

A Project report

on

AUCTION DAPP FOR FUNDRAISING

*Submitted in partial fulfillment of the requirements
for the award of the degree of*

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING

by

R. SREENITH (174G1A0591)

B. NITHESH KUMAR REDDY (164G1A0560)

S. GOWRI SANKAR (169L1A0577)

Under the Guidance of

Dr. G. K. Venkata Narasimha Reddy P.HD.,
Professor & HOD



Department of Computer Science & Engineering

SRINIVASA RAMANUJAN INSTITUTE OF TECHNOLOGY :: ANANTHAPURAMU
(Affiliated to JNTUA, Accredited by NAAC with 'A' Grade, Approved by AICTE, New Delhi
& Accredited by NBA(EEE,ECE&CSE))

2020-21

SRINIVASA RAMANUJAN INSTITUTE OF TECHNOLOGY
(Affiliated to JNTUA, Accredited by NAAC with 'A' Grade, Approved by AICTE, New Delhi
& Accredited by NBA(EEE,ECE&CSE))
Rotarypuram Village , B K Samudram Mandal , Ananthapuramu – 515701



Certificate

This is to certify that the project report entitled **Auction Dapp for Fundraising** is the bonafide work carried out by **R. Sreenith** bearing Roll Number **174G1A0591**, **B. Nithesh Kumar Reddy** bearing Roll Number **164G1A0560**, **S. Gowri Sankar** bearing Roll Number **169L1A0577** in partial fulfilment of the requirements for the award of the degree of **Bachelor of Technology** in **Computer Science & Engineering** during the academic year 2020-2021.

Guide

Dr. G. K. Venkata Narasimha Reddy Ph.D.
Professor & HOD

Head of the Department

Dr. G. K. Venkata Narasimha Reddy Ph.D.
Professor & HOD

Date:

Place: Ananthapuramu

EXTERNAL EXAMINER

ACKNOWLEDGEMENT

The satisfaction and euphoria that accompany the successful completion of any task would be incomplete without the mention of people who made it possible, whose constant guidance and encouragement crowned our efforts with success. It is a pleasant aspect that we have now the opportunity to express my gratitude for all of them.

It is with immense pleasure that we would like to express my indebted gratitude to my Guide **Dr. G.K. Venkata Narasimha Reddy, P.h.D., Professor & Head of the Department, Computer Science & Engineering**, who has guided me a lot and encouraged me in every step of the project work. We thank him for the stimulating guidance, constant encouragement and constructive criticism which have made possible to bring out this project work.

We express our deep-felt gratitude to **Dr.P.Chitralingappa,M.Tech,Ph.D, Associate Professor & Mrs.M.Sowmya,M.Tech, Assistant Professor**, project coordinators valuable guidance and unstinting encouragement enable us to accomplish our project successfully in time.

We are very much thankful to **Dr. G. K. Venkata Narasimha Reddy, Ph.D., Professor & Head of the Department, Computer Science & Engineering**, for his kind support and for providing necessary facilities to carry out the work.

We wish to convey my special thanks to **Dr. G. Bala Krishna, Ph.D., Principal of Srinivasa Ramanujan Institute of Technology** for giving the required information in doing my project work. Not to forget, we thank all other faculty and non-teaching staff, and my friends who had directly or indirectly helped and supported me in completing my project in time.

We also express our sincere thanks to the Management for providing excellent facilities.

Finally, we wish to convey our gratitude to our family who fostered all the requirements and facilities that we need.

Project Associates

DECLARATION

We, R. Sreenith bearing reg no: 174g1a0591, B. Nithesh Kumar Reddy bearing reg no: 164g1a0560, S. Gowri Sankar bearing reg no: 169l1a0577, students of SRINIVASA RAMANUJAN INSTITUTE OF TECHNOLOGY, Rotarypuram, hereby declare that the dissertation entitled “AUCTION DAPP FOR FUNDRAISING” embodies the report of our project work carried out by us during IV Year Bachelor of Technology under the guidance of Dr. G.K. Venkata Narasimha Reddy, P.hD., Department of CSE and this work has been submitted for the partial fulfillment of the requirements for the award of Bachelor of Technology degree.

The results embodied in this project report have not been submitted to any other Universities of Institute for the award of Degree.

R.Sreenith

Reg no: 174G1A0591

B. Nithesh Kumar Reddy

Reg no: 164G1A0560

S. Gowri Sankar

Reg no: 169L1A0577

Contents

List of Figures	viii
List of Screens	ix
List of Abbreviations	x
Abstract	xi
Chapter 1 Introduction	1
1.1 Objective	1
1.2 Penny Social Functionality	2
1.3 Blockchain Technology	3
1.3.1 Key Elements of Blockchain	4
1.3.2 Benefits of Blockchain	5
1.3.3 Structure of Blockchain	7
1.4 Types of Blockchain	8
1.4.1 Public Blockchain	8
1.4.2 Private Blockchain	8
1.4.3 Consortium Blockchain	9
1.4.4 Hybrid Blockchain	9
1.5 Ethereum Blockchain	9
1.6 Features of Blockchain	10
1.6.1 Ether	10
1.6.2 Smart Contracts	10
1.6.3 Ethereum Virtual Machine (EVM)	11
1.6.4 Proof of Work (POW)	11
1.6.5 Proof of Stake	12
1.6.6 GAS	13

Chapter 2	Literature Survey	13
2.1	Existing System	13
2.2	Proposed System	14
Chapter 3	Requirements	16
3.1	System Requirements	16
3.2	Hardware Requirements	16
3.3	Software Requirements	16
3.3.1	Installing Metamask	16
3.3.2	Remix-IDE	20
3.3.3	Solidity	21
3.3.4	Etherscan	21
Chapter 4	Design	22
4.1	UML Introduction	22
4.1.1	Usage of UML in Present	22
4.2	English Auction	23
4.3	Sequence Diagram	24
4.4	Class Diagram	25
Chapter 5	Implementation	26
5.1	creating a metamask Ethereum wallet	26
5.1.1	Using Rinkeby Test Network	27
5.1.2	Address of Metamask	28
5.2	Remix	28
5.2.1	Smart Contract in Remix- IDE	29
5.2.2	Compiling Smart Contract	29
5.2.3	Deploying Smart Contract	30
5.3	Deploying Smart Contract into Dapp Plug-in	31
5.3.1	ABI CODE	31
5.3.2	Dapp Plug-in	32
5.4	Etherscan Blockchain Explorer	33
5.4.1	Transaction History of Dapp	34
5.4.2	Transaction Overview	34

Chapter 6:	Testing	36
6.1	Test Directory	37
6.2	Solidity Unit-Testing Plug-in	37
Chapter 7:	Results and Discussions	40
conclusion		41
References		42

List of Figures

Fig. No	Description	Page.No
1.1	Elements of Blockchain	3
1.2	Structure of Blockchain	7
4.1	Work Model of English Auction	23
4.2	Sequence Diagram	24
4.3	Class Diagram	25

List of Screens

Fig. No	Description	Page. No
3.1	Fetching Metamask from Browser	17
3.2	Metamask	17
3.3	Chrome extension for Metamask	18
3.4	Adding Metamask extension	18
3.5	Creating Password for MEW	19
3.6	Setting Backup Phrase	20
5.1	Metamask Ethereum Wallet	26
5.2	Setting Rinkeby Network	27
5.3	Address of Metamask	28
5.4	Smart Contract in Remix-IDE	29
5.5	Compiling Smart Contract	30
5.6	Deploying Smart Contract in Remix-IDE	31
5.7	ABI Code	32
5.8	Dapp Plug-in	33
5.9	Transaction History	34
5.10	Transaction Overview	35
6.1	Solidity Dapp Plug-in	36
6.2	Test Directory	37
6.3	Deploying Smart Contract in Remix-IDE8	37
6.4	Running all Test cases	38

List of Abbreviations

MEW	Metamask Ethereum Wallet
LOT	Ledger of Transactions
POW	Proof of Work
POS	Proof of Stake
EVM	Ethereum Virtual Machine
ICO	Initial Coin Offering
SC	Smart Contract
DAPP	Decentralized Application
DLT	Distributed Ledger Technology

ABSTRACT

Ethereum is a permissionless, non-hierarchical network of computers (nodes) which build and come to consensus on an ever growing series of blocks or batches of transactions, known as Blockchain. It is a community-run technology powering the cryptocurrency, ether (ETH) and thousands of decentralized applications. Ethereum is a general-purpose blockchain that is more suited to describing business logic, through advanced scripts, also known as smart contracts. The Ethereum virtual machine and smart contracts are key elements of Ethereum, and constitute its main attraction. In Ethereum, smart contracts represent a piece of code written in a high-level language like (Solidity, LLL, Viper) and stored as bytecode in the blockchain, in order to run reliably in a stack-based virtual machine, in each node, once invoked.

An Auction is a typical example, but it is complex enough to provide a Dapp. It demonstrates the trustless nature of the blockchain, in which we can manage funds automatically and securely, without requiring legal recourse or relying on a trusted third party. The smart contract is composed of the address of Chairperson, Bid, Winners and Sheet of tickets or tokens. In short, our auction Dapp will be a web application that enables users to join auctions using ether.

CHAPTER 1

INTRODUCTION

A Decentralized application which is also called as Dapp refers to an application executed in a decentralized network. The Ledger of transactions known as LOT present in blockchain are permanent and can't be erased or lost that can happen in the centralized data. The LOT acts as a proof of validity. The smart contracts are like the terms and conditions of policy which are obeyed by users using them. With the implementation of smart contracts, blockchain has potential to be implemented in all areas. An auction place for peer to peer trading is created. The motivation behind this is to create a place where seller and consumer can deal directly without involving middleman. This helps in reducing the prices as no taxes are involved and increase performance as it has low latency and high throughput. Auctions are made in the smart contracts. Every transaction done on blockchain is stored with unique hash value in public ledger or LOT. The hash value once created can't be erased by any user except for the original author who even can only update a transaction but can't delete it, thus having its own validity of product.

1.1 Objective:

The objective of this project is to build an application where consumer and buyer no need to pay hefty prices for middleman and can also prove their authenticity of ownership without the requirement of liquid documents and bonds. The authenticity of ownership can be easily tampered if they are documents or bonds, which have become very common these days. If the owner lost documents through by any means like fire accidents, bugs, robbery, children etc, he can lose his ownership with the documents, which can be prohibited by using blockchain. It has public ledger where every transaction is saved in hash code which can't be tampered as it requires validity of proof, Even the owner can update the transaction but cannot delete it from the ledger. This ledger is stored in all the systems that are connected to the Blockchain and so even the owner lost it, it is still present in the other systems. Thus, storing it permanently in the blockchain.

The user can stay unknown to other user that is his address (unique hash key for every user) is stored in any transaction done by him. The only thing we can get his hash address thus cannot trace the user maintain privacy.

1.2 Penny Social Functionality:

Here we chose the problem of Chinese auction or Penny social. It is a conventional approach used for fundraising for a cause. Penny social is a fundraising event that combines elements of raffles and silent auctions.

The organizers collect items to be auctioned off for raising funds. Before the auction, the items for auctions are received and arranged each with a bowl to place the bid. A chairperson is a special person among the organizers. She/he heads the effort and is the only person who can determine the winner by random drawing at the end of the auction.

