**SUMMER TRAINING REPORT**

**On­­­**

**“Chat Application using Blockchain Technology**

**Using Microsoft Azure Services”**

**Submitted for the requirement of**

**Project course**

BACHELOR OF ENGINEERING

**COMPUTER SCIENCE & ENGINEERING**

**Submitted By: Sahul Kr. Parida(20BCS4919)**

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

**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING**



**CHANDIGARH UNIVERSITY, GHARUAN**

**MOHALI, PUNJAB**

**July 2022**

**BONAFIDE CERTIFICATE**

Certified that this project report **“Chat Application using Blockchain Technology”** is the bonafide work of **“Sahul Kumar Parida(20BCS4919) of WM904-B”** who carried out the project work.

Submitted for the project viva-voice examination held on \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

### INTERNAL EXAMINER EXTERNAL EXAMINER

**DECLARATION**

I hereby declare that I have completed my two months internship at **Microsoft Azure** from 15 May,2022 to 15 July,2022 under the guidance of **FUTURE READY TALENT**. I hereby undertake that the work done undertaken by me is the genuine work of mine.

NAME OF STUDENT: SAHUL KUMAR PARIDA

DATE: 15/08/2022

### ACKNOWLEDGEMENT

We express my heartfelt and sincere gratitude to GOD Almighty, for his blessings for the successful completion of our project work without any hindrance.

The support and help of a few people not only enabled us to complete our work successful and also made it a worthwhile experience. I thank the management for providing all the support and facilities.

We also express my sincere thanks to my parents for their guidance and wholehearted

support.

We convey my sincere thanks to all other faculties for their help and encouragement. I thank

all my friends who have helped me during the work, with their inspiration and co-operation.

Once again I convey my gratitude to all those persons who had directly or indirectly

influenced on the work.

**ABOUT THE COMPANY**

Microsoft Azure, often referred to as Azure is a cloud computing platform operated by Microsoft for application management via Microsoft-managed data centers. It provides software as a service (SaaS), platform as a service (PaaS) and infrastructure as a service (IaaS) and supports many different programming languages, tools, and frameworks, including both Microsoft-specific and third-party software and systems. Its parent company is Microsoft Corporation. Microsoft Corporation is an American multinational technology corporation producing computer software, consumer electronics, personal computers, and related services headquartered at the Microsoft Redmond campus located in Redmond, Washington, United States. Its best-known software products are the Windows line of operating systems, the Microsoft Office suite, and the Internet Explorer and Edge web browsers. It is one of the Big Five American information technology companies, alongside Alphabet, Amazon, Apple, and Meta.

**CERIFICATE**



**TABLE OF CONTENT**

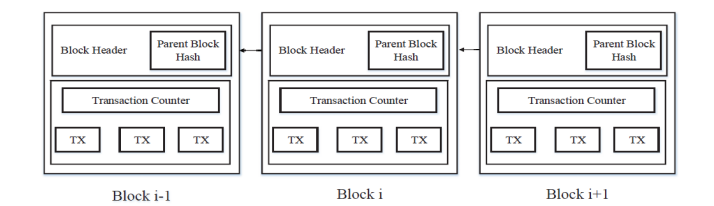
|  |  |  |
| --- | --- | --- |
| Sr no. | Topic | Page No. |
|  | ABSTRACT | 6 |
|  | GRAPHICAL ABSTRACT | 7 |
| 1 | Introduction | 8 |
| 2 | Background and History | 9 - 17 |
| 3 | Problem Definition | 18 - 21 |
| 4 | Tools of Implementation on Blockchain | 22 - 27 |
| 5 | Development and Implementation | 28 - 36 |
| 6 | Conclusion | 37 |
| 7 | References | 38 |

**ABSTRACT**

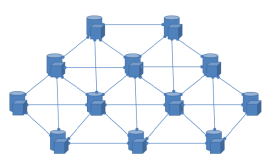
Blockchain is a way of storing data such that it will be very difficult or we can say nearly impossible to alter, hack or change the system. As the Time is passing the blockchain technology is becoming popular and blockchain based applications also called DApps will be used in this paper are becoming quick and more accepted in all places. In this paper we are going to build a chat application based on a decentralised network to remove its total dependency from centralized players. The actual message and data will be stored on the peer-to-peer network like Ethereum. We will use a blockchain and distributed hash table (DHT) to provide decentralised storage and efficient look up. which will provide what we are going to do which is a hot topic in each lecturers and trade. except for the importance of DApps, we tend to still have little or no understanding of DApps and nature. to shut the data gap, this paper presents a file for the primary powerful blockchain-based DApps study up to now, supported the in depth 995 Ethereum DApps information and twenty-nine, 846, 075 dealings logs higher than them. we tend to do the descriptive analysis of DApps preferences, summarize patterns of however DApps use good contracts to access basic blockchain, and check the problems facing the post and mistreatment DApps. supporting the findings, we propose a lot of Results for DApp users to pick out the appropriate DApps. The Dapp developers need to do more in order to get more efficient DApps and also it should also be supported by its vendors.

Keywords—Blockchain, DApps (Decentralized applications), Ethereum, Smart contract.

**GRAPHICAL ABSTRACT**

****

An example of blockchain which consists of a continuous sequence of blocks



Decentralized nodes are only connected to peers

**Chapter 1 Introduction**

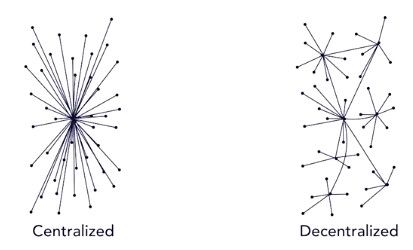
## Introduction

In 2008 cryptocurrency was invented by Satoshi Nakamoto and the currency is known as bitcoin [2], blockchain technology has been quickly developed, and has attracted a lot of attention from both academics and industry. The blockchain has been relocated is being distributed, as well as a digital public book used for recording transactions on multiple nodes for any record involved; it cannot be changed by reversal, without all modification to the following blocks. Because of its benefits of power allocation, consistency, security, and transparency, the blockchain has been one in every of the foremost promising infrastructure technologies within the next generation of online-based programs, like social services, net of Things (IoT), name systems, and security [3]. blockchain may be a variety of distributed software packages that provides procedure power for applications exploitation multiple calculation nodes. From blockchains they {are doing} not have one management however are maintained according to official compatibility models, use in blockchains they are actually for shared use, a special form of package during which the applying is created not controlled by one business. Traditionally, DApps normally see applications running on the Peer-to-Peer (P2P) network of laptops instead of one single computer. Many fashionable DApps are developed and cosmopolitan, like BitTorrent file sharing, BitMessage instant electronic communication, and Popcorn video streaming time. Blockchains give a standard release of calculation with a way of intelligent agreements, which makes it easier to progress DApp for varied application eventualities. For instance, The Ethereum blockchain provides complete Alan Mathison Turing intelligence developers' contracts to implement programs for a common purpose. As a result, with the advent of blockchains, many blockchain-based DApps are already available and are accepted in almost every place. According to a recent report, the largest market-based blockchain based on DApp, Ethereum The DApp market, has reached billions of dollars since January 2019[4]. The blockchain app based on DApps is getting popular these days, understanding of such blockchain based systems is very less. The reports published by industries have great emphasis on the normal usage of the statistics for example number of transactions and the amount of daily active users. Education research work is mainly focused on the underlying blockchain system [5] and the mechanism of smart contract done on block chain networks [6]. Some studies dig into the features and development practices of blockchain-based DApps.

