**Roll number – 19BCE072**

**Name – Gondaliya Krish D**

**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 :**

// SPDX-License-Identifier: MIT

pragma solidity >= 0.8.0;

contract TicketSystem {

uint256 ticketPrice;

address owner;

mapping(address = > uint256) public ticketHolders;

constructor(\_ticketPrice) {

owner = msg.sender;

ticketPrice = \_ticketPrice;

}

function buyTickets(address \_user, uint256 \_quantity) payable public {

require(msg.value >= \_quantity \* ticketPrice, "Not enough coins");

ticketHolders[\_user] += \_quantity;

}

function transferTickets(address \_to, uint256 \_quantity) public {

require(msg.sender == owner, "Only owner can transfer tickets");

require(ticketHolders[\_to] >= \_quantity, "Not enough tickets");

ticketHolders[\_to] -= \_quantity;

ticketHolders[msg.sender] += \_quantity;

}

function getTickets(address \_user) public view returns(uint256) {

return ticketHolders[\_user];

}

function getOwner() public view returns(address) { return owner; }

function getPrice() public view returns(uint256) { return ticketPrice; }

function getTotalTickets() public view returns(uint256) {

uint256 total = 0;

for (uint256 i = 0; i < ticketHolders.length; i++) {

total += ticketHolders[i];

}

return total;

}

**Conclusion :**

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