A Major Project Final Report on

Centralized Blockchain-Based Banking Transactions using Digital Currency

Submitted in Partial Fulfillment of the Requirements
for the Degree of Bachelor of **Engineering in Computer Engineering**
under Pokhara University

Submitted by:

Amrit Bhusal, 171303

Satendra Yadav, 171351

Sushant Babu Luitel, 171338

Yukta Prasad Sharma, 171344

Under the supervision of

Asst. Prof. Roshan Kumar Sah

Date:

24 September, 2022

Department of Computer Engineering

NEPAL COLLEGE OF INFORMATION TECHNOLOGY

Balkumari, Lalitpur, Nepal

Acknowledgement

We wish to express our sincere gratitude towards the **Department of Computer Engineering, Pokhara University** for including the 8th semester major project as part of the syllabus. Also, we would like to extend our gratitude towards **Mrs. Resha Deo,** Head of department, Computer Engineering and Project Coordinator **Prof. Roshan Chitrakar** for supporting and assisting us in the completion of the Project.

Special thanks and gratitude to our project supervisor **Asst. Prof. Roshan Kumar Sah** under whose guidelines we were able to complete our project. We are wholeheartedly thankful to him for giving us his valuable time and attention and for providing us a systematic way for completing our project in time.

Last but not the least, our thanks goes to all our fellow friends and the college staff who willingly helped us with their abilities and valuable suggestions.

Every attempt has been made to include each and every aspect of the project in this report so that readers can clearly understand the project. We would be pleased to get the feedback on it.

Sincerely,

Amrit Bhusal, 171303

Satendra Yadav, 171351

Sushant Babu Luitel, 171338

Yukta Prasad Sharma, 171344

Abstract

Blockchain is a technology redesigning the Information and Technology industry with its characteristics like immutable data storing capability, peer-to-peer connection, and distributed ledger. There is merely a field, blockchain is not applicable. The Digital Transaction system is one of them. Blockchain technology has evolved to influence highly on digital payment methods and e-commerce. Cryptocurrencies like Bitcoin, Ethereum, etc have shown the scope of blockchain-based currencies in digital transactions. Considering this trend, the governments and the central banks are trying to catch up to have regulated and stable digital currencies representing present-day fiat currencies. Fiat currencies are the government and central bank-issued currencies like US Dollar, Nepali Rupees, etc. Unlike the unstable and irregulated cryptocurrency, Bitcoin, a Government-authorized currency is issued by the Central bank serving as a legal tender. This system is to provide transaction services using regulated digital currencies issued by any authorized Institution using a private blockchain.

Keywords: Blockchain, Digital transaction, hash, Digital Currency

Technical Terms: Ethereum, Distributed Ledger

Table Of Contents

List Of Figures

List Of Tables

List of Abbreviations

CBDC	Central Bank Digital Currency
CNCF	Cloud Native Computing Foundation
dApps	Decentralized Application
DC/EP	Digital Currency Electronic Payment
DCI	Digital Currency Initiative
DSA	Digital Signature Algorithm
ECC	Elliptical Curve Cryptography
ECDSA	Elliptical Curve Digital Signature Algorithm
Nonce	Number only used once
P2P	Peer to Peer
PoA	Proof-of-Authority
PoS	Proof-of- Stake
PoW	Proof-of-Work
RSA	Rivest-Shamir-Adleman
SDLC	Software Development Life Cycle
SHA	Secure Hash Algorithm
TSS	Time Stamping Service
UML	Unified Modeling Language
UTXOs	Unspent Transaction Output
UI	User Interface

1. Introduction

Since the first crypto coin, Bitcoin, was launched in 2009,which was the first decentralized cryptocurrency. Since then digital Currency has gone a long way and achieved enormous popularity. Apart from Bitcoin there are many more digital currencies like Ethereum, Binance, Ripple coin etc. which are taking the digital currency upto the new height. Digital transactions have been found to be simpler, faster, and more cost-effective. The usage of e-commerce and online services is boosted by the use of digital payment. Digital currency eliminates the traditional method of using fiat currency and the paperwork that has to be done while transacting the money. Cryptocurrencies are fast gaining in popularity. They pose a challenge to the government and current banking system since they encourage an unregulated, decentralized structure. They also do not require banks for regulation. One solution is a blockchain-based digital transaction system with government-issued and regulated digital currency.

Talking about the differences between Bitcoin, centralized digital currency and fiat currency, Bitcoin is decentralized which means there is no central authority and are not controlled by specific organizations and individuals. The one who mines the Bitcoin gets rewarded with a mining price. While fiat currency is the physical form of country authorized money. Fiat currency has central authority, issued and controlled by the government issuing it. Currencies like NRs, Inr, USD, Euro etc are some form of fiat currency. Centralized digital currency is the digital form of fiat currency which is authorized and controlled by the government issuing it. Just like fiat currency, digital currency must be a stable coin so that its value does not fall drastically over time. There are mainly three aspects that define centralized digital currency and they must be digital assets, backed by the central bank and controlled by the central bank as well.

As the concept of Digital Currency opened doors for a lot of possible application areas, the banking system is not an exception. Many physical fiat currency requirements, such as the development of real production facilities, are not applicable to digital currencies. Physical currency is also resistant to physical faults, whereas digital currencies are not. With a very broad application of blockchain, this can be a potentially large concept to replace the traditional ways of transactions and banking. Blockchain based digital currency enables transaction of digital currency without relying on trusted third parties(mainly banks).

Blockchain is still a new technology, it has taken the world of the internet by storm. It is expected to modify the internet and make it more decentralized. It is actually a set of blocks, linked in the form of chains. Each block contains a set of transactions which includes cryptocurrency transactions and the transaction data gets replicated to all the nodes to form identical transaction details in the ledger. Blockchain transactions are cryptographically sealed and secured which ensure security.

1.1 Problem Statement

The number of financial transactions and dependency of payment systems on digital transaction systems is increasing. The current Fiat Currency services contain high intermediation charges. Paper money is expensive to create, issue, and manage. Also, they have a high chance of being lost and damaged. The manufacturing and issuance process takes lots of time. In the present context, every transaction detail is not tampered proof thus having a high chance of being tampered with and data loss due to various physical damage and loss. Blockchain-based systems solve these issues. We find the following problems in the existing systems :

- In present day digital transactions, data is stored in normal databases which can be tampered. A false intentioned party can change it.
- Data stored in the central server.
- Popular Cryptocurrency are hard to implement in day to day retail payment due to its high fluctuate behavior,
- Such cryptocurrency lacks Financial regulations and can't be tracked. Taxation is hard to implement.

1.2 Project Objectives

This project's main goal is to provide service of bank transactions digitally and validate them to redefine trust in digital payment systems. However, the project has other objectives too.

The key objectives of this project can be listed as:

1. To study and implement if Blockchain and digital currency has a future potential to upgrade existing banking systems,
2. To develop a tamper-proof, distributed ledger to record financial transactions reducing the chances of data loss,
3. To implement Blockchain at the Enterprise level to enhance security and immutability,
4. To explore the usage of blockchain technology for the development of cryptographically secure digital transactions,
5. To analyze blockchain in the context of its functions, applications, and research trends

1.3 Significance of the Study

The findings of this study has greatly contributed to the benefit of society as the study contributes to creating a digital society by solving problems in digital ways. Digital Payment has become quite popular in the past few years, as suggested by many recent studies. As the digital transaction system is more secure and has more advantages over the traditional transaction system. On the other hand, it has become more profitable for the organizations as it increases the accuracy of results and it minimizes the cost and time dramatically along with proper data security and authorization.

We are focused on developing a platform that can provide data security and an easy digital transaction platform for users. Through this platform, banks have a separate block explorer CMS to track transactions on the top of their private blockchain and transactions are made using digital currency. Using this platform transaction data are stored securely on the network without worries about data theft and tampering i.e. making transactions immutable easy, fast, and secure.

2. Literature Review

Digital transactions are found to be easier, quicker, and more economical transactions. Digital payment allows distant payment instantly promoting the use of e-commerce and internet services. The popularity of cryptocurrencies is growing rapidly. They are a threat to the government and present-day banking system as it promotes an unregulated and decentralized system. Also, they do not need banks for regulation. Cryptocurrencies like Bitcoin and Ethereum are advanced and better in security but are unstable. These unstable cryptocurrencies bring financial instability. With no financial regulations on them, they might be used for criminal purposes as well.

