NEW HORIZON COLLEGE OF ENGINEERING

Autonomous College Permanently Affiliated to VTU, Approved by AICTE & UGC
Accredited by NAAC with 'A' Grade, Accredited by NBA

A MINI PROJECT REPORT

for

Mini Project in BLOCKCHAIN (20NHOP714)

on

PaySmart

Submitted by

Shreyas Chaudhary

USN: 1NH19CS743, Sem-Sec: 7-E

In partial fulfillment for the award of the degree of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING

Academic Year: 2022-23

ABSTRACT

Crowdfunding is a method for an individual or group to generate cash for a project by having many individuals contribute small amounts of money, usually over the internet. In the current environment, an entrepreneur needs money to implement his ideas whenever he wants to. Before, when crowdfunding platforms weren't available, business owners could only target a small group of people for project funding, and it was very challenging to find people who could contribute to the project. Nowadays, there are a lot of crowdfunding platforms, such as Kickstarter, that allow entrepreneurs toput their ideas on the site, where backers can see the project and give money to it. These crowdfunding platforms make it easier for entrepreneurs to reach out to a big group of people all over the world who may help them fund their businesses. Despite the many benefits of the existing crowdfunding system, there are several issues with it, such as the high cost of upkeep, lack of transparency in the system, and lack of confidence. By offering a more transparent system where every transaction can be stored on the blockchain using Smart Contracts, our program can eliminate the difficulties associated with the old crowdfunding method.

ACKNOWLEDGEMENT

The satisfaction and euphoria that accompany the successful completion of any task would be impossible without the mention of the people who made it possible, whose constant guidance and encouragement crowned our efforts with success.

I have great pleasure in expressing my deep sense of gratitude to **Dr. Mohan Manghnani**, Chairman of New Horizon Educational Institutions for providing necessary infrastructure and creating good environment.

I take this opportunity to express my profound gratitude to **Dr. Manjunatha**, Principal NHCE, for his constant support and encouragement.

I am grateful to **Dr. R. J. Anandhi**, Dean of Academics, for her unfailing encouragement and suggestions, given to me in the course of my project work.

I would also like to thank **Dr. B. Rajalakshmi**, Professor and Head, of the Department of Computer Science and Engineering, for her constant support.

I express my gratitude to **Ms. Yogitha,** Sr. Assistant Professor, and my project guide, for constantly monitoring the development of the project and setting up precise deadlines. Her valuable suggestions were the motivating factors in completing the work.

Finally, a note of thanks to the teaching and non-teaching staff of the Dept of Computer Science and Engineering, for their cooperation extended to me, and my friends, who helped me directly or indirectly during the project work.

SHREYAS
CHAUDHARY
1NH19CS743

CONTENTS

CHAPTER 1

INTRODUCTION

1.1 Background

The recent advent of technology is affecting all parts of human life and is changing the way we use and perceive things previously. Just like the changes technology has offered in various other sectors of life, it is also finding new ways for improvement in healthcare sector. The main benefits that advancement in technology is offering are to improve security, user experience and other aspects of the healthcare sector. These benefits were offered by Electronic Health Record (EHR) and Electronic Medical Record (EMR) systems. However, they still face some issues regarding the security of medical records, user ownership of data, data integrity etc.The solution to these issues could be the use of a novel technology, i.e., Blockchain. This technology offers to provide a secure, temper-proof platform for storing medical records and other healthcare-related information. Before the advent of modern technology, healthcare sector used paper-based system to store medical records, i.e., using handwritten mechanism. This paper-based medical record system was inefficient, insecure, unorganized, and not temper-proof. It also faced the issue of data- duplication and redundancy as all the institutions that patients visited had various copies of patient‟s medical records, patients‟ appointment management, billing and accounts, and lab tests. They are available in many of the EHR system being used in the healthcare sector. The basic focus is to provide secure, temper-proof, and shareable medical records across different platforms. Even though notion behind usage of EHR systems in hospitals or healthcare was to improve the quality of healthcare, these systems faced certain problems and didn‟t meet the expectations associated with them data entries have been compromised in these systems since October 2009. Another study conducted by Argaw et al. [8], explains that hospitals have become a target of cyber-attacks and an increasing trend has been witnessed by the researchers while conducting this study that a lot of research work has been done in this domain by storing the data on that medium.

Moreover, our proposed work is intending to solve the above-mentioned information asymmetry and data breaches problem faced by the EHR system. This paper is organized as follows section II of this paper summarizes the basics of blockchain technology and its dependencies; section III narrates the related work done in this domain. The section IV explains the design and architecture of the proposed framework and section V explains the performance of this framework. The last section provides the conclusion and references. the double spending problem of bitcoin but soon this novel technology was being used in many other applications. Blockchain is a chain of blocks that are connected and are continuously growing by storing transactions on the blocks. This platform uses a decentralized approach that allows the information to be distributed and that each piece of distributed information or commonly known as data have shared ownership. Blockchains holds batches of transactions that are hashed thus providing them security and they are managed by peer-to-peer networks. A blockchain has certain benefits such as security, anonymity, and integrity of data with no third-party intervention. These benefits make it a reasonable choice to store patient's medical records on it because the innovation of technology In the healthcare industry has made the security of patients' medical data a top priority. Several researchers have also identifiedthat using blockchain technology in healthcare would be a feasible solution.

FIG: 1 BLOCKCHAIN IN FREE HOLD

1.2 BLOCKCHAIN TECHNOLOGY AND ITS DEPENDENCIES:

This technology was introduced by Satoshi Nakamoto [13], for his popular work of digital currency or crypto-currency, i.e., bitcoin. Nakamoto used blockchain technology to solve the double spending problem of bitcoin but soon this novel technology was being used in many other applications. Blockchain is a chain of blocks that are connected together and are continuously growing by storing transactions on the blocks. This platform uses a decentralized approach that allows the information to be distributed and that each piece of distributed information or commonly known as data has shared ownership. Blockchains holds batches of transactions that are hashed thus providing them security and they are managed by peer-to-peer networks. A blockchain has certain benefits such as security, anonymity, and integrity of data with no third-party intervention. These benefits make it a reasonable choice to store patients' medical records on it because the innovation of technology in the healthcare industry has made the security of patients "s medical data a top priority. Several researchers have also identified that using blockchain technology in healthcare would be a feasible solution [14] [15][16]. A. ARCHITECTURE To understand the blockchain architecture let us use the following figure 1 which explains the whole process of a transaction being send from a user on the blockchain network. 1. A new transaction being sent by a user on the blockchain network suggests that a new block is created. A block in the blockchain is used for keeping transactions in them and these blocks are distributed to all of the connected nodes in the network. That transaction placed inside a block is broadcasted to all of the nodes in the network. All the nodes in the network have a copy of the complete blockchain that helps them in the verification process. When a block containing the user transactionis broadcasted to all the connected nodes, they verify that the block is not tampered by any means. If this verification results in success, then the nodes add that block in their own copy of blockchain. 2. This whole process of the block is added on the blockchain is done by the nodes reaching a consensus where they decide which blocks are valid to be added on the blockchain and which are not. This validation is performed by the connected nodes using some known algorithms to verify the transaction and to ensure that sender is an authenticated part of the network. When a node succeeds in performing the validation that node is rewarded with

crypto-currency. This process of validating the transaction is known as mining and the node performing this validation is known as miner. 3. After validation is done that block is added to the blockchain. 4. After the whole process of validation is performed the transaction is completed. This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/. This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2019.2946373, IEEE Access Some basic concepts of blockchain technology can be understood in the following descriptions.

