Two base station communication through IPFS

/\*\*

   \* @title ContractName

   \* @dev ContractDescription

   \* @custom:dev-run-script file\_path

   \*/

// SPDX-License-Identifier: MIT

pragma solidity 0.8.7;

contract MyContract {

    uint256 public peopleCount = 0;

    mapping(string => Person) public people;

    address owner = 0x5B38Da6a701c568545dCfcB03FcB875f56beddC4;

    address otherOwner = 0xAb8483F64d9C6d1EcF9b849Ae677dD3315835cb2;

    string ipfsHash;

    modifier onlyOwner() {

    require(msg.sender == owner || msg.sender == otherOwner);

        \_;

    }

    struct Person {

        uint id;

        string \_name;

        uint \_reputation;

        string \_uniqueID;

    }

//    constructor() public {

//        owner = msg.sender;

//    }

    function addPerson (string memory \_name, uint \_reputation, string memory \_uniqueID, address \_blockhash) public onlyOwner {

        // here \_blockhash = \_blockAddress of addBlock - represents hash of the latest block added by node

        people[\_uniqueID] = Person(peopleCount, \_name, \_reputation, \_uniqueID);

        incrementCount();

    }

    function incrementCount() internal {

        peopleCount += 1;

    }

    //upload node data to IPFS - Go to first owner account. Go to 'Deploy & Run Transactions'. Click Save in Transactions Recorded. Save the resulting json file on local system. Upload on IPFS

    function sendHash (string memory X) public onlyOwner {

        ipfsHash = X;

    }

    //request node data from IPFS - Go to second owner account. Click Get hash. Paste on IPFS and get json file. Copy file in Remix IDE. Go to 'Deploy & Run Transactions'. Click Run in Transactions Recorded. Check required node

    function getHash() public view onlyOwner returns (string memory) {

        return ipfsHash;

    }

}

//0x5B38Da6a701c568545dCfcB03FcB875f56beddC4

//0x7465737400000000000000000000000000000000