



COURSE NAME: BLOCKCHAIN

GROUP NUMBER:06

**PROJECT TITLE:FOOD AUTHENTICATION 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, MOORTHY G**

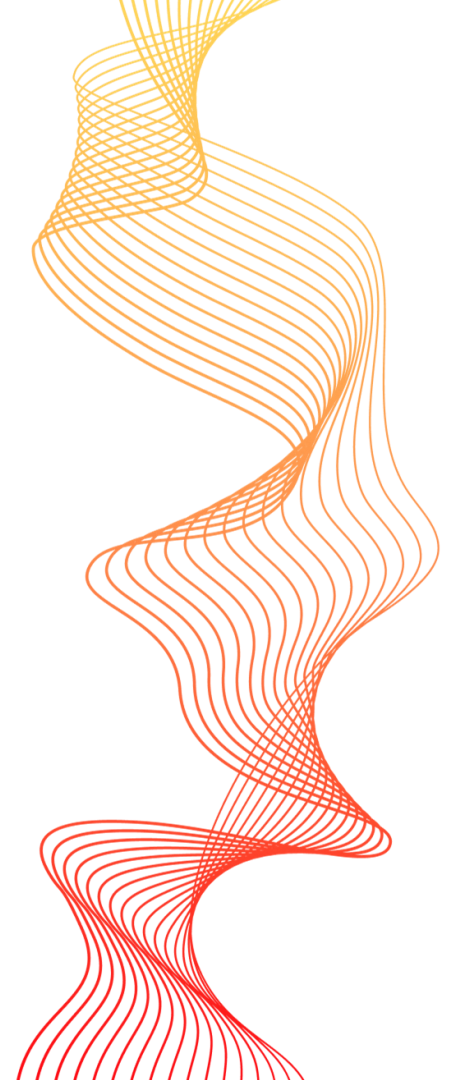
GUIDED BY:Ms. S.H. ANNIE SILVIYA

SPOC NAME: Dr. SRIDHAR S



Using Ethereum Blockchain for Food Authentication

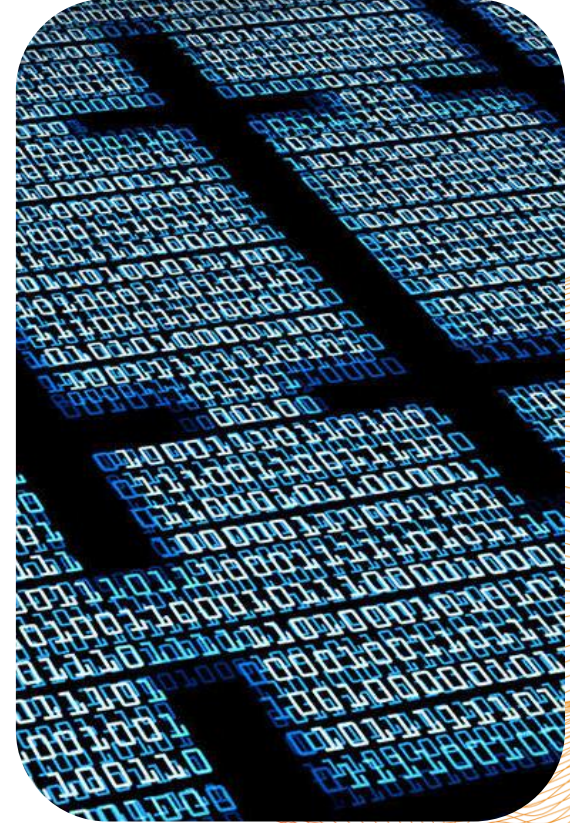
This presentation discusses the use of Ethereum blockchain and smart contracts for authenticating food items and ensuring their safety for consumption.





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.