B. BLOCK As explained earlier blockchain are formed together by a number of blocks connected together in a peer-to-peer network thus making a decentralized application. The header of these blocks contains hashes of previous blocks in them. A block contains three things in it which are data, hash of current block and hash of previous block. The data could be anything as it depends on the type of blockchain. As in case of bitcoin, the data consists of coins that are actually electronic cash [13]. The hash that is stored in these blocks contains a SHA-256 cryptographic algorithm which is used for unique identification of a block on the chain.

C. CONSENSUS ALGORITHM Each block that is added on the chain would need to follow some consensus rules for it to be added on the blockchain. For this purpose blockchain technology uses consensus algorithms. The most common consensus algorithm used is Proof of Work (PoW) algorithm and it was used by Nakamoto [13], in bitcoin network. The basic working of this algorithm is that there are number of nodes or participants on a blockchain network so when a transaction is requested to be added on the network by any participating node it needs to be calculated. This process is called mining and the nodes that are performing these calculations are miners .

D. KEY FEATURES OF BLOCKCHAIN Decentralization: With blockchain the information is distributed across the network rather than at one central point. This also makes the control of information to be distributed and handled by consensus reached upon by shared input from the nodes connected on the network. The data that was before concentrated at one central point is

now handled by many trusted entities. Data transparency: Achieving data transparency in any technology is to have a trust-based relationship between entities. The data or record at stake should be secured and temper-proof. Any data being stored on the blockchain is not concentrated at one place and is not controlled by one node but is instead distributed across the network. The ownership of data is now shared and this makes it to be transparent and secure from any third-party intervention. Security and Privacy: Blockchain technology uses cryptographic functions to provide security to the nodes connected to its network. It uses SHA-256 cryptographic algorithm on the hashes that are stored on the blocks. SHA stands for Secure Hashing Algorithm, these hashes provide security to the blockchain as data integrity is ensured by them. Cryptographic hashes are strong one-way functions that generate a checksum for digital data that cannot be used for data extraction. This makes blockchain a decentralized platform made secure by cryptographic approaches which makes it to be a good option for the privacy protection of certain applications.

E. CHALLENGES FACED BY BLOCKCHAIN TECHNOLOGY Scalability and storage capacity: Storage of data on the blockchain causes two main problems, i.e., confidentiality and scalability. The data on the blockchain is visible to everyone that is present on the chain this makes the data vulnerable which is not a desired outcome for a decentralized platform. The data stored on the blockchain would contain patient medical history, records, lab results, X-rays reports, MRI results and many other reports, all of this voluminous data is to be stored on the blockchain that would highly affect the storage capacity of blockchain [18]. Lack of social skills: The way the blockchain technology works is understandable by very few people. This technology is still in its initial phases and is constantly evolving. Moreover, the shift from trusted EHR systems to the blockchain technology would take time as hospitals, or any other healthcare institutes need to completely shift their systems to blockchain. Lack of universally defined standards: As this technology is still in the initial phases and is constantly evolving so there is no defined standard for it. Due to this the implementation of this technology in healthcare sector would also take more time and effort. As it would require certified standards from international authorities that overlook the standardization process of any technology [19]. These universal standards would benefit in

deciding upon the data size, data format and type of data that could be stored on the blockchain. Moreover, the adaptation of this technology would become easier due to the defined standards, as they could be easily enforced in the organizations.

Chapter-2

IMPORTANCE OF SOLIDITY AND SMART CONTRACT

2.1 Introduction to Solidity

Solidity is contract oriented high-level programming language which is used for writing smart contacts. It was developed by Gavin Wood, Christian Reitwiessner, Alex Beregszaszi, Liana Husikyan, Yoichi Hirai and several former Ethereum core contributors. Solidity is used for implementing smart contracts on various blockchain platforms. This platform has a lot of similarity with JavaScript (another scripting language). Solidity does the process of verifying and enforcing the constraints at compile-time as opposed to the run-time. Solidity is basically there to develop and execute code in the Ethereum Virtual Machine.

