// SPDX-License-Identifier: MIT

pragma solidity ^0.6;

contract banking

{

 mapping(address=>uint) public user\_account;

 mapping(address=>bool) public user\_exists;

 function create\_account() public payable returns(string memory)

 {

 require(user\_exists[msg.sender]==false,'Account already created');

 if(msg.value==0)

 {

user\_account[msg.sender]=0;

user\_exists[msg.sender]=true;

 return "Account created";

 }

 require(user\_exists[msg.sender]==false,"Account already created");

user\_account[msg.sender]=msg.value;

user\_exists[msg.sender]=true;

 return "Account created";

 }

 function deposit(uint amount) public payable returns(string memory)

 {

 require(user\_exists[msg.sender]==true,"Account not created");

 require(amount>0,"Value for deposit is Zero");

user\_account[msg.sender]=user\_account[msg.sender]+amount;

 return "Deposited Successfully";

 }

 function withdraw(uint amount) public payable returns(string memory)

 {

     require(user\_account[msg.sender]>amount,"Insufficient Balance");

 require(user\_exists[msg.sender]==true,"Account not created");

 require(amount>0,"Amount should be more than zero");

user\_account[msg.sender]=user\_account[msg.sender]-amount;

 return "Withdrawl Successful";

 }

 function transfer(address payable userAddress, uint amount) public payable returns(string memory)

 {

 require(user\_account[msg.sender]>amount,"Insufficient balance in Bank account");

 require(user\_exists[msg.sender]==true,"Account is not created");

 require(user\_exists[userAddress]==true,"Transfer account does not exist");

 require(amount>0,"Amount should be more than zero");

user\_account[msg.sender]=user\_account[msg.sender]-amount;

user\_account[userAddress]=user\_account[userAddress]+amount;

 return "Transfer Successful";

 }

 function user\_balance() public view returns(uint)

 {

 return user\_account[msg.sender];

 }

 function account\_exist() public view returns(bool)

 {

 return user\_exists[msg.sender];

 }

 }