Project Design Phase-I

proposed solution

Date	19 September 2022
Team ID	NM2023TMID00434
Project Name	Ethereum Decentralised Identity Smart Contarct
Maximum Marks	4 Marks

proposed solution:

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.0;

contract Decentralized Identity {

  struct Identity {
    address owner; // Ethereum address of the identity owner
    string name;   // Name associated with the identity
    // Add more identity attributes as needed
  }

  mapping(address => Identity) public identities; // Mapping from Ethereum address to Identity

  event Identity Created(address indexed owner, string name);
  event Identity Updated(address indexed owner, string newName);

  // Create a new identity
  function create Identity(string memory _name) public {
    require(bytes(_name).length > 0, "Name must not be empty");
    require(identities[msg.sender].owner == address(0), "Identity already exists");

    Identity storage newIdentity = identities[msg.sender];
    newIdentity.owner = msg.sender;
    newIdentity.name = _name;
```

```
        emit IdentityCreated(msg.sender, _name);
    }

    // Update the name associated with an identity
    function updateIdentity(string memory _newName) public {
        require(bytes(_newName).length > 0, "New name must not be empty");
        require(identities[msg.sender].owner != address(0), "Identity does not exist");

        identities[msg.sender].name = _newName;

        emit IdentityUpdated(msg.sender, _newName);
    }

    // Get identity details for a given address
    function getIdentity(address _owner) public view returns (address, string memory) {
        Identity memory identity = identities[_owner];
        return (identity.owner, identity.name);
    }
}
```