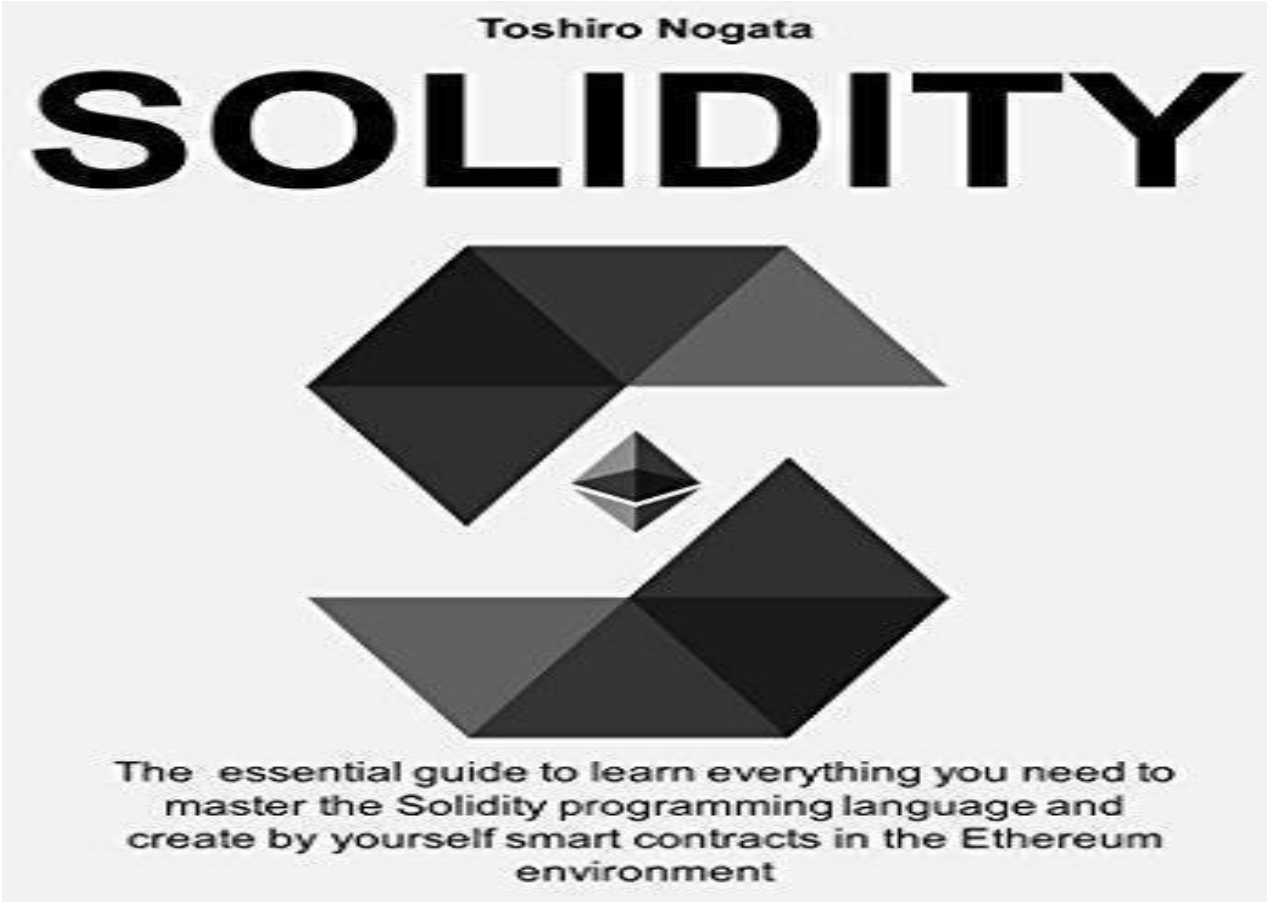


FIG:2 SOLIDITY

2.2 What are Smart Contracts?

Smart contracts are the best means to exchange money, property, shares, or anything of value in a clear way without any issues. This basically eliminates the role of middleman. The best way to go into depth of smart contracts is by comparing the technology to a vending machine. If we go through the normal process, you will visit the lawyer or notary, pay them the amount. After that, you will wait to get your documents but if we go with the smart contracts, you will just simply drop bitcoin into the vending machine (also known as ledger) and your escrow, driver's license, or whatever drops into your account. To the more, if we consider the traditional contract, there are rules and penalties for the same but in smart contracts, these are not stated but obligations are automatically attached to it

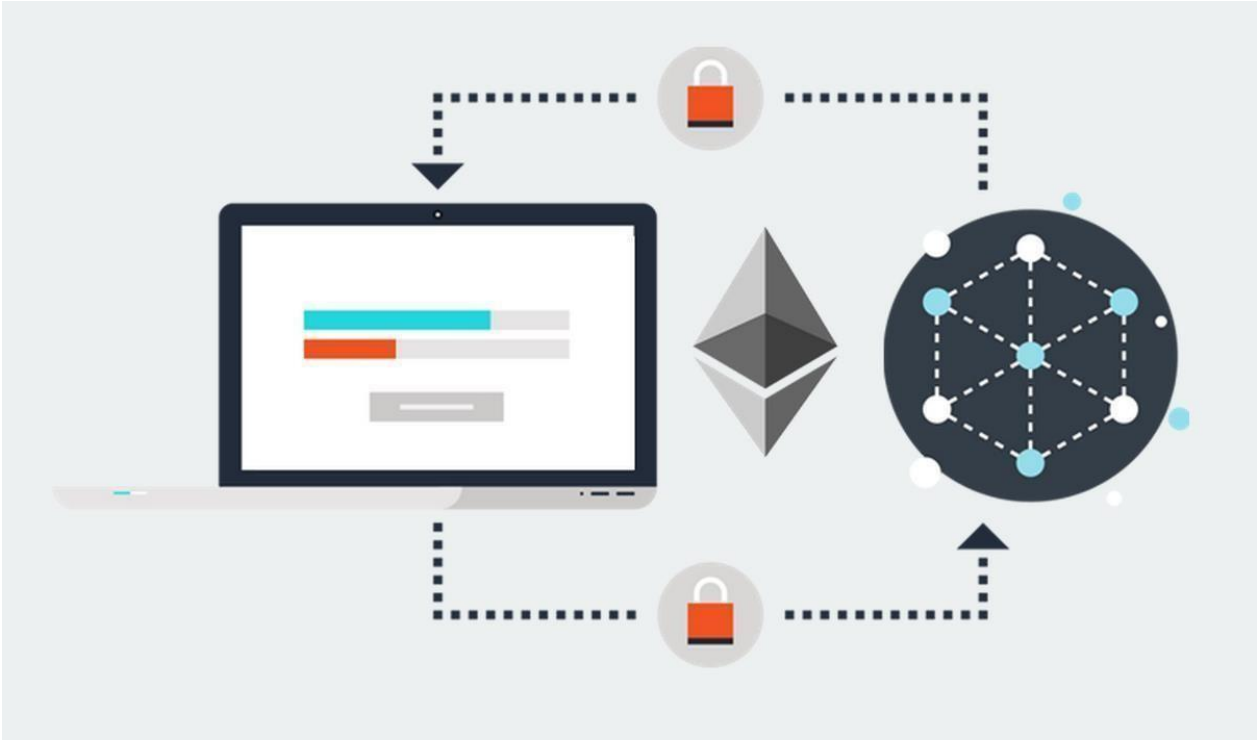


FIG:3 SOLIDITY CONTRACT

One of the most striking cases is about upgradeable contracts. It is not possible to change the code of a working smart contract once it has been started. This is due to immutable nature of the

blockchain. But if we use the method for assigning calls, a proxy can be arranged by pointing another contract which contains the actual business logic. This work actually helped to enhancethe working of the contract by providing a different target address to the proxy contract. For example, implementing a newly positioned version of the marked contract while fixing some bugs. The similar method can be adjusted to use other contracts as libraries. This actually decreases deployment costs, as the contract using the library does not require to execute of all the code itself

2.3 ARCHITECTURE OF SOLIDITY

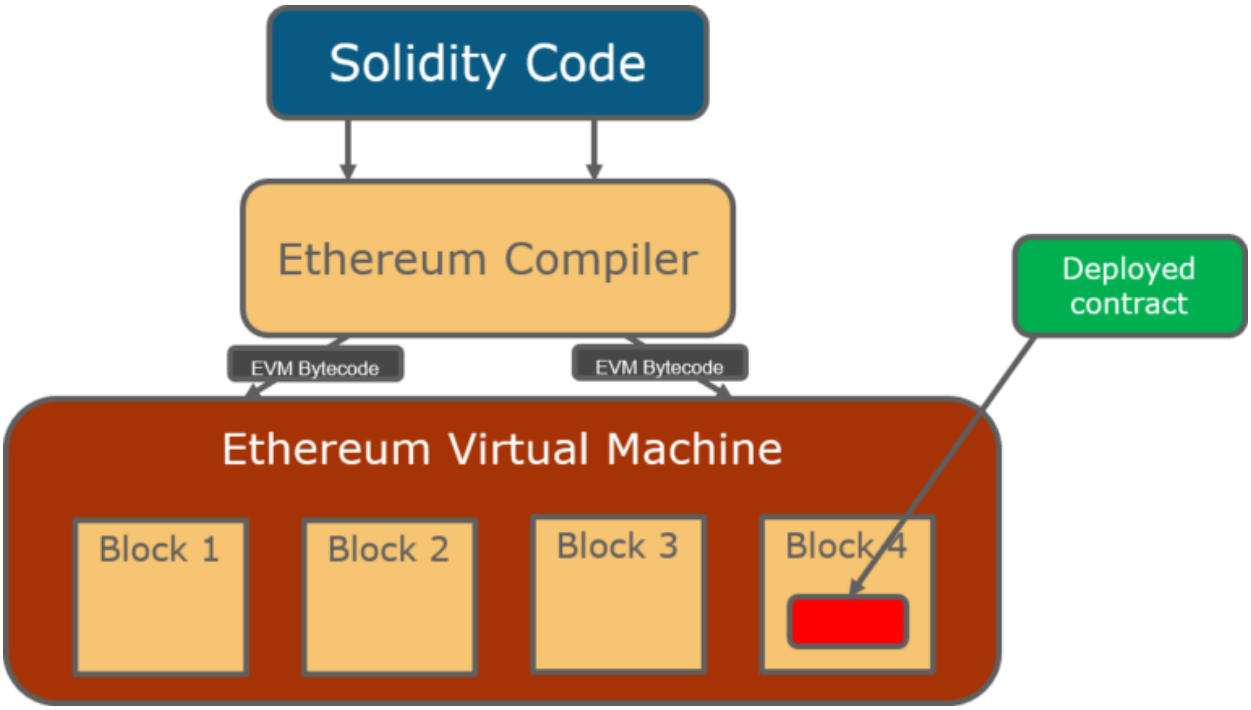


FIG:4 ARCHITECTURE

Solidity is compiled to bytecode (or portable code) that is executable on the Ethereum Virtual Machine (EVM), the runtime environment for smart contracts in Ethereum. It sets the rules and

penalties related to an agreement, much like a traditional contract does, but can also enforce those obligations automatically.