# **Chapter 2 Background and History**

### Blockchain

Blockchain or distributed ledger technology, DLT, is a new technology to store and manage data across the Internet and other computing networks. It was created as a result of the introduction of the Bitcoin cryptocurrency. Today, the application of blockchain and its potential far exceed its genesis in Bitcoin. It supports not just digital money and trusted data movement and storage, but the exchange of value, an Internet of value. [3] The terminology blockchain reflects the logic of the mechanism. New transactions in a network are bundled into blocks, which are added to existing blocks - forming a chain with cryptographic signatures. These signatures security link the blocks to each other [6]. Since the cryptographic signature depends on the chain of all previous blocks, changing an existing block in the chain would invalidate all the following blocks in the eyes of the rest of the network [7]. As a result, when a new transaction is proposed no participant would accept transactions coming from a modified version of this chain: this constitutes blockchains groundbreaking fraud-detection system [8].



Centralized vs. Decentralized

### Smart Contracts

A smart contract is a code or an automated process that handles the transaction in between two parties without a third party. Contracts are very much like classes in object-oriented languages. Contracts may have state variables, functions, events, and struct types. Inside the contract we may have modifiers. Once the smart contract governing the organization is deployed on the blockchain network, it become independent of its developers and cannot be influenced by any outside entity. Its rules, financial records and transactions are controlled and maintained by the blockchain and therefore eliminate the need of a middleman.

## Technical Structure

### Cryptography

Cryptography is a Greek word, and it means secret writing. Using this technique, we are able to send messages in cryptic way, and nobody will be able to edit it. This process is called encryption. The message will be decrypted at the final destination. Cryptography involves the use of code and protocols to establish secure communications [10].

### Public and Private Keys

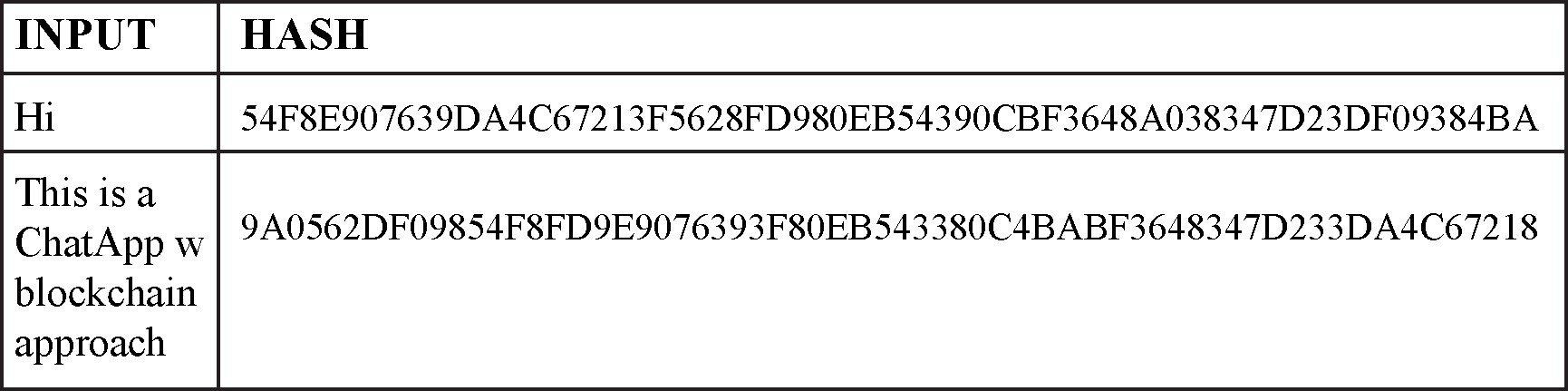
Public Key Infrastructure or PKI is one form of cryptography. It requires two keys, a public and a private key. Public key is visible to everyone. Private key is only visible by the authorized user. Since blockchain technology aims to provide a distributed and unalterable ledger of information, it has qualities considered highly suitable for the storage and management of public keys [11]. Private key is a random mixture of numbers and letters a thorough f. When a private key is generated its pair is also created which is the public key at the same time. Creating public key from private key is easy, however the opposite is extremely difficult. It is almost impossible to figure out the private key from public key. When an electronic message is sent out, it’s public key is sent out with it, and it is signed with a private key. The recipient has the public key. The authentication of the message can be verified by checking that the sender created the signature with the private key pair. The recipient opens the message with the private key. Unlocking the message without the private key is impossible.

### Nonce and Hash Function

A nonce is a number that is used once and for a specific purpose. It will not be used again. There is always a chance that data entered in the database may have the same identifier. Adding a nonce to identifiers will make it extra unique. Nonce is essential for adding blocks to the blockchain. The miner will mine the block to find. The valid nonce and the data create a specific hash.

Hash is a fingerprint of a digital data. It is data of a fixed-size. A hash function is a mathematical process that takes data of any size. Hash function will perform on it, and returns a hash. A hash algorithm turns an arbitrarily large amount of data into a fixed length hash.

The same hash will always result from the same data, but modifying the data by even one bit will completely change the hash. Like all computer data, hashes are large numbers, and are usually written as hexadecimal [4]. It is difficult to find another hash value for the same string.



Unique Fixed Size Hash Output

In the blockchain, hashes are used as identifiers for blocks, transactions, and addresses. The hashing algorithm used in the blockchain is called SHA-256. SHA, which stands for

Secure Hash Algorithm, generates an almost unique fixed-size 256-bit hash. [3]