A set of bidders buy sheets of tickets with their money. The bidder's sheet has a stub that identifies the bidder's number, and tokens bought. The bidders examine the items to bid, place the one or more tickets in the bowl in front of the items they desire to bid for until all the tickets are used.

After the auction period ends the chairperson, collects the bowls, randomly selects a ticket from each item's bowl to determine the winning bidder for that item. The item is transferred to the winning bidder. Total money collected is the fund raised by the penny social auction.

1.3 Blockchain Technology:

Blockchain is a peer-to-peer, distributed ledger that is cryptographically secure, append-only, immutable (extremely hard to change), and updateable only via consensus or agreement among peers.

Block is among the primary components of the blockchain. Each block comprises a set of transactions. The blocks were chained together by storing a unique hash value of the preceding block in the existing block. This link blocks together like a chain. The hash function is used to validate the data integrity of the content of each block. The hash

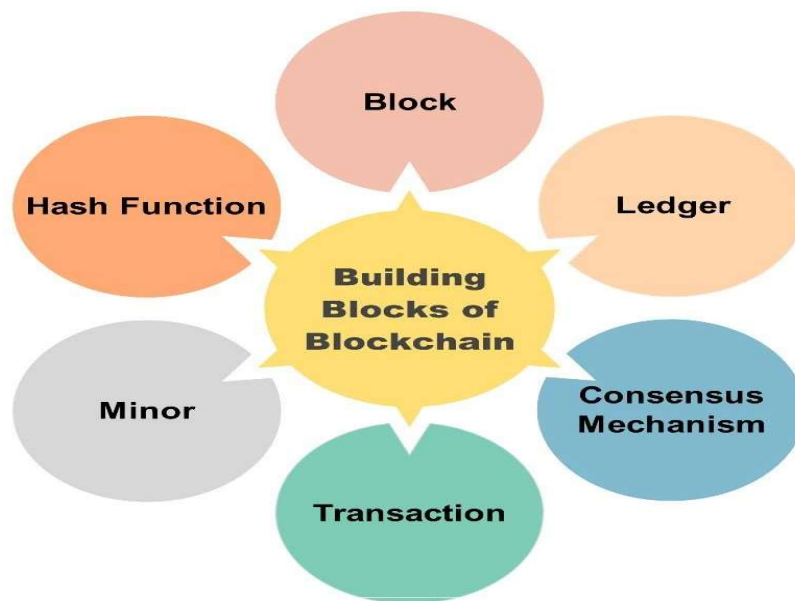


Fig.1.1. Elements of Blockchain

function is a mathematical problem that minors need to crack to find a block. The reason to use the hash function is that it is collision-free in which it is very hard to create two identical hashes for two different digital data. So, assigning a hash value for each block can serve as a way to identify the block and also validate its contents.

A transaction is the smallest unit of process or operation in which a set of transactions are combined and stored in a block. A certain transaction cannot be added to the block unless the majority of the participating nodes in the blockchain network record their consent. The size of a transaction is important for minors as small transactions require less power and are easier to validate. Minors are computers/agents that attempt to solve a complex mathematical problem (typically, a form of hash functions) to explore a new block. Discovering a new block is started by broadcasting new transactions to all nodes, and then each node combines a set of transactions into a block and operates to discover the block's proof-of-work. If a node discovers a block, then the block will be broadcasted to all the nodes to be verified.

1.3.1 Key Elements of Blockchain:

Peer-to-peer:

The term peer-to-peer means that there is no central controller in the network, and all participants talk to each other directly. This property allows for cash transactions to be exchanged directly among peers without third-party involvement.

Distributed ledger Technology:

All network participants have access to the distributed ledger and its immutable record of transactions. With this shared ledger, transactions are recorded only once, eliminating the duplication of effort that's typical of traditional business networks.

Cryptographically Secure:

In a blockchain, the ledger is cryptographically secure, which means that cryptography has been used to provide security services that make this ledger secure against tampering and misuse. These services include non-repudiation, data integrity, and data origin authentication.

Immutable records:

No participant can change or tamper with a transaction after it's been recorded to the shared ledger. If a transaction record includes an error, a new transaction must be added to reverse the error, and both transactions are then visible.

Smart Contract:

To speed transactions, a set of rules — called a smart contract — is stored on the blockchain and executed automatically. A smart contract is a piece of code stored on the blockchain that enables the automation of complex business logic.

1.3.2 Benefits of Blockchain:**Decentralization:**

This is a core concept and benefit of the blockchain. There is no need for a trusted third party or intermediary to validate transactions; instead, a consensus mechanism is used to agree on the validity of transactions.

Transparency and Trust:

Because blockchains are shared and everyone can see what is on the blockchain, this allows the system to be transparent. As a result, trust is established. This is more relevant in scenarios such as the disbursement of funds or benefits where personal discretion about selecting beneficiaries needs to be restricted.

Immutability:

Once the data has been written to the blockchain, it is extremely difficult to change it back. It is not genuinely immutable, but because changing data is so challenging and nearly impossible; this is seen as a benefit to maintaining an immutable ledger of transactions.

High availability:

As the system is based on thousands of nodes in a peer-to-peernetwork, and the data is replicated and updated on every node, the system becomes highly available. Even if some nodes leave the network or become inaccessible, the network as a whole continues to work, thus making it highly available. This redundancy results in high availability.

Highly secure:

All transactions on the blockchain are cryptographically secured and thus provide network integrity.

Faster Dealings:

Blockchain can play a vital role by enabling the quick settlement of trades. Blockchain does not require a lengthy process of verification, reconciliation, and clearance because a single version of agreed-upon data is already available on a shared ledger between financial organizations.

Cost-Saving:

As no trusted third party or clearinghouse is required in the blockchain model, this can massively eliminate overhead costs in the form of the fees which are paid to parties.

1.3.3 Structure of Blockchain:

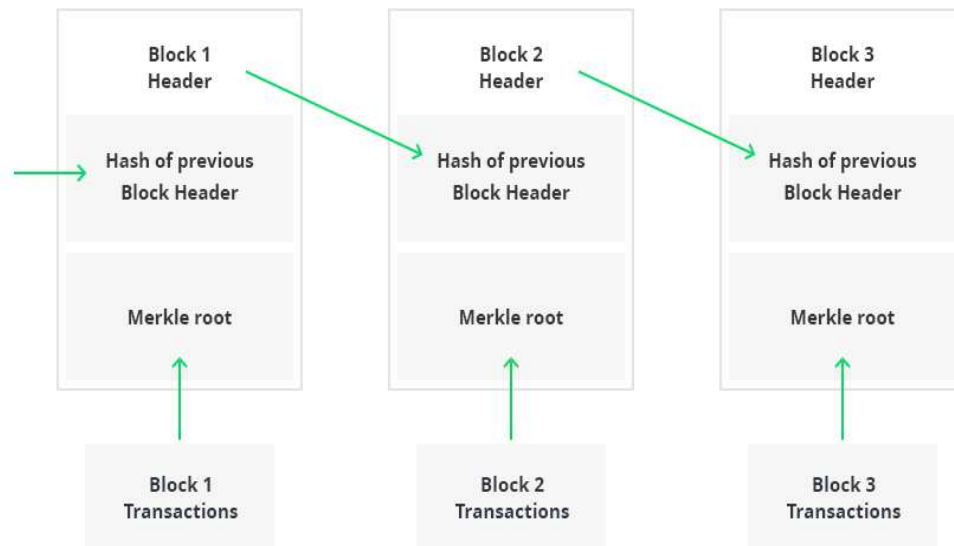


Fig.1.2. Structure of Blockchain

As Blockchain is a Decentralized, Digitized, public and shared ledger of information resistant to tampering. It is a singly linked chain of blocks of verified, signed transactions which are replicated globally on millions of nodes. So forth it is known as a “Distributed Ledger Technology”

In the distributed network of blockchain architecture, each participant within the network maintains, approves, and updates new entries. The system is controlled not only by separate individuals, but by everyone within the blockchain network. Each member ensures that all records and procedures are in order, which results in data validity and security. Thus, parties that do not necessarily trust each other are able to reach a common consensus.

The structure of blockchain technology is represented by a list of blocks with transactions in a particular order.

1.4 Types of Blockchain:

There are primarily two types of blockchains; Private and Public blockchain.

However, there are several variations too, like Consortium and Hybrid blockchains. Every blockchain consists of a cluster of nodes functioning on a peer-to-peer (P2P) network system. Every node in a network has a copy of the shared ledger which gets updated timely. Each node can verify transactions, initiate or receive transactions and create blocks.

1.4.1 Public Blockchain:

A public blockchain is a non-restrictive, permission-less distributed ledger system. Anyone who has access to the internet can sign in on a blockchain platform to become an authorized node and be a part of the blockchain network. A node or user which is a part of the public blockchain is authorized to access current and past records, verify transactions or do proof-of-work for an incoming block, and do mining. The most basic use of public blockchains is for mining and exchanging cryptocurrencies. Thus, the most common public blockchains are Bitcoin and Litecoin blockchains. Public blockchains are mostly secure if the users strictly follow security rules and methods.

1.4.2 Private Blockchain:

A private blockchain is a restrictive or permission blockchain operative only in a closed network. Private Blockchains are usually used within an organization or enterprises where only selected members are participants of a blockchain network. The level of security, authorizations, permissions, accessibility is in the hands of the controlling organization. Thus, private blockchains are similar in use as a public blockchain but have a small and restrictive network. Private Blockchain networks are deployed for voting, supply chain management, digital identity, asset ownership, etc.

1.4.3 Consortium Blockchain:

A consortium blockchain is a semi-decentralized type where more than one organization manages a blockchain network. This is contrary to what we saw in a private

blockchain, which is managed by only a single organization. More than one organization can act as a node in this type of blockchain and exchange information or do mining. Consortium blockchains are typically used by banks, government organizations, etc.

1.4.4 Hybrid Blockchain:

A hybrid blockchain is a combination of the private and public blockchain. It uses the features of both types of blockchains that is one can have a private permission-based system as well as a public permission-less system. With such a hybrid network, users can control who gets access to which data stored in the blockchain. Only a selected section of data or records from the blockchain can be allowed to go public keeping the rest as confidential in the private network. The hybrid system of blockchain is flexible so that users can easily join a private blockchain with multiple public blockchains.

1.5 Ethereum Blockchain:

Ethereum is a technology that's home to digital money, global payments, and applications. The community has built a booming digital economy, bold new ways for creators to earn online, and so much more. It's open to everyone, wherever you are in the world – all you need is the internet. Ethereum provides a software platform where we can develop Decentralized Applications. It uses ether (crypto currency) in the transactions. It is mainly used to describe business logic.

Ethereum is a permissionless, non-hierarchical network of computers (nodes) which build and come to consensus on an ever-growing series of "blocks", It comes under the public blockchain domain.

1.6 Features of Ethereum Blockchain

Ethereum has various set of features and some of the important features of Ethereum Blockchain are discussed below:

1.6.1 Ether:

Ether (ETH) is Ethereum's cryptocurrency. It is the fuel that runs the network. It is used to pay for the computational resources and the transaction fees for any transaction executed on the Ethereum network. Like Bitcoins, ether is a peer-to-peer currency. Apart from being used to pay for transactions, ether is also used to buy gas, which is used to pay for the computation of any transaction made on the Ethereum network.