2.4 Compiling and deploying the smart contract

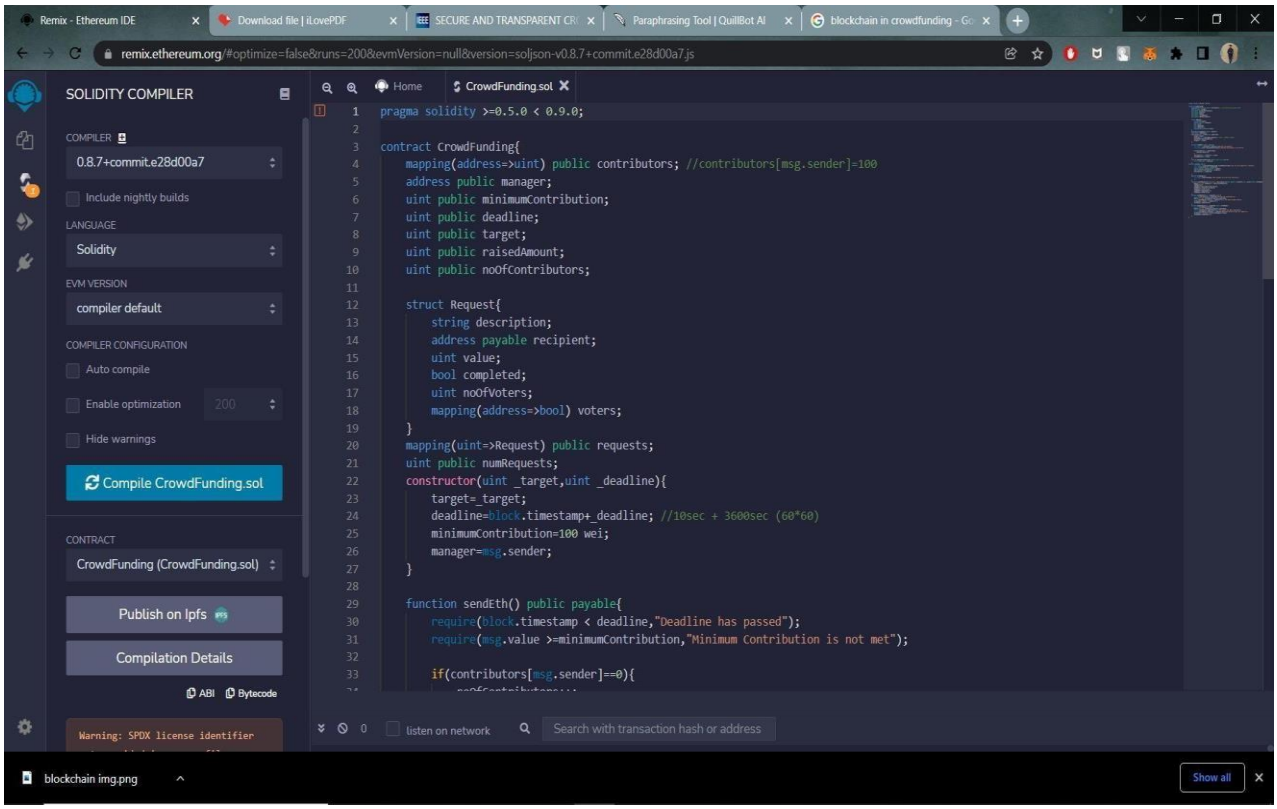


FIG 5: Smart contract compilation

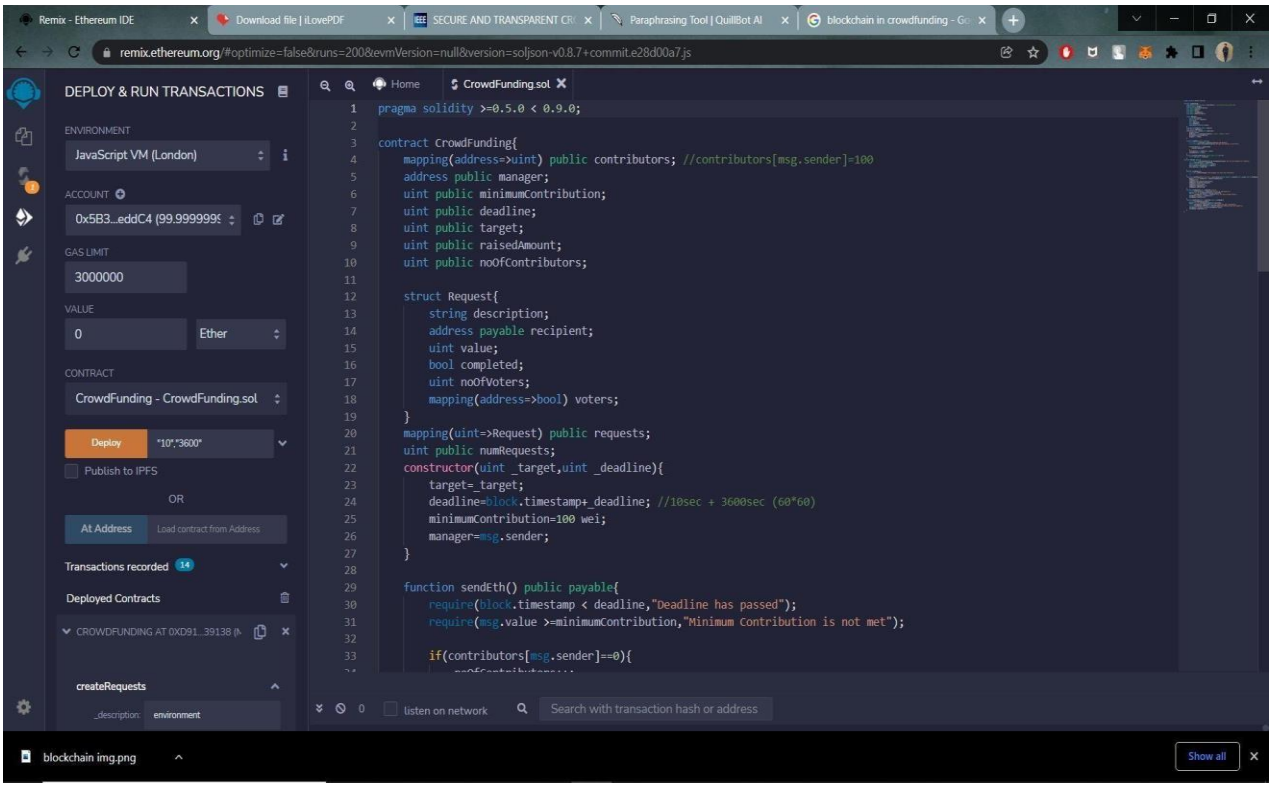


FIG 6: Smart contract Deploy

Deploying and Testing a Smart Contract Once you've compiled a contract in Remix, you can use the "run" tab to deploy it. The "environment" drop-down gives three options for where to deploy the contract: 1. JavaScript VM - This lets you run your contract directly in the browser using a JavaScript implementation of the Ethereum virtual machine (EVM). This is great for simple testing but doesn't allow anyone else to interact with your contract. 2. Injected Web3 - Web3 is the interface for interacting with an Ethereum node. If you're using the MetaMask browser.Extension. It injects an implementation of Web3 into every web page. This option will let you use that injected implementation to deploy to a test network or the main Ethereum network. 3. Web3 Provider - This option connects directly to an Ethereum node via HTTP. If you're running your own node, you can use this option to connect to it.