Though they have some issues, they still are secure and better technologically. So, cannot be ignored. There should be a solution for resolving the existing problems and making use of their benefit. Blockchain-based digital transaction systems with government-issued and regulated digital currencies is one of the solutions. Among many ongoing projects related to this solution, below mentioned are a few of them,

1) **"How to Time-Stamp a Digital Document"[1]**

A paper published by Stuart Haber and W. Scott Stornetta in 1990 entitled "*How to Time-Stamp a Digital Document*" is the research paper proposing ideology on how digital data and documents can be preserved and secured from being faked, copied and tampered without owners authority. In a way we can say it theorized about enforcing copyright. Though Bitcoin by Satoshi Nakamoto is considered to be the first implementation of blockchain, this paper is the origin of Blockchain technology. In this paper researchers Authors have described how digital documents like research papers, articles and other papers can be safeguarded and copyrighted using time-stamp. Time stamp is a sequence of characters or encoded information identifying when a certain event occurred These character sequences are time of day, or seconds sometimes. i.e if a paper is uploaded, the time period when it was uploaded or issued is the exact time-stamp. Their solution takes time-stamping as a security key which cannot be altered. The solution includes improvements on hash and digital signatures. Hash was improved by the use of a family of cryptographically secure collision-free hash functions. Another improvement was made by using digital signatures which is an algorithm scheme for a party, the signer, to tag messages in a way that uniquely identifies the signer. With a secure signature that was used, when the time

stamping service (TSS) receives the hash value, it appends the date and time, then signs this compound document and sends it to the client. After checking the digital signatures, the client is assured that time stamping service actually did process the request, appended hash was correctly received and tha correct time is included. Time stamping service was basically included to service records at which date and time the document was received and retains the copy of document for safekeeping and whenever the integrity of the user's document is ever challenged, it can be compared with the copy stored by TSS.

Haber and Stornetta intended to introduce a mathematically sound and computationally practical solution to the time-stamping problem. A TSS can be made trusted using Collision Free Hash and Signature.

- **Hash** : Hash functions compress bit-strings of arbitrary length to bit strings of a fixed length *l*, and it is easy to pick a member of the family at random. It is computationally infeasible, given one of these functions,

  $$h : \{0,1\}^* \rightarrow \{0,1\}^l$$

  to find a pair of distinct strings x, x' satisfying $h(x) = h(x')$. Such a pair is called collision for h. The hash functions are one way functions and it is infeasible to compute another string x' given string x with $x \neq x'$ satisfying,

  $$h(x) \neq h(x'), \quad \text{for a randomly chosen } h.$$

  Hash function in the paper has been used to send hash value, $h(x) = y$ of

  document to TSS with y as time-stamping equivalent to time-stamping x .

- **Signature** : Signature scheme is an algorithm for a party, the signer, to tag messages in a way that uniquely identifies the signer. One way functions can be used to design a signature scheme satisfying the very strong notion of security that was first defined by Goldwasser, Micali, and Rivest. TSS receives a hash value, appends data and time, then signs the document and sends it to the client using signature. TSS assures client checking the signature, ensures hash was correctly received and correct time has been included.

Since, neither the signature nor hash functions could prevent issuing false time-stamping, a further mechanism was suggested. The legitimate time-stamp of a Algorithm A with document x, and timing information T in bit string can be written as,

$$c = A(x,T)$$

The above ***c*** is to be prevented from being forged to produce same time information ***T*** , and same certificate ***c*** later. For this two approach was proposed,

i) Linking : This approach is to constrain a centralized but possibly untrustworthy TSS to produce genuine time-stamps, in such a way that fake ones are difficult to produce. If bits from the previous sequence of client requests in signed certificates are included in present then we know time-stamp occurred after these requests. Linking leaves no room to insert a new document with forged timing information to the service, making it ineligible.

Let $\sigma(.)$ be procedure to sign document by TSS where, it issues signed and sequentially numbered time-stamp certificates. ***H***, be a hash function, ***l***-bit string ***y*** be time-stamping request and ***ID*** as client id number. Also, client request be, $(\gamma n, IDn)$, in nth request. TSS

- *Sends signed certificate*, $S = \sigma(Cn)$ *where,* $Cn = (n, tn, IDn, yn; Ln)$
  Here, ***Ln*** is the linking information, coming from certificates previously issued.

  $Lo = (t(n-1), ID(n-1), \gamma(n-1), H(n-1))$.
- *TSS sends* ***ID(n+1)*** *for the next request*.

Client checks signature ***S***, and ***ID(n+1)*** on arrival. Timestamp of the document can be verified as ***(S, ID(n+1))*** for any collision between client and TSS. The challenger of time-stamp can also call ***ID(n+1)*** to produce their time-stamp, ***(S', ID(n+2))*** and check copy of ***yn*** in ***L(n+1)*** and authenticate by including ***H(Ln)*** of linking information to verify with ***L(n+1)*** .

The paper also suggests another variation of linking, in which each request is linked to next ***k*** requests.

- *Linking information,* ***Ln*** *is*, $Ln = [(t(n-k), ID(n-k), \gamma(n-k), H(L(n-k))) \ldots (t(n-1), \gamma(n-1), H(L(n-1))]$.
- *Tss sends a client list* ***(ID(n+1), …, ID(n+k))*** *after next* ***k*** *requests have been processed.*

ii) Distributed Trust : The distributed trust approach is to distribute the required trust among the users of service. When a client needs to time-stamp, hash value ***y*** can be used as seed for a pseudorandom generator, whose output can be interpreted in a standard as a ***k-tuple*** of client ID numbers, ***G(y) = (ID1, ID2, ..., ID(k))*** . Client then sends a request ***(y, ID)*** to each client.

Sending client receives a signed message, $\sigma j = (t, ID, y)$. The final time-stamp of the client now consists, $[(y, ID), (S1, \ldots, S(k)]$. This scheme suggests the number of dishonest clients is always less than the majority of clients and suggests choosing seeds to be impossible to find.

This paper concludes by giving the idea that, if both the schemes/approaches be used together can be a secure practical solution to time-stamping of digital documents. This way it gave rise to block chain technology.

2) Bitcoin: A Peer-to-Peer Electronic Cash System[2]

This paper was proposed by Satoshi Nakamoto in 2008. The Bitcoin white paper has found out the new definition of money when the belief on the traditional financial system was being salvaged. This paper basically describes the technical specifications and motivation regarding the cryptocurrency. In this S. Nakamoto has pointed out the strong case for the invention of an online payment system that could replace traditional payment systems. The main purpose of bitcoin was to develop a computer technology that enables multiple parties to make online payments directly to each other (P2P cash system) without involvement of financial organizations such as Banks. This paper better explains how the transaction works, use of the network for transactions and the efforts that should be rewarded while mining.

This paper covers many sections related to the P2P cash system which includes transaction, Timestamp server, PoW mechanism, Network, Incentive, reclaiming disk spaces, simplified payment verification, combining and splitting values, privacy, calculation.

The transaction section of this paper explains how transactions are made and what are the things that should be considered while transacting. Electric coins are transacted through the system. Electric coin is nothing but a chain which consists of a digital signature. If the owner of the Bitcoin wants to transact to the other owner, they have to sign the transaction digitally by hash. Hashing in Bitcoin is done by an encryption algorithm which uses two mathematical related .i.e, public key and private key. The public key encrypts the transaction along with use of the owner's private key to digitally sign the transaction. Here the public key acts as the recipient address to whom the Bitcoin owner wants to send the coin.

The next section is Timestamp server in which Satoshi Nakamoto describes how the Bitcoin network uses a distributed Timestamp server to show the order of transactions generated. The

computer connected in the network acts as nodes which are distributed all over the world and makes the network secure. When the transactions are made, the timestamp server adds a timestamp to the hash of the block containing the transaction at the same time to all the nodes in the network. The timestamp gives us proof that the transaction existed at that time and every timestamp includes the previous timestamp in its hash.

Proof-of-work section describes the algorithm that the P2P computer network uses to mine Bitcoin to create a practically unchangeable history of the transaction. It is used to solve the cryptographic puzzle in the Bitcoin network. The nodes in the Bitcoin network start scanning, testing and discarding billions of nonce each second to find the nonce that meets the target which is set by the network at the time of grouping the transactions. When nonce is found by the nodes, then it is broadcasted to other nodes in the network, validated and a valid new block is added to the network.

The Network section of this paper outlines the transaction process. When the sender makes transactions then it is broadcasted to all the nodes in the network. The nodes gather the information and try to find the Pow for it to check the transaction has not been previously spent and the new block is broadcasted to the network and is added as valid block and accepted by the nodes to create new block using hash of the last block. The longest chain in the network is considered as the correct chain.

Next section is about Incentives in which S. Nakamoto proposes that the nodes participating in the network that supply computing power should be rewarded if they are the first node to create the block. Users have to pay transaction fees when they make a transaction which at a later point would become sole reward once enough coins were in circulation. Nodes in the bitcoin network are likely to stay honest than the defraud network as a huge amount of computing power is required to defraud the network.

The next section reclaiming the disk describes the size of the Bitcoin blockchain which grows in size as it is immutable which requires large amounts of memory for storage. So to encounter this problem S. Nakamoto proposed that once a transaction is buried under a sufficient number of blocks, the spent transaction before it is discarded to save disk space. For this all the transactions are reduced to a single hash- a root hash which can be done by Merkle tree.

Simplified Payment Verification section about payment verification which is possible without a user running a full node in the network by building a Bitcoin implementation that relies on connecting to a trusted full node and downloading only the block header and the client computer verifies correct connecting of chain header and a sufficient level of difficulty to ensure that it is correct blockchain.

