**Name**: Raj K Patel

**Roll No**: 20BCE218

**Course**: 2CSDE93 - Blockchain Technology

**Practical No**: 8

**Aim**: To design and develop end-to-end decentralized applications (Dapps).

**Code:**

***// SPDX-License-Identifier: MIT***

pragma solidity ^0.8.0;

contract IdentityManagement { address public government;

struct Student { uint256 Id; string FirstName; string LastName; uint8 Percentage;

bool IsEligibleForScholarship; address Address;

**}**

struct College { uint256 Id; uint Fees; string Name; string Location; address Address;

**}**

struct Scholarship { uint256 Id; uint256 StudentId; uint256 CollegeId;

string ScholarshipName; uint256 Amount; address payable To; string Status;

**}**

mapping(uint256 => Student) internal studentRecords;

Student[] internal students;

mapping(uint256 => College) internal collegeRecords; College[] internal colleges;

mapping(uint256 => Scholarship) internal scholarshipRecords; Scholarship[] internal scholarships;

struct StudentFeesPaid {

mapping(uint256 => uint) studentFeesPaid;

**}**

mapping(uint => StudentFeesPaid) college\_student\_fees\_paid;

modifier onlyGovernment() { require(

msg.sender == government,

"Only government can perform this action"

**);**

**\_;**

**}**

constructor() {

government = msg.sender;

**}**

function enrollCollege( uint256 \_id, uint256 \_fees, string memory \_name,

string memory \_location, address payable \_address

) public onlyGovernment {

require(\_address != address(0), "Invalid college address"); require(

collegeRecords[\_id].Id == 0,

"College with the given ID already exists"

**);**

require(bytes(\_name).length > 0, "College name cannot be empty"); require(

bytes(\_location).length > 0, "College location cannot be empty"

**);**

College memory c = College({ Id: \_id,

Fees: \_fees, Name: \_name,

Location: \_location, Address: \_address

**});**

collegeRecords[c.Id] = c; colleges.push(c);

**}**

function getCollegeDetails( uint256 \_Id

) public view returns (College memory) { require(

collegeRecords[\_Id].Id != 0,

"No college with the given ID found"

**);**

return collegeRecords[\_Id];

**}**

function addStudentRecord( uint256 \_id,

string memory \_firstName, string memory \_lastName, uint8 \_percentage, address \_address

) public onlyGovernment { require(

studentRecords[\_id].Id == 0,

"Student with the given ID already exists"

**);**

Student memory student = Student({ Id: \_id,

FirstName: \_firstName, LastName: \_lastName, Percentage: \_percentage, IsEligibleForScholarship: false, Address: \_address

**});**

students.push(student); studentRecords[student.Id] = student;

**}**

function getStudentDetails( uint256 \_Id

) public view returns (Student memory) { require(

studentRecords[\_Id].Id != 0,

"No student with the given ID found"

**);**

return studentRecords[\_Id];

**}**

function isStudentEligibleForScholarship( uint256 \_Id

) public view returns (bool) { require(

studentRecords[\_Id].Id != 0,

"No student with the given ID found"

**);**

Student memory s = getStudentDetails(\_Id); if (s.Percentage >= 80) {

return true;

} else {

return false;

**}**

**}**

***//***

function createScholarship( uint256 \_StudentId, uint256 \_CollegeId

) public onlyGovernment { require(

studentRecords[\_StudentId].Id != 0, "No student with the given ID found"

**);**

require(

collegeRecords[\_CollegeId].Id != 0, "No college with the given ID found"

**);**

require(

scholarshipRecords[\_StudentId].StudentId == 0, "Scholarship with the given ID already exists"

**);**

require(

isStudentEligibleForScholarship(\_StudentId), "Student is not eligible for a scholarship"

**);**

College memory c = collegeRecords[\_CollegeId]; uint min\_amount = 2000000000000000000;

uint scholarship\_amount = 0;

if (c.Fees < min\_amount) { scholarship\_amount = c.Fees;

} else {

scholarship\_amount = min\_amount;

**}**

Scholarship memory new\_scholarship = Scholarship({ Id: scholarships.length + 1,

StudentId: \_StudentId, CollegeId: \_CollegeId,

ScholarshipName: "Merit Scholarship", Amount: scholarship\_amount,

To: payable(c.Address), Status: "Pending"

**});**

scholarshipRecords[\_StudentId] = new\_scholarship; scholarships.push(new\_scholarship);

**}**

function disburseScholarship( uint256 \_StudentId

) public payable onlyGovernment {

Scholarship

storage scholarship\_of\_registered\_stduent = scholarshipRecords[

\_StudentId

**];**

require(

scholarship\_of\_registered\_stduent.StudentId != 0, "Scholarship for the given student not found"

**);**

require(

keccak256(

abi.encodePacked(scholarship\_of\_registered\_stduent.Status)

) == keccak256(abi.encodePacked("Pending")), "Scholarship is not pending"

**);**

scholarship\_of\_registered\_stduent.Status = "Awarded"; scholarship\_of\_registered\_stduent.To.transfer(

scholarship\_of\_registered\_stduent.Amount

**);**

StudentFeesPaid storage getCollege = college\_student\_fees\_paid[ scholarship\_of\_registered\_stduent.CollegeId