CHAPTER 3

SYSTEM DESIGN AND REQUIREMENTS

The related work section includes the work done in the domain of health care being implemented using blockchain technology. As mentioned, they provided certain solutions for solving the prevalent problems in blockchain technology. The studies in the discussion were mainly addressing the problems of scalability and data sharing through blockchain. They propose the solution of using an underlying database, pertaining to some ONC requirements and any other defined standards to solve them. In contrast to those solutions, our proposed framework offers to solve this problem of scalability by using off-chain scaling mechanism of IPFS. Moreover, Ethereum is used for the overall implementation of the proposed framework. Ethereum and its dependencies are also discussed in the previous sections of this paper.

A. SYSTEM DESIGN

System design is the most important and vital part of any framework as it is used for the development of the system from its theory. This section includes the modules, architecture and various elements that are combined to form the whole system's framework. As defined earlier the purpose behind this proposed framework is to create a decentralized system that is tamper-proof, secure, and confidential blockchain-based system for electronic health records. As visible in below figure 2, the proposed framework or system has three entities or modules. These modules when combined would keep our system working. These entities or modules have further concepts that need to be understood they are explained as follows.

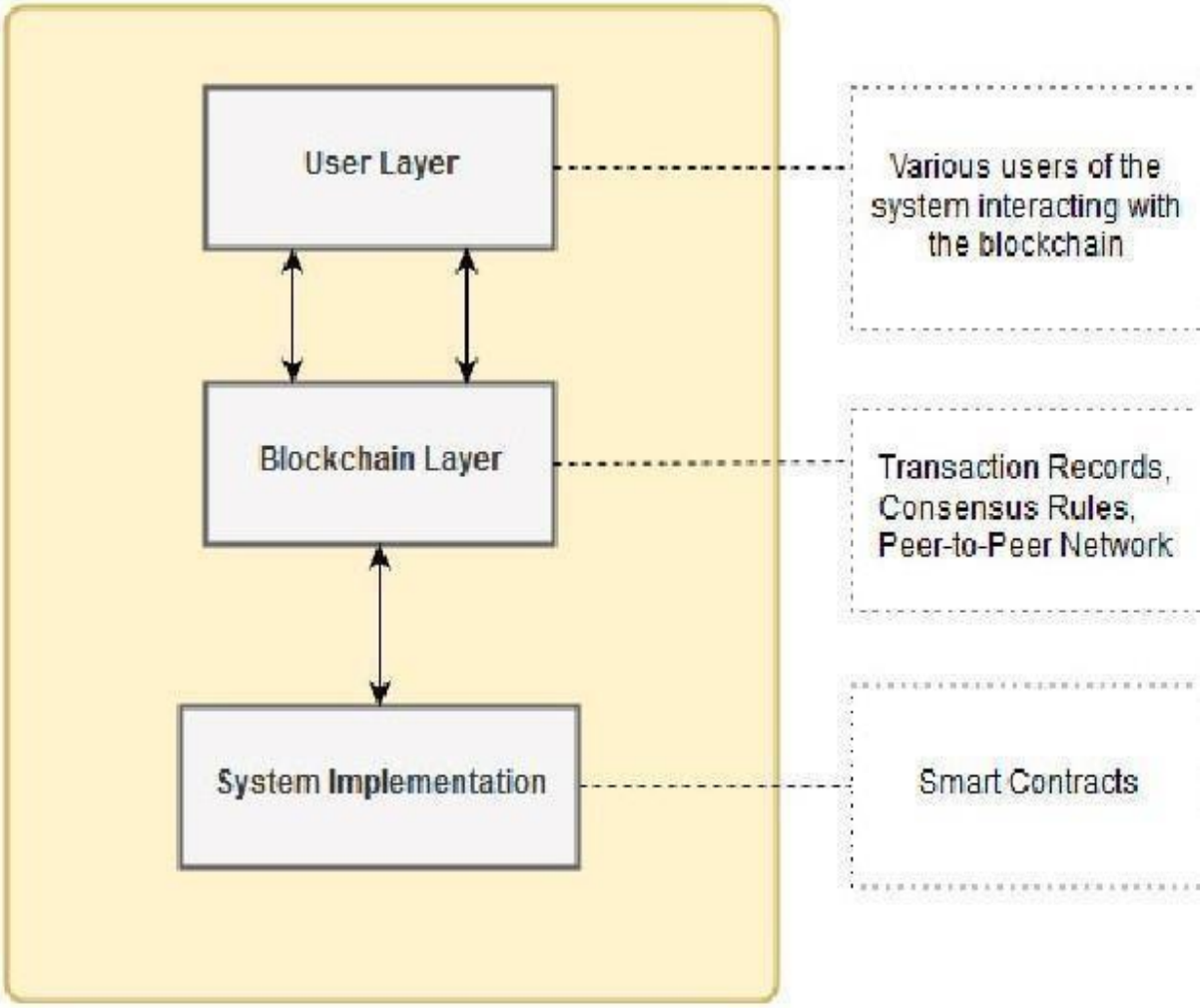


FIGURE 2. System design of proposed framework.

The proposed framework consists of users that could be patients, doctors, administration and nursing staff. They were given granular access as they should have varying level of authority on the system.

1) USER LAYER

A user of a system is defined as an individual who makes effective use of the system and its resources. A user has various roles and features on the system, making him identifiable on the system. The users of this system could be patients, doctors, administrative staff etc. The main task of these users would be to interact with the system and perform basic tasks such as creating, reading, updating, and deleting medical records. The users using this system would be accessing thesystem's functionality through a browser which in technical terms we refer to as DApp browser, as it is

containing the GUI (Graphical User Interface) of the DApp, i.e., our proposed system framework. The GUI contains all the functions that could be accessed by a particular user. The user according to the assigned role could use this GUI for interacting with the other layer of the system, i.e., the blockchain layer. 2) BLOCKCHAIN LAYER The next layer of the system is the blockchain layer; this layer contains the code or mechanism for the interaction of the user with the DApp which is functioning on the blockchain. This layer contains three elements inside it. They are • Blockchain Assets: In Ethereum blockchain, the transaction is the process by which external users can update the state of the record or information stored on the Ethereum blockchain network. These transactions are treated as assets by the Ethereum blockchain as they are pieces of information that a user can send to another user or simply store for using it later. • Governance Rules: Blockchain technology in general follows some consensus rules for its transactions to be done and computed. For this purpose, it needs some consensus algorithms to keep the blockchain temper-proof and secure. Ethereum blockchain uses Proof of Work (PoW) consensus algorithm, the reason behind using it is also for ensuring that governance of blockchain is maintained in a trusted manner which is through consent from all the trusted nodes attached to the blockchain network. • Network: Ethereum blockchain uses the peer-to-peer network. In this network, all the nodes are connected as peers. With no node acting as the central node controlling all the functions of the network. The reason behind using this network was because the idea was to create a distributed platform not a centralized. So, using a network where all the connected nodes have equal status and right was the best choice this technology could have done. TRANSACTION The system includes following transactions:

- Add records would create the patient's medical records in the DApp. It contains the fields of ID, name, co-morbid, blood group, and IPFS hash. The patient's basic medical records are stored along with the IPFS hash that contains the file uploaded containing the lab results or other medical records of a patient.
- Update records would update the medical records of patient. This can only change the basic information of the patient not the IPFS hash. IPFS hash is non-updateable to ensure security of records.
- View records would let the user view the medical records of a patient stored in DApp. The view records function is used both by doctors and patients. The patient can view his records by the system authenticating that patient views only his own medical records. For

this purpose system uses the public account address of the patient to ensure that only the relevant medical records are shown to the patient.