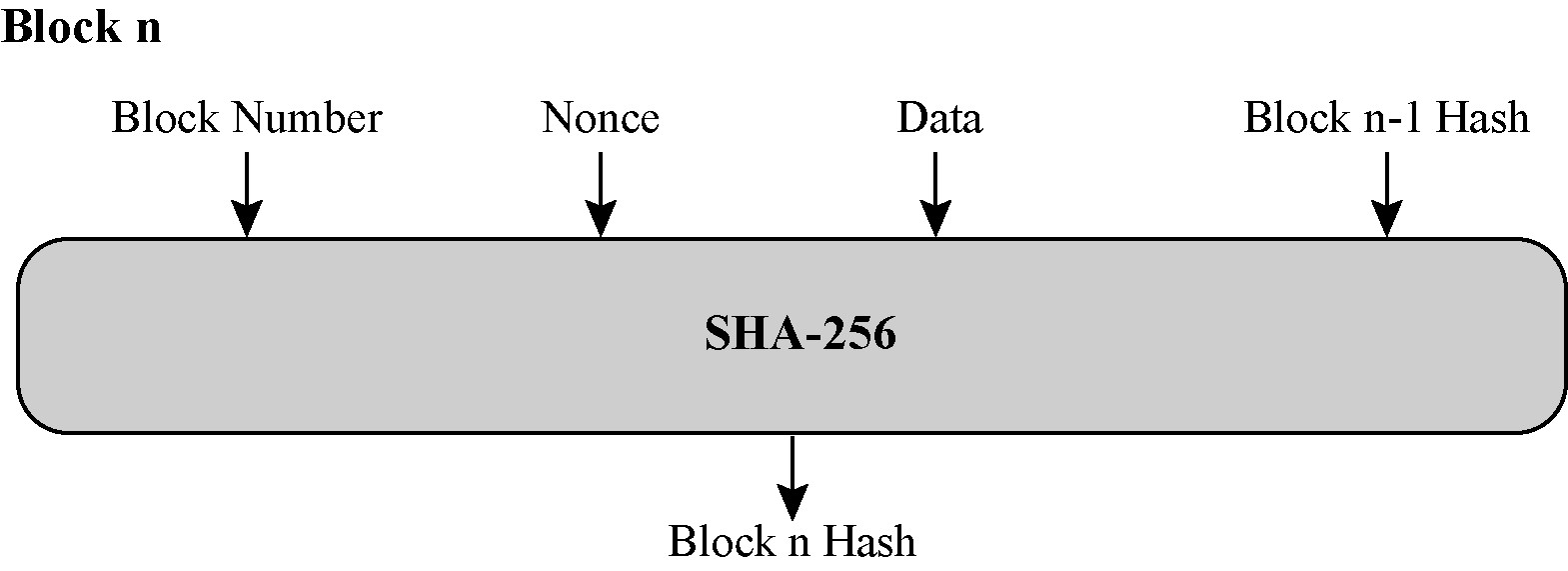
SHA256 Hash Function

### Blocks

Blocks are files that permanently record and hold transaction data. Blocks are linear sequences. Miners constantly process new blocks. These blocks are added to end of the chain of the blockchain. As more blocks are added to the blockchain, it gets harder to change, edit or remove the blocks that were added earlier and are deeper in the chain. This is how the transactions become irreversible. Each block in blockchain contains ledgers and transaction data. The block data is hashed by cryptographic hash functions.

### Block Structure

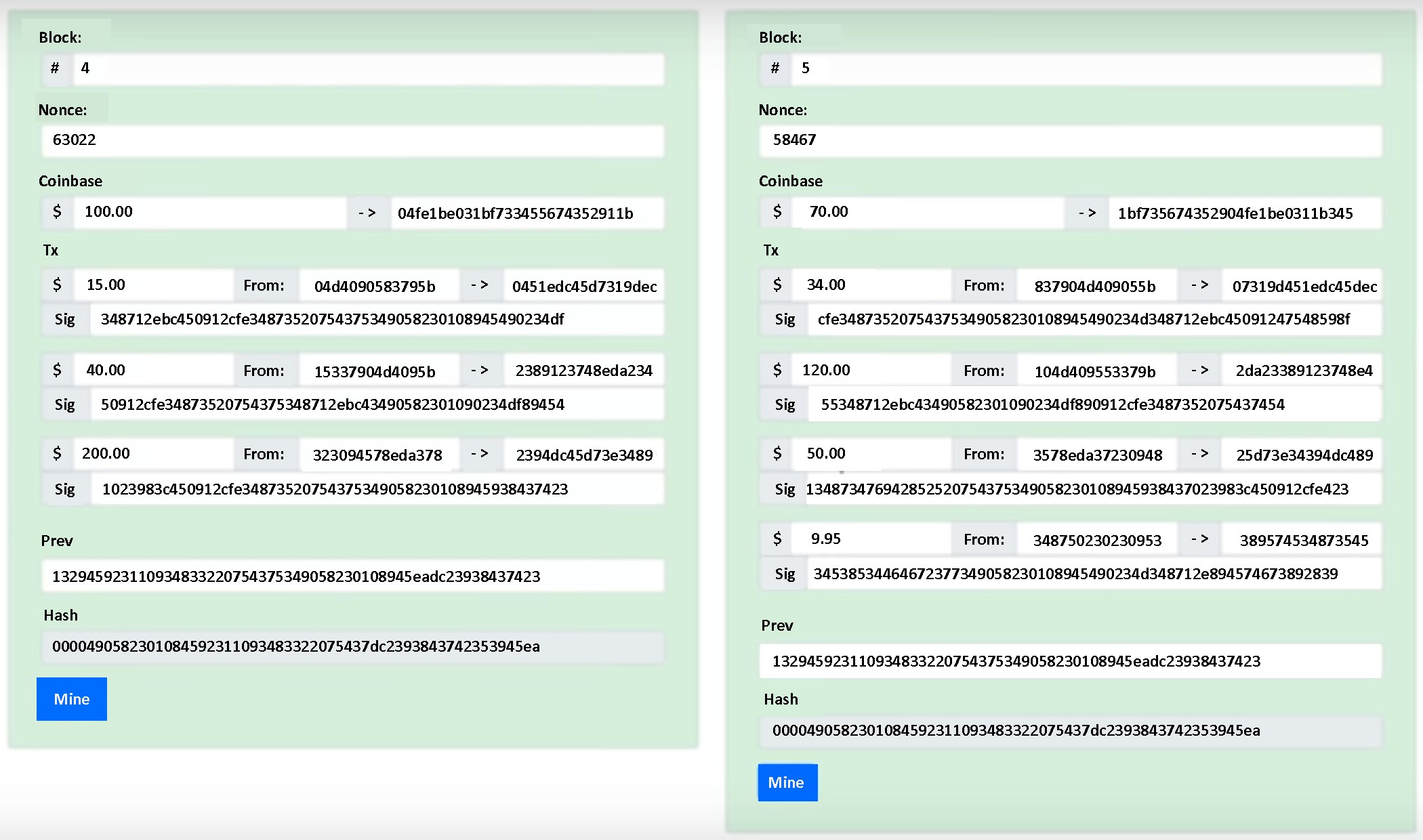
A block on the blockchain consists of block number, data (transactions and ledger), nonce, hash of the current block, and the hash of previous block. Previous hash is passed to the cryptic hash function and gives us the hash of the current block. The hash of this block will be used as a previous hash of the next block (Figure 4).



Hashing a Block using SHA-256

### Forming a Chain

The first block in the blockchain is a Genesis Block. After the genesis block, each block will be validated and then the validated block will be added to the blockchain. Each block contains a pointer to the previous hash. A hash pointer is similar to a pointer. Hash pointer contains the address of the previous block and the hash of the data inside the previous block. The chain of the blocks contains the ledger of all the systems.

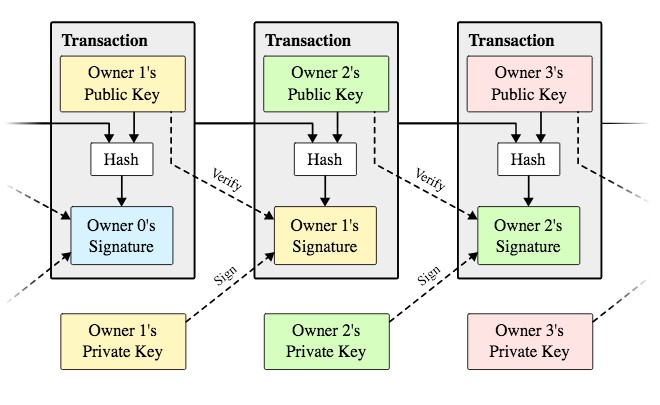


How Blocks in the Blockchain Form

### Transactions

Traditional blockchain application like Bitcoin, consist of transactions that represent an exchange of money between two entities (or users). Each valid transaction is recorded in a block, which can contain multiple transactions, for efficiency. Immutability is achieved by leveraging strong cryptographic properties such as hashing. [13] An electronic coin is a chain of digital signatures. Each owner transfers the coin to the next by digitally signing a hash of the previous transaction and the public key of the next owner and adding these to the end of the coin. A payee can verify the signatures to verify the chain of ownership. The problem of course is the payee can't verify that one of the owners did not double spend the coin. A common solution is to introduce a trusted central authority, or mint, that checks every transaction for double spending. After each transaction, the coin must be returned to the mint to issue a new coin, and only coins issued directly from the mint are trusted not to be double spent.