Also, if you want to deploy a contract on Ethereum, you will need gas, and you would have to pay for that gas in ether. So gas is the execution fee paid by a user for running a transaction in Ethereum. Ether can be utilized for building decentralized applications, building smart contracts, and making regular peer-to-peer payments.

1.6.2 Smart Contracts:

Smart Contracts are revolutionizing the way how traditional contracts worked, which is why you need to know about them in this Ethereum tutorial. A smart contract is a simple computer program that facilitates the exchange of any valuable asset between two parties. It could be money, shares, property, or any other digital asset that you want to exchange. Anyone on the Ethereum network can create these contracts. The contract consists primarily of the terms and conditions mutually agreed on between the parties (peers).

The primary feature of a smart contract is that once it is executed, it cannot be altered, and any transaction done on top of a smart contract is registered permanently—it is immutable. So even if you modify the smart contract in the future, the transactions correlated with the original contract will not get altered; you cannot edit them.

The verification process for the smart contracts is carried out by anonymous parties of the network without the need for a centralized authority, and that's what makes any smart contract execution on Ethereum a decentralized execution.

The transfer of any asset or currency is done in a transparent and trustworthy manner, and the identities of the two entities are secure on the Ethereum network.

Once the transaction is successfully done, the accounts of the sender and receiver are updated accordingly, and in this way, it generates trust between the parties.

1.6.3 Ethereum Virtual Machine (EVM):

EVM is designed to operate as a runtime environment for compiling and deploying Ethereum-based smart contracts. EVM is the engine that understands the language of smart contracts, which are written in the Solidity language for Ethereum. EVM is operated in a sandbox environment—basically, you can deploy your stand-alone environment, which can act as a testing and development environment, and you can test your smart contract “n” number of times, verify it, and then once you are satisfied with the performance and the functionality of the smart contract, you can deploy it on the Ethereum main network.

Any programming language in the smart contract is compiled into the bytecode, which the EVM understands. This bytecode can be read and executed using the EVM. One of the most popular languages for writing a smart contract in Solidity. Once you write your smart contract in Solidity, that contract gets converted into the bytecode and gets deployed on the EVM. And thereby EVM guarantees security from cyber attacks.

1.6.4 Proof of Work (POW):

Every node in the Ethereum network has: The entire history of all the transactions, it has the history of the smart contract, which is the address at which the smart contract is deployed, along with the transactions associated with the smart contract the handle to the current state of the smart contract

The goal of the miners on the Ethereum network is to validate the blocks. For each block of a transaction, miners use their computational power and resources to get the appropriate hash value by varying the nonce. The miners will vary the nonce and pass it through a hashing algorithm—in Ethereum, it is the Ethash algorithm.

This produces a hash value that should be less than the predefined target as per the proof-of-work consensus. If the hash value generated is less than the target value, then the block is considered to be verified, and the miner gets rewarded.

When the proof of work is solved, the result is broadcast and shared with all the other nodes to update their ledger. If other nodes accept the hashed block as valid, then the block gets added to the Ethereum main blockchain, and as a result, the miner receives a reward, which as of today stands at three ethers. Plus the miner gets the transaction fees that have been generated for verifying the block. All the transactions that are aggregated in the block the cumulative transaction fees associated with all the transactions are also given as a reward to the miner.

1.6.5 Proof of stake:

In Ethereum, a process called proof of stake is also under development. It is an alternative to proof of work and is meant to be a solution to minimize the use of expensive resources spent on mining using proof of work. In proof of stake, the miner—who is the validator—can validate the transactions based on the number of crypto coins he or she holds before actually starting the mining. So based on the accumulation of crypto coins the miner has beforehand, he or she has a higher probability of mining the block. However, proof of stake is not widely used as of now compared to proof of work.

1.6.6 GAS:

We need gas to run applications on the Ethereum network. To perform any transaction within the Ethereum network, a user has to make a payment—shell out ethers—to get a transaction done, and the intermediary monetary value is called gas. On the Ethereum network, gas is a unit that measures the computational power required to run a smart contract or a transaction. So if you have to do a transaction that updates the blockchain, you would have to shell out gas, and that gas costs ethers.

For every transaction, there is gas and its correlated gas price. The amount of gas

required to execute a transaction multiplied by the gas price equals the transaction fees.

“Gas limit” refers to the amount of gas used for the computation and the amount of ether a user is required to pay for the gas.

CHAPTER 2

LITERATURE SURVEY

2.1 Existing System:

In Penny social activities, there is more reliance on organizers by Institution and having trust in organizers is the issue of concern. This aspect may lead to single point of failure. Privacy and security issues are also a concern in such case. It is more prone to hacks and data leaks.

There are higher transaction fees and audit trails are always a question mark.

Imran Bashir, in his book Mastering Blockchain: Deeper insights into decentralization discussed that Dapps are software programs that can run on their respective blockchains, use an existing established blockchain, or use only the protocols of an existing blockchains. Data and records of operations of the application are must be cryptographically secured and stored on a public, decentralized blockchain to avoid any central points of failure. [1]

In White paper, “A next-generation smart contract and decentralized application platform”, Ethereum community showcases how Ethereum blockchain with a built-in Turing-complete programming language, allows anyone to write smart contracts and decentralized applications where they can create their own arbitrary rules for ownership, transaction formats and state transition functions. [5]

In the paper, “Ethereum: a secure decentralized generalized transaction ledger, Byzantium version,” elaborates on Establishment of consensus by a Dapp can be achieved using consensus algorithms such as POW and PoS. Only PoW has been found to be incredibly resistant to 51% attacks, as is evident from Blockchain. Furthermore a Dapp can distribute tokens (coins) via mining, fundraising and development. [7]

In one cause penny social they discusses on Penny socials are also very easy to adapt with additional fundraising ideas and themes. For example, a penny social can be its own standalone event plus, you have complete flexibility over what items you'll offer and how you'll price the raffle tickets. [3]

2.2 Proposed System:

Our project the proposed system is the Institution can keep track of all the activities on its own without relying on third party (organizers) using Dapp which is powered by Blockchain Technology.

Here, we use the ethereum blockchain which mainly describes the business logic between two recipients. It provides enhanced transparency between Institution and Bidders which increases the resilience of overall network. It majorly prevents the fraud and data theft and improves the robustness and integrity. It also provides powerful audit trails and lower transaction fees.

CHAPTER 3

REQUIREMENTS

3.1 System Requirements

The System Requirements like software and hardware requirements are mentioned below

3.2 Hardware Requirements

Any Contemporary PC.

3.3 Software Requirements

The software requirements required for developing this project are as follows:

- Operating System : Windows 10
- Tools : Metamask, Remix-IDE, Etherscan
- Languages Used : Solidity

3.3.1 Installing Metamask

Step 1:

1. Metamask is a free and secure browser extension that allows web applications to read and interact with the Ethereum blockchain.

To create a new wallet with Metamask you need to install the extension first. You can install Metamask for Chrome, Firefox, Brave and Opera browsers.

Open <https://metamask.io> or search for “Metamask extension” using your favorite search engine.

