FAKE PRODUCT DITECTON BY ,
USING BLOCKCHAIN

Smart Contract in Solidity for Product Authentication

CODE:

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

contract ProductAuthentication {

  struct Product {
    string name;
    string manufacturer;
    string batchNumber;
    uint256 timestamp;
    address owner;
    bool isAuthentic;
  }

  mapping(bytes32 => Product) public products;

  event ProductRegistered(bytes32 indexed productId, string name, string manufacturer,
uint256 timestamp);
  event ProductVerified(bytes32 indexed productId, bool isAuthentic);

  function registerProduct(
    string memory _name,
    string memory _manufacturer,
    string memory _batchNumber
  ) public returns (bytes32) {
    bytes32 productId = keccak256(abi.encodePacked(_name, _manufacturer,
_batchNumber, block.timestamp));
    require(products[productId].timestamp == 0, "Product already registered.");
    products[productId] = Product({
      name: _name,
      manufacturer: _manufacturer,
      batchNumber: _batchNumber,
      timestamp: block.timestamp,
      owner: msg.sender,
      isAuthentic: true
    });
```

```
        emit ProductRegistered(productId, _name, _manufacturer, block.timestamp);
        return productId;
    }

    function verifyProduct(bytes32 _productId) public view returns (bool) {
        Product memory product = products[_productId];
        require(product.timestamp != 0, "Product does not exist.");
        emit ProductVerified(_productId, product.isAuthentic);
        return product.isAuthentic;
    }

    function reportFakeProduct(bytes32 _productId) public {
        Product storage product = products[_productId];
        require(product.timestamp != 0, "Product does not exist.");
        require(msg.sender == product.owner, "Only the product owner can report a fake.");
        product.isAuthentic = false;
        emit ProductVerified(_productId, false);
    }

    function getProductDetails(bytes32 _productId) public view returns (string memory,
string memory, string memory, uint256, bool) {
        Product memory product = products[_productId];
        require(product.timestamp != 0, "Product does not exist.");
        return (product.name, product.manufacturer, product.batchNumber,
product.timestamp, product.isAuthentic);
    }
}
```

prepared by prasanth peethala