**Roll number – 19BCE123**

**Name – Jash Mavani**

**Batch - EL2**

**Course Code & Name – 2CSDE93 & Blockchain Technology**

## 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 )

## Smart Contract :

```
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) payable public

{

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

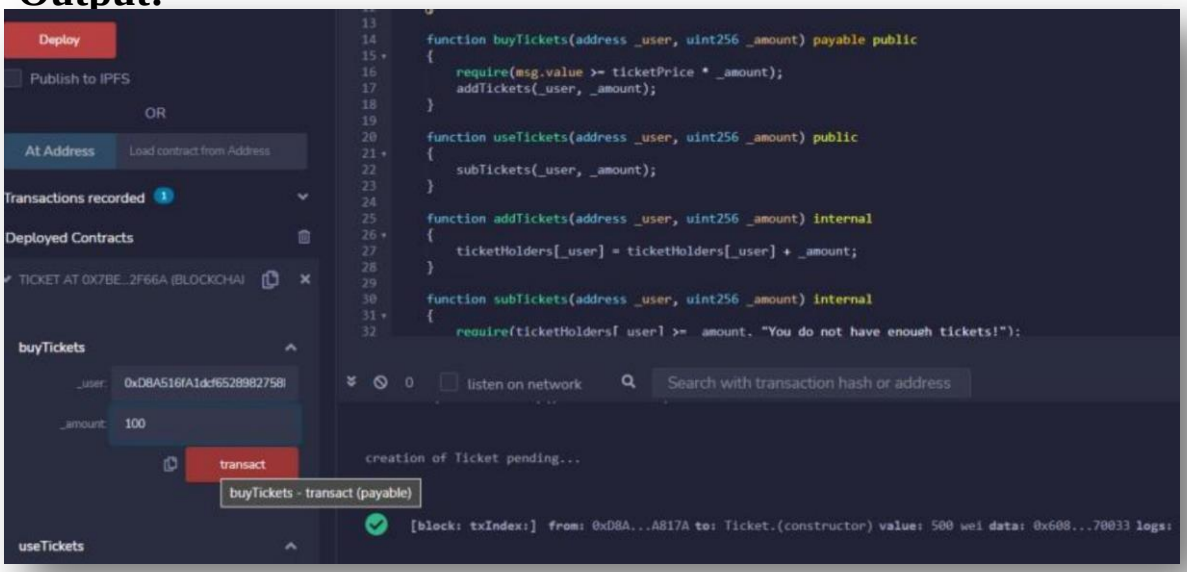
{

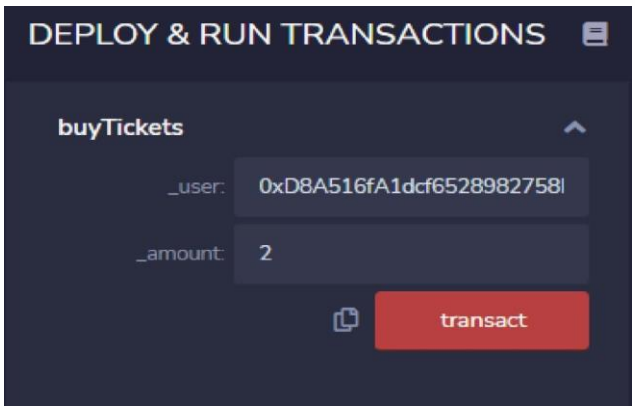
require(msg.sender == owner, "You are not the owner!"); (bool success, ) = owner.call{value: address(this).balance}(""); require(success);

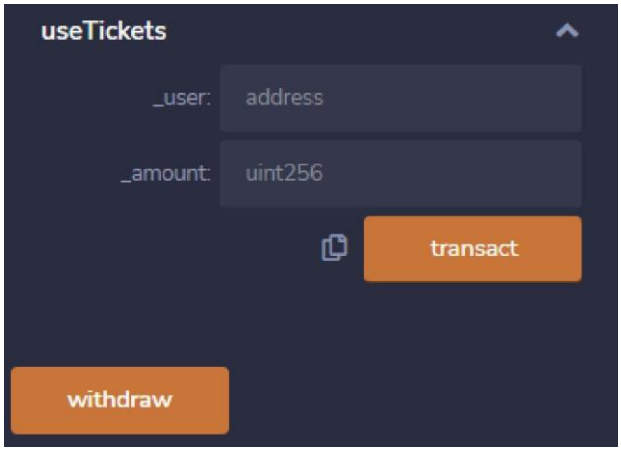
}

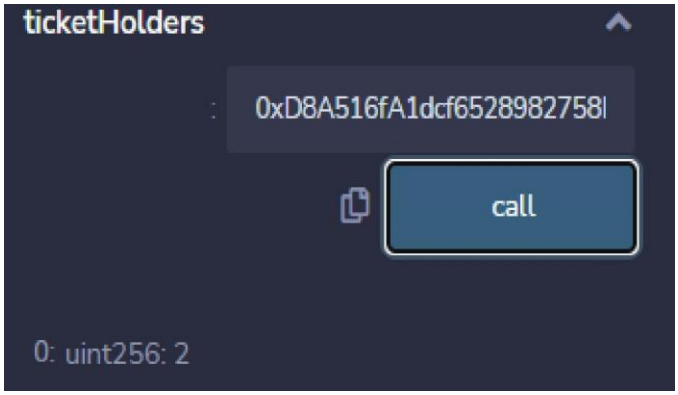
}

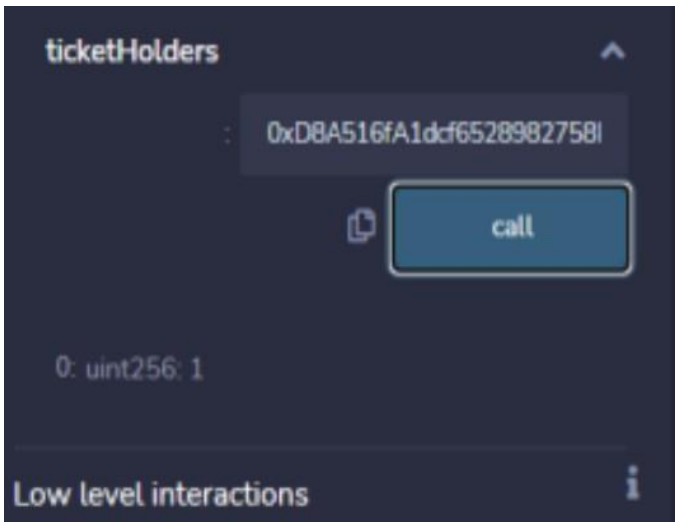
# Output:






**Conclusion :**

After completion of this practical, I learnt how to write smart contract for sales system using RemixIDE.