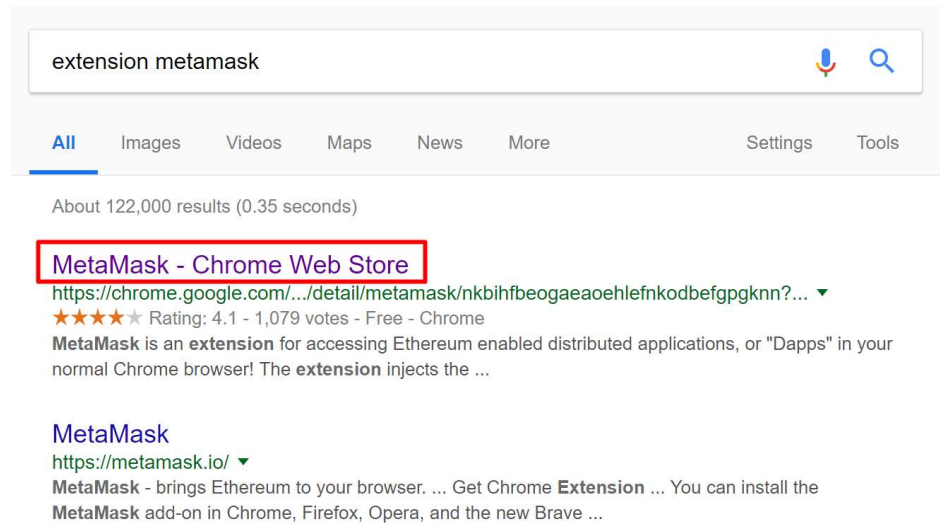


Fig.3.1. Fetching metamask from browser

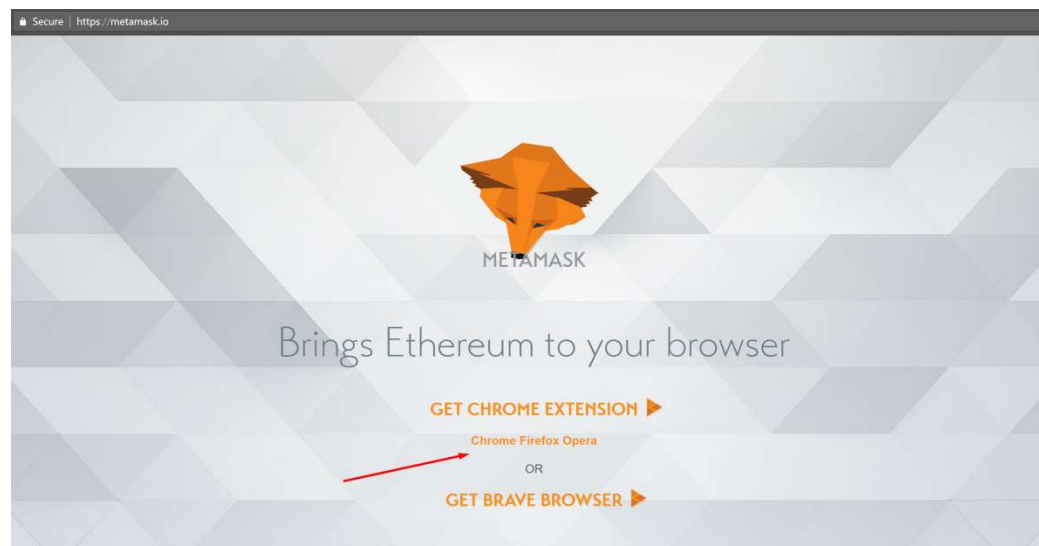


Fig.3.2. Metamask

2. Click Chrome to install MetaMask as a Google Chrome extension.
3. Click Add to Chrome.
4. Click Add Extension.

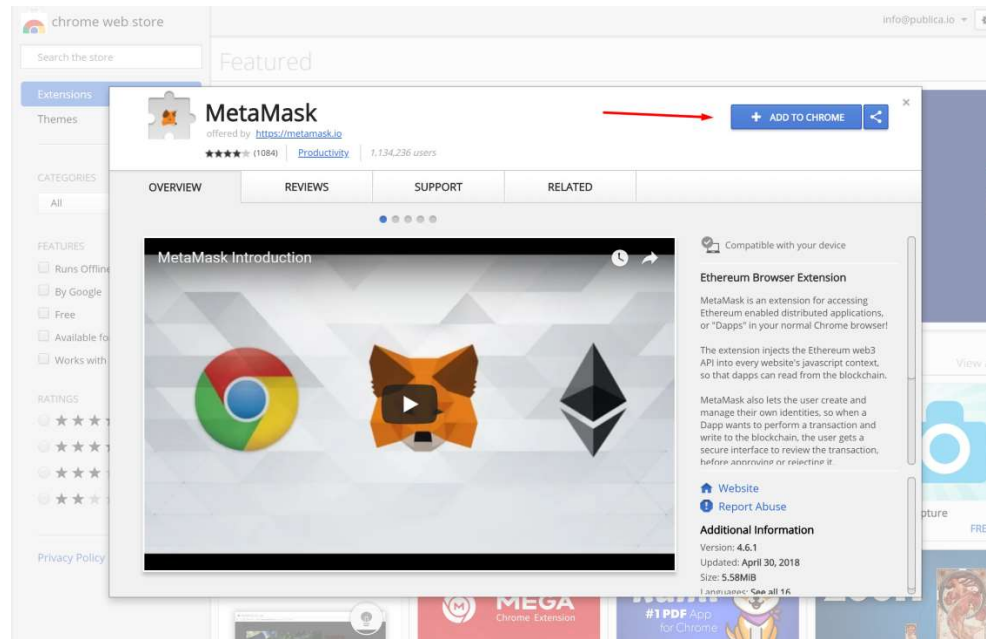


Fig.3.3. Chrome extension for metamask

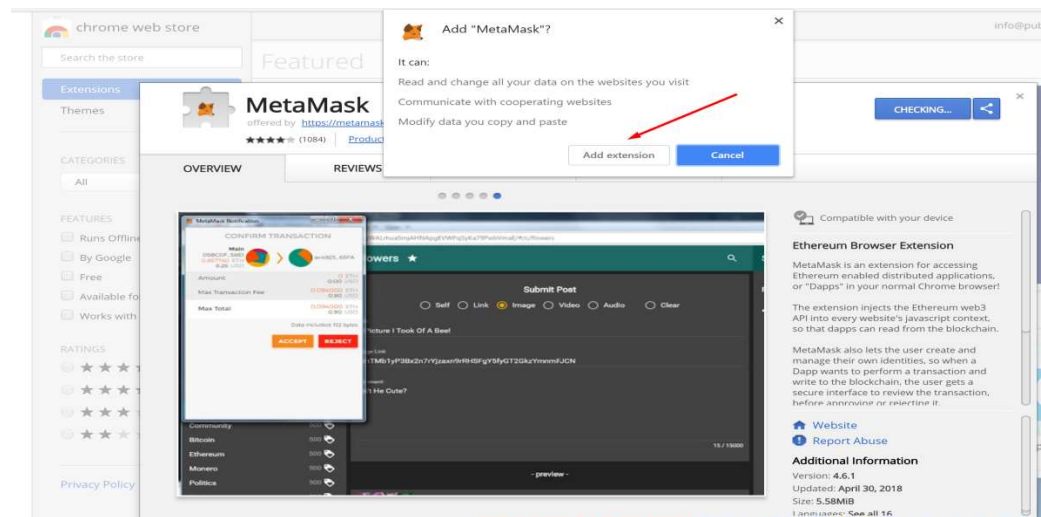


Fig.3.4. Adding metamask extension

With that, you have successfully created the extension for Metamask Wallet.

Step 2: Create an account in Metamask Wallet

The next step is to create an account.

1. Click on the MetaMask icon in the upper right corner to open the extension.
2. To install the latest version of MetaMask, click Try it now.
3. Click Continue.
4. You will be asked to create a new password. Create a strong password and click Create.

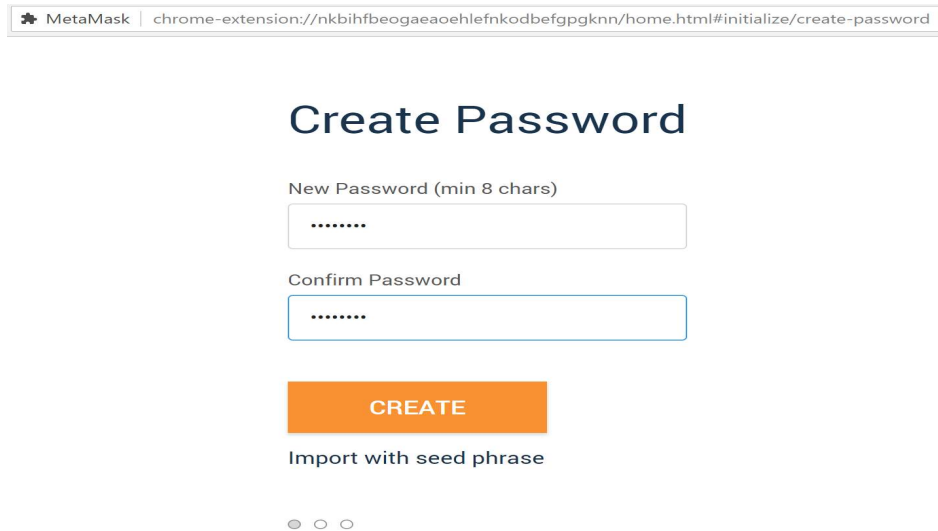


Fig.3.5. Creating Password for MEW

5. Proceed by clicking next, and then accept Terms of Use.
6. Click Reveal secret words.
7. You will see a 12 words seed phrase. Save seed words as a file or copy them to a safe place and click next.

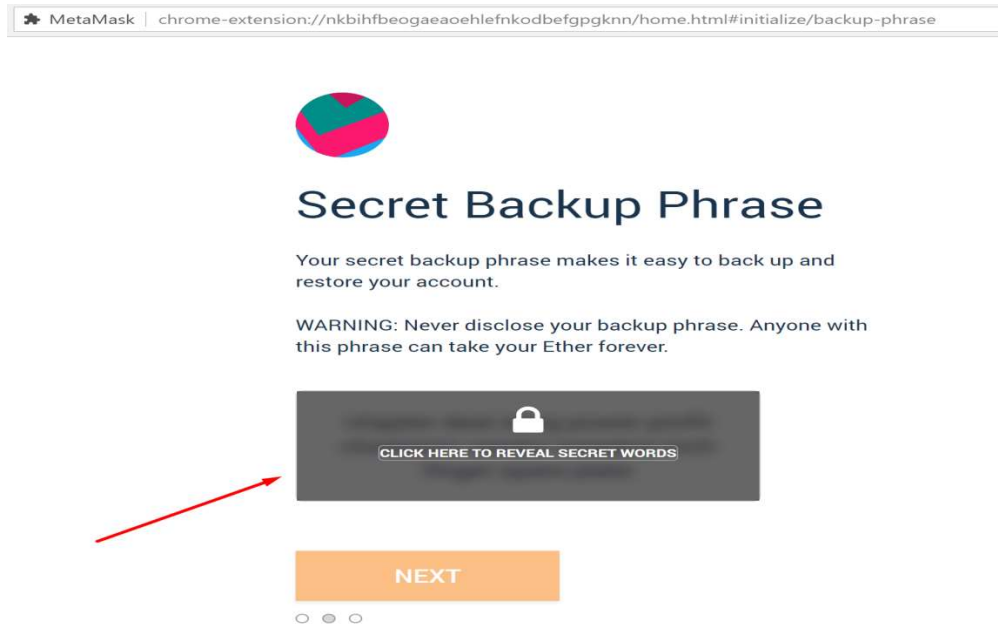


Fig.3.6. Setting Backup phrase

8. Reveal secret words and copy your secret backup phrase to a safe place.
9. Verify your secret phrase by selecting the previously generated phrase. When done, click Confirm.
10. By “solving this puzzle” you are confirming that you know your secret phrase.
11. On solving the puzzle you will get a message Quoting Congratulations! You have successfully created your MetaMask account. A new Ethereum wallet address was automatically generated for you!

3.3.2 Remix –IDE:

Remix IDE is an open source web and desktop application. It fosters a fast development cycle and has a rich set of plugins with intuitive GUIs. Remix is used for the entire journey of contract development as well as being a playground for learning and teaching Ethereum. Remix IDE is part of the Remix Project which is a platform for development tools that use a plugin architecture. It encompasses sub-projects including Remix Plugin Engine, Remix Libs, and of course Remix-IDE. Remix IDE is a powerful

open source tool that helps you write Solidity contracts straight from the browser. It is written in JavaScript and supports both usage in the browser, in the browser but run locally and in a desktop version. Remix IDE has modules for testing, debugging and deploying of smart contracts and much more.

3.3.3 Solidity:

Solidity is a contract-oriented, high-level language for implementing smart contracts. It was influenced by C++, Python and JavaScript and is designed to target the Ethereum Virtual Machine (EVM).

Solidity is statically typed, supports inheritance, libraries and complex user-defined types among other features.

3.3.4 Etherscan:

Etherscan is the leading Blockchain Explorer, Search, API and analytics Platform for Ethereum, a decentralized smart contracts platform. Built and launched in 2015 it is one of the earliest and longest running independent project built around Ethereum and its community with the mission of providing equitable access to blockchain data.

CHAPTER 4

DESIGN

4.1 UML Introduction:

The unified modeling language allows the software engineer to express an analysis model using the modeling notation that is governed by a set of syntactic, semantic and pragmatic rules. A UML system is represented using five different views that describe the system from distinctly different perspective.

UML is specifically constructed through two different domains, they are:

- UML Analysis modeling, this focuses on the user model and structural model views of the systems.
- UML Design modeling, which focuses on the behavioral modeling, implementation modeling and environmental model views.

4.1.1 Usage of UML in Project

As the strategic value of software increases for many companies, the industry looks for techniques to automate the production of software and to improve quality and reduce cost and time to the market. These techniques include component technology, visual programming, patterns and frameworks. Additionally, the development for the World Wide Web, while making some things simpler, has exacerbated these architectural problems. The UML was designed to respond to these needs. Simply, systems design refers to the process of defining the architecture, components, modules, interfaces and data for a system to satisfy specified requirements which can be done easily through UML diagrams.

4.2 English Auction:

This auction is for the individual products that have huge demand in the market. Interested bidders start bidding for a product at the base price and keep on increasing until there is only one bidder left.