In the Bitcoin network, combining and splitting of values apply to transactions that contain multiple inputs and outputs. The privacy of the transaction in the Bitcoin network is public and everyone in the network can see the transaction made but are unaware of the fact that between whom transactions are made. Users can identify themselves in the Bitcoin network with a public key and can access the transaction with their private key.

The final section of Bitcoin whitepaper is calculation which illustrates the unlikelihood that the Bitcoin network would be successfully attacked by a fraudster. This shows how complicated it would be for an attacker to start a new chain compared to a valid chain.

Bitcoin is a P2P system for trustless, electronic cash systems that uses PoW mechanism to record and store the history of transactions publicly and is highly safe against the attack as long as the majority of computing power is controlled by the honest nodes. The nodes in the network only accept valid blocks and reject invalid blocks based on consensus mechanisms.

3) OpenCBDC[3]

OpenCBDC, being an open-source project, engaged in collaborating technical research to understand the concept for space of design for potential central bank digital currency. OpenCBDC was the major outcome of Project Hamilton which is a collaborative research project between MIT DCI(Digital currency Initiative) and the Federal Reserve Bank of Boston. The major goal of this project was to investigate whether the general-purpose central bank digital currency was technically feasible or not and could be used by an economy the size of the United States. Project Hamilton also aimed to analyze the technical challenges, risk opportunities, and trade-offs.

OpenCBDC-tx, an experimental transaction processor, was the first phase of project Hamilton. It is a modular transaction processor for hypothetical CBDC for implementation of two architectures and has the capacity to process 1.7 million plus transactions per second. The key

concept of CBDC-tx was decoupling transaction and validation, ensuring a secure and flexible transaction format, and making transactions much more efficient. The central operator (central Bank) runs the central transaction processor and the mode of payment authorization was Digital signatures using UTXOs(Unspent Transaction Output). The goal of this project was to assist the central bank, industry, academics, policymakers, and others make decisions on how to best design this type of technology while promoting neutral and collaborative global research. With the help of OpenCBDC, we can hope to engage a community of engineers and scholars, alongside government, business, and civil society leaders in research and development to learn how the digital currency system can be best designed to advance privacy, user agency, innovation, and financial equity.

4) China: Digital Yuan[4]

In 2022 China launched e-CNY, also referred to as digital Yuan as the beta version of the digital Yuan app which is currently employed in 23 cities across China. The general public of these cities can download the app for IOS and Android through Chinese App stores and use the app for payment of products and services. Users are required to sign-up within the system which might be done through several traditional commercial and online banks.

The e-CNY is officially called the Digital Currency Electronic Payment (DC/EP), which is a digitized version of China's legal currency, the renminbi (RMB). It's issued by China's central bank, the People's Bank of China (PBOC). it's designed for high-frequency, small-scale retail purchases and transactions. The e-CNY forms a part of the monetary base (M0) of the country, which implies the digital currency makes up a little of the 'cash' that's in circulation.

Although e-CNY is a kind of digital currency, it's not a kind of cryptocurrency like bitcoin and Ethereum, but it's a form of central bank Digital Currency (CBDC) and is issued by the People's Bank of China (PBOC). Such that, it's not a decentralized currency, and also it doesn't operate on a blockchain system. The value of e-CNY is the same as RMB. The worth of e-CNY does rely upon the worth of RMB so it can replace physical currency.

The digital wallet, called shuzi qianbao in Chinese, is a digital wallet that's accustomed to storing and keeping track of e-CNY. This wallet is accessed through the digital yuan app also called shuzi renminbi in Chinese. The digital Yuan app contains a primary means through which the overall public can use e-CNY. Functionally, e-CNY is comparable to the other online

payment platforms. The digital yuan app also allows the transaction to shop for goods and services like every other online payment platform. However, e-CNY is basically a digital banknote and makes up a little of the country's overall cash supply. It means spending the e-CNY is unlike employing a credit card or a digital payment platform because the money on these platforms directly comes through user accounts. e-CNY completely removes the third-party system that reduces the additional charge required in transactions through the digital payment platform.

5) Central Bank and Future of Digital Money[5]

The author of this whitepaper is M. Bouchaud, Tom Lynos, M. Saint Olive and Ken Timit. The ConsenSys whitepaper entitled “Central Bank and Future of Digital Money” has explained the overview and proposal for central bank digital currency on Ethereum blockchain. This paper has addressed the cryptocurrencies, stable coins and evolution of digital money over the last decades with the evolution of central bank digital currency. This ConsenSys whitepaper more direct and innovate ways and keep pace with technological change. CBDC could help and simplify and reduce the cost of cross-border remittance as well while forming the basis of a more efficient and more secure way of interbank payment. This Paper has considered Ethereum as the best suited blockchain for the implementation of CBDC for more secure and reliable way interbank and cross-border payments.

This ConsenSys whitepaper has better explained the topics about the cryptocurrency and stable coins, CBDC, benefits of digital currencies for central bank and the digital economy, the basic requirements that are necessary for implementation of CBDC and the architecture for Ethereum based central bank money. This paper has focused on the evolution of digital assets over time and the future oriented monetary policy and regulatory tools for the regulation of digital assets. This paper include the requirement that are necessary for successful implementation of CBDC and the requirements like distribution(the issuer of CBDC “the central bank” decides on how to circulate it), sound governance for controlled and regulated infrastructure with clear governance structures in terms of design, development, maintenance, funding, upgrades etc., privacy and transparency, token-based or account-based CBDC, performant and operationally robust, legally sound etc.

The architecture of CBDC proposed by these papers depends on a private, permissioned, Ethereum based network in which central bank appointed intermediaries act as node. As Ethereum provides both permissionless and permissioned networks, it can be very useful in creation of tokens and permissioned Ethereum is the best platform for CBDC. Central bank could easily design and implement tokens that can be widely circulated yet whose issuance and destruction remain firmly under control of the central bank.

The technical requirements for the proposed architecture are:

- Full control of money supply by central bank
- Quasi-real-time asset transfer at negligible cost
- High transaction throughput
- Large number of network participants
- Privacy of consumer data and transaction
- Confidentiality of business data
- Compliance with KYC/AML and related regulations
- Assets recovery and acceptable environmental impact

Architecture of Ethereum based CBDC has basically 4 layers. The layers are:

- **Layer 1/ Base settlement layer:** One base settlement layer on permissioned Ethereum blockchain
- **Layer 2 :** comprises network state channels between intermediaries that would enable fast payments.
- **Layer 3:** intermediary operates on its own side chain, where the central bank is a participant and can ensure that the supply of money remains consistent with the supply of CBDC allocated to the intermediary in layer 1(base settlement layer).
- **Layer 4:** this layer consists of end users that may be dApps and wallets.

6) CBDC (Central Bank Digital Currency)[6]

CBDC is short for central bank Digital Currency, an electronic kind of central bank money that citizens can use to make digital payments and store value. they're based on the worth of that country's Fiat currency. Many countries are attempting to develop CBDCs, and a few have even implemented them.

Fiat money is a government-issued currency that's not backed by physical commodities like gold or silver. It's considered a style of a tender that is used to exchange goods and services. Traditionally, paper currency came within the type of banknotes and coins, but technology has allowed governments and financial institutions to supplement physical paper money with a credit-based model during which balances and transactions are recorded digitally.

The introduction and evolution of cryptocurrency and blockchain technology have created further interest in the cashless environment and digital currencies. Thus, governments and central banks worldwide are exploring the chance of using government-backed digital currencies. When, and if, they're implemented, these currencies would have the complete faith and backing of the govt. that issued them, a bit like paper currency. Cryptocurrencies, as we all know them today, are extremely volatile and lack government backing — CBDCs overcome these concerns while using the identical underlying distributed ledger technology of cryptocurrencies. Governments recognize CBDCs as tender within the issuing central bank's jurisdiction, meaning anyone can use them for payments and each merchant must accept them.

If a country issues a CBDC, its government considers it to be a monetary system, a bit like fiat currencies; both CBDC and physical cash would be legally acknowledged as a variety of payments and act as a claim on the financial organization or government. A financial organization's digital currency increases the protection and efficiency of both wholesale and retail payment systems. On the wholesale side, a financial institution's digital currency facilitates the quick settlement of retail payments. It could improve the efficiency of constructing payments for the purpose of sale or between two parties (p2p).

No physical coins or notes are available to individuals in a digital society, and all money is exchanged in a very digital format. If a country intends to become a cashless society, a digital currency with government / financial organization backing could be a credible alternative. The pressure for governments to adopt a CBDC is powerful because the marketplace for private

e-money is on the increase. If it becomes mainstream, beneficiaries are at a drawback because e-money providers aim to maximize their profits rather than the general public. Issuing a CBDC would give governments a footing over the competition from private e-money.