- Delete records would make the user able to delete the record of any patient. The user herewould be the doctors they are given the right to delete any patient's record stored on the blockchain.
- Grant access for each of the above-mentioned transactions, certain users would need to have access to them, i.e., only the doctor or nursing staff can make changes in the records of the patient or add them. So, adding and updating records would only be accessible to theseentities. Moreover, patients can view their medical records, but win be given access toadd or update them.

3) SYSTEM IMPLEMENTATION:

As already explained in the previous sections, the system was implemented by using Ethereum and its dependencies. This section explores system implementation in more detail to get an insight on the system various functions. SMART CONTRACTS As explained earlier, smart contracts are an important part of DApps as they are used for performing basic operations. Following contracts are included in this framework: • Patient Records • Roles These contracts are used for giving access to the users on the DApp and performing CRUD operations on the records of patient. The Patient Records smart contract is made purely for implementing the functionality of the proposed framework. It performs the CRUD operations along with the defining roles for access of these functions. The second contract mentioned above, i.e., Roles is a predefined smart contract by the OpenZeppelin smart contract library. This library contains several smart contracts performing various functionalities that could be used for creating your own smart contracts. The reason behind using this library was to make use of the benefits it provides, i.e., tested and community-reviewed code. The Roles smart contract belongs to the Asset library, which is a sub-library of the OpenZeppelin library. The asset library contains various other contracts for defining the access rules, but the roles library provides a granular role definition mechanism which was the mainreason behind selection of this smart contract.

The algorithm for defining the Patient Records smart contract is given below. It defines all the operations that are being performed in it and the various conditions that are associated with them. It also explains how the roles are being maintained for granting access to a particular functionality.

3.1 Algorithm

```
pragma solidity >=0.5.0 < 0.9.0;

contract CrowdFunding{
    mapping(address=>uint) public contributors; //contributors[msg.sender]=100
    address public manager;
    uint public minimumContribution;
    uint public deadline;
    uint public target;
    uint public raisedAmount;
    uint public noOfContributors;

    struct Request{
        string description;
        address payable recipient;
        uint value;
        bool completed;
        uint noOfVoters;
        mapping(address=>bool) voters;
    }
    mapping(uint=>Request) public requests;
    uint public  numRequests;
    constructor(uint _target,uint _deadline){
        target=_target;
        deadline=block.timestamp+_deadline; //10sec + 3600sec (60*60)
        minimumContribution=100 wei;
        manager=msg.sender;
    }

    function sendEth() public payable{
        require(block.timestamp < deadline,"Deadline has passed");
        require(msg.value >=minimumContribution,"Minimum Contribution is not met");

        if(contributors[msg.sender]==0){
            noOfContributors++;
        }
        contributors[msg.sender]+=msg.value;
        raisedAmount+=msg.value;
    }
    function getContractBalance() public view returns(uint){
        return address(this).balance;
    }
function refund() public{
```

```
        require(block.timestamp>deadline && raisedAmount<target,"You are not eligible
        forrefund");
        require(contributors[msg.sender]>0);
        address payable user=payable(msg.sender);
        user.transfer(contributors[msg.sender]);
        contributors[msg.sender]=0;

    }
    modifier onlyManger(){
        require(msg.sender==manager,"Only manager can calll this function");
        _;
    }
    function createRequests(string memory _description,address payable _recipient,
    uint
_value) public onlyManger{
        Request storage newRequest = requests[numRequests];
        numRequests++;
        newRequest.description=_description;
        newRequest.recipient=_recipient;
        newRequest.value=_value;
        newRequest.completed=false;
        newRequest.noOfVoters=0;
    }
    function voteRequest(uint _requestNo) public{
        require(contributors[msg.sender]>0,"YOu must be contributor");
        Request storage thisRequest=requests[_requestNo];
        require(thisRequest.voters[msg.sender]==false,"You have already voted");
        thisRequest.voters[msg.sender]=true;
        thisRequest.noOfVoters++;
    }
    function makePayment(uint _requestNo) public onlyManger{
        require(raisedAmount>=target);
        Request storage thisRequest=requests[_requestNo];
        require(thisRequest.completed==false,"The request has been completed");
        require(thisRequest.noOfVoters > noOfContributors/2,"Majority does not support");
        thisRequest.recipient.transfer(thisRequest.value);
        thisRequest.completed=true;
    }
}
```

3.2 HARDWARE SPECIFICATION

REQUIREMENTS:

1. Intel core i5 generation
2. Speed of 1.8 GHz
3. Ram OF 4GB
4. Hard disk of 10GB

3.3 SOFTWARE SPECIFICATION

REQUIREMENTS:

1. Windows 10 operating system/macOS
2. Programming language: Solidity
3. IDE: Remix Ethereum IDE
4. Migration and Testing: Node.js

CHAPTER-4

CODE AND IMPLEMENTATION

4.1 SMART CONTRACT DEPLOYMENT

The following contract will implement the simplest form of cryptocurrency. It is possible to generate coins out of thin air, but only the person that created the contract can do that(it is trivial to implement a different issuance scheme). Furthermore, anyone can send coinsto each other without needing to register with a username and password. All you need is an Ethereum keypair.

This code basically lets you record patient information. **pragma**

solidity ^0.4.19:

This indicates that the source code is written for Solidity version **^0.4.19** or anything newer that does not break functionality. This is to ensure that the code doesn't behave differently with the new compiler versions. **contract Healthcare:**