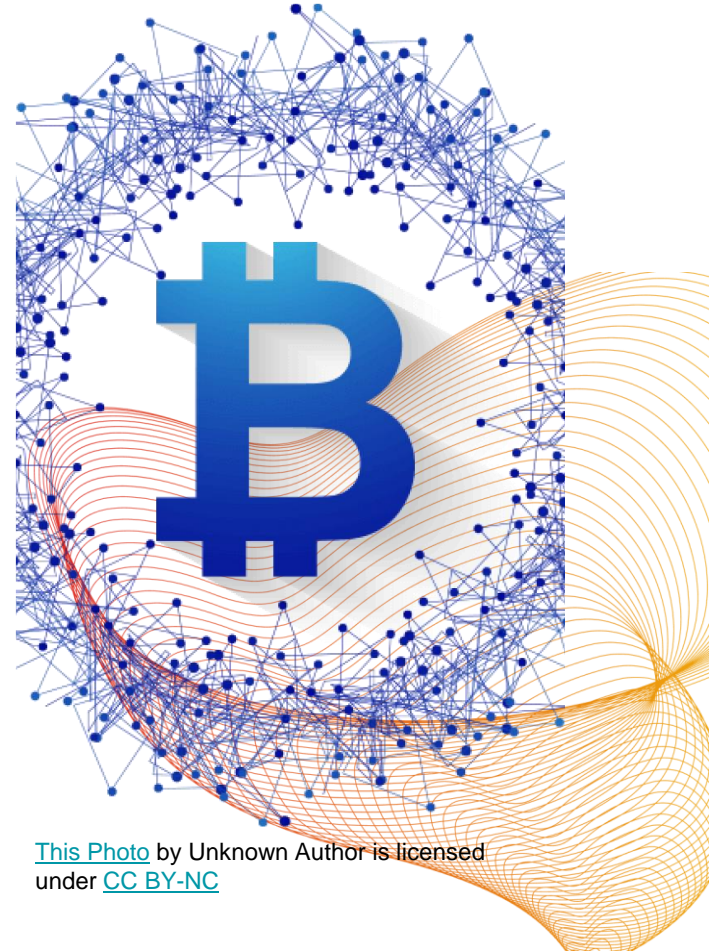


[This Photo](#) by Unknown Author is licensed under [CC BY-SA](#)



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.



[This Photo](#) by Unknown Author is licensed under [CC BY-NC](#)



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.



[This Photo](#) by Unknown Author is licensed under [CC BY-ND](#)



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

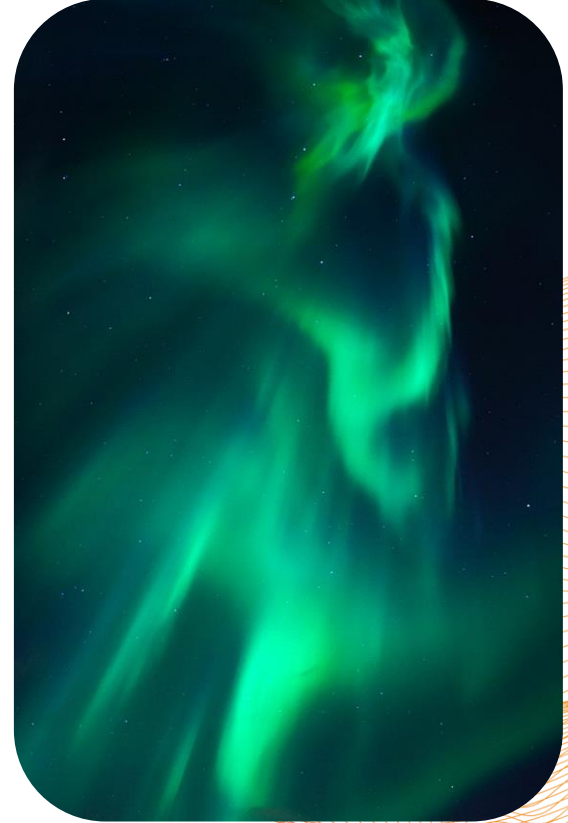
- Authenticity and origin of food items can be verified.
- Transactions are secure and transparent, ensuring trust between buyers and sellers.
- Reduction in the need for intermediaries, leading to cost savings.





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:

The screenshot displays a Solidity development environment. On the left, the **FILE EXPLORER** shows a workspace named **default_workspace** containing a project **Food Items Project 6**. Inside this project, there is an **artifacts** folder and several JSON files: **build-info**, **FoodAuthentication_metadata.json**, **IdentityRegistry.json**, **FoodAuthentication.json**, and **IdentityRegistry_metadata.json**. The main **Food Items.sol** file is open in the editor.