The solution is a timestamp server.

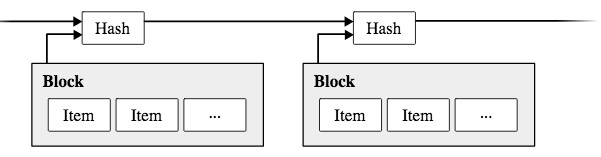


How transactions in blockchain work

### Timestamp Server

Timestamp server takes the hash of a block of items and time stamped it. It will widely publish the hash. The timestamp is a way of proving that the data must have existed at the time. Each timestamp includes the previous timestamp in its hash. This will form a chain.

Next timestamp will reinforce the ones before it as shown in figure.



Timestamp server

### Blockchain Consensus Protocols

In order for the blockchains to function globally, a practical, efficient and secure consensus algorithm is required for a shared public ledger. All blockchain-based applications use distributed consensus algorithm. Research on consensus mechanisms has proposed a large range of systems, from proof-of-work to proof-of-stake systems to the hybrid systems in between.

#### Proof of work

The mining proof-of-work (PoW) exists as a cryptographically secure nonce that proves beyond reasonable doubt that a particular amount of computation has been expended in the determination of some token value. It is utilized to enforce the blockchain security by giving meaning and credence to the notion of difficulty (and, by extension, total difficulty). However, since mining new blocks comes with an attached reward, the proof-of-work not only functions as a method of securing confidence that the blockchain will remain canonical into the future, but also as a wealth distribution mechanism [30]. Proof of work is accessible to as many nodes as possible. There is minimum need for specialized or uncommon hardware. The goal is to make the mining from electricity at the same rate for any node in the globe. A proof of work is used to implement a distributed timestamp server on a peer-to-peer basis. The proof-of-work involves scanning for a value that when hashed, such as with SHA-256, the hash begins with a number of zero bits. The average work required is exponential in the number of zero bits required and can be verified by executing a single hash.

#### Proof of Stake

Proof of stake is the most common alternative to proof of work. The validators (stakeholders) invest in the coins instead of investing in computational equipment and energy. All the coins of the system exist on day one and no mining is required. The chance of the validators to be picked for next block depends on the fraction of the coins they own in the system. The decentralized consensus mechanism has advanced security measures by eliminating the need for a trusted third party in any interaction [49]. One possible decentralized ledger implementation with security not based on expensive computations relies on proof of stake algorithms. The idea behind proof of stake is simple: instead of mining power, the probability to create a block and receive the associated reward is proportional to a user’s ownership stake in the system.

An individual stakeholder who has p fraction of the total number of coins in circulation creates a new block with p probability [14]. The nodes with the highest number of stakes in the network have the most interest to keep a secure network. If the network has successfully attacked the value of the cryptocurrency would drop due to the attacks. The attackers have to acquire most of the network (currency) in order to implement a successful attack. Doing so is very expensive and discourages the attackers.

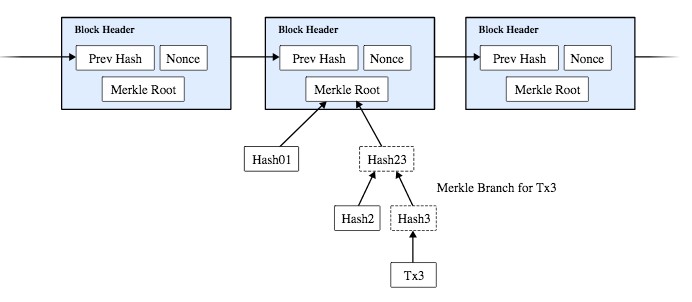
#### Proof of Capacity

Proof of capacity is also known as proof of space. The nodes on the network prove that they have enough storage to solve a computational problem. Proof of capacity algorithm targets computational problems such as hard-to-pebble graphs that need large amount of memory storage to solve the problem.

### Network

In a network, the new transactions are broadcasted to all the nodes. The new transactions are collected into a block by the nodes. Every node will work really hard to solve the problem, therefore provide the proof-of-work for its block. The first node that finds the proof-of-work will broadcast it to all the nodes. All the other nodes will only approve the new block only if all the transactions in it are valid, also not already spent. Once the nodes accept the new block, it will be added to the chain.

The longest chain is the correct chain, and accepted by the nodes. This is the chain that other nodes will add blocks to it. There are times that the two nodes broadcast the next block at the same time. As the result the blocks will not be the same. Some of the participating nodes will receive one or another first. The nodes will work on the first block they received. The other branch will be saved in case it becomes long. If one of the branches become longer and longer, all the nodes will move on to the longer node, and abandon the shorter one. The tie will be broken. The new transactions do not necessarily be broadcast to all modes. Block broadcasts are also ok with dropped messages. The node that did not receive a block will request it when it receives the next block. That is when the node realizes that it missed it. Once the latest transaction in a coin is buried under enough blocks, the spent transactions before it can be discarded to save disk space. To facilitate this without breaking the block's hash, transactions are hashed in a Merkle tree, with only the root included in the block's hash. Old blocks can then be compacted by stubbing off branches of the tree. The interior hashes do not need to be stored [5].



Structure of a Blockchain

### Distributed Ledger

A distributed ledger is an asset database, and a form of decentralized database. A massive network of participating nodes holds distributed ledger. These nodes have a full copy of the ledger. The nodes can be virtual or real. The distributed ledger runs on the blockchain technology.

New transactions in a network are bundled into blocks, which are added to the existing blocks, therefore forming a chain with cryptographic signatures. These signatures securely link the blocks to each other. Since the cryptographic signature depends on the chain of all previous blocks, changing an existing block in the chain would invalidate all the following blocks in the eyes of the rest of the network. As a result, when a new transaction is proposed no participant would accept transactions coming from a modified version of this chain: this constitutes blockchains ground-breaking fraud-detection system [16].

### Chapter 3 Problem Definition

**Aim and Objective**

The aim of this paper is to provide information about our project and share the knowledge of data security though blockchain technology.

We are going to reduce the possibility of any change in the system or any alter which can be done as fact creating an immutable system. The system will work even if a node fails so it will also be crash proof.

**Literature Survey**

It is a paper in which the author has proposed all the uses and any possible ways the blockchain could be used along with decentralization. Also, the author emphasizes on, what the future blockchain applications will be. There is also a detailed report on advantages it provides, different areas in which blockchain can improve computing and how it is better and the current traditional systems. To the most effective of our information, our work isn't the primary comprehensive study to grasp blockchain-based DApps. during this section, we have a tendency to 1st look over the similar work of blockchain ecosystems which uses it to run, then going to throw some lights on P2P applications, which are the normal sort of DApps

1. Blockchain application

For its decentralization, tenacity, data and verifiability, blockchain ar typically utilized in anonymous commerce, persistence services and cross structure transactions. Thus, finance services use block chain widely, IoT (Internet of Things), information security, edge computing and software package engineering.

2. Peer-to-peer Applications

DApps are applications which are based on decentralization which is also known as P2P network, which requires a lot of research work and efforts, including security, application and performance.

3. Security

There are two parts in security of P2P network: P2P network security and unethical users of the network which can pose danger to the security of the network. For security of the P2P network, researchers detect attacks, use of some more technologies like trustworthy computing and style new rules. Some harmful behaviours in P2P applications are found additionally.

