pragma solidity ^0.5.0;

contract p5 {

    string public name;

    uint public productCount = 0;

    mapping(uint => Product) public products;

    struct Product {

        uint id;

        string name;

        uint price;

        address payable owner;

        bool purchased;

    }

    event ProductCreated(

        uint id,

        string name,

        uint price,

        address payable owner,

        bool purchased

    );

    event ProductPurchased(

        uint id,

        string name,

        uint price,

        address payable owner,

        bool purchased

    );

    constructor() public {

        name = "Dapp University Marketplace";

    }

    function createProduct(string memory \_name, uint \_price) public {

        // Require a valid name

        require(bytes(\_name).length > 0);

        // Require a valid price

        require(\_price > 0);

        // Increment product count

        productCount ++;

        // Create the product

        products[productCount] = Product(productCount, \_name, \_price, msg.sender, false);

        // Trigger an event

        emit ProductCreated(productCount, \_name, \_price, msg.sender, false);

    }

    function purchaseProduct(uint \_id) public payable {

        // Fetch the product

        Product memory \_product = products[\_id];

        // Fetch the owner

        address payable \_seller = \_product.owner;

        // Make sure the product has a valid id

        require(\_product.id > 0 && \_product.id <= productCount);

        // Require that there is enough Ether in the transaction

        require(msg.value >= \_product.price);

        // Require that the product has not been purchased already

        require(!\_product.purchased);

        // Require that the buyer is not the seller

        require(\_seller != msg.sender);

        // Transfer ownership to the buyer

        \_product.owner = msg.sender;

        // Mark as purchased

        \_product.purchased = true;

        // Update the product

        products[\_id] = \_product;

        // Pay the seller by sending them Ether

        address(\_seller).transfer(msg.value);

        // Trigger an event

        emit ProductPurchased(productCount, \_product.name, \_product.price, msg.sender, true);

    }

}