CODING AND SOLUTION

DATE	24 NOV 2023
TEAM ID	NM2023TMID11919
PROJECT NAME	FOOD TRACING SYSTEM
MAX.MARK	4 MARK

CODING

```
// SPDX-License-Identifier: MIT
pragma solidity "0.8.0;
contract FoodTracking {
    address public owner;

    enum FoodStatu {
        Unverified,
        Verified,
        Consumed
    }
    struct FoodItem {

        string itemid;
        string productName;
        string origin;
        uint256 sentTimestamp;
        FoodStatus status;
    }
mapping(string> FoodItem) public foodItems;
 e vent FoodItemSent(
    string Indexed itenid,
```

```
        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");
      }

function sendFoodItem(
string memory itemid,
string memory productName,
string memory origin

     external onlyOwner{
require(

bytes(foodItems[itemId].itemid), length, Ites already exists"
      }
 equire(foodItems[ItemId].status Foodstatus.
 Unverified, Item is already verified or consumed"
```

FEATURES

This Solidity contract named FoodTracking is designed to track food items. Let me provide a breakdown of the key components:

1. Contract Structure:
   - It starts with SPDX-License-Identifier to specify the license for the code.
   - The pragma statement sets the version of the Solidity compiler to be used (0.8.0 in this case).

2. State Variables:
   - owner: A public variable to store the address of the contract owner.
   - FoodStatus: An enumeration representing the status of a food item (Unverified, Verified, Consumed).
   - FoodItem: A struct to store details about a food item, including its ID, name, origin, timestamp of sending, and status.
   - foodItems: A mapping that associates food item IDs with their corresponding FoodItem structs.

3. Events:
   - FoodItemSent: Triggered when a food item is sent, emitting details like item ID, product name, origin, and timestamp.
   - FoodItemVerified: Triggered when a food item is verified, emitting the item ID.
   - FoodItemConsumed: Triggered when a food item is consumed, emitting the item ID.

4. Constructor:
   - Sets the contract owner to the address deploying the contract.

5. Modifier:
   - onlyOwner: A modifier to restrict certain functions to be callable only by the contract owner.

6. Function: sendFoodItem
   - Allows the contract owner to send a new food item with specified details.
   - Checks if the item ID is not empty and if the item doesn't already exist.
   - Checks that the status of the item is currently "Unverified."