4. Performance

The applications which are using P2P network are vulnerable to become a huge task for the local network, more prominently file sharing systems in P2P network. Researchers continuously watch the performance of the P2P networks and try to evolve the applications based on the P2P network by optimization of application layerer and network layer. Application. P2P technology is preponderantly utilized in several areas, like electronic communication app, file sharing system, development, security and many more. Cryptocurrencies, for example in 2009 there was a currency that came into existence known as bitcoin it is also based on P2p network, there are a large number of cryptocurrency which are being issued, which ultimately resulted in the growth of blockchain technology, more importantly public blockchain technology. as the CoinMarketCap says, today their square measure over a pair of, cryptocurrencies within the world. Few researches square measure concerning blockchain-based DApps, however these DApps have nice influence on the blockchains on that they run. Some work has been performed to assist developers develop Blockchain-based DApps.

**Problem Statement**

The system which we are using right now is having a centralized approach to have good sharing of information and to do communication among the similar application. In this type of application, the data is stored in a centralized database system there is a huge possibility of losing data if the central server fails, there are uncountable counterfeit of data and information. Products publish their advertisement on social media network without any known root transgressor (like WhatsApp and Telegram), the data stored at the central server can be altered or hacked. So, to deal with all these problems we are going to make secure application using blockchain.

**Scope**

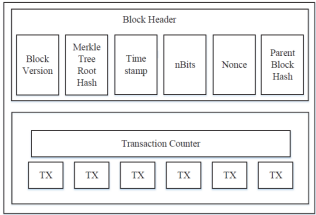
The scope of the project is to make a software that will provide all the features which is provided by the current system and add some new features which is relevant in the today's world and is the need of people like adding blockchain wallet, cloud, etc are the future aspect of the application. So it is required that we should overcome the drawbacks of old system. The software which is going to be made will be very secure and reliable than the current application.

**Proposed System**

All the user data is stored on a block that is connected to other blocks forming a chain. A decentralized application does not have a centralized server. It is a peer-to-peer network. Also, the data that is stored in the block is almost impossible to view as very secure encryption and hashing functions (256 bits) are used. Also, if a hacker tries to make changes to the information in the block, then, he/she will have to make changes to all the copies of that block on the whole blockchain network and that can be quite impossible. Though the block is on all nodes, they cannot access the information in it, only the person for whom the information can access it.

**System Architecture**

A block body consists of a dealing counter and transactions. The utmost range of dealings that a block will contain depends on the block size and therefore the size of every transaction. Blockchain uses associate degree uneven cryptography mechanism to validate the authentication of transactions. Digital signature supported uneven cryptography is employed in associate degree untrusty atmosphere. Now, we tend to shortly justify the digital signature.



Block Structure

Block

A block includes the block header and therefore the block body. The block header consists of:

a) Block version: gives instruction about the which block validation process.

b) Merkle tree root hash: it is the place where the hash value of all the changes made are present. c) Merkle tree root hash: it is the place where the hash value of all the changes made are present.

d) n-Bits: target threshold of a legitimate block hash.

e) Nonce: a 4-byte field, which usually starts with 0 and increases for every hash calculation.

f) Parent block hash: a 256-bit hash value which points to the block ahead.

**Advantages**

1) Secure: The application which will be made will be more secure. The security will be there because of usage of the technology blockchain. As we have discussed in the paper that blockchain consist of a distributed ledger mean every user of the system has a copy of data. Hence it is impossible to change data.

2) Fast and Efficient: It will give plenty of time to propagate the transaction as there are multiple nodes in the blockchain network.

3) Decentralization: It is achieved by the consensus algorithm in blockchain technology. It will maintain data consistency and durability of data in a decentralized environment.

4) Immutability: Large numbers of users will store large amounts of data so it is nearly not possible to change the data in a blockchain public network which uses a different securing algorithm.

**Chapter 4 Tools of Implementation on Blockchain**

## Structure of Smart Contracts

In most instances, smart contracts are written in Solidity, but there are other options available such as Lisk, Bamboo and Viper. A basic coding structure of smart contract in Solidity:

* Version pragma will tell the compiler what version of the Solidity to use to compile the contract. By using the right version, the code compiles at it was originally intended.

pragma solidity ^version; //version of the required compiler

* The Contract can inherit properties of other contracts. Contract is the highest element of the code.
* State variables will be permanently stored in smart contract. State variables are similar to class variables. State variables are defined by their type and level of visibility. State variables cannot be external.
* Functions (similar to functions in object oriented programming) are the codes that execute the actions of the contract. Functions can be called as many times as necessary. Default Functions are public but they can be defined public or private.
* Function Modifiersmodify semantics of functions. They can be seen as helpers to avoid repetition in methods' conditions (in terms of access or execution) [16]. Modifiers are conditioned to a function or conditions before we run the function. They are used to amend the semantics of a function in a declarative way. These can be used to automatically check a condition prior and/or after the execution of a function [21]. A function can have few function modifiers.
* Events are the logs created by the contract. Events are visible for everyone to see on the blockchain. Events expect address from and address to
* Mapping is used to structure value types, such as booleans, integers, addresses, and structs. It consists of two main parts: a \_KeyType and a \_ValueType; they appear in the following syntax [25]:

mapping (\_KeyType => \_ValueType) mapName;

mapping (address => uint) public balances;



* Block is the block of information within that Ethereum. Number, difficulty, coinbase (current block miner’s address, which is a function and return address payable), gaslimit (current block’s gas limit), and timestamp are members of each block.
* Msg which is the data in the receiving message. Sender, value (number of wei sent with the message), data (complete calldata) and sig (first four bytes of the calldata also known as function identifier) are the members of msg. The values of all members of msg, including msg.sender and msg.value can change for every external function call. This includes calls to library functions [26].

## Dependencies and Developer Setup

### Node.js and Node Package Manager

Node.js provides the tools to build JavaScript applications that run outside the scope of the browser. Node.js is single threaded and all the users are sharing the same thread. Events are raised and recorded in the event queue, and then handled in the order that were raised [36]. Node.js is asynchronous, non-blocking and can handle more than one event at a time, therefore it is fast. Node package manager or NPM is the first installed dependency. NPM is a way to reuse codes from other developers and share codes with them as well. The bits of reusable codes are called packages and sometimes modules. It is a module-based approach. A package is a directory with one or more files in it that also has a file called package.json with some metadata about that package. NPM comes with Node.js. The command “npm install” will install all the dependencies for the package.json and creates the node\_modules folder.

### React

React is one of the most popular JavaScript libraries for creating user interfaces, and is a vital skill for many front-end and full stack developers [34]. React creates reusable components, and these components display data as it changes over time. React makes it painless to create interactive UIs. It will efficiently update and render just the right components when the data changes [24]. React manages all of the view and template logic in the same file. React page is a collection of components. In ChatApp project create-react-app is also used.

### Web3

The Web3 class is a wrapper to house all Ethereum related modules [29]. Web3 is a JavaScript library that allows the frontend application to talk to the blockchain and interact with DApps. Web3 is an open source standard project to interact from the client side applications to the Ethereum network. One of the challenges with developing the client side of the decentralized application is finding a way to effectively wiring up the frontend application with the blockchain, web3 helps us with this challenge. Web3.eth is an important package that allows the user to interact directly with the blockchain. Web3.eth gives the required tools to the user to access Coinbase of a particular block, look at transactions, look at different accounts and check their balances and much more.