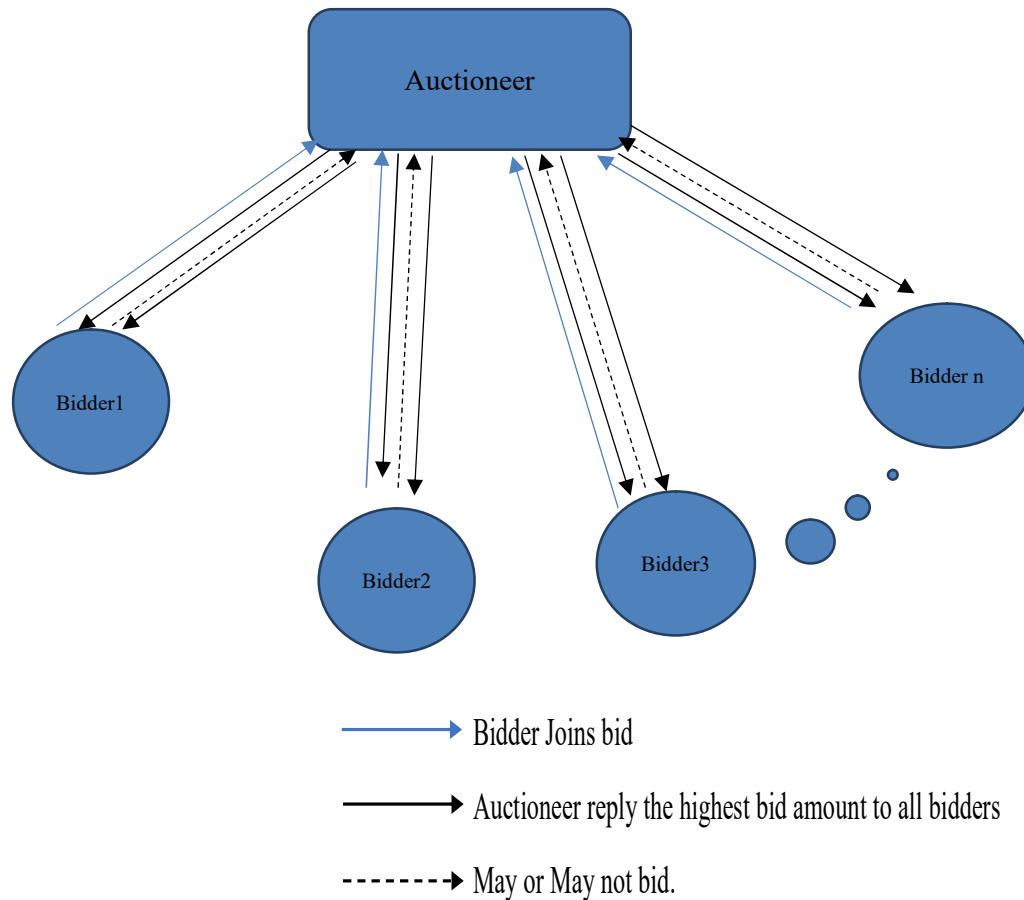


Fig.4.1. Work Model of English Auction

First the bidders who want to join the auctioning can join auction at given time. The auctioneer starts the auction with the base price. Interested bidders can bid the amount starting with the price which is a sum of base price and bid price mentioned at start of auction. The bidder's bids will all reach the auctioneer and auctioneer will keep updating the bid amount to all the bidders who joined the auction. This helps the bidders

where to start with price. When all bids reach the auctioneer he selects the highest bid and update other bidders with the highest amount. The interested bidders after receiving the highest bid can still bid more if they are interested.

4.3 Sequence Diagram:

A sequence diagram shows object interactions arranged in time sequence. It depicts the objects involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario. Sequence diagrams are typically associated with use case realizations in the logical view of the system under development. Sequence diagrams are sometimes called event diagrams or event scenarios.

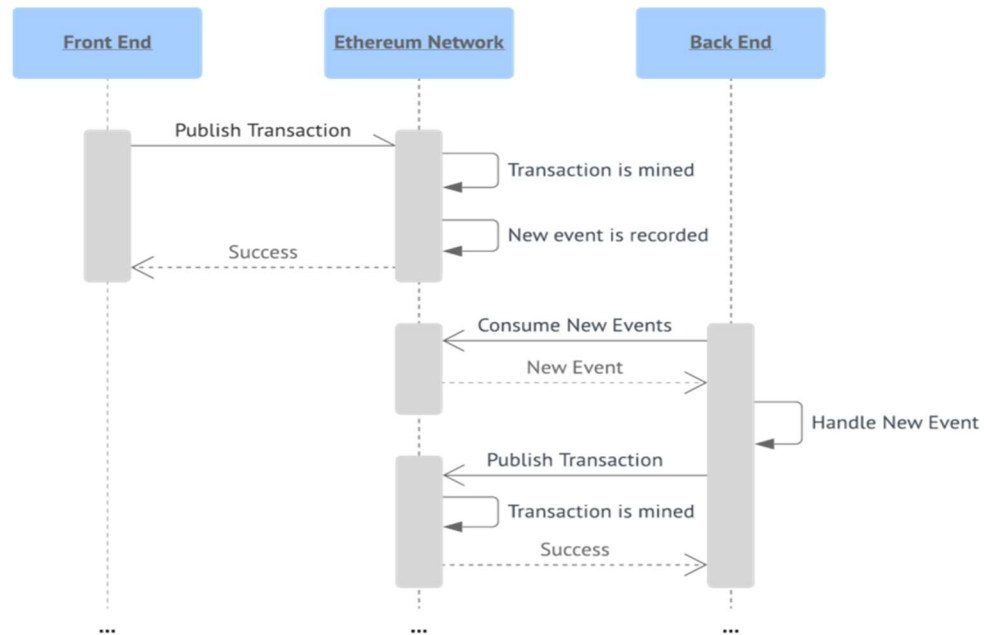


Fig.4.2. Sequence diagram

The above Sequence Diagram shows the example flow of server's reaction to the user's action in de-centralized network.

4.4 Class Diagram:

Class diagram is a static diagram. It represents the static view of an application.

Class diagram is not only used for visualizing, describing, and documenting different aspects of a system but also for constructing executable code of the software application.

Class diagram describes the attributes and operations of a class and also the constraints imposed on the system. The class diagrams are widely used in the modeling of object oriented systems because they are the only UML diagrams, which can be mapped directly with object-oriented languages.

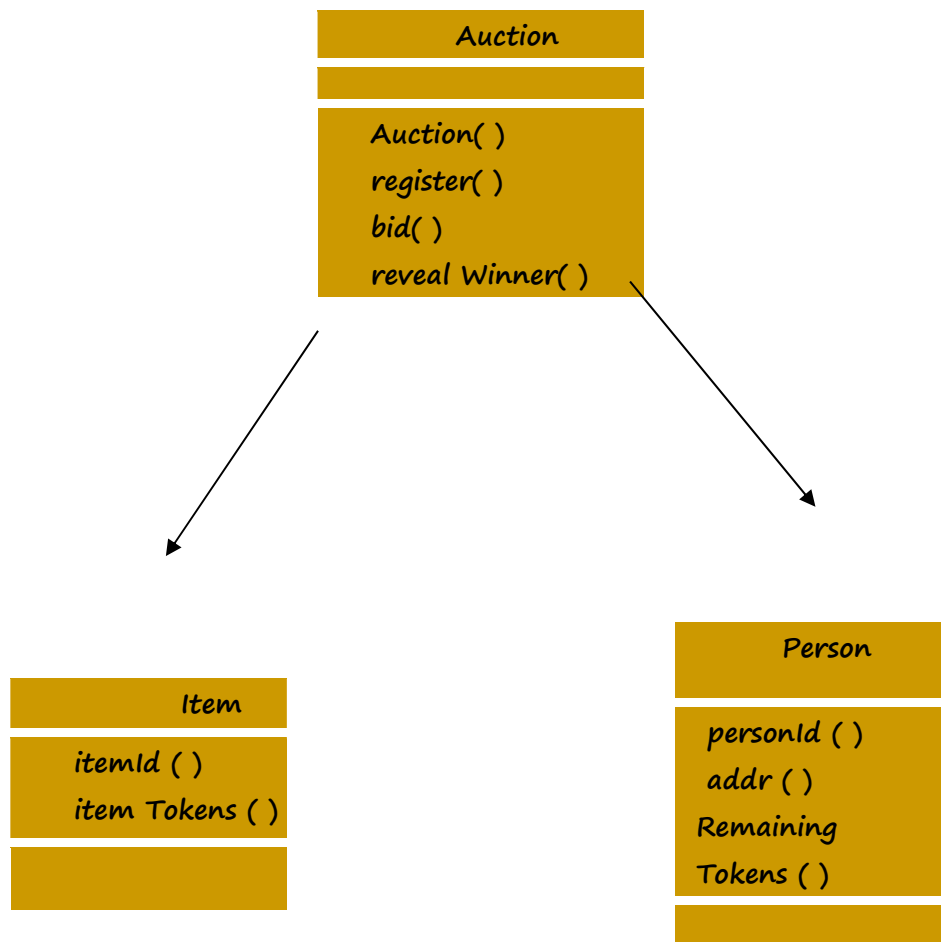


Fig.4.3. Class diagram

CHAPTER 5

IMPLEMENTATION

In order to implement our de-centralized application which uses Ethereum blockchain, there are some steps to be followed which involve as follows:

1. Creating a Metamask Ethereum Wallet (MEW)
2. Writing Smart Contract in Remix-IDE
3. Compiling and Enabling the smart contract
4. Deploying the smart contract into Dapp-plugin-in
5. Exploring Etherscan Blockchain Explorer

5.1 Creating a Metamask Ethereum Wallet (MEW)

After installing the metamask, we need to give the password to our account and we can send ethers from one account to another using the metamask wallet.

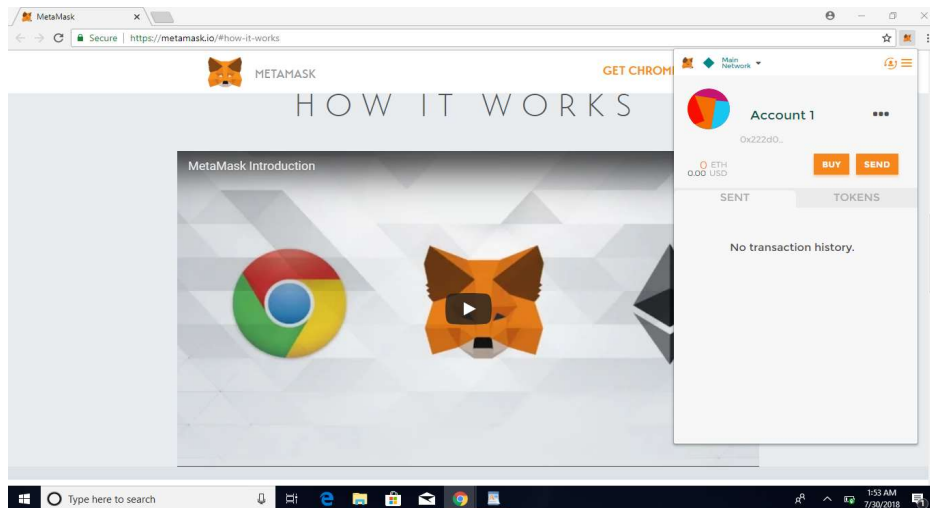


Fig.5.1. Metamask Ethereum Wallet

5.1.1 Using Rinkeby Test Network:

You are now in the Ethereum Main network. To start experimenting with Metamask, you can switch to one of the testnet networks by clicking “Main Network” in the left hand corner of the wallet pop up screen, and selecting one of the testnets such as Ropsten Test Network or Kovan Test Network.

If we want to connect to ethereum network and perform transactions, we should pay the gas fee, paying gas fee is essential because it acts as transaction fee to Miners who are involved in validating ethereum blocks.

So, we need to buy ethers in order to perform transactions, as it is not possible to get real ethers we use Valueless ethers which comes in the Test network such as Rinkeby network.

From these Test Networks, you can safely “buy” and “send” test Ether from a faucet and begin experimenting with the blockchain. Click around within the test network and start understanding how the Ethereum blockchain works.