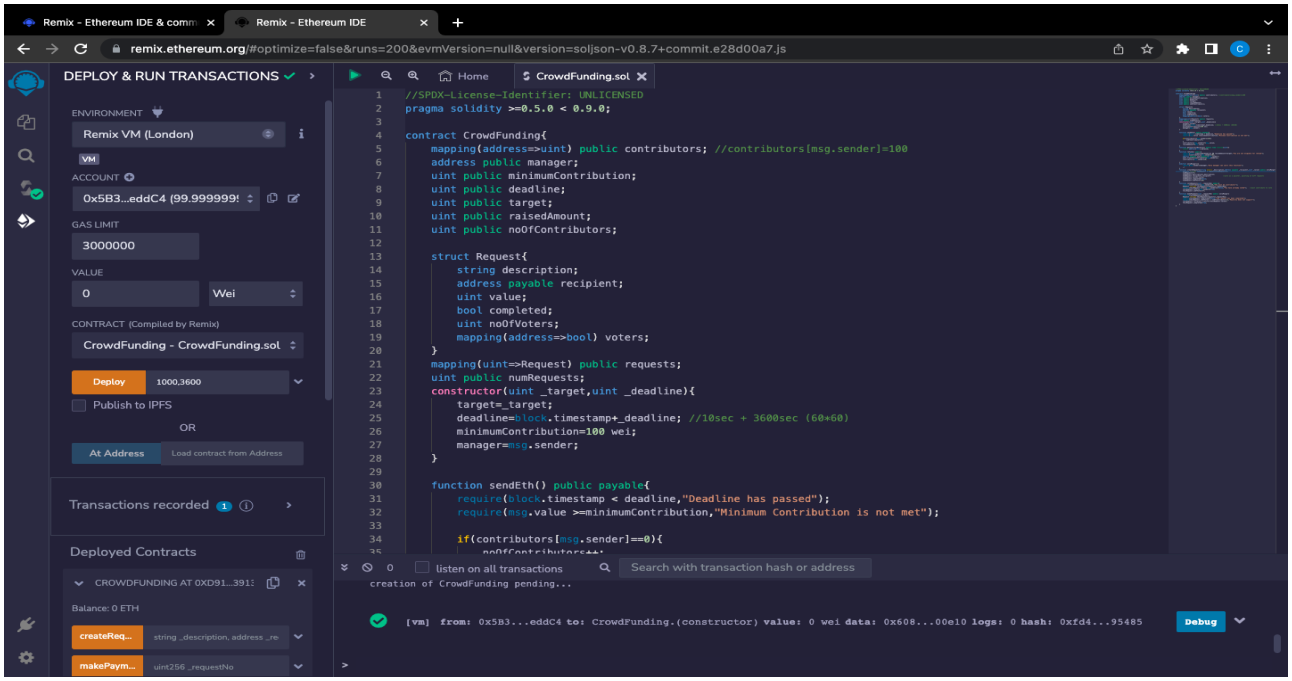
Everything related to healthcare goes inside this contract. Essentially, a contract in solidity is the collection of functions and state (code and data) sitting at an address on the Ethereum blockchain.

```
pragma solidity >=0.5.0 < 0.9.0;

contract CrowdFunding{
    mapping(address=>uint) public contributors; //contributors[msg.sender]=100
    address public manager;
    uint public minimumContribution;
    uint public deadline;
    uint public target;
    uint public raisedAmount;
    uint public noOfContributors;

    struct Request{
        string description;
        address payable recipient;
        uint value;
        bool completed;
        uint noOfVoters;
        mapping(address=>bool) voters;
    }
    mapping(uint=>Request) public requests;
    uint public  numRequests;
    constructor(uint _target,uint _deadline){
        target=_target;
        deadline=block.timestamp+_deadline; //10sec + 3600sec (60*60)
        minimumContribution=100 wei;
        manager=msg.sender;
    }

    function sendEth() public payable{
        require(block.timestamp < deadline,"Deadline has passed");
        require(msg.value >=minimumContribution,"Minimum Contribution is not met");

        if(contributors[msg.sender]==0){
            noOfContributors++;
        }
        contributors[msg.sender]+=msg.value;
        raisedAmount+=msg.value;
    }
    function getContractBalance() public view returns(uint){
        return address(this).balance;
    }
function refund() public{
        require(block.timestamp>deadline && raisedAmount<target,"You are not eligible for
refund");
```

```
        require(contributors[msg.sender]>0);
        address payable user=payable(msg.sender);
        user.transfer(contributors[msg.sender]);
        contributors[msg.sender]=0;

    }
    modifier onlyManger(){
        require(msg.sender==manager,"Only manager can calll this function");
        _;
    }
    function createRequests(string memory _description,address payable _recipient,uint
_value) public onlyManger{
        Request storage newRequest = requests[numRequests];
        numRequests++;
        newRequest.description=_description;
        newRequest.recipient=_recipient;
        newRequest.value=_value;
        newRequest.completed=false;
        newRequest.noOfVoters=0;
    }
    function voteRequest(uint _requestNo) public{
        require(contributors[msg.sender]>0,"YOu must be contributor");
        Request storage thisRequest=requests[_requestNo];
        require(thisRequest.voters[msg.sender]==false,"You have already voted");
        thisRequest.voters[msg.sender]=true;
        thisRequest.noOfVoters++;
    }
    function makePayment(uint _requestNo) public onlyManger{
        require(raisedAmount>=target);
        Request storage thisRequest=requests[_requestNo];
        require(thisRequest.completed==false,"The request has been completed");
        require(thisRequest.noOfVoters > noOfContributors/2,"Majority does not support");
        thisRequest.recipient.transfer(thisRequest.value);
        thisRequest.completed=true;
    }
}
```