Web3 is used for:

o Performing transactions

o Managing user accounts

o Smart Contract Interaction

To get web3.js into the project

$ npm install web3

$ node

> var Web3 = require(‘web3’)

Create a web3 instance and setting a provider [28]:



In this project web3 was used to interact with Ethereum blockchain by calling methods, and usage of objects, utils and some other packages.

### Application Binary Interface (ABI)

ABI is a JSON encoding of a smart contract. Etherscan website has a copy of ABI which is up for grab. The web3-eth-abi package allows decoding and encoding parameters from an application binary interface. This will be used for calling functions of a deployed smart-contract.

### Truffle Framework

Smart contracts don’t get compiled on Ethereum network. They are already compiled bytecode when they exist in the blockchain. So we need to compile and migrate them and one way is using Truffle [26]. Truffle also has a test server. It provides the tools to test the smart contracts and check for the bugs. By running Truffle a contract is initiated and it takes a deposit. Here are some uses of Truffle:

* Build decentralized applications on the Ethereum blockchain
* Tools to write smart contacts with the Solidity programming language
* Test the smart contracts
* Deploy smart contracts to the blockchain
* A place to develop client-side application

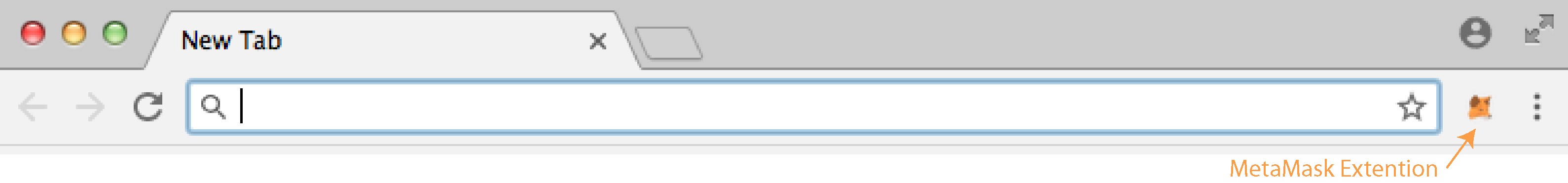
Migrations are JavaScript files that help deploy contracts to the Ethereum network. They will stage the deployment tasks.



Running Migrations Using Truffle

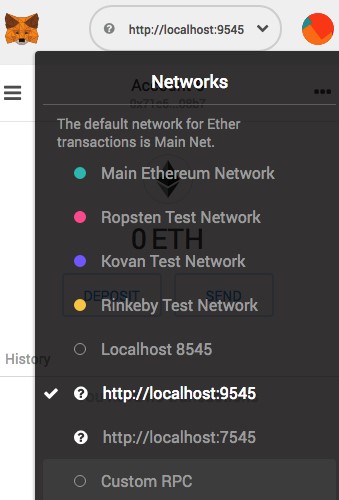
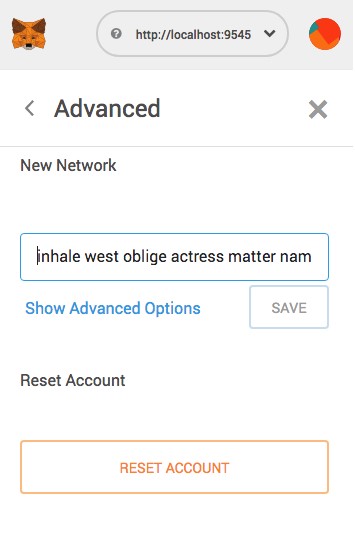
### Metamask for Authentication and User Authorization

The next dependency is the Metamask extension for Google Chrome. Metamask is used to connect to blockchain. MetaMask is used for verification of the owner of a blockchain address, which can send and receive messages. Metamask is used to connect to the local Ethereum blockchain with the personal account, and interact with the smart contract. It enables the user to have a wallet that is built into the browser so Ethereum can be sent and it can be signed any time Ethereum has to be sent to the network.

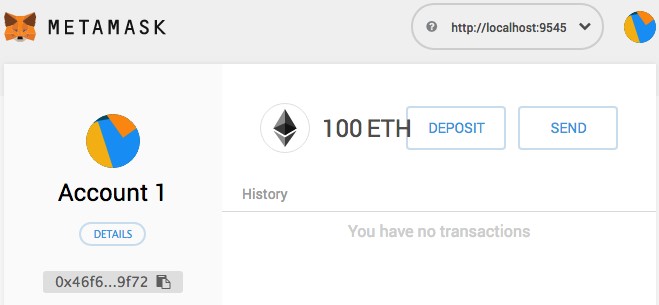


MetaMask Extension for Chrome Browser

Copy the mnemonic phrases and in MetaMask extension, create a custom RPC as shown in figure 17 and figure 18 and point it to http://localhost9545.

Create custom RPC Reset the account



MetaMask is setup and ready to test the DApp.

The localhost:9545

It is possible to test the apps without having to sign them in local environment, but once the developer starts running in the test environment, the contracts have to be signed [26].

Brave Browser is an alternative to Chrome which has Metamask built in.

### Rinkeby Test Network

Once the smart contract is tested locally it needs to be tested in a testnet and eventually mainnet. Geth is used for both. Depending on where geth is pointed to, it will test it on testnet or mainnet. Go-Ethereum supports connecting to a proof-of-authority based test network called Rinkeby. This network is lighter, more secure, but is only supported by Go-Ethereum. Network ID value for Rinkeby is 4.

# **Chapter 5 The Development and Implementation**

The ChatApp has been developed on the base of last chapter's dependencies. ChatApp has a smart contract and a web interface. The smart contract will allow accessing the blockchain. The smart contract is where the data is stored. The user interface is where the encrypted messages are sent from one address to another. Only the sender and the receiver of the message can decrypt the message, however the encrypted messages will forever be on the Ethereum blockchain. ChatApp does not require a centralized server and only depends on the Ethereum network. In order to get in touch with other people, the user shares his public key with other users while keeping his private key secret. Messages are encrypted using an encryption key, which is the same key that will be used to decrypt the message. Only the sender and the receiver can generate the same encryption key. The sender generates the key from sender’s private key and receiver’s public key. The receiver will compute the key from his private key and the sender’s public key. The contract generates events that will be stored on the network. It does not store the messages or contacts. All the encryption and decryption will be done with web app and not through the contract. The messages and friends list will be aggregated based on history of events generated by the contract. ChatApp members and the relationship between members are the only information saved in the contract’s storage.

Basic structures of ChatApp are

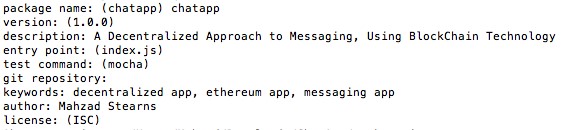
* Joining by signing the private key (There is a transaction fee involved)
* Accept friend request (transaction fee)
* Add friends using their public key (transaction fee)
* Send messages to friends and receive messages from friends (transaction fee)

• Receive messages from friends (free)

## JSON File

All the dependencies in the project will be specify in package.json file. A package.json file is a manifest that contains information about the app. It will easily distribute the application code without having to worry about distributing all the dependencies as well [36].

$ npm init



Npm also provides us a way to automate running, testing, debugging our applications or, really, running any Unix or DOS commands [36]. Scripts property is part of package.json. "Npm start" is used to start the web application. Scripts create mock data on the blockchain for manually testing the app.



The dependencies that are used for ChatApp are: solc, web3, react, react-dom, create react-app, ethereumjs-tx, ethereumjs-wallet, truffle-hdwallet-provider, ganache-cli, next, mocha and flux.

