COURSE NAME: BLOCK CHAIN

GROUP NUMBER: 01

PROJECT TITLE: IDENTITY DETAILS

PROJECT SUBMITTED TO: ANNA UNIVERSITY/NAAN MUDHALVAN

YEAR: 2ND

DEPARTMENT: CSE

SEMESTER: 04

GROUP MEMBERS: NEHA.R, JASMINE.N

GUIDED BY: Dr.CH.SARADA DEVI

SPOC NAME: Dr.R.PARKAVI

*Report on IdentityDetail*s *SmartContract*

*Introduction:*

*The IdentityDetails contract is a Solidity smart contract designed to store identity information. The contract allows users to add their identity details, which includes their name, date of birth, age, email address, phone number, location address, city, state, country, and postal code.*

# *Contract Overview:*

*The IdentityDetails contract is implemented using the Solidity programming language and follows the version ^0.8.0. The contract has a single struct named "Identity" which contains all the necessary fields for storing identity information.*

*The contract also has a mapping named "identities" which is used to store the identity information for each user. The mapping uses the user's Ethereum address as the key and the "Identity" struct as the value.*

*Additionally, the contract has an "owner" variable, which is set to the address of the contract creator. The contract owner can access the "getIdentity" function, which returns the identity information for a given Ethereum address.*

# *Functions:*

***1.******addIdentity :*** *This function is used to add identity information to the contract. The function takes ten parameters: name, date of birth, age, email address, phone number, location address, city, state, country, and postal code. The function stores the identity information in the "identities" mapping using the Ethereum address of the user as the key.*

***2. getIdentity******:*** *This function is used to retrieve identity information from the contract. The function takes a single parameter, which is the Ethereum address of the user whose identity information is to be retrieved. The function can only be called by the contract owner.*

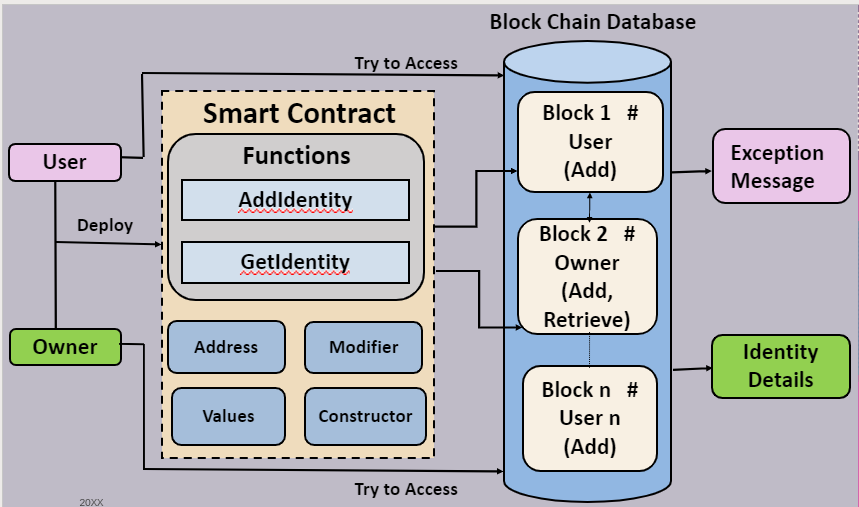
*Modifiers:*

*The contract has a single modifier named "onlyOwner" which restricts access to the "getIdentity" function to the contract owner.*

# *Conclusion:*

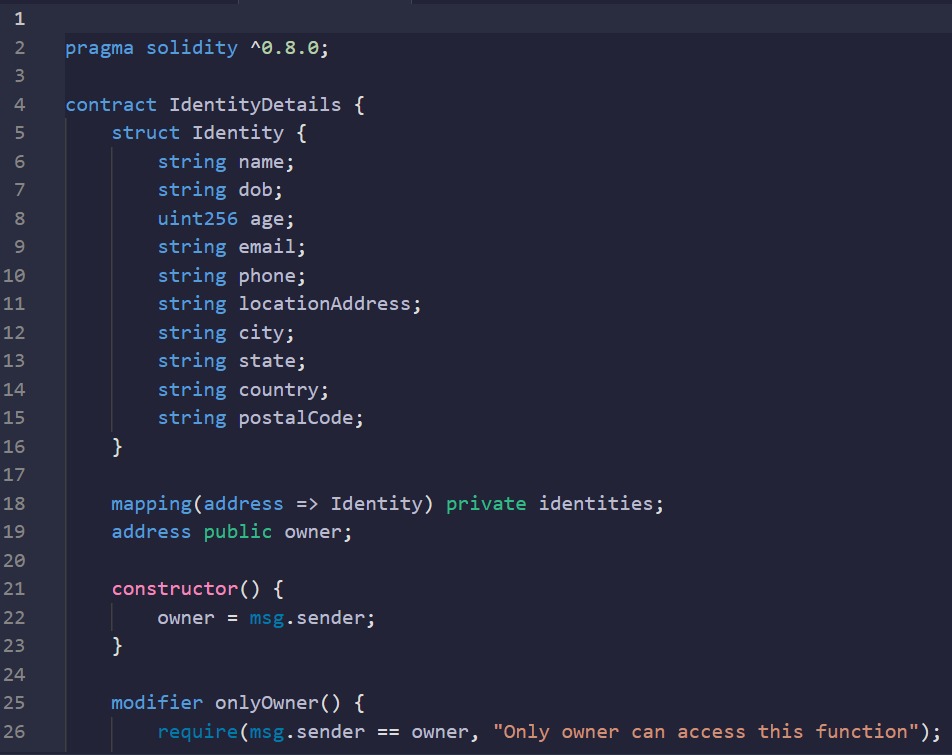
*The IdentityDetails contract is a useful smart contract for storing identity information on the Ethereum blockchain. The contract is implemented using Solidity, and it allows users to add their identity details and retrieve them later. The contract is simple yet effective, and it can be easily modified or extended to meet the needs of different projects.*

