**PRACTICAL: 6**

**AIM:** Write a Smartcontract to store details of PATIENTS’s details which includes the following things

* Write code using 0.8.0 or higher version of solidity.
* Define struct to define different properties of adding patient’s details.
* Define mapping function to struct
* Create a constructor to initialize first transaction using admin only.
* Create function to add details of patient such as patient\_id, patient\_name,
* decease type, doctor\_name and patient\_contact.
* Initialize proper datatypes of all variable.
* Display the patient’s details by patient\_id
* Display all patient’s details

Use online Ethereum REMIX IDE to compile and deploy the contract.

Integrate the metamask wallet to perform the transaction of certificate.

**CODE:**

// SPDX-License-Identifier: MIT

pragma solidity ^0.8.20;

contract PatientRecords {

    address public admin;

    struct Patient {

        uint256 patient\_id;

        string patient\_name;

        string disease\_type;

        string doctor\_name;

        string patient\_contact;

    }

    mapping(uint256 => Patient) private patients;

    uint256[] private patientIds;

    event PatientAdded(uint256 patient\_id, string patient\_name, string disease\_type, string doctor\_name, string patient\_contact);

    modifier onlyAdmin() {

        require(msg.sender == admin, "Only admin can perform this action");

        \_;

    }

    constructor() {

        admin = msg.sender;

    }

    function addPatient(

        uint256 \_patient\_id,

        string memory \_patient\_name,

        string memory \_disease\_type,

        string memory \_doctor\_name,

        string memory \_patient\_contact

    ) public onlyAdmin {

        require(patients[\_patient\_id].patient\_id == 0, "Patient already exists");

        patients[\_patient\_id] = Patient(\_patient\_id, \_patient\_name, \_disease\_type, \_doctor\_name, \_patient\_contact);

        patientIds.push(\_patient\_id);

        emit PatientAdded(\_patient\_id, \_patient\_name, \_disease\_type, \_doctor\_name, \_patient\_contact);

    }

    function getPatient(uint256 \_patient\_id) public view returns (Patient memory) {

        require(patients[\_patient\_id].patient\_id != 0, "Patient not found");

        return patients[\_patient\_id];

    }

    function getAllPatients() public view returns (Patient[] memory) {

        Patient[] memory allPatients = new Patient[](patientIds.length);

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

            allPatients[i] = patients[patientIds[i]];

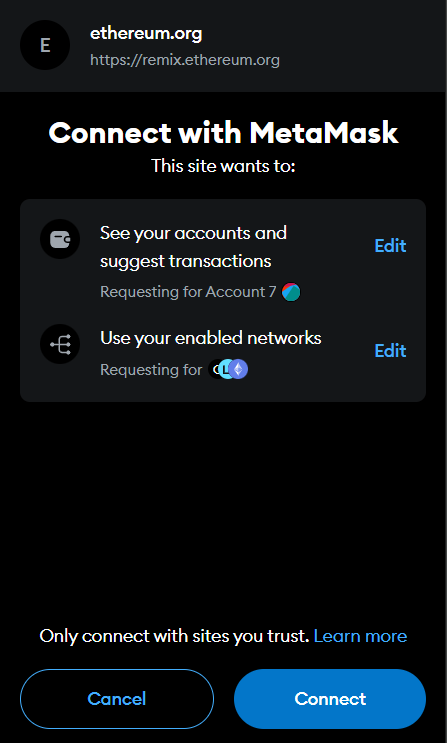
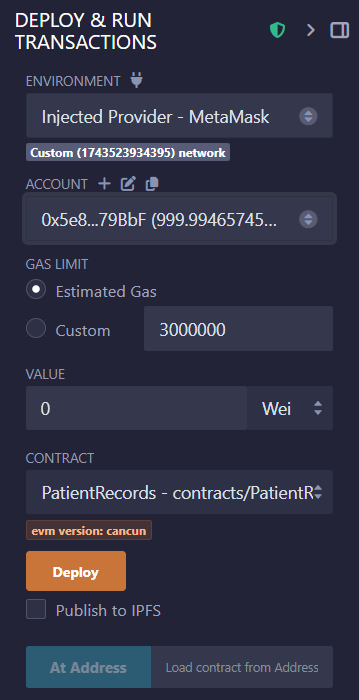
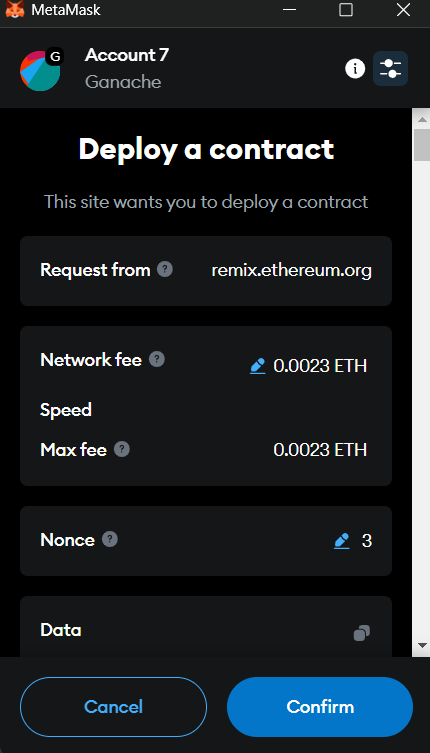
        }