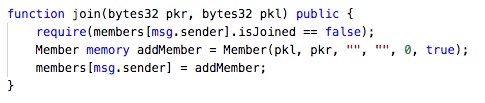
## The Smart Contract

Communication between contracts takes more gas than method calls within a single contract. Also, every separately deployed contract in the system increases the overall attack surface. When the contracts are compiled, only that final contract needs to be deployed to the blockchain. Its bytecode contains the whole inheritance chain, structs, state variables, methods and mappings. Solidity contracts support multiple inheritances.

Address: is the address on the Ethereum network. There is one address for sender and one address for the receiver. Message is a special object that comes in called message and the message is going to contain some data including the sender, which will be the address of who’s sending the money, and also the value, which will be the value of what's being sent. The msg.sender variable is always present upon execution of a contract’s public methods (including the constructor) and could represent a wallet address or the address of another smart contract.

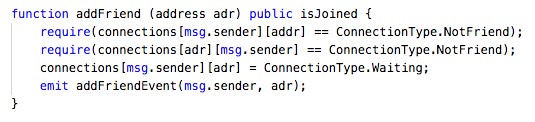


* The Join Function

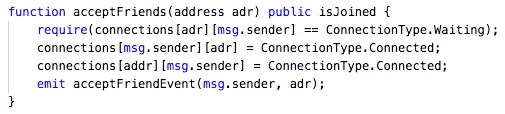


* Add Friend Function

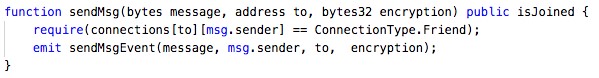
The function addFriend will add friends by their addresses. One has to be joined to be able to add friends.



* Accept Friends Function



* Send Message Function



* Encryption

In a blockchain system, any key holder can use their private key to sign a piece of data.

This results in a signature. Whoever obtains the signature can use this to:

* Recover the public key (account address) of the Author
* Verify whether the message is the same as the one signed by Author const message = web3.sha3('The Message'); const signature = await web3.eth.sign(account, message); const { v, r, s } = ethUtil.fromRpcSig(signature);

The first line creates a SHA3 hash of the message we want to sign. This results is 32 bytes (256 bits) hash.

Second line then uses Ethereum’s JSON RPC to tell the Ethereum wallet (which controls the private key) to sign the message on a given account, resulting in a signature.

## Functionalities

* **Sending Transactions:** Method eth\_sendTransaction,creates new message call transaction or a contract creation, if the data field contains code.The parameters are:
* **Object** - The transaction object
* **to**: data, 20 bytes - (optional when creating new contract) The address the transaction is directed to. The to field in ChatApp is the ContractAddress. There is no from, because when the transaction is signed with the private key, it is obvious who’s sending it
* **gas**: quantity - (optional, default: 90000) Integer of the gas provided for the transaction execution. It will return unused gas.
* **gasPrice**: quantity - (optional, default: To-Be-Determined) Integer of the gasPrice used for each paid gas. It is the reward that is sent to the miner and it is the cost of each unit of gas when we send the transaction. Every time Ethereum is built on the blockchain gas has to be paid.
* **gasLimit**: is safeguard for the maximum money allowed to be sent
* **value:** quantity - (optional) Integer of the value sent with this transaction. For

ChatApp this value is zero, since no ether will be transferred.

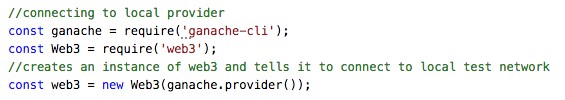
o **data:** data - The compiled code of a contract or the hash of the invoked method signature and encoded parameters. We send data which is the bytecode of the smart contract

* **nonce:** quantity - (optional) Integer of a nonce. Nonce is the account’s transaction count. It has to be hexadecimal. It is a safeguard that prevents the double spending problem. For a brand new account the nonce value is zero.

## Web3

In the Ethereum world, web3 is used as an absolute solution for communicating between a JavaScript app and the Ethereum network. Web3 is the library for programmatic access to a deployed contract on the blockchain. All functions in our contract are asynchronous that will be executed using web3. Different instances of Web3 library were built in the project, so each instance could connect to a different Ethereum network. One instance of Web3 at a time is used. Web3 expects a defined network, with which it would communicate. We call it a provider.

The provider is coming from the local test network but can be swapped out for another provider from a different network.



## The Testing Architecture

It is very important to test the smart contract before we deploy it to network because the blockchain is immutable and also transactions on the blockchain cost gas. The only way to fix a bug is to deploy a new version of the contract; the old version will always stay on the blockchain. Testing the smart contract before deploying it helps to make sure the functions have the expected behaviors.

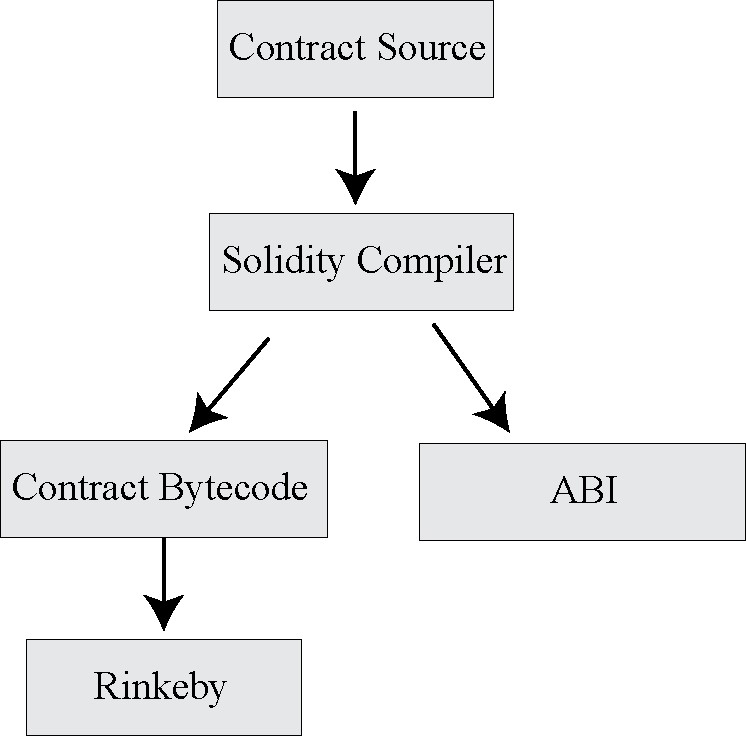
By compiling the contract we get the ABI and bytecode as output. We deploy the bytecode to the local test network. The test will call different functions inside the contract.

We generate a local network solely for deploying and testing the smart contract. This involves a custom script. A library named Ganache, which is discussed in previous chapter, will create the local network. The JavaScript interface (ABI) will be fed to web3. A series of tests were written to make sure the app is running as it was expected without errors before it is deployed to the main network.

## Build, Compile and Deploy

### Solidity Compiler

Solidity compiler compiles the smart contract into bytecode and ABI. The compile and deploy structure is shown in figure 20. Solidity Compiler was installed as part of npm package.



Compile & Deploy to Rinkeby Structure

The reason we can’t simply require our ChatApp.sol file is because Node will try to execute files as JavaScript and would throw an error with a .sol file.

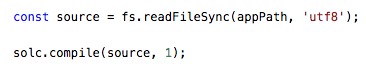
The compile.js file will look into the contract folder and compile the solidity files. The contents of the file will be read of the hard drive. Therefore, we make use of two separate built-in modules.