CHAPTER-5

RESULTS



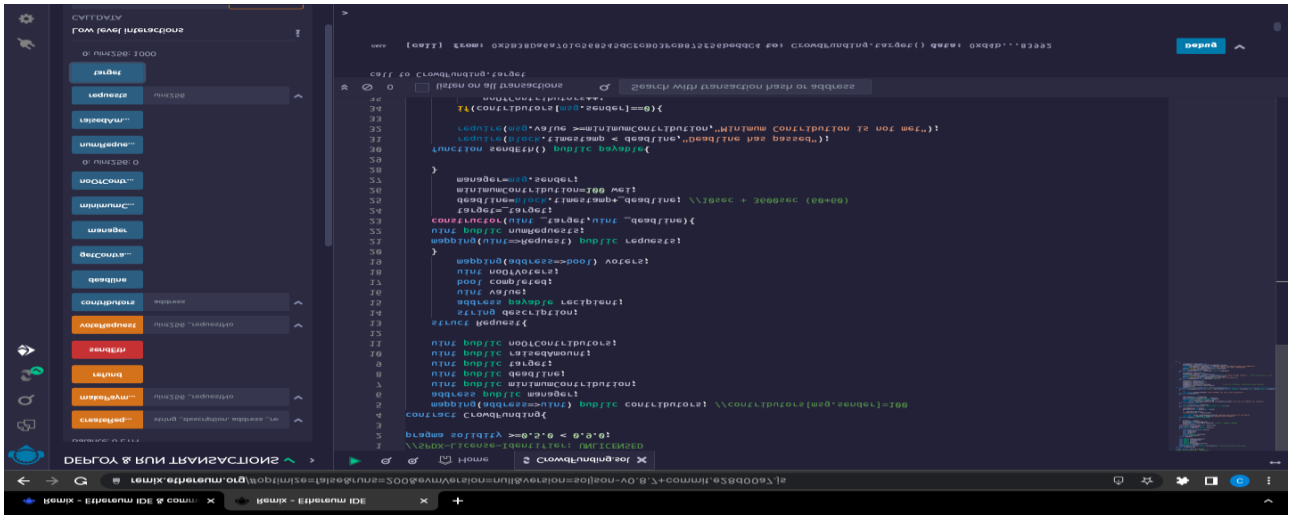


FIG:7 PaySmart USING REMIX IDE

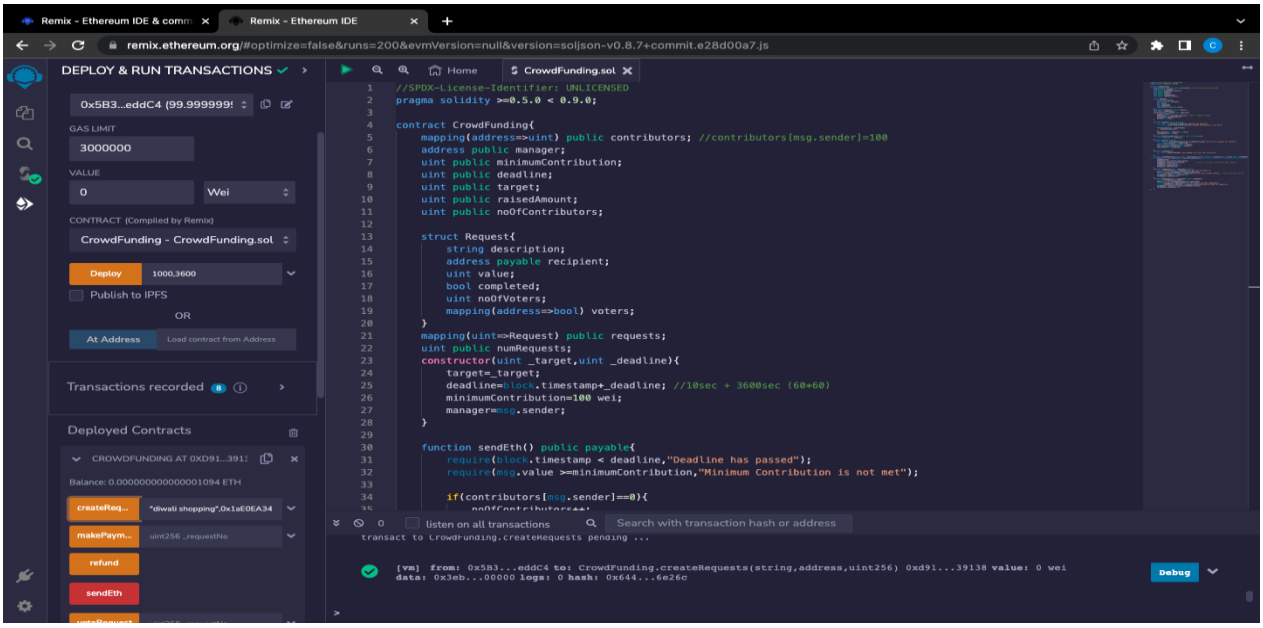


FIG:8 CREATING REQUEST

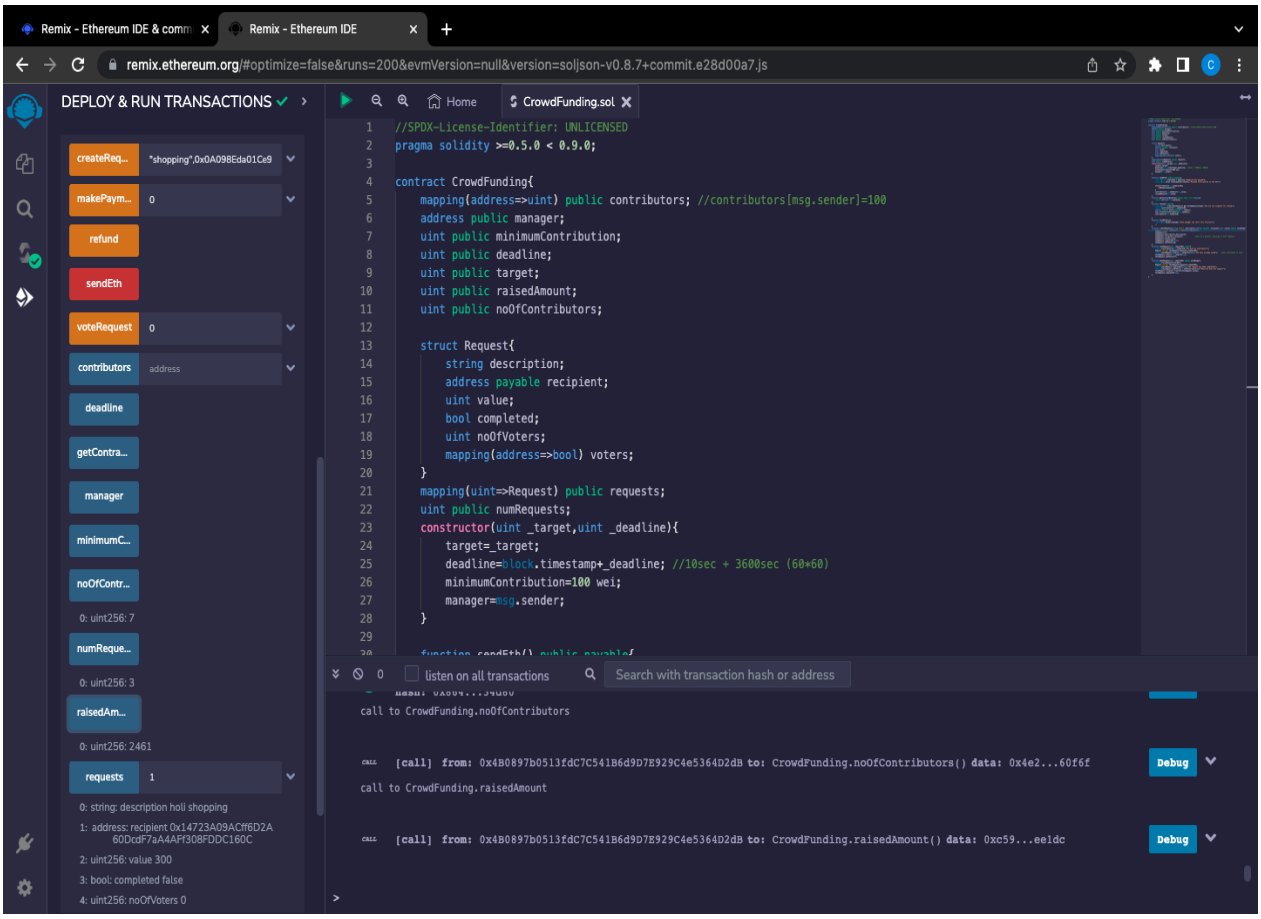


FIG:9 Making payment and voting

CHAPTER-6

CONCLUSION

Crowdfunding is a method for an individual or group to generate cash for a project by having many individuals contribute small amounts of money, usually over the internet. In today's environment, whenever an entrepreneur wants to put his idea into action, he requires money to do so. Previously, when crowdfunding platforms were not accessible, entrepreneurs could only target a small number of people for project finance and reaching out to those who could fund the project was extremely difficult Nowadays, there are a lot of crowdfunding platforms, such as Kickstarter, that allow entrepreneurs to put their ideas on the site, where backers can see the project and give money to it. These crowdfunding platforms make it easier for entrepreneurs to reach out to a big group of people all over the world who may help them fund their business. Despite the many benefits of the existing crowdfunding system, there are several issues with it, such as the high cost of upkeep, lack of transparency in the system, and lack of confidence. By offering a more transparent system where every transaction can be stored on the blockchain using Smart Contracts, our program can eliminate the difficulties associated with the old crowdfunding method.

BIBLIOGRAPHY

1] Blockchain using Solidity contracts

2] Blockchain.com

3] A. Kosba, A. Miller, E. Shi, Z. Wen, C. Papamanthou, "Hawk: The blockchain model of cryptography and privacy-preserving smart contracts", 2015.