We get the working ether from the Rinkeby faucets i.e., Rinkeby Authenticated account.

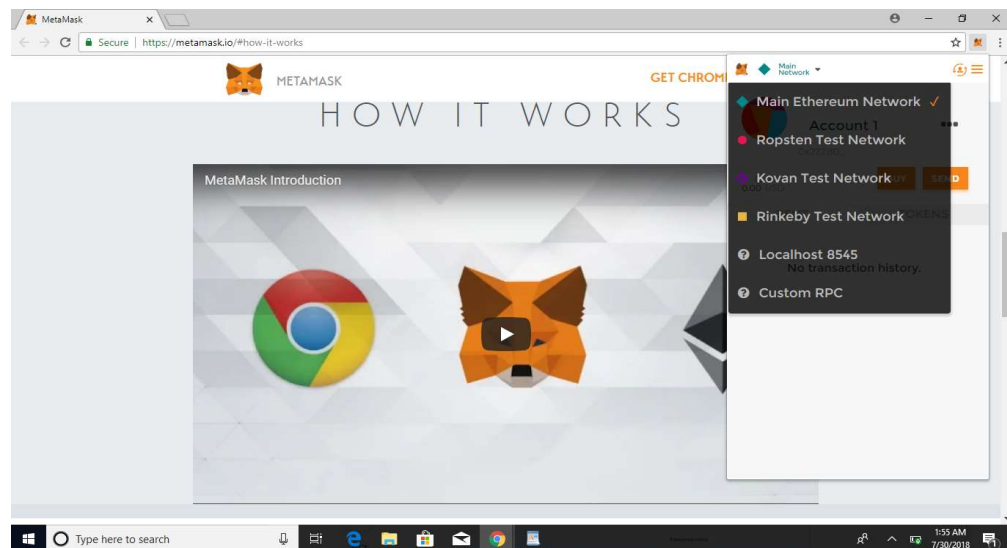


Fig.5.2. Setting Rinkeby Network

5.1.2 Address of Metamask:

Metamask allows you to run Ethereum Dapps right in your browser without running a full ethereum node. Every Metamask wallet has public address.

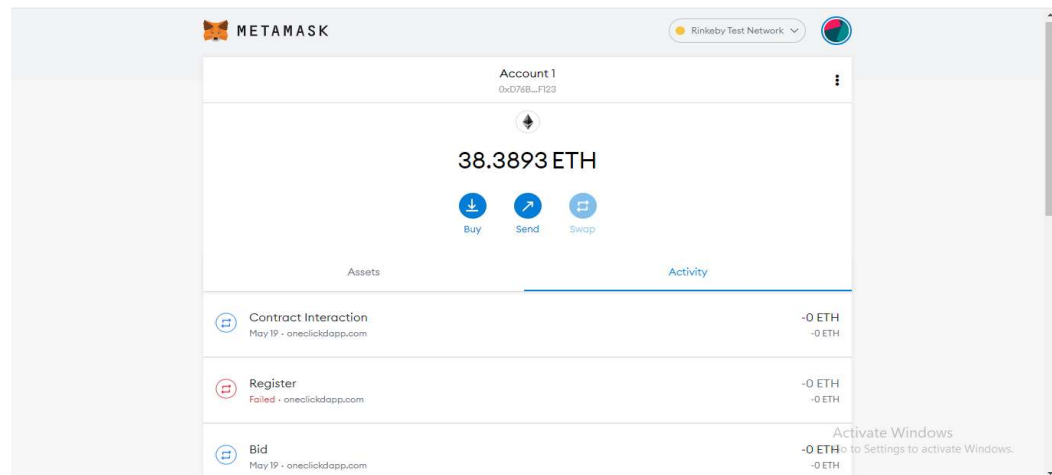


Fig.5.3. Address of metamask

5.2 Remix:

Remix IDE is an open source web and desktop application. It fosters a fast development cycle and has a rich set of plug-ins with intuitive GUIs. Remix is used for the entire journey of contract development as well as being a playground for learning and teaching Ethereum.

Remix IDE is part of the Remix Project which is a platform for development tools that use a plug-in architecture. It encompasses sub-projects including Remix Plug-in Engine, Remix Libs, and of course Remix-IDE.

Remix IDE is a powerful open source tool that helps you write Solidity contracts straight from the browser.

It has modules for testing, debugging and deploying of smart contracts and much more. Remix-IDE is available at remix.ethereum.org and more information can be found

in these docs.

5.2.1 Smart contract in Remix-IDE:

In Work space we need to create a file named Auction. We should start writing the business logic for smart contract.

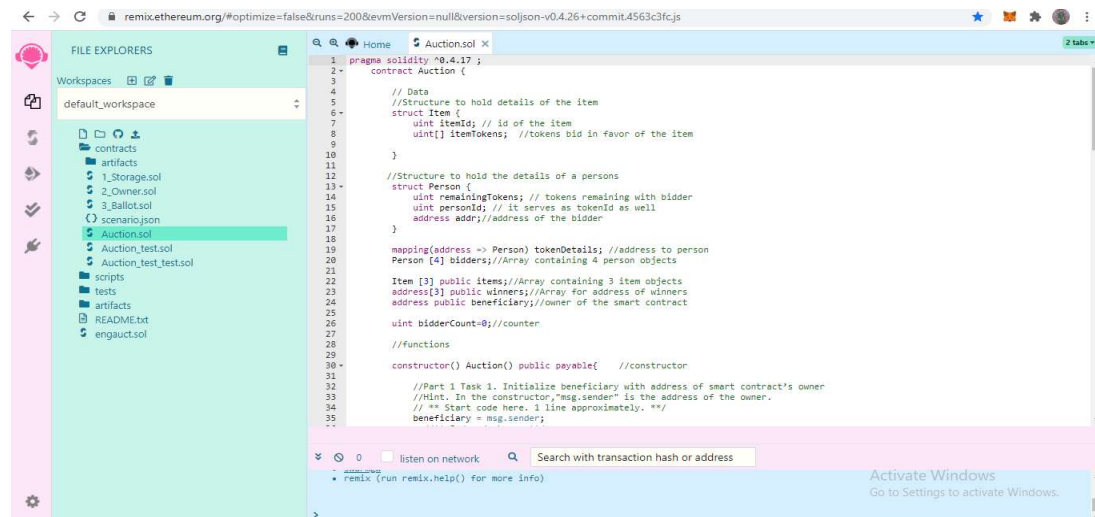


Fig.5.4. Smart Contract in Remix-IDE

5.2.2 Compiling the smart contract:

After the completion of writing whole smart contract, we need to make sure that the correct version of compiler is being used and we should start compiling the smart contract. There are various versions of compilers involved in remix-ide which creates flexibility in writing the code in solidity.

In the EVM version column we should use the Compiler Default and the start compiling the smart contract. After the successful compilation of contract we can see the tick symbol on the plug-in manager.

If there are any errors involve in the code we can see them at the end to the plug-in manager.

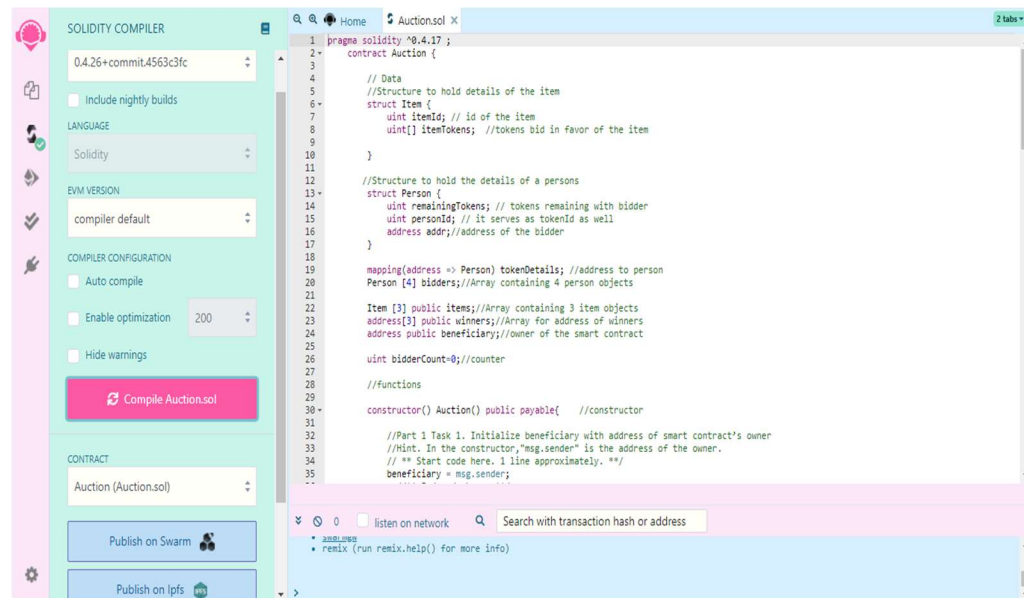


Fig.5.5. Compiling Smart Contract

5.2.3 Deploying smart contract:

Here, we need to select the environment that we are running the smart contract, our smart contract is running on the Injected Web3 environment i.e., on the rinkeby test network. After that we should check the account address and the Gas Limit in plug-in manager.

We will see that our smart contract is named as the Auction- contracts/Auction.sol on clicking the Deploy which is in red colour in the plug-in manager the smart contract gets deployed.

After deploying the smart contract, we can see the deployed smart contract address. Besides, we can check the block number and transaction address.

All the hash values included in the chain can be seen here. We can even the execution of our smart contract.

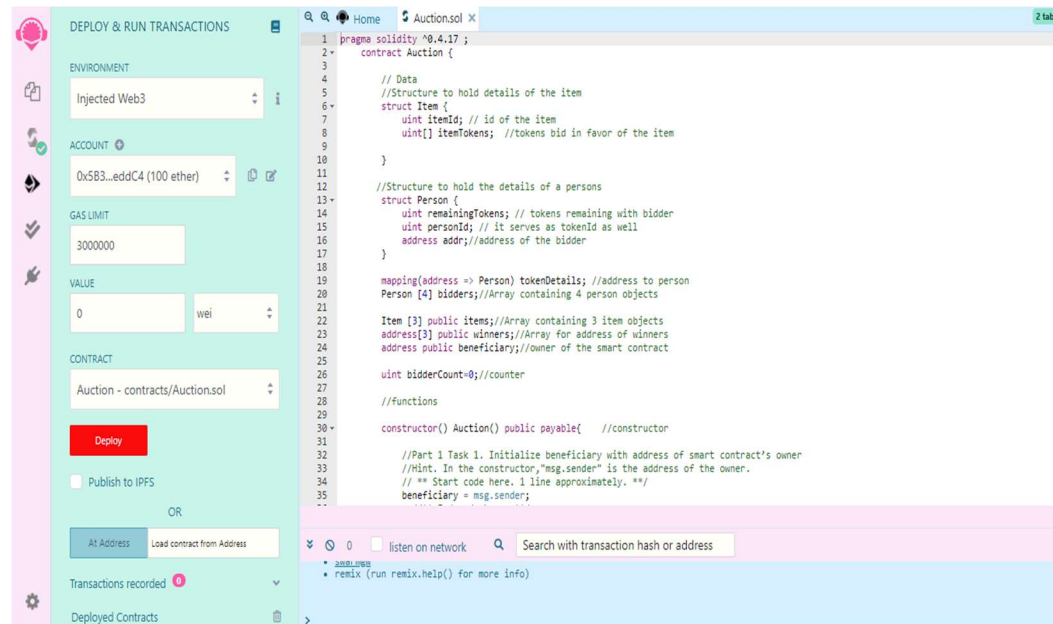


Fig.5.6. Deploying Smart Contract in Remix-IDE

5.3 Deploying smart contract into Dapp-plugin:

After the completion of compiling and running the smart contract we need to create a interface which can be done by importing the Dapp- plug-in. we use abi code in the dapp plug-in so that it can obtain the interface.