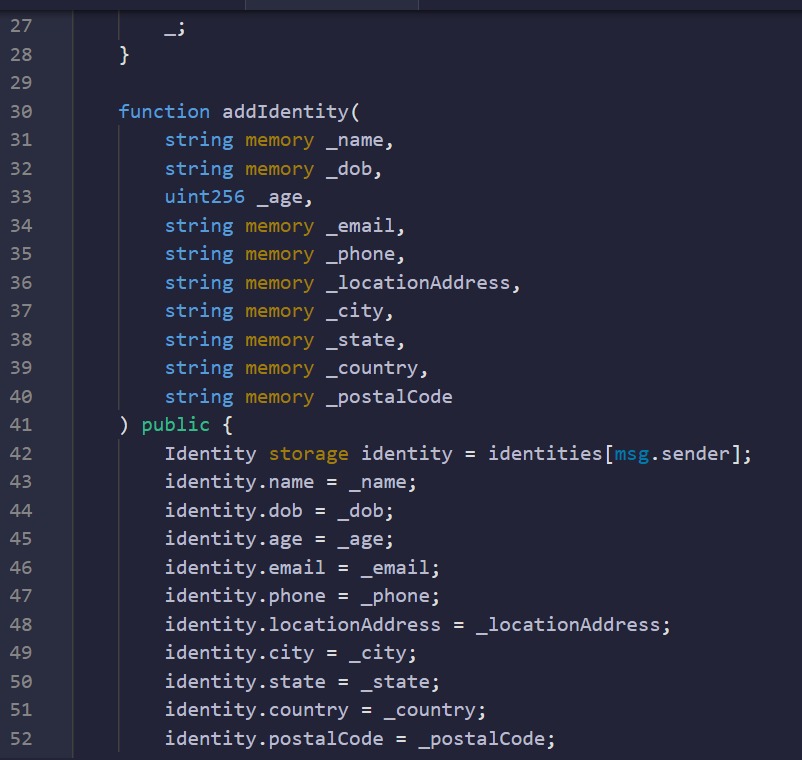
***Below is the Architecture diagram for the IdentityDetails Solidity smart contract designed to store and retrieve identity information***



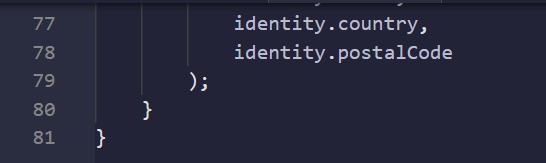
* *The above is the architecture diagram for the IdentityDetails Etherium Smartcontract which uses Blockchain to store and retrieve data.*
* *The SmartContract can be accessed by any number of users but only one person(owner) can retrieve the details. The diagram represents two users, owner and user. The owner is the one who deploys the contract first. Only the owner can retrieve the details of other users. The users can only add their details.*
* *The Smart Contract consists of various elements such as:*
* *Functions (AddIdentity,GetIdentity)*
* *Modifier*
* *Constructor*
* *Etherium Addresses*
* *It uses a struct and a mapping to store the identity information, and has functions to add and retrieve this information.*
* *The `addIdentity()` function allows a user to add/update their own identity information to the mapping, while the `getIdentity()` function allows the contract owner to view any user's identity information.*
* *The contract also has a modifier that restricts access to the retrieval function to the contract owner.*
* *It has a constructor which sets the owner variable to the address that deployed the contract.*
* *After the smart contract is executed and the input is given into the functions and transacted, the data of owner is stored in block 1 in the blockchain and the data of user2 is stored in block 2.*
* *This block chain is stored in the blockchain database.*
* *When a user tries to access the details of other users using the GetIdentity function, an exception message will be shown. If the user is the owner, and if he tries to access the BlockChain, Identity details of other users will be displaye*d.

*PROGRAM:*









*OUTPUT:*