The code in **Food Items.sol** is as follows:

```
1 // SPDX-License-Identifier: MIT
2 pragma solidity ^0.8.0;
3
4 contract FoodAuthentication {
5     // Structure for food item details
6     struct FoodItem {
7         string name;
8         string origin;
9         uint256 sentTimestamp;
10        address owner;
11        bool isConsumed;
12    }
13
14    // List of food items
15    mapping(uint256 => FoodItem) public foodItems;
16
17    // Number of food items
18    uint256 public foodItemCount;
19
20    // Event for food item creation
21    event foodItemCreated(
22        uint256 indexed id,
23        string name,
24        string origin,
25        uint256 sentTimestamp,
26        address owner
27    );
28
29    // Event for food item consumption
30    event foodItemConsumed(
31        uint256 indexed id,
```

At the bottom, the console shows a successful transaction log entry:

```
[vm] from: 0x583...edc4 to: FoodAuthentication.(constructor) value: 0 wei data: 0x608...20033 logs: 0 hash: 0x9ae...c305a
```

On the right side of the interface, there is a sidebar with a tree view of the project's file structure and a list of transactions.



FILE EXPLORER

WORKSPACES +

default_workspace

Food Items Project 6

artifacts

build-info

FoodAuthentication_metadata.json

IdentityRegistry.json

FoodAuthentication.json

IdentityRegistry_metadata.json

Food Items.sol

Food Items.sol

```
27    };
28
29    // Event for food item consumption
30    event foodItemConsumed(
31        uint256 indexed id,
32        address consumer
33    );
34
35    // Function to add food item
36    function addFoodItem( infinite gas
37        string memory _name,
38        string memory _origin
39    ) public {
40        foodItemCount++;
41        foodItems[foodItemCount] = FoodItem(
42            _name,
43            _origin,
44            block.timestamp,
45            msg.sender,
46            false
47        );
48        emit foodItemCreated(
49            foodItemCount,
50            _name,
51            _origin,
52            block.timestamp,
53            msg.sender
54        );
55    }
56
57    // Function to get food item details
58    function getFoodItemDetails(
59        uint256 indexed id
60    ) public view returns (FoodItem memory) {
61        return foodItems[id];
62    }
63}
```