3. Methodology

3.1 System Overview

The overall system overview of 'Centralized Blockchain Based Banking Transactions using Digital Currency' is shown below. The main components of this system are :

1. User Wallet UI
2. Bank UI
3. Blockchain Technology
4. Distributed Network

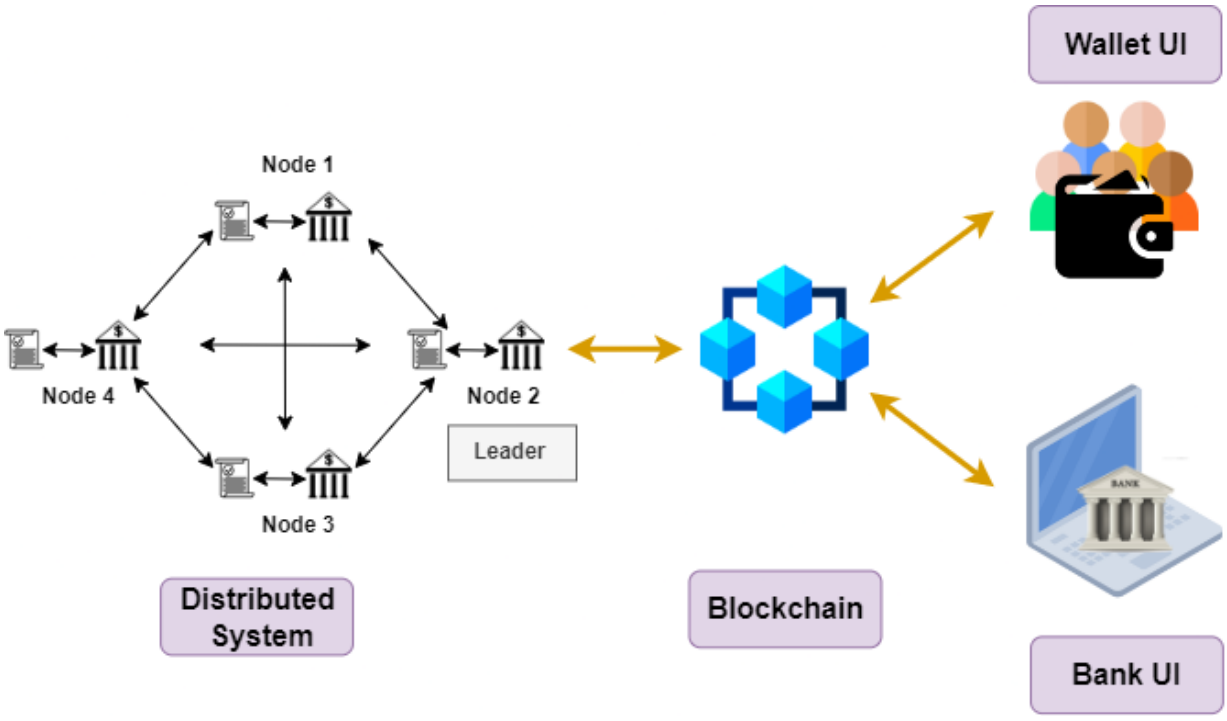

Figure 1. System Overview

Distributed system uses a Raft Algorithm based ETCD database which uses Level DB as its core database. ETCD is a CNCF (Cloud Native Computing Foundation) project. It stores data in key-value pairs. Distributed system is handled by the Bank and only the bank maintains these node servers.

Blockchain is the part where every logic and service has been written. It handles JSON type block data to store data in an array called Chain. The block data once stored in data in a chain considering its timestamp, it can't be changed, which is an immutable property of blockchain.

User Interface facilitates the Users and Banks to use Blockchain systems. It connects them with our system letting them use it.

1. User Wallet UI

   A separate wallet for users to make daily transactions is a must needed feature of the system. Through the system, users are able to register for the system and apply to create bank accounts. Users can make transactions to other verified accounts and view the transaction logs he has made.

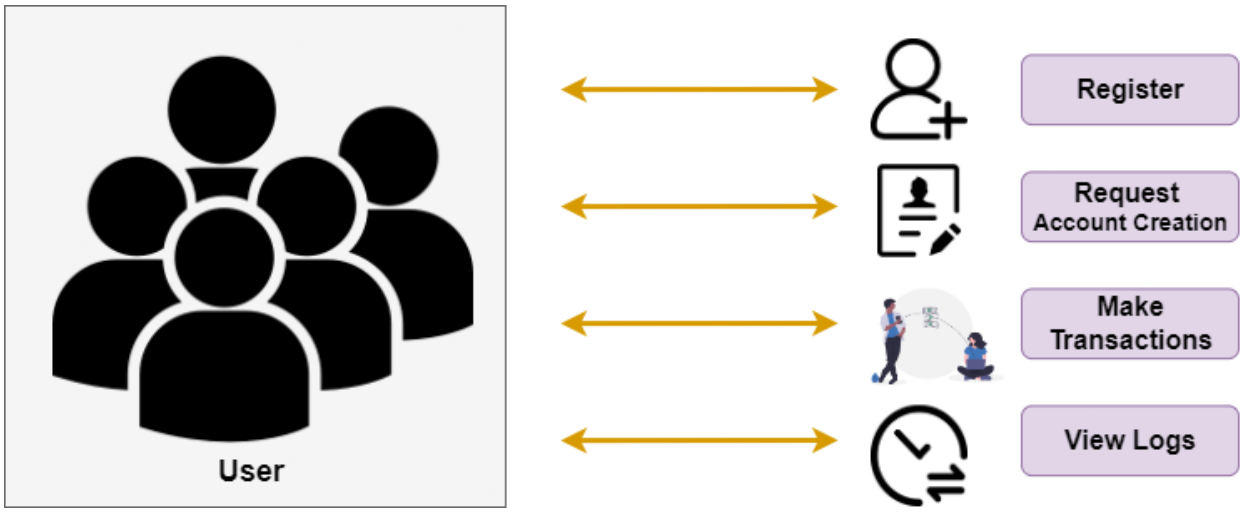

Figure 2. User Wallet UI

2. Bank Module UI

   Banks are the major monitoring and governing body of the system. First, the coin is issued by the banks and initial transactions are made to accounts of bank branches. From those accounts, now the digital coin is distributed to the users. Bank will get the following services in the system:

   a. Create Coin
   b. Make Transaction
   c. Validate Accounts
   d. Block Explorer

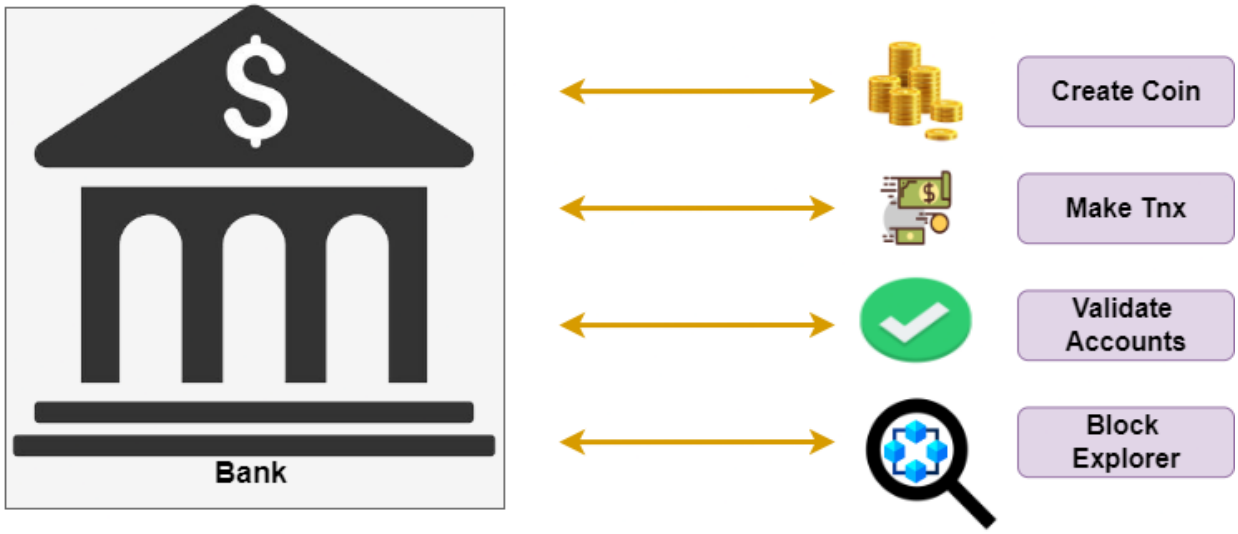

Figure 3. Bank module UI

3. Blockchain

   The system is based on blockchain technology. The blockchain type here is private blockchain maintained by the banks. Usage of blockchain enhances the security and transparency of blockchain. Blockchain is the part where every logic and service has been written. It handles JSON type block data to store data in an array called Chain. The block data once stored in data in a chain considering its timestamp, it can't be changed, which is an immutable property of blockchain.