        return allPatients;

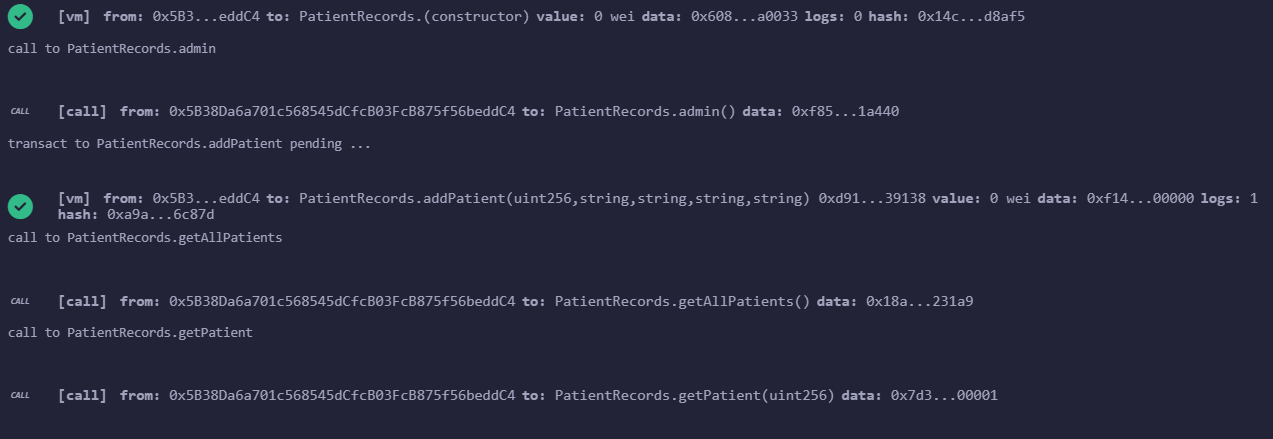
    }

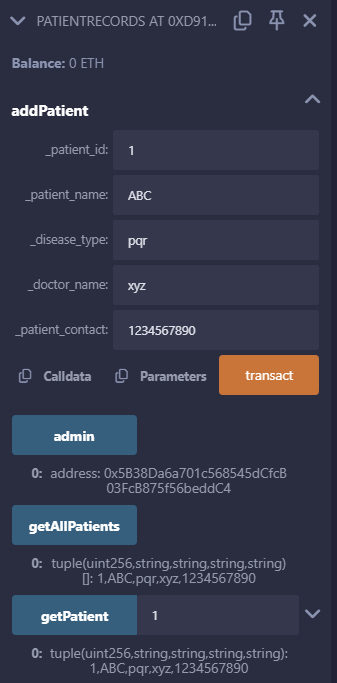
}

**OUTPUT:**







**LEARNING OUTCOME:**

A Solidity smart contract for patient details management ensures secure and immutable storage of medical records on the Ethereum blockchain. It uses structs to define patient attributes and mappings for efficient data retrieval. Functions are implemented to add, update, and retrieve patient details while maintaining data integrity. The contract is deployed using Remix IDE, and Metamask integration enables secure blockchain transactions, ensuring transparency, privacy, and tamper-proof record-keeping.