The path module will build a directory path from the current compiled js file into the contract file. By using the path module, we are guaranteed to get cross-platform compatibility. This will generate a path that points directly to ChatApp.sol file.

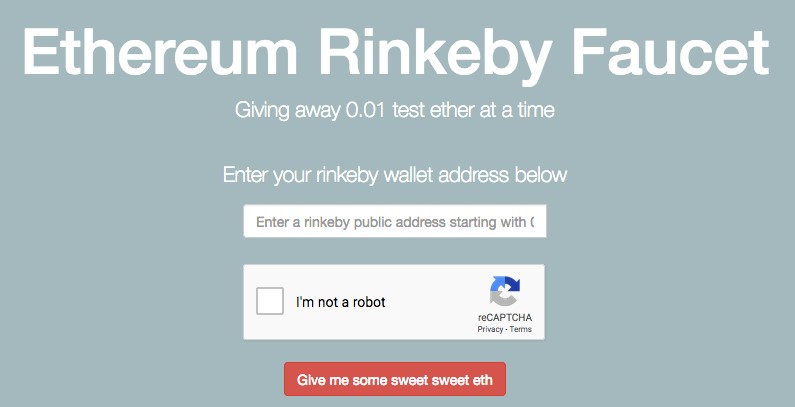


To read the raw source from the contract we create the constant source. Now we can write the compiled statement, this is where the solc comes to work.



### Smart Contract Deployment to Main/Test Networks

In order to write procedures on the Ethereum network, gas is required. Rinkeby test network has been used in this project. Ethereum Rinkeby faucet provides free gas to the provided address.



Rinkeby Faucet

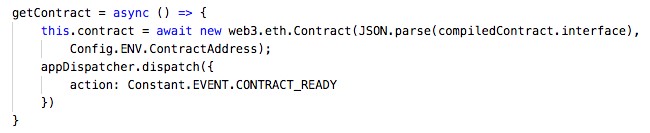
The web3 library is the sole mean for interacting with Ethereum network. It’s the portal to everything with Ethereum.



Constructor function allows us to interact with existing contracts on the blockchain, or to create and deploy new contracts. The first argument to the contract’s constructor is the ABI. This is where we get a conversion between the JavaScript world and Solidity world. The interface property is passed to JSON.parse. Solidity compiler gives out the interface in JSON. A JavaScript object needs to be given to the contract and this is where it is taken care of. It will tell web3 that there is a contract out there and gives it the interface it expects. There is no information about the deployment or creation. The deployment of the contract is asynchronous therefore await is used.

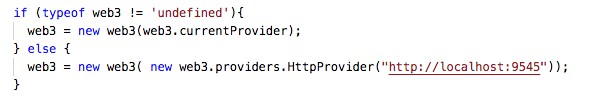
## Connecting to the Blockchain and Interacting with it

The web3.eth.Contract object makes it easy to interact with the smart contract on the Ethereum blockchain. When a new contract object is created, the JSON interface of the respective smart contract has been given and web3 will auto convert all calls into lowlevel ABI calls over RPC for the developer. This will allow the developer to interact with smart contract, as they were JavaScript objects.

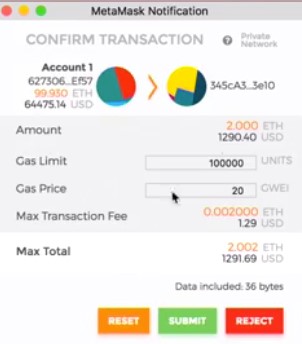


## Application of Metamask

Since interaction with MetaMask is used in this project, MetaMask will inject it's own web3, therefore web3 is defined.



This way if we have MetaMask running it will use the MetaMask’s version of web3. Web3 will allow the contract to directly interact with the Ethereum blockchain. In Metamask login with Mnemonic code and use the public key from Truffle development to interact with the contract. This will ask for signature and will launch the MetaMask.



Signature Request from MetaMask

**Chapter 6 Conclusion**

**Future Scope**

The goal of this project was to analyze inefficiencies of the traditional central based messaging applications and address them by using Ethereum smart contracts in trustless and secure decentralized application. This has been achieved by developing a decentralized application that can be run by any user to send and receive encrypted messages. Adding capabilities like picture and video sharing, customized chat groups and exploring a new technology in blockchain called Whisper for messaging are some of the future directions.

**Conclusion**

In this project, we are developing an application that makes use of blockchain in a very efficient way. Also, by eliminating the centralized approach, we can assure the safety and availability of data and communication. Decentralized applications tend to make the interaction between two people more efficient and simpler. The chatting process nowadays has a mediating node, while our software does not have any mediating device/node i.e., every person is connected by peer-to-peer network. balance dimensionally. If you must use mixed units, clearly state the units for each quantity that you use in an equation.

**References**

[1] Zheng, Z., Xie, S., Dai, H., Chen, X., & Wang, H. ―An Overview of Blockchain Technology: Architecture, Consensus, and Future Trends‖. 2017 IEEE International Congress on Big Data (BigData Congress) (2017).

[2] S. Nakamoto, ―Bitcoin: A peer-to-peer electronic cash system, ‖ (2008)

[3] Zheng, Z., Xie, S., Dai, H. N., Chen, X., & Wang, H. ―Blockchain challenges and opportunities: a survey. International Journal of Web and Grid Services‖, 14(4), 352. (2018). [4] https://www.dapp.com/article/annual-dapp-market-report2018, Gregorian calendar month 2019 ―Dapp.com 2018 Dapp Market Report‖ https://www.dapp.com/article/. (2019)

[5] R. Li and H. Asaeda, "A Blockchain-Based Data Life Cycle Protection Framework for InformationCentric Networks," in IEEE Communications Magazine, vol. 57, no. 6, pp. 20-25, June 2019

[6] Kosba, A. Miller, E. Shi, Z. Wen and C. Papamanthou, "Hawk: The Blockchain Model of Cryptography and PrivacyPreserving Smart Contracts," 2016 IEEE Symposium on Security and Privacy (SP), 2016.

[7] Wörner, A., Meeuw, A., Ableitner, L. et al. Trading solar energy within the neighborhood: field implementation of a blockchain-based electricity market. Energy Inform 2, 11 (2019)

[8] Chris Reed, Uma M Sathyanarayan, ShuhuiRuan, Justine Collins, Beyond BitCoin—legal impurities and off-chain assets, International Journal of Law and Information Technology, Volume 26, Issue 2, Summer 2018

[9] J. Lockl, V. Schlatt, A. Schweizer, N. Urbach and N. Harth, "Toward Trust in Internet of Things Ecosystems: Design Principles for Blockchain-Based IoT Applications," in IEEE Transactions on Engineering Management, vol. 67, no. 4, pp. 1256-1270, Nov. 2020

[10] Nguyen HL., Eisenbarth JP., Ignat CL., Perrin O. (2018) Blockchain-Based Auditing of Transparent Log Servers. In: Kerschbaum F., Paraboschi S. (eds) Data and Applications Security and Privacy XXXII. DBSec 2018. Lecture Notes in Computer Science, vol 10980. Springer, Cham

[11] J. Xu, S. Wang, B. K. Bhargava and F. Yang, "A BlockchainEnabledTrustless Crowd-Intelligence Ecosystem on Mobile Edge Computing," in IEEE Transactions on Industrial Informatics, vol. 15, no. 6, pp. 3538-3547, June 2019