4. Distributed System:

   The Distributed System in this project represents the network of different nodes connected to each other. As this is a centralized system, nodes are the multiple representatives of banks. We can consider nodes as the validators of banking systems. Each transaction is broadcasted over a distributed network and each node will store the copy of that transaction. Distributed system uses a Raft Algorithm based ETCD database which uses Level DB as its core database. ETCD is a CNCF (Cloud Native Computing Foundation) project. It stores data in key-value pairs. Distributed system is handled by the Bank and only the bank maintains these node servers.

Data Structure in Blockchain

Blockchain is a linear back-linked listed chain of multiple blocks and each block contains data, transactions and other details. Blocks are connected linearly and are secured cryptographically. Block contains multiple types of data to aggregate transactions to be added in the blockchain network. The table explains the block structure in blockchain:

Table 2. Data Structure of Blocks

Key	**Type**	**Purpose**
Block Height	Integer	Index number of block in chain
Transactions	Array	Array of Transactions
Size	Bytes	The size of the block, in bytes
Hash	AlphaNumeric	Hash of the current block
Parent Hash	AlphaNumeric	A reference to the hash of the previous (parent) block in the chain
Merkle Root	AlphaNumeric	A hash of the root of the merkle tree of this block's transactions
Timestamp	Integer	Time of creation of block
Block Data	String	Additional Data
Accounts	Array	Array of verified accounts added to the respective block

Summing up our Block Structure, it looks like:

```
Block {
blockHeight    :
transactions :          [
"id":
 "input": {
            "timestamp":
            "amount":
            "sender":
            "signature": {
              "r":
              "s":
              "recoveryParam":
            }
 },
 "outputs": [
            {
                  "amount":
                  "address":
            },
            {
              "amount":
              "address":
            }
          ],
 "remark":
 "hash":
 "blockNumber":
]
```

size :

hash :

parent_hash :

merkle_root :

timestamp :

blockData : “”

accounts : [0]

0 {}

name :

citizenshipNumber :

accountAddress :

role :

}

3.2 Context Diagram

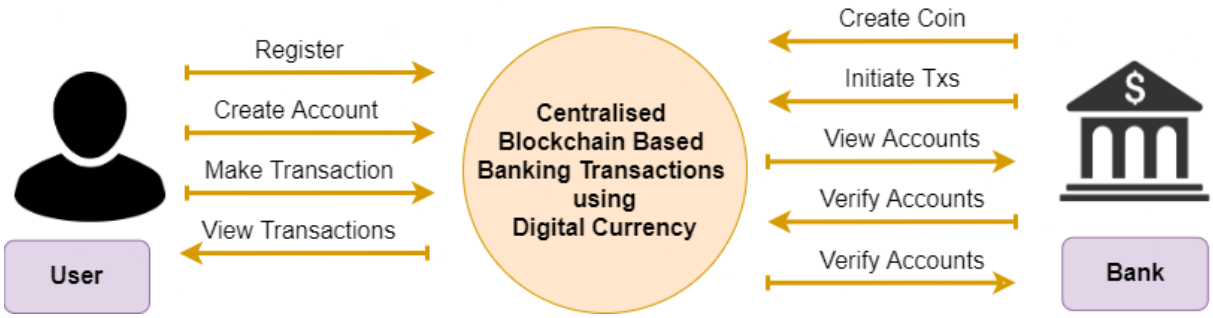

Figure 4. Context Diagram

Above diagram represents the overall architecture of the “Centralized Blockchain-Based Banking Transactions using Digital Currency”. Bank issues and manages coins and validates user’s accounts. Initial transaction has to be made by the Bank so that it can move the coins created to other accounts. Users can login and perform transactions (send/receive currency) via their respective address. First, the user needs to have an account or create an account. To make the account address workable, banks should verify the accounts first. A Blockchain network is used to store user data and user transactions. The wallet acts as a medium to connect users and system services (banking transactions).

3.2.1 User (Wallet Module):

Users are actors or ultimate consumers of the service. The features for users are: Creating account, Making transactions, Viewing remaining amount, Viewing Transaction History. In the issuer module, following processes are carried out:

1. User creates an account through the web app just like the bank account but it is generated through ECDSA algorithm that generates Private Key and Public Key. User needs to fill in his name, Citizenship No and other details. Users should keep the Private Key as there is no second method to generate again. Public Key will be his wallet address/account address.

   Eg:

   Name: **Balram Kunwar**

   Citizenship Number: **122-34-223**

   Account/Address : ***0x5CA9499c3Bf66F0DfcA00c3798B8c4E40D2E4ce1***

2. Now the account creation request is sent to the bank. Bank will get logs of account creation requests. Banks verify the accounts and add them to blockchain.
3. Once verified, User now can see the amount he holds
4. User can make transactions to another verified account
5. User can view the logs of previous transactions

3.2.2 Bank (Service Module):

Bank is responsible for maintaining transactions, users and services. The Centralised Bank manages the coins all over the territory. In Bank module , following work is done :

1. The Bank is responsible for creating coins/digital currency.
2. The Bank contains the digital money and can be sent to any verified account.
3. The account requests by users are verified or rejected from the bank side. The verified accounts are sent over the blockchain network
4. The verified accounts are now used for transactions by the users.
5. Every Transaction is automatically validated and mined by the system and distributed over the blockchain network.

3.3 UML Diagram

3.3.1 Use Case Diagram

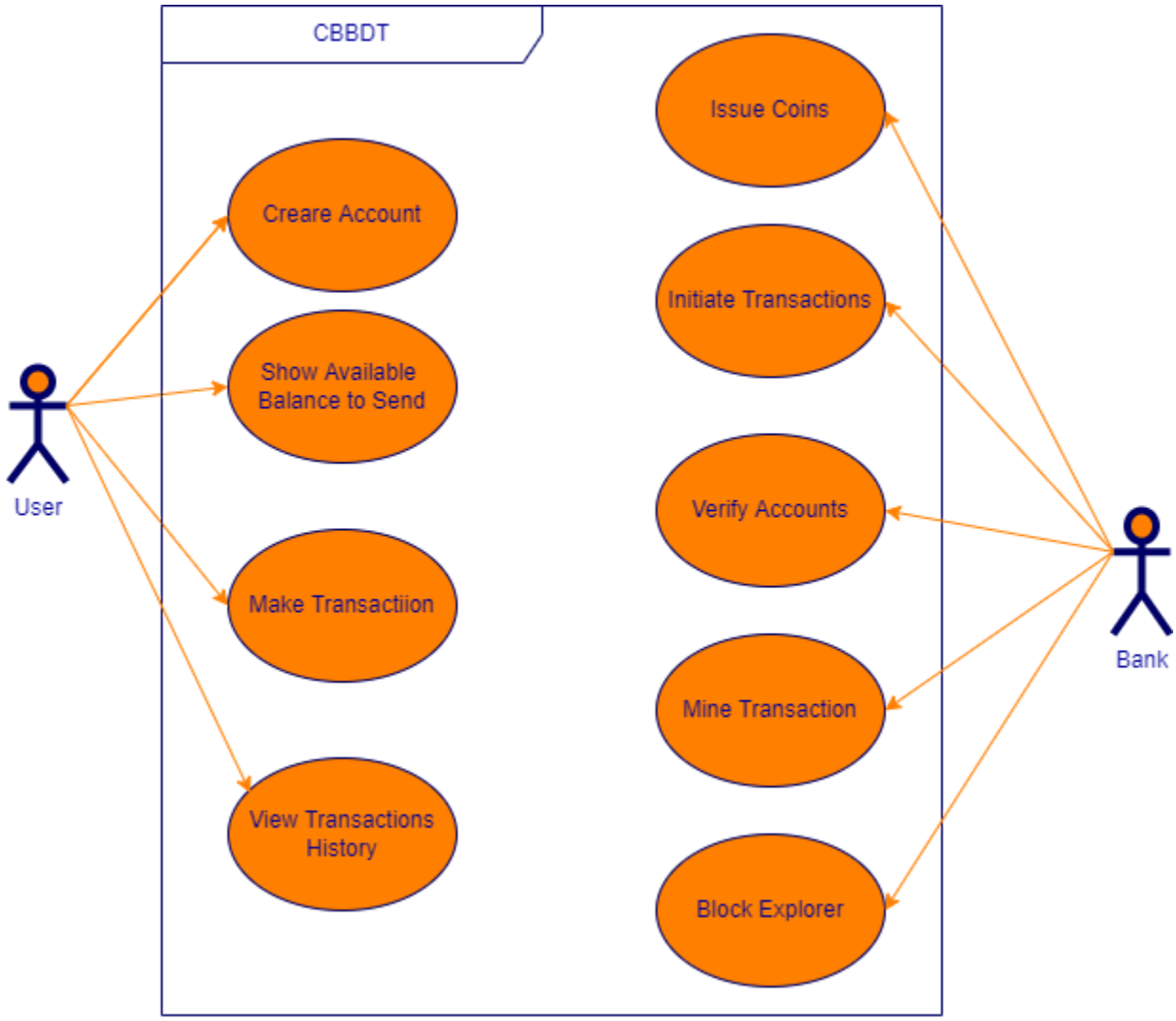

Figure 5. Use Case Diagram

The Bank generates and manages digital currency(or coin) which is used as a digital representation of fiat currency. It stores user-provided information and transactions in the blockchain network after verification. Banks manage the user accounts, observe the financial transactions and imply regulations on them. The first transaction in a blockchain is the amount generated by a bank. Secondly, the bank can transfer the amount into any verified accounts(we can take it as a transaction between central to other banks or branches). And after that, people using this service are now able to perform banking transactions.

