
    // Structure for food item details
    struct FoodItem {
        string name;
        string origin;
        uint256 sentTimestamp;
        address owner;
        bool isConsumed;
    }

    // List of food items
    mapping(uint256 => FoodItem) public foodItems;

    // Number of food items
    uint256 public foodItemCount;

    // Event for food item creation
    event foodItemCreated(
        uint256 indexed id,
        string name,
        string origin,
        uint256 sentTimestamp,
        address owner
    );

    // Event for food item consumption
    event foodItemConsumed(
        uint256 indexed id,
```

```
[vm] from: 0x5B3...eddC4 to: FoodAuthentication.(constructor) value: 0 wei data: 0x608...20033 logs: 0 hash: 0x9ae...c305a
```



```
    );

    // Event for food item consumption
    event foodItemConsumed(
        uint256 indexed id,
        address consumer
    );

    // Function to add food item
    function addFoodItem(
        string memory _name,
        string memory _origin
    ) public {
        foodItemCount++;
        foodItems[foodItemCount] = FoodItem(
            _name,
            _origin,
            block.timestamp,
            msg.sender,
            false
        );
        emit foodItemCreated(
            foodItemCount,
            _name,
            _origin,
            block.timestamp,
            msg.sender
        );
    }

    // Function to get food item details
```

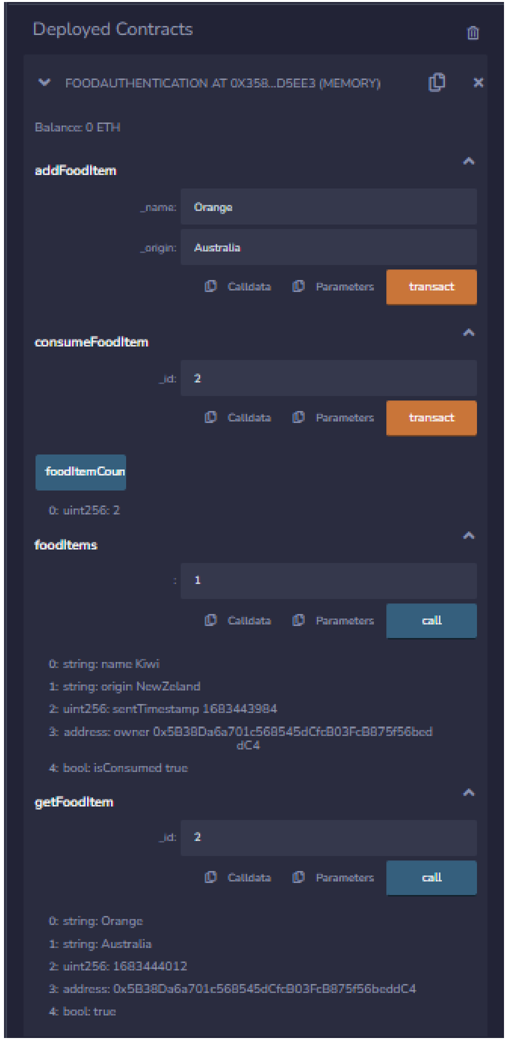
```
[vm] from: 0x5B3...eddC4 to: FoodAuthentication.(constructor) value: 0 wei data: 0x608...20033 logs: 0 hash: 0x9ae...c305a
```



```
    // Function to get food item details
    function getFoodItem(uint256 _id)
        public
        view
        returns (
            string memory,
            string memory,
            uint256,
            address,
            bool
        )
    {
        FoodItem memory foodItem = foodItems[_id];
        return (
            foodItem.name,
            foodItem.origin,
            foodItem.sentTimestamp,
            foodItem.owner,
            foodItem.isConsumed
        );
    }

    // Function to consume food item
    function consumeFoodItem(uint256 _id) public {
        FoodItem storage foodItem = foodItems[_id];
        require(msg.sender == foodItem.owner, "Only the owner can consume the food item");
        require(!foodItem.isConsumed, "Food item has already been consumed");
        foodItem.isConsumed = true;
        emit foodItemConsumed(_id, msg.sender);
    }
}
```

```
creation of FoodAuthentication pending...

[vm] from: 0x5B3...eddC4 to: FoodAuthentication.(constructor) value: 0 wei data: 0x608...20033 logs: 0 hash: 0x9ae...c305a
```

OUTPUT:



Deployed Contracts

FOODAUTHENTICATION AT 0X358...D5EE3 (MEMORY)

Balance: 0 ETH

addFoodItem

_name: Orange

_origin: Australia

consumeFoodItem

_id: 2

```
0: uint256: 2
```

foodItems

: 1

```
0: string: name Kiwi
1: string: origin NewZeland
2: uint256: sentTimestamp 1683443984
3: address: owner 0x5B38Da6a701c568545dCfcB03FcB875f56bed
                                dC4
4: bool: isConsumed true
```

getFoodItem

_id: 2

```
0: string: Orange
1: string: Australia
2: uint256: 1683444012
3: address: 0x5B38Da6a701c568545dCfcB03FcB875f56beddC4
4: bool: true
```

Conclusion

- Ethereum blockchain and smart contracts can be used for authenticating food items and ensuring their safety for consumption.
- Benefits include transparency, security, and cost savings.
- Challenges and limitations must be considered, and future possibilities are promising.