5.3.1 ABI CODE:

An Ethereum smart contract is bytecode deployed on the Ethereum blockchain. There could be several functions in a contract. An ABI is necessary so that you can specify which function in the contract to invoke, as well as get a guarantee that the function will return data in the format you are expecting. ABI stands for application binary interface.

In general, an ABI is the interface between two program modules, one of which is often at the level of machine code. The interface is the de facto method for encoding/decoding data into/out of the machine code. In Ethereum, it's basically how you can encode Solidity contract calls for the EVM and, backwards, how to read the data out

of transactions.

```
{
  "constant": false,
  "inputs": [],
  "name": "register",
  "outputs": [],
  "payable": true,
  "stateMutability": "payable",
  "type": "function"
},
{
  "constant": true,
  "inputs": [],
  "name": "beneficiary",
  "outputs": [
    {
      "name": "",
      "type": "address"
    }
  ],
  "payable": false,
  "stateMutability": "view",
  "type": "function"
},
{
  "constant": false,
  "inputs": [
    {
      "name": "_itemId",
      "type": "uint256"
    },
    {
      "name": "_count",
      "type": "uint256"
    }
  ],
  "name": "bid".
}
```

Fig.5.7. ABI Code

5.3.2 Dapp plug-in:

After handling the abi code from the ethereum virtual machine, we need to use the same abi code in order to create the interface which has all the functions involved in the smart contract and execution can be shown in the Dapp plug-in.

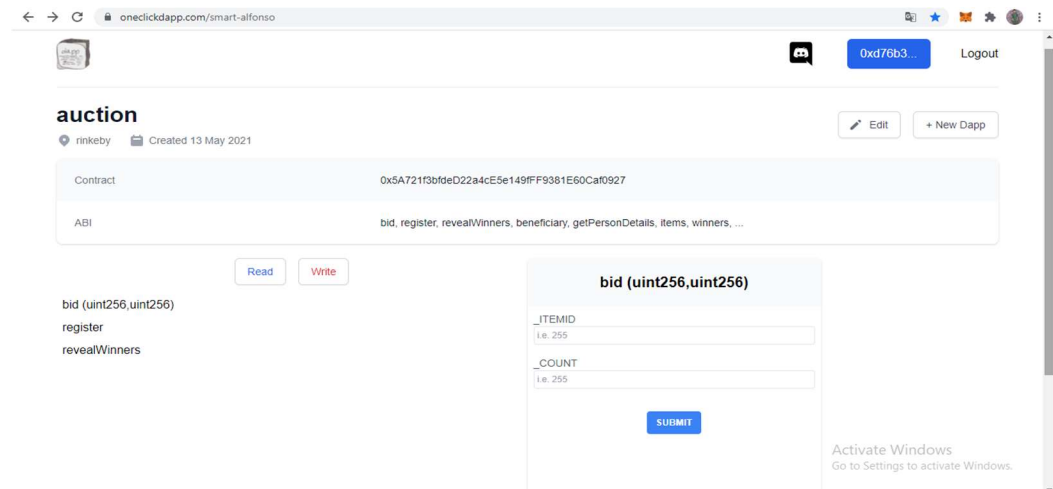


Fig.5.8. Dapp Plug-in

In the dapp it has the read and the write functions, where we can input values and check the output result of bids that have taken place in the auction. On correct execution of transaction it shows the result as “success” which means that the successful transaction has taken place.

For every transaction, the screen pops-up with the metamask wallet. In the metamask wallet we should sign the on-going transaction and pay the gas fee to the miners. On completion of the entire transaction related to bidding we can see the bidder with highest bid and confirm our output.

5.4 Etherscan Blockchain Explorer

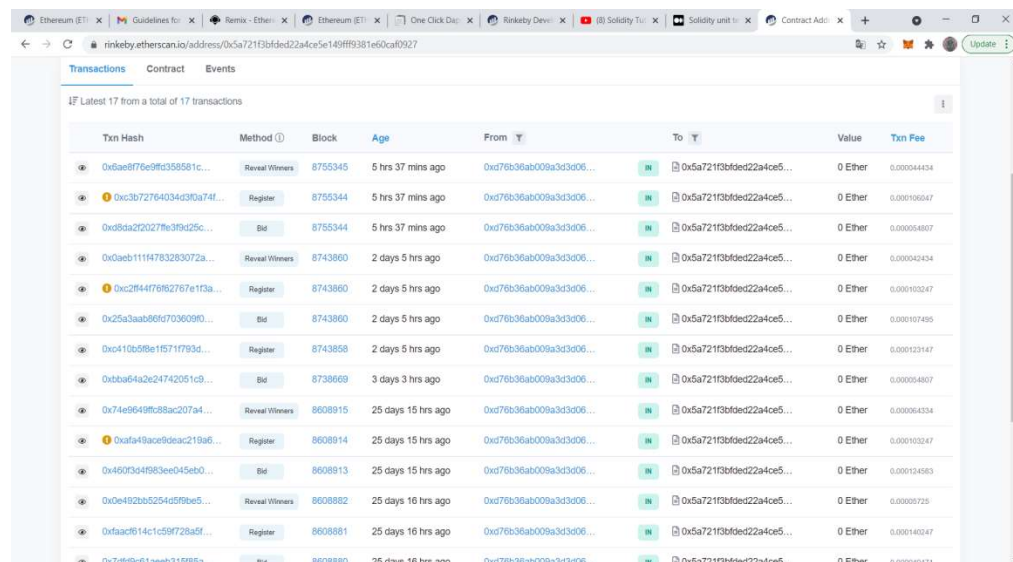
Etherscan is the leading block explorer and analytics platform for Ethereum, a decentralized smart contracts platform. A block explorer is a search engine that allows users to easily lookup, confirm, and validate transactions that have taken place on the Ethereum Blockchain.

They are a team of individuals grown and independently operated by their passions about the sorts of decentralized information and infrastructure applications that run on Ethereum. Etherscan is not funded, operated, or managed by the Ethereum Foundation but instead exists as an independent entity.

Etherscan is a useful resource for all Ethereum Network users to track transactions, check smart contracts, find out stats, and generally stay on top of what's happening in the Ethereum blockchain. What's more, it's free to use and you don't have to register to use the main features

5.4.1 Transaction History of Dapp:

All the transactions done are shown in the Etherscan Block explorer and we can see the addresses of contract and metamask wallet in the transaction history.



The screenshot shows the Etherscan interface with the 'Transactions' tab selected. It displays a list of 17 transactions for the contract address 0x5a721f3bde22a4ce5149ff9381e60ca1927. The table includes columns for Txn Hash, Method, Block, Age, From, To, Value, and Txn Fee. Transactions are categorized as 'Reveal Winners', 'Register', and 'Bid'.

Txn Hash	Method	Block	Age	From	To	Value	Txn Fee
0x8ae876e9fc358581c...	Reveal Winners	8755345	5 hrs 37 mins ago	0xd78b36ab009a3d3d06...	0x5a721f3bde22a4ce5...	0 Ether	0.000044334
0xc3b72764034d90a74f...	Register	8755344	5 hrs 37 mins ago	0xd78b36ab009a3d3d06...	0x5a721f3bde22a4ce5...	0 Ether	0.000108047
0xd5da2f20278e39d25c...	Bid	8755344	5 hrs 37 mins ago	0xd78b36ab009a3d3d06...	0x5a721f3bde22a4ce5...	0 Ether	0.000054807
0x0aeb1114783283072a...	Reveal Winners	8743860	2 days 5 hrs ago	0xd78b36ab009a3d3d06...	0x5a721f3bde22a4ce5...	0 Ether	0.000040334
0xc28447682767e1f3a...	Register	8743860	2 days 5 hrs ago	0xd78b36ab009a3d3d06...	0x5a721f3bde22a4ce5...	0 Ether	0.000103247
0x25a3aab86f703609f0...	Bid	8743860	2 days 5 hrs ago	0xd78b36ab009a3d3d06...	0x5a721f3bde22a4ce5...	0 Ether	0.000107486
0xd410b58e1f571f793d...	Register	8743858	2 days 5 hrs ago	0xd78b36ab009a3d3d06...	0x5a721f3bde22a4ce5...	0 Ether	0.000123147
0xb6a642e24742051c9...	Bid	8738669	3 days 3 hrs ago	0xd78b36ab009a3d3d06...	0x5a721f3bde22a4ce5...	0 Ether	0.000054807
0x74e9498fc88ac207a4...	Reveal Winners	8608915	25 days 15 hrs ago	0xd78b36ab009a3d3d06...	0x5a721f3bde22a4ce5...	0 Ether	0.000064334
0xa149ace9deac219a6...	Register	8608914	25 days 15 hrs ago	0xd78b36ab009a3d3d06...	0x5a721f3bde22a4ce5...	0 Ether	0.000103247
0x4603c4f583ee045eb0...	Bid	8608913	25 days 15 hrs ago	0xd78b36ab009a3d3d06...	0x5a721f3bde22a4ce5...	0 Ether	0.000134383
0x0e492ab5254d5f9be5...	Reveal Winners	8608882	25 days 16 hrs ago	0xd78b36ab009a3d3d06...	0x5a721f3bde22a4ce5...	0 Ether	0.00008725
0xfac9b14c1c59f728a5f...	Register	8608881	25 days 16 hrs ago	0xd78b36ab009a3d3d06...	0x5a721f3bde22a4ce5...	0 Ether	0.000140247
0x7df59c61aeeb31585a...	Bid	8608880	25 days 16 hrs ago	0xd78b36ab009a3d3d06...	0x5a721f3bde22a4ce5...	0 Ether	0.000040471

Fig.5.9. Transaction history

5.4.2 Transaction Overview:

In the Transaction overview, we can see the Transaction hash, Block number, Status of the transaction (whether it is successful or not) we can also see the miners address, contract address and the wallet's address.

It also includes Block Timestamp, Transaction fee which is paid to the miner in form of the ether. On multiplying the Gas Limit with the Gas Fee we can get the Transaction fee.

We can note the gas used by particular transaction and can deal with the gas limit. Nonce can be displayed in the transaction receipt. State of the transaction can also be seen the receipt.

The screenshot shows the Etherscan interface for a transaction on the Rinkaby Testnet. The transaction is successful and has been confirmed by 1 block. The gas limit is 44,434, and the gas used is 44,434 (100%).