Users or bank account holders make use of the system wallet to experience the services provided. Through a wallet, they are able to create an account online, check balance, and transfer the amount. They are also able to see the transaction history linked to their account.

3.3.2 Activity Diagram

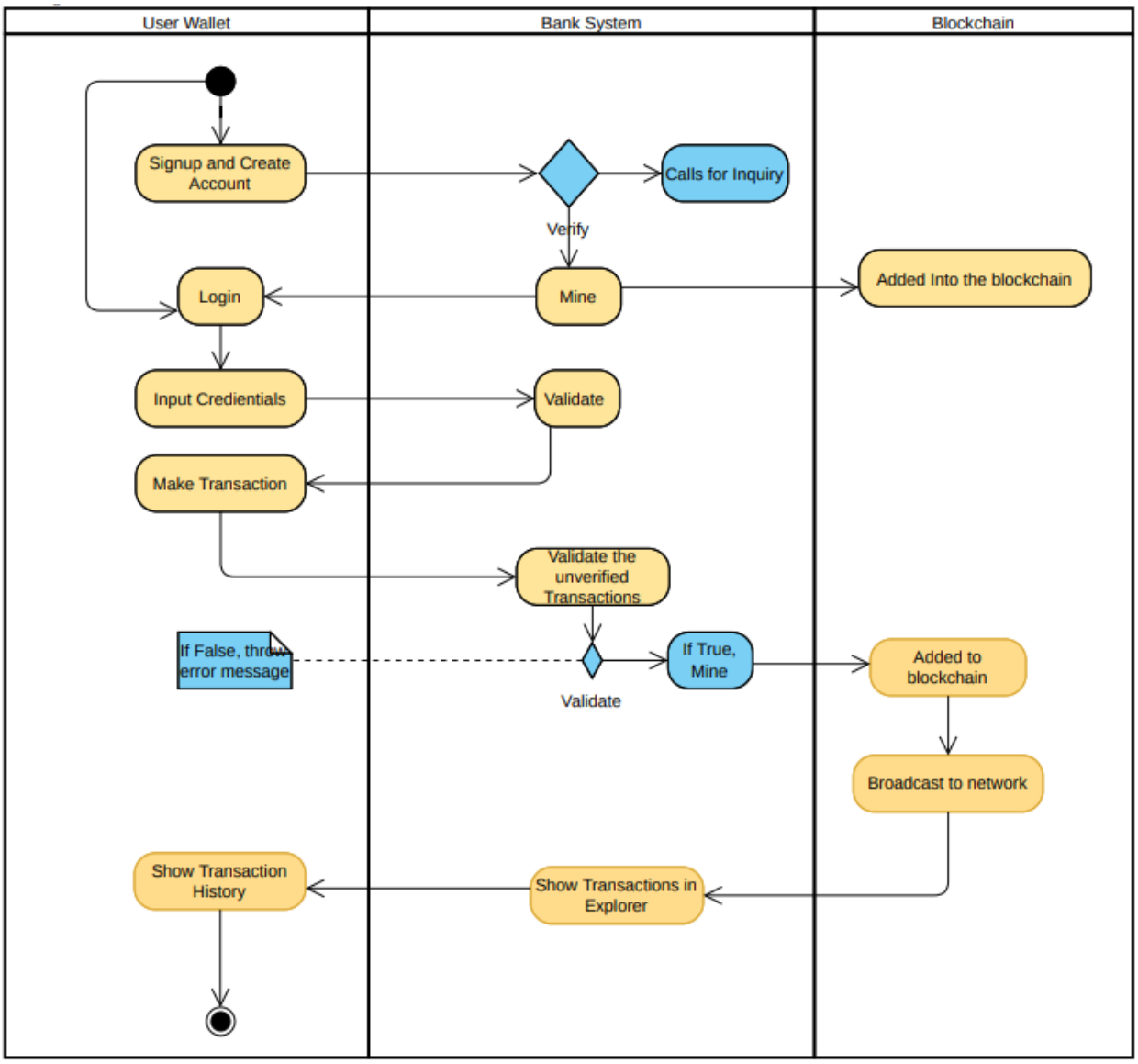

Figure 6. Activity Diagram

The Bank creates and maintains digital coins that serve as a digital substitute for fiat money. After being verified, it saves data and transactions supplied by users on the blockchain network. Banks oversee user accounts, keep track of transactions, and impose rules on them. The amount produced by a bank is the initial transaction on a blockchain. Second, any confirmed account can

get the funds from the bank (we can take it as a transaction between central to other banks or branches). After that, users of this service can now conduct financial transactions. The system wallet is used by users or people who have bank accounts to use the services. They can register for an online account and check their balance using a wallet. They also can view the transaction logs associated with their account.

3.4 Tools and Technologies

Blockchain:

Blockchain is a decentralized, distributed, and digital ledger consisting of records called blocks. The blockchain data structure is a time-stamped list of blocks, which records and aggregates data that have ever occurred within the blockchain network. The blockchain provides an immutable data storage container, which only allows inserting data without updating or deleting any existing data on the blockchain to prevent tampering. Every data recorded in the blockchain can be separately verified without any interruption by using its hash value, since it is open, can be publicly verified and the data once entered cannot be altered which helps in preventing forgery and misuse. In the blockchain, every block of data is linked to the previous block with the help of the hash value of the preceding block. The only genesis block which is the initial block does not have a hash of the preceding block. Blockchain has widely been popularized by crypto currencies like Bitcoin and people have misunderstood that Blockchain is Bitcoin. The fact is Bitcoin is only a use case of Blockchain. Blockchain is far more beyond crypto currencies and is expected to replace WEB at present with WEB 3.0 .

It could be of 2 types based on authority:

- **Centralized Blockchain,** In this type of blockchain network there is central authority that handles, controls and governs the network. There is involvement of third parties. It is intuitive and easy to use as it has a command chain, reduced cost and consistent output.
- **Decentralized Blockchain,** In this type of blockchain network there is no central authority that controls and governs the network. Every type of control stays with the user itself. The data in this network is immutable and highly secured.

It could be of 3 types based on permission to access the network.

- **Public Blockchain,** Public blockchain is an open network which can be accessed by every general public. Public blockchain is based on PoW consensus mechanisms and are open source and cannot be permissioned. Ethereum and Bitcoin are the best examples of it. Anyone participating in the network can read, write, and audit the activities.
- **Private Blockchain,** Private blockchain are generally used within an organization preventing others from accessing the network. Permissions are kept centralized. It lets organizations design and change network rules, architecture based on their necessity.In this blockchain, the write permission is kept centallized to an organization and read permission can be made public. Ripple, Multichain are examples of it.
- **Consortium Blockchain,** Consortium Blockchain are designed to have both Public and Private networks. Here, instead of a single organization, multiple organizations collaborate to take part in the network. It is a permissioned blockchain. Rubix and Hyperledger Fabric are some of its examples.

Table 3. Comparison of three kinds of Blockchain

Comparison	**Public Blockchain**	**Private Blockchain**	**Consortium Blockchain**
Openness	High	Low	Medium
Permissioned	No	Yes	Yes
Speed	Low	High	Medium
Scalability	Low	High	Medium
Decentralization	High	Low	Medium
Cost	Low	High	Medium
Throughput	Low	High	Medium
Example	Bitcoin, Ethereum	Ripple, Multichain	Fabric, Rubix

Central Bank Digital Currency:

CBDC is short for central bank Digital Currency, an electronic kind of central bank money that citizens can use to make digital payments and store value. they're based on the worth of that country's Fiat currency. Few among many countries implemented CBDC.The concept of

implementation of CBDC is based on both centralized as well as decentralized blockchain networks and can be implemented publicly and privately as well but not mandatory that it has to be on blockchain. Basically there are two types of CBDC.

- **WholeSale CBDCs,** these are similar to holding the reserve as a fund in the bank. The central bank grants an institution an account to deposit funds or settle monetary funds through interbank payments.
- **Retail CBDCs,** these are the government backed digital currencies which are used by consumers and businesses. Retail CBDCs eliminates the intermediary risk, the risk that private currency issuer might become bankrupt and lose customer assets.

Most of CBDCs projects are research based and are wholesale type CBDCs but our system is retail based CBDCs whose central authority is only with the central bank..