**];**

getCollege.studentFeesPaid[ scholarship\_of\_registered\_stduent.StudentId

] = scholarship\_of\_registered\_stduent.Amount;

**}**

function getScholarshipStatus( uint256 \_StudentId

) public view returns (string memory) { require(

scholarshipRecords[\_StudentId].Id != 0, "Scholarship with the given ID does not exists"

**);**

Scholarship memory scholarship = scholarshipRecords[\_StudentId]; return scholarship.Status;

**}**

function updateScholarshipStatusToCancel( uint256 \_StudentId

) public onlyGovernment { require(

scholarshipRecords[\_StudentId].Id != 0, "Scholarship with the given ID does not exists"

**);**

Scholarship storage \_scholarship = scholarshipRecords[\_StudentId];

\_scholarship.Status = "Cancel";

**}**

function updateScholarshipStatusToPaid( uint256 \_StudentId

) public onlyGovernment { require(

scholarshipRecords[\_StudentId].Id != 0, "Scholarship with the given ID does not exists"

**);**

Scholarship storage \_scholarship = scholarshipRecords[\_StudentId];

\_scholarship.Status = "Paid";

**}**

function updateScholarshipStatusToFailed( uint256 \_StudentId

) public onlyGovernment { require(

scholarshipRecords[\_StudentId].Id != 0, "Scholarship with the given ID does not exists"

**);**

Scholarship storage \_scholarship = scholarshipRecords[\_StudentId];

\_scholarship.Status = "Failed";

**}**

function updateScholarshipStatusToActive( uint256 \_StudentId

) public onlyGovernment { require(

scholarshipRecords[\_StudentId].Id != 0, "Scholarship with the given ID does not exists"

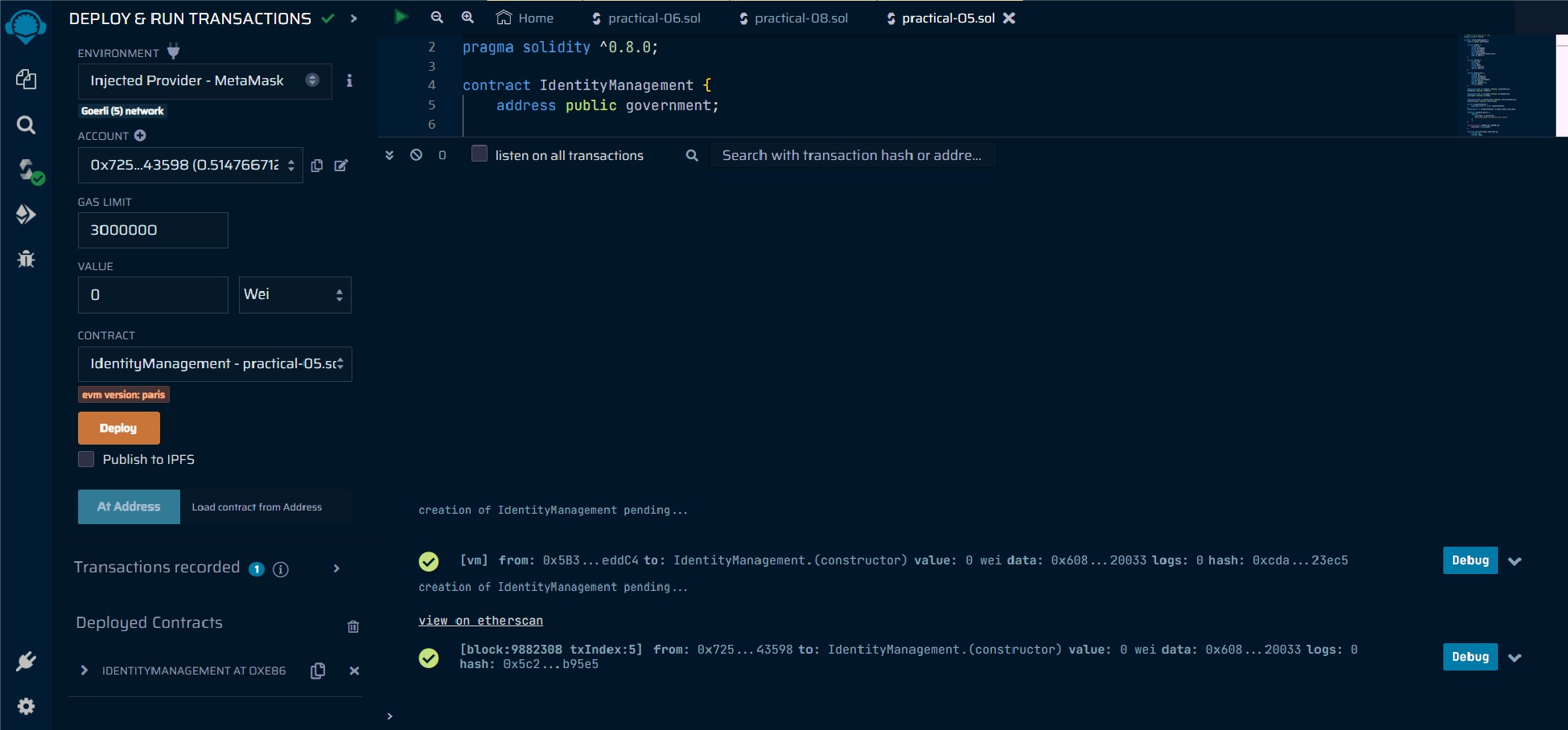
**);**

Scholarship storage \_scholarship = scholarshipRecords[\_StudentId];

