**Roll number – 19BCE072**

**Name – Gondaliya Krish D**

**Blockchain Technology**

**Practical – 8**

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

**Solidity Code :**

// SPDX-License-Identifier: MIT

pragma solidity ^0.8.0;

contract Autonomous {

address public caller;

struct Owner {

string name;

uint age;

uint location;

bool exists;

bool is\_info\_exchange;

}

struct Avs {

string name;

uint location;

bool exists;

}

mapping(address => Owner) owners;

mapping(address => Avs) avs;

constructor() payable {

caller = msg.sender;

}

function viewBalance() view public returns(uint) {

return address(this).balance;

}

function addOwner(address \_id, string calldata \_name, uint \_age, uint

\_location) external {

owners[\_id].name = \_name;

owners[\_id].age = \_age;

owners[\_id].location = \_location;

owners[\_id].exists = true;

owners[\_id].is\_info\_exchange = false;

}

function getOwner(address \_id) view public returns (string memory, uint,

uint, bool) {

return (owners[\_id].name, owners[\_id].age, owners[\_id].location,

owners[\_id].is\_info\_exchange);

}

function addAvs(address \_id, string calldata \_name, uint \_location) external

{

avs[\_id].name = \_name;

avs[\_id].location = \_location;

avs[\_id].exists = true;

}

function getAvs(address \_id) view public returns (string memory, uint) {

return (avs[\_id].name, avs[\_id].location);

}

function critcial\_information\_exchange(address ownerAddress, address payable

avsAddress) external returns (uint) {

require(owners[ownerAddress].exists == true, "Owner does not exist.");

require(avs[avsAddress].exists == true, "Avs does not exist.");

require(owners[ownerAddress].is\_info\_exchange == false, "Critcial Information

already exchanged.");

//Information exchange

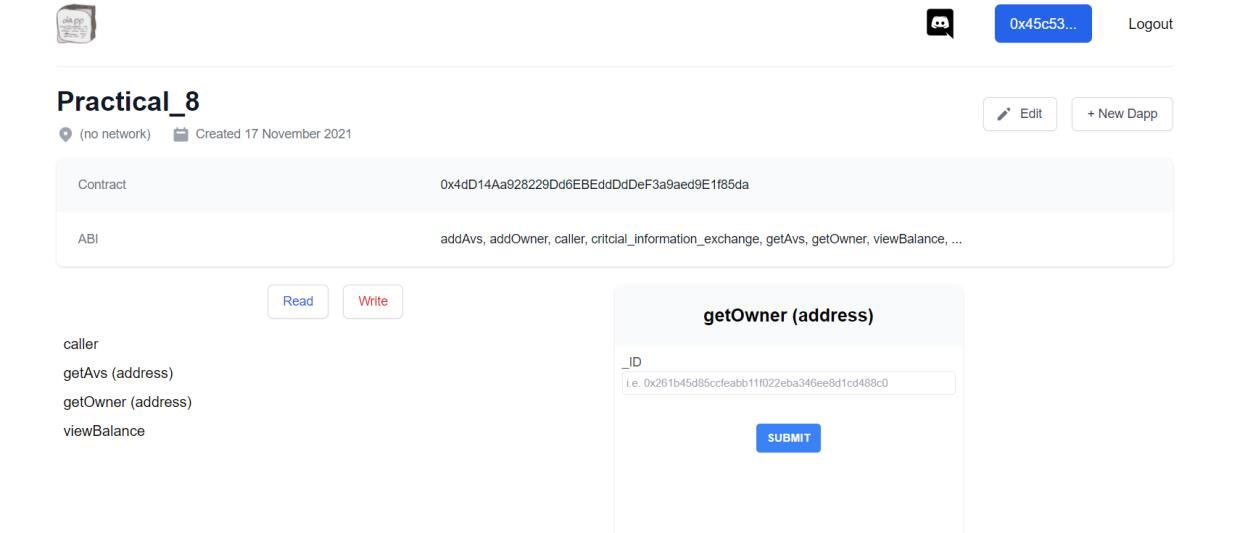
owners[ownerAddress].is\_info\_exchange = true;

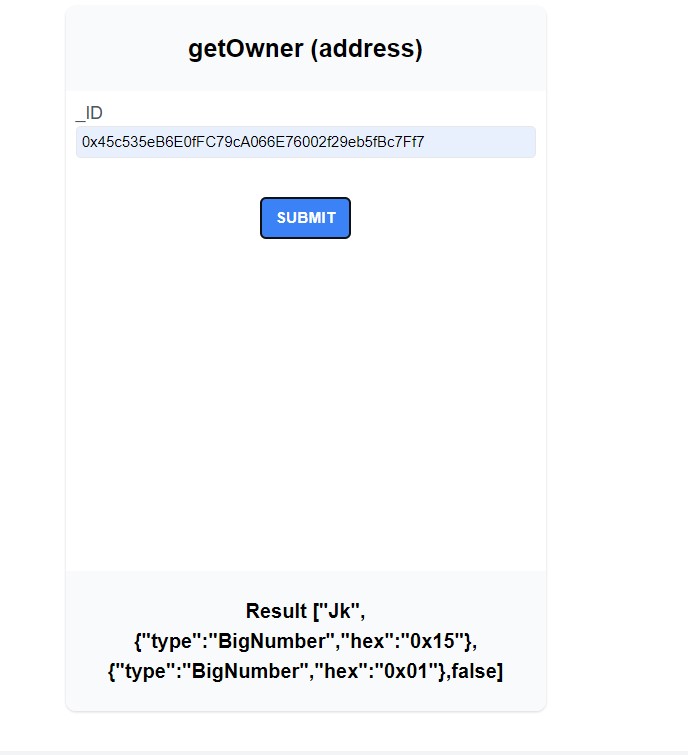
return 1;

}

}

**Input / Output and ScreenShots**





**Conclusion**

After completion of this practical I learnt how to create a Decentralized app using Dapp.