Table 4. A Summary of Blockchain based on CBDC

Project-Name	**CBDC types**	**Blockchain**	**Blockchain types**	**Use-case**	**Country/ year**
DNBcoin [7]	-	Bitcoin	Public	-	Dutch/2015
RScoin [8]	-	Bitcoin	Public	Domestic Payments	England/2016
CADcoin [9]	Wholesale	Ethereum/ Corda	Consortium	Interbank payments	Canada/2016
E-Krona [10]	Retail	Corda	Consortium	Domestic retail and interbank	Swedish/ 2018
Project Inthanon [11]	Retail and wholesale	Corda, Hyperledger	Private	Interbank-settlements	Thailand/ 2018
Project LionRock[12]	Retail and Wholesale	Corda	Consortium	Interbank Settlements	Hk, China/ 2018

Nonce:

It stands for “Number only used once”. It is a 32-bit (4-Byte) number added to the hash of the block or the encrypted block in blockchain, when rehashed meets the difficulty level restriction. These are the numbers that miners need to solve inorder to get rewarded from the network for

adding the valid block to the blockchain network. Nonce are used as the number that is used to verify the transaction and other data contained in the block. It is used on authentication protocol and cryptographic hash function. The basics of nonce shows a lot about the importance of nonce in blockchain as miners won't be able to complete the transaction and add blocks to the network without nonce. In blockchain, the miners perform lots of hash functions with many different nonce values until a valid output is obtained. The probability of guessing the right nonce at once is almost zero and miners use a trial and error approach (the approach which is characterized by repeated varied attempts which is continued until success) in which every hash calculation function takes new values of nonce.

Merkle Tree:

Merkle tree also referred to as 'binary hash tree' is a data structure that verifies the integrity of transactions stored in the blockchain, with fewer amounts of data. Leaf node of the merkle tree is labeled with the cryptographic hash of the data block and the non-leaf node is labeled with the cryptographic hash of the child nodes. It is a mathematical data structure made up of hashes of various data blocks that summarizes all the transactions in the block and are used to encrypt the blockchain data more efficiently and securely. It helps in quick and secure data verification across big datasets in the blockchain network and verifies the validation of data in the blocks.

Merkle root is a mathematical method that confirms all the facts of a merkle tree. They are used in cryptocurrency to ensure that the data blocks sent through the P2P network are whole.

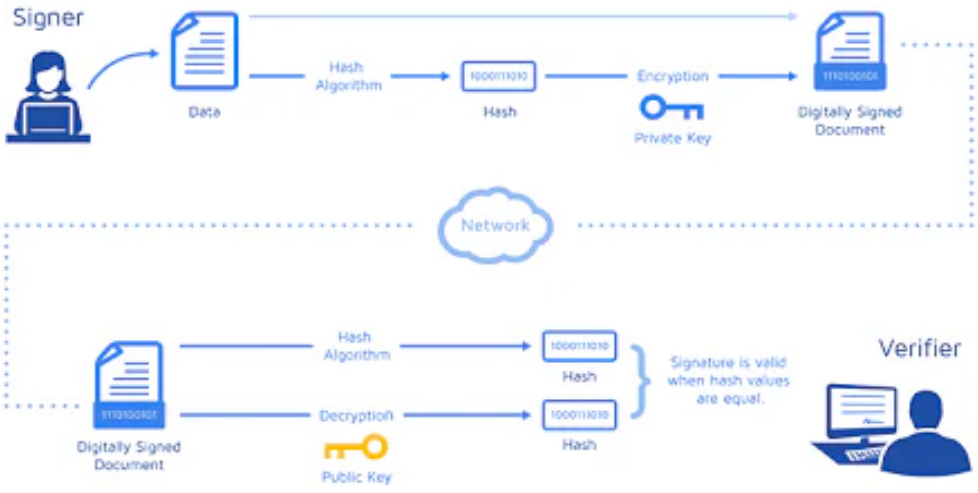

Figure 7. Working of Merkle Tree [13]

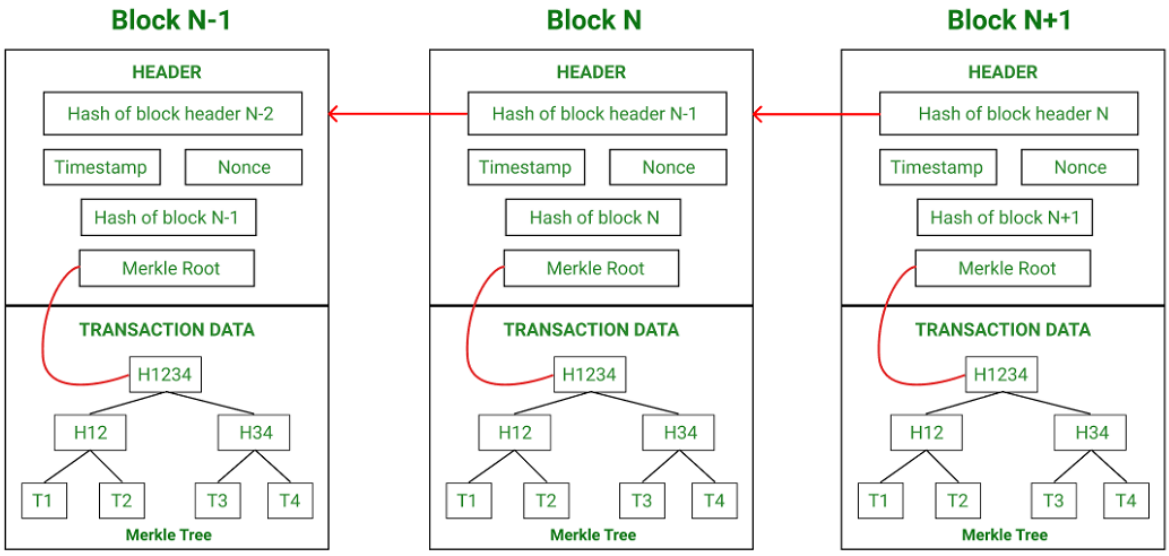

Figure 8. Block comprising of Block header and merkle tree [13]

A merkle tree totals all the transactions in a block and generates digital signatures through encrypted hash value of the entire set of operations in the block, allowing the user to verify whether it includes transactions in the block. Merkle trees are made by hashing pairs of nodes repeatedly until only one hash remains and this hash is merkle root or root hash. They are built from the bottom using the transaction hash of individual transactions. Each non leaf node is the hashes of its previous hash and every leaf node are hashes of its transactional data.

Let us have a look at examples considering the following scenario in which there are four transactions occurring in the same block. Let those four transactions be A, B, C and D. When these transactions are hashed we get the hashes of the transaction as Hash A, Hash B, Hash C and Hash D for transaction A, B, C and D respectively. The Hashes are then paired together as Hash A pairs with Hash B and Hash C gets paired With Hash D. Thus formed hashes pairs when combined together form merkle root with Hash ABCD.

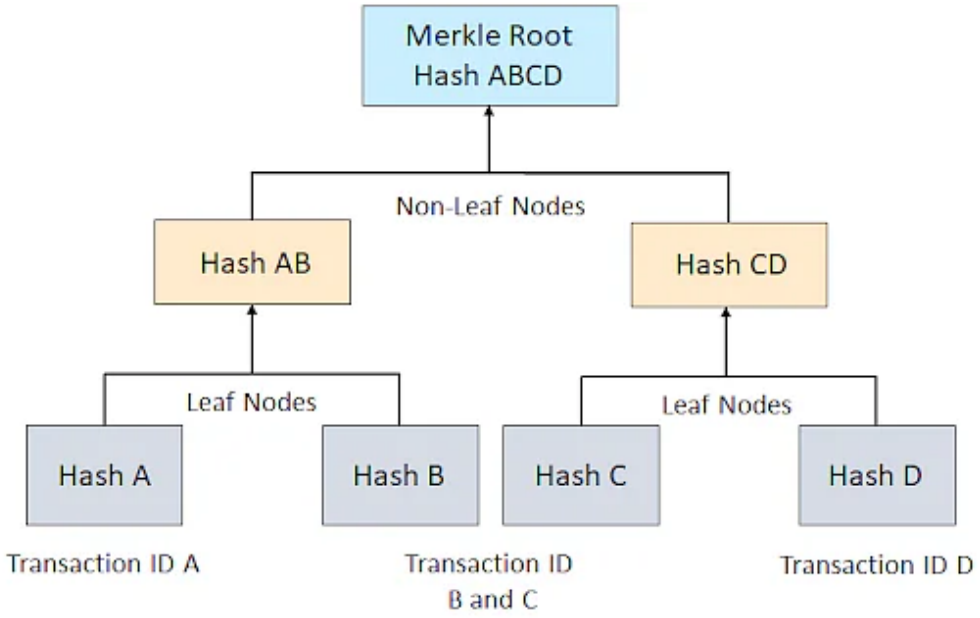

Figure 9. Merkle tree for transaction A, B, C and D [13]

Consensus In Blockchain:

Consensus is a process of agreement between nodes in a final state of data. In order to achieve consensus, different algorithms are used and it is easy to reach consensus between two nodes (in Client-Server system) but when multiple nodes are participating in a distributed system, they need to agree on a single value and it becomes difficult to achieve consensus. This method of achieving consensus in between multiple nodes is considered as distributed consensus.