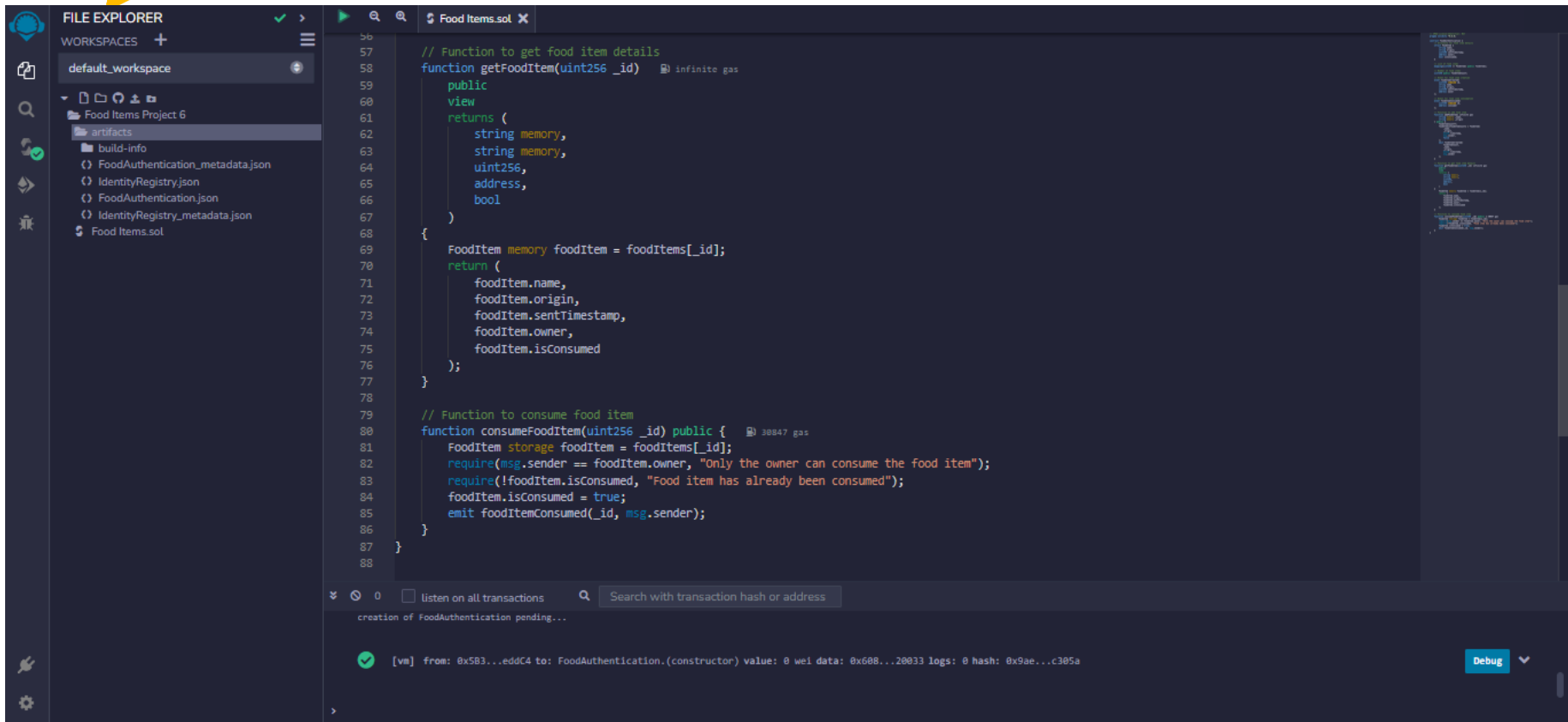
0

☐ listen on all transactions

Search with transaction hash or address

✓ [vm] from: 0x583...eddC4 to: FoodAuthentication.(constructor) value: 0 wei data: 0x608...20033 logs: 0 hash: 0x9ae...c305a

Debug



FILE EXPLORER

WORKSPACES +

default_workspace

Food Items Project 6

- artifacts
- build-info
- FoodAuthentication_metadata.json
- IdentityRegistry.json
- FoodAuthentication.json
- IdentityRegistry_metadata.json
- Food Items.sol

```
56
57
58 // Function to get food item details
59 function getFoodItem(uint256 _id) infinite gas
60
61     public
62     view
63     returns (
64         string memory,
65         string memory,
66         uint256,
67         address,
68         bool
69     )
70
71 {
72     FoodItem memory foodItem = foodItems[_id];
73     return (
74         foodItem.name,
75         foodItem.origin,
76         foodItem.sentTimestamp,
77         foodItem.owner,
78         foodItem.isConsumed
79     );
80 }
81
82 // Function to consume food item
83 function consumeFoodItem(uint256 _id) public { 30847 gas
84     FoodItem storage foodItem = foodItems[_id];
85     require(msg.sender == foodItem.owner, "Only the owner can consume the food item");
86     require(!foodItem.isConsumed, "Food item has already been consumed");
87     foodItem.isConsumed = true;
88     emit foodItemConsumed(_id, msg.sender);
89 }
```

0 ☐ listen on all transactions Search with transaction hash or address

creation of FoodAuthentication pending...

✓ [vm] from: 0x5B3...eddC4 to: FoodAuthentication.(constructor) value: 0 wei data: 0x608...20833 logs: 0 hash: 0x9ae...c305a

Debug



OUTPUT:

Deployed Contracts

FOODAUTHENTICATION AT 0x358...D5EE3 (MEMORY)

Balance: 0 ETH

addFoodItem

_name: Orange

_origin: Australia

Calldata

Parameters

transact

consumeFoodItem

_id: 2

Calldata

Parameters

transact

foodItemCount

0: uint256: 2

foodItems

1

Calldata

Parameters

call

0: string: name Kiwi

1: string: origin NewZeland

2: uint256: sentTimestamp 1683443984

3: address: owner 0x5038Da6a701c568545dCfcB03FcB875f56beddC4

4: bool: isConsumed true

getFoodItem

_id: 2

Calldata

Parameters

call

0: string: Orange

1: string: Australia

2: uint256: 1683444012

3: address: 0x5038Da6a701c568545dCfcB03FcB875f56beddC4

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.