Field	Value
Transaction Hash	0x6ae8f76e9f1d358581c4c1e58d3949dc266e0f8aad42cb423f9550ca8cb324e
Status	Success
Block	8755345 (1 Block Confirmation)
Timestamp	22 secs ago (Jun-13-2021 06:40:46 AM +UTC)
From	0xd76b35ab009a3d3d0615703148dafdc1c09f123
To	Contract 0x5a721f3bf0ed22a4ce5e148ff9381e60ca0927
Value	0 Ether (\$0.00)
Transaction Fee	0.000044434 Ether (\$0.00)
Gas Price	0.00000001 Ether (1 Gwei)
Gas Limit	44,434
Gas Used by Transaction	44,434 (100%)

Fig.5.10. Transaction Overview

CHAPTER 6

TESTING

The main objective of testing is to uncover a host of errors, systematically and with minimum effort and time. Testing is a process of executing a program with the intent of finding an error.

1. A successful test is one that uncovers an as yet undiscovered error.
2. A good test case is one that has a high probability of finding error, if it exists.

After writing the smart contract, we need to test the written solidity code. In order to do that we need to install the solidity unit testing plug-in from the plug-in manager. After activating the unit test plug-in from the manager we need to perform the required tests. Alternatively, just select Solidity environment from Remix IDE Home tab.

This will activate the solidity unit test plug-in along with the solidity compiler, Deploy and run transactions and solidity analysis plug-in.

After successful loading solidity plug-in looks like this:

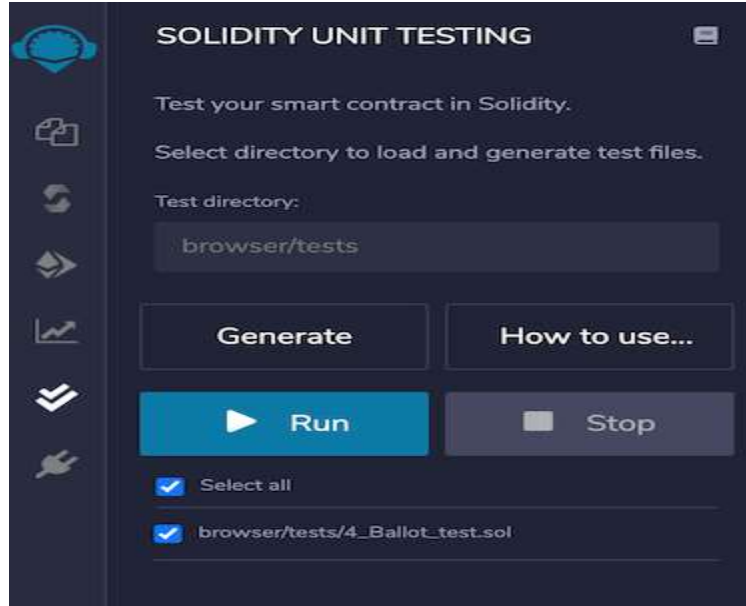


Fig.6.1. Solidity Dapp plug-in

6.1 Test Directory:

Plug-in asks you to provide a directory which will be your workspace only for this plug-in. To select directory, as soon as you add `/` to the path, it shows the possible options.

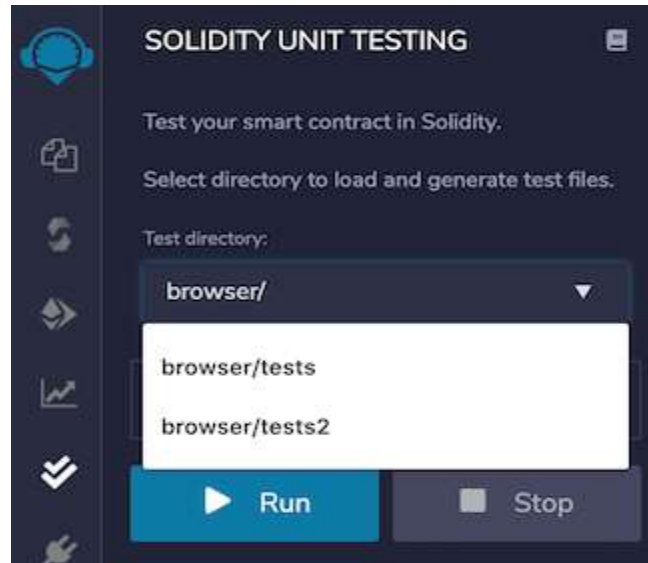


Fig.6.2. Test Directory

Once selected, this directory will be used to load test files and to store newly generated test files.

6.2 Solidity Unit Test Plug-in:

Select a solidity file which you want to test and click on the button Generate, It will generate a test file dedicated to selected file in the test directory. If no file is selected, it will still create a file with generic name as newfile_Test.sol

This file contains sufficient information to give better understanding about developing tests for a contract.

Generic file looks as:


```

pragma solidity >=0.4.22 <0.8.0;
import "remix_tests.sol"; // this import is automatically injected by Remix.
import "remix_accounts.sol";
// Import here the file to test.

// File name has to end with '_test.sol', this file can contain more than one testSuite contracts
contract testSuite {

    /// 'beforeAll' runs before all other tests
    /// More special functions are: 'beforeEach', 'beforeAll', 'afterEach' & 'afterAll'
    function beforeAll() public {
        // Here should instantiate tested contract
        Assert.equal(uint(1), uint(1), "1 should be equal to 1");
    }

    function checkSuccess() public {
        // Use 'Assert' to test the contract,
        // See documentation: https://remix-ide.readthedocs.io/en/latest/assert_library.html
        Assert.equal(uint(2), uint(2), "2 should be equal to 2");
        Assert.notEqual(uint(2), uint(3), "2 should not be equal to 3");
    }

    function checkSuccess2() public pure returns (bool) {
        // Use the return value (true or false) to test the contract
        return true;
    }

    function checkFailure() public {
        Assert.equal(uint(1), uint(2), "1 is not equal to 2");
    }

    /// Custom Transaction Context
    /// See more: https://remix-ide.readthedocs.io/en/latest/unittesting.html#customization
    /// #sender: account-1
    /// #value: 100
    function checkSenderAndValue() public payable {
        // account index varies 0-9, value is in wei
        Assert.equal(msg.sender, TestsAccounts.getAccount(1), "Invalid sender");
        Assert.equal(msg.value, 100, "Invalid value");
    }
}

```

Fig.6.3. Generic file

Once you are done with writing tests, select the file(s) and click on **Run** to execute the tests. The execution will run in a separate environment. After completing the execution of one file, a test summary will be show as below:

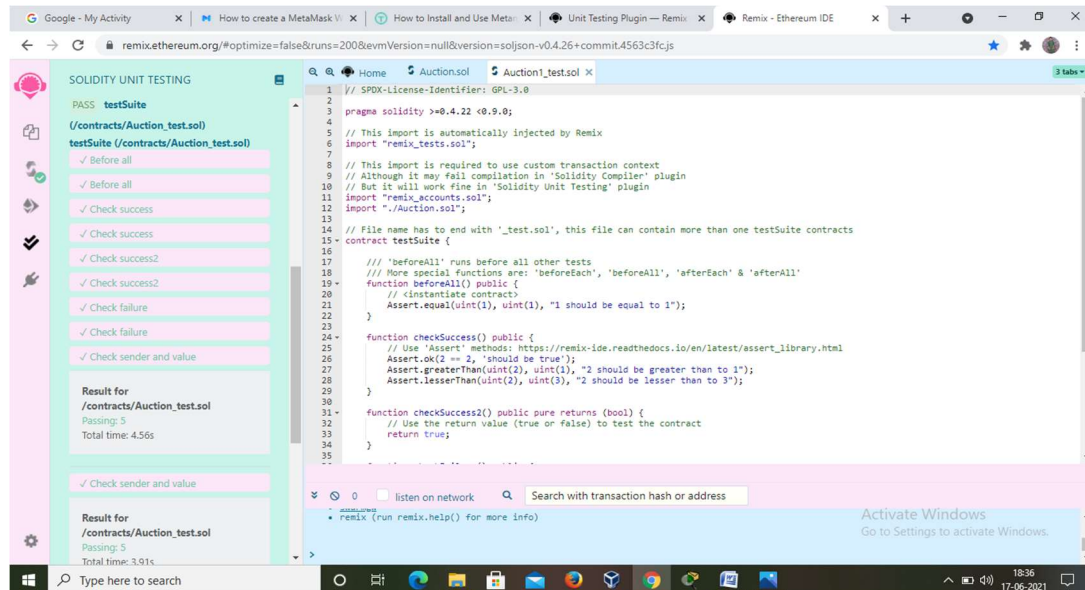


Fig.6.4. Running all test cases

With this we can perform the unit testing operation on the given code and generate the tests related to the smart contract written using the solidity language. From the above screen, test cases are generated and we can see all the cases are successfully passed. Unit test is successfully performed on the auction contract.

CHAPTER 7

RESULTS AND DISCUSSIONS

The transactions costs in blockchain are insignificant or null when compared it to costs on Real time applications. The latency rate of Bitcoin is 600 sec and its block length is 900 KB whereas the latency rate and block length of remaining cryptocurrencies are as follows:

Litecoin- Latency rate = 150 sec, Block length = 20 KB

Monero- Latency rate = 210 sec, Block length = 55 KB

Ethereum- Latency rate = 15 sec, Block length = 35 KB

When the average transaction costs for Auction in real Life and blockchain are compared, the latency for the auctions is drawn to a graph to show the difference in response time and is compared to real life latency.

Latency rate helps in maintaining efficiency and speed of transactions in block chain. The one with the less latency rate that is fast response has more efficiency. Ethereum is more formidable, with its less Latency rate. Latency rate is inversely proportional to the block length as it takes more time to reach desired transaction. But more block length makes it more secure.

CONCLUSION

In this project, the decentralised approach has been furnished and the difference in transaction costs on blockchain and real life are compared, it is a significant difference. The blockchain model is more trustworthy than any other free application for auctioning. The latency rates can be improved further if there are many systems in the blockchain thus increasing speed and efficiency. The blockchain can be our future mode of transactions, once it gets stabilised as it is more secure, unable to tamper, decentralised and easy to interact with the users directly.

REFERENCES

Book

- [1] Imran Bashir, Mastering Blockchain: Deeper insights into decentralization, cryptography, Bitcoin and popular Blockchain frameworks, 2017.
- [2] Luc Desrosiers, Nitin Gaur, and petr Novotny, Hands-on Blockchain with Hyperledger: Building Decentralized Applications with Hyperledger Fabric and composer, 2018.

World Wide Web:

- [3] One cause- penny social <https://www.onecause.com/blog/penny-social/>
- [4] Tutorial for building an ethereum Dapp with Integrated Web3 monitoring <https://www.moesif.com/blog/blockchain/ethereum/Tutorial-for-building-Ethereum-Dapp-with-Integrated-Error-Monitoring/>

Journal Article:

- [5] Ethereum Community, “A next-generation smart contract and decentralized application platform,” White Paper, available at: <https://www.github.com/ethereum/wiki/wiki/White-Paper>
- [6] V. Buterin, “Ethereum white paper: a next generation smart contract & decentralized application platform,” 2013, available at: https://www.the-blockchain.com/docs/Ethereum_white_paper-a_next_generation_smart_contract_and_decentralized_application_platform-vitalik-buterin.pdf
- [7] G. Wood, “Ethereum: a secure decentralized generalized transaction ledger, Byzantium version,” 2018, available at: <https://ethereum.github.io/yellowpaper/paper.pdf>