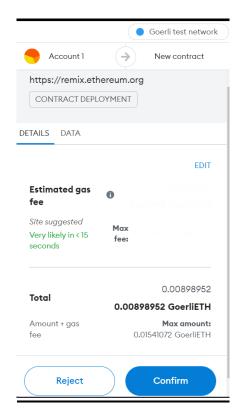19BCE248

BCT

Practical 9

AIM: To write a Solidity contract that implements a distributed ticket sales system. Anybody can create an event (specifying the initial price and number of tickets). Anybody can then purchase one of the initial tickets or sell those tickets peer-to-peer. At the event, gate agents will check that each attendee is listed in the final attendees list on the blockchain. (Ethereum programming).
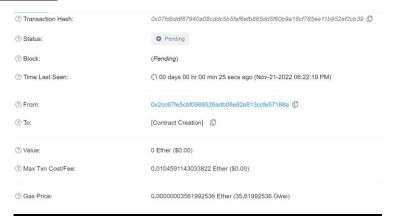
Code:

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

contract Ticket {
    uint256 ticketPrice = 100 wei;
    address owner;
    mapping(address => uint256) public ticketHolders;

    constructor() payable {
        owner = msg.sender;
    }

    function buyTickets(address _user, uint256 _amount) public payable {
        require(msg.value >= ticketPrice * _amount);
        addTickets(_user, _amount);
    }

    function useTickets(address _user, uint256 _amount) public {
        subTickets(_user, _amount);
    }

    function addTickets(address _user, uint256 _amount) internal {
        ticketHolders[_user] = ticketHolders[_user] + _amount;
    }

    function subTickets(address _user, uint256 _amount) internal {
        require(
            ticketHolders[_user] >= _amount,
            "You do not have enough tickets!"
        );
        ticketHolders[_user] = ticketHolders[_user] - _amount;
```

```
    }

    function withdraw() public {
        require(msg.sender == owner, "You are not the owner!");
        (bool success, ) = owner.call{value: address(this).balance}("");
        require(success);
    }
}
```

Deploying on etherscan.io

Deployed on etherscan:

Running of the Project:

Balance: 1 ETH

buyTickets

_user: `0x5B38Da6a701c568545dCfcB0`

_amount: `20`

Calldata   Parameters   **transact**   b

---

ticketHold...   `0x5B38Da6a701c568545dCfc`

0: uint256: 20

---

useTickets

_user: `"0x5B38Da6a701c568545dCfcB`

_amount: `"5"`

Calldata   Parameters   **transact**

---

ticketHold...   `0x5B38Da6a701c568545dCfc`

0: uint256: 15