\_scholarship.Status = "Active";

**}**

**}**



Q Search by Address /Txn Hash/ Block/Token

Etherscan Home Blockchain v Tokens v NFTs v Misc v

Transaction Details < >

-State Morev

I This is a Goerli Testnet transaction only]

<i)TransactionHash:

<i)Status:

<i)Block:

Ox4f61cac913226f 55c37d9898851de68a04383626e59ad909fa35ab645844aca b c9

osuccess

0 9891952 7612BlockConf1rmations

<i) Timestamp: @1 day7 hrs ago (Oct-19-2023 03:46:00 AM +UTC)

i> Method:

<i)From:

<i)To:

Ox725016361 FOAD1OOb5A790EaeE2440Be4D843598 (9

( Ox049dc0b690141e7dc22df8f590b9136cd58c2a97 Created I [1)0

<i)Value:

®Transaction Fee:

* 0 ETH ($0.00) 0.00698043252792173 ETH so.oo

Q Search by Address/ Txn Hash/ Block/ Token

Etherscan Home Blockchain " Tokens " NFTs .,.,. Misc "'

* Contract Ox049dc08690141E7dc22dFBF59089136cD58c2a97 cP ""

More v

Overview More Info Multi Chain

ETH BALANCE

* + 0 ETH

CONTRACT CREATOR

Ox725016...4DB43598 cP at txn Ox4f61cac913226f55c...

MULTICHAIN ADDRESSES

N/A

1111111 TokenTransfers{ERC-20)

0

Contract

Events

J. Latest 1 from a total of 1 transactions

*(y* TransactionHash

Method® Block

.,.

From To

Value

TxnFee

@ Ox4f61cac913226f55c...

9891952

1 day7hrs ago

Ox725016.. 4D843598 cP 1N lflll Create: ldentityManage...

0 ETH 0.00698043

[Oownload:CSVExport,!.]

{): A contract address hosts a smart contract, which is a set of code stored on the blockchain that runs when predetermined conditions are met. learn more about addresses inour Knowledge Base

Transactions TokenTransfers{ERC-20) rlflllllll Events

Ill ReadContract WriteContract

0 Contract Source Code Verified {Exact Match)

Q Search by Address /Txn Hash/ Block/Token

1. Search Source Code

-:e- ♦

I

SohdIty Compiler Bugs chck for more info

Contract Name: ldentityManagement Optimization Enabled: No with 200 runs

Compiler Version vO.B.1B+commit.87f61d96 Other Settings: default evmVersion, MIT license

QI Contract Source Code (Solidity) Open In v Outline v More Options v

1• *Vn*

2 *\*Submitted* for *verification at Ether5can. ia on 2023-10-19*

1 *'I*

1. // *SPDX-licen5e-Identifier-: /.IIT*
2. pragma solidity A0.8.18;

'8·contr-actidentityManagement{

9 address public government;

10

structStudent{ uint256Id; stringFirstName;

1. stringlastName;
2. uint8Percentage; boolisEligibleForScholarship;

17 address Address;

18

https://goerli.etherscan."io/address/Ox049dc0b690141e7dc22df8f590b9136cd5&2,197#

Q Search by Address/ Txn Hash/ Block/ Token

ETH BALANCE

* + 0 ETH

CONTRACT CREATOR

0x725016...4DB43598 cP at txn 0x4f61cac913226f55c...

MULTICHAIN ADDRESSES

N/A

Transactions TokenTransfers{ERC-20) - Events

Code 111111 WriteContract econnected-Web3

[si Read Contract Information [Expand all] [Reset]

1. getCoHegeDe!ails
2. getScholarshipS!atus
3. getStuden!Details
4. government
5. isStudentEligibleForScholarship

Q Search by Address/ Txn Hash/ Block/ Token

ETH BALANCE

* + 0 ETH

CONTRACT CREATOR

0x725016...4D843598 cP at txn 0x4f61cac913226f55c...

MULTICHAIN ADDRESSES

N/A

Transactions TokenTransfers{ERC-20) - Events

Code ReadContract --

econnected-Web3 [Expandall] [Reset]

1. addStuden!Reoord(Ox926aOd10)
2. createScholarship(Ox80823de7)
3. dist>urseScholarship(Ox50bac3e9)
4. enrolICollege(Ox3eee07dd)
5. updateScholarshipSta!usToActive(OxdeOcc2b2)
6. updateScholarshipSta!usToCancel(Ox4b71880c)
7. updateScholarshipSta!usToFai d(Oxa8ec2194)
8. updateScholarshipSta!usToPaid(Oxf&c174e)

javascript:;