Consensus mechanism is a set of protocols or steps that are taken by all or most nodes in order to agree on a proposed value. Consensus mechanism has gained much popularity with the advent of Bitcoin and blockchain and makes sure that nobody in the network spends the same money twice as it only allows legitimate transactions in the network.There are certain requirements which must be met in order to get desired results from consensus mechanisms. Following are the requirements:

- **Agreement:** All the honest nodes in the network should agree on the same value.
- **Termination:** All the honest nodes in the network should terminate the execution of the consensus process and reach the consensus decision.

- **Validation:** All the honest nodes in the network should agree on the same value and these values should be the same as proposed by at least one of the honest nodes.
- **Fault Tolerance:** Consensus algorithms should be able to run in the fault nodes.
- **Integrity:** No nodes in the network should be able to make the decision more than once. The nodes can make decisions only once in a single consensus cycle.

Types of Consensus in Blockchain:

- **Proof-of-Work(PoW):**

  Proof-of-work is a type of consensus mechanism which relies on the proof that enough computational resources have been spent before proposing a value for acceptance by the network. This mechanism is used in Bitcoin and other cryptocurrency and this algorithm is pretty much successful against different attacks.PoW consensus is based on SHA-256 hash and was first introduced as part of Bitcoin. PoW consensus mechanism involves solving the complex mathematical problem inorder to create a new block in the blockchain network and this process is called as mining.There needs a lot of computational power and energy to solve these mathematical problems. Moreover, the involvement of computational power in the network makes the blockchain secure. Process of verifying the transaction and adding it to the block, organizing the blocks and broadcasting it over the network doesn't take much time and energy. When the miner finds the right solution to problems then the node broadcasts it to the whole blockchain network and the miner receives a cryptocurrency reward based on the PoW protocol.

- **Proof-of-Stake(PoS):**

  It is an alternative to Proof-of-work consensus. PoS consensus is used in ethereum. The PoS algorithm works on the idea that a validator or node has enough stake in the system and they don’t need to invest in the expensive hardware to solve the complex mathematical problems. The validator stakes their coin in order to get a chance to validate a block and coin owners with staked coins become validators. Validators are selected randomly to mine or validate the block. The blocks are validated by more than one validator and when a specific number of validators verifies that the block is valid and

legitimate, the block is finalized and added to the network and the validators are rewarded inreturn of validating the block.

- **Proof-of-Authority(PoA):**

  PoA is a consensus mechanism that provides high performance and fault tolerance. In Proof-of-authority, rights to generate and add new blocks to the network are provided to the node who have proven their authority to do so. These nodes are known as validators and they work on systems that allow them to put transactions into the blocks. PoA is best suited for both private and public blockchain networks, like PoA networks where trust is distributed. In PoA consensus mechanism, it leverages a value of transaction IDs which means that validators are not staking their coins but their own authority and reputation instead. PoA is secured by trust on the identities selected. Here the validation process is automated and does not require validators to be constantly monitoring their computer but it does require maintaining the computer uncompromised. The interval of time at which new blocks are generated and added to the network is predictable and it does not require nodes to spend computational resources for solving mathematical problems.

- **Raft Consensus Algorithm:**

  The Raft algorithm was developed by Diego Ongaro and John Ousterhout. Raft is a distributed consensus algorithm which was designed to solve the problem of getting multiple servers to agree on a shared state even in the face of failure. Shared state is a data structure which is supported by replicated logs and the system needs to be fully operational as long as the majority of servers are up. Raft works by electing a leader in a cluster of servers and the leader is responsible for managing the replication of logs and for accepting the clients requests. The data flow is one-directional and flows from the leader to servers.

  Raft breaks down consensus into three sub-problems:

  - Leader Election: If one of the existing leader fails, a new leader needs to be elected.
  - Log Replication: The leader should keep the logs of all servers synchronized with its own log through replication.

- Safety: If any one of the servers has committed log entry at a particular index then no other servers can apply for the log entry at the same index.

  Election safety ensures that at most one leader can be elected in a given term. A leader only appends new entries but never overwrites new entries. Log matching, leader completeness and state machine safety etc, Raft ensures that these properties are true every time.

Raft Basics: Raft may have several servers and each server may exist in either of the states, that is, Leader, follower and candidate. Normally there is only one leader in the server for a certain term number and all other servers are followers. Followers in the servers remain passive and it only responds to the request of the leader and candidate. The leader controls all the client requests and if the client requests a follower then followers redirect to the leader. Candidate is used to select the leader if the leader fails or the term number is over through the involvement of followers.

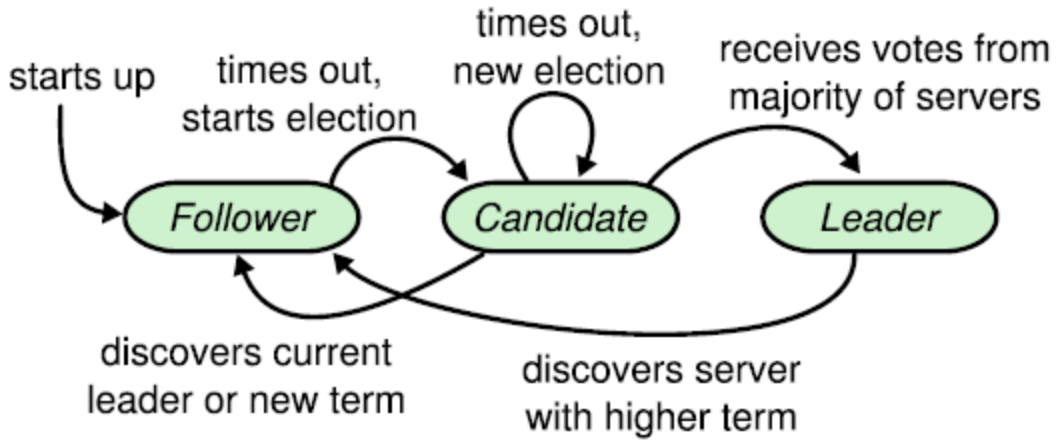

Figure 10. Server States transition [15]

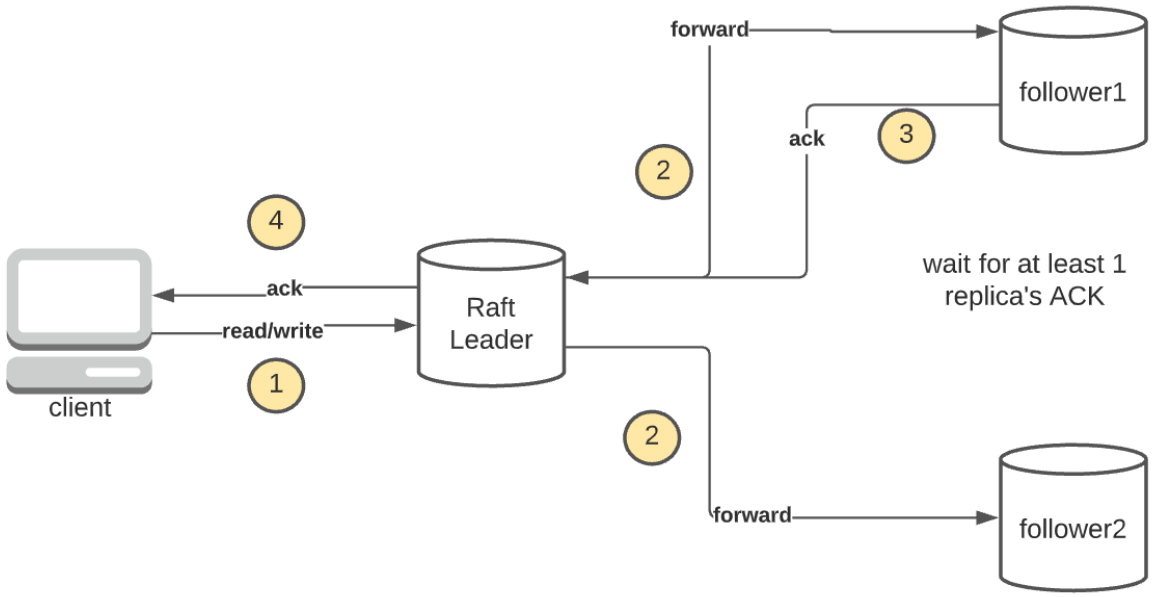

Figure 11. Working mechanism of Raft Algorithm [15]

Cryptography:

Cryptography is a process of developing techniques and protocols to prevent the third party to gain the access of the data or private messages that have to be transmitted so that only the person for whom it is intended to send can read. It is a practice and study of technique that prevents the public from reading private messages. Various aspects in security of information that are the central to cryptography such as:

- Data Confidentiality: Cryptography process makes sure that data sent through the network is confidential and are not accessible to any third parties.
- Data Integrity: Integrity means that data sent through the network has not been tampered.
- Data Authentication: Authentication of data means to make sure that data are from the trusted source and integrity of data is also validated.
- Non-repudiation: It refers to the process which guards against any denial of message sent by the sender with the help of digital signature. Since every input message has its own digital signature which prevents collision and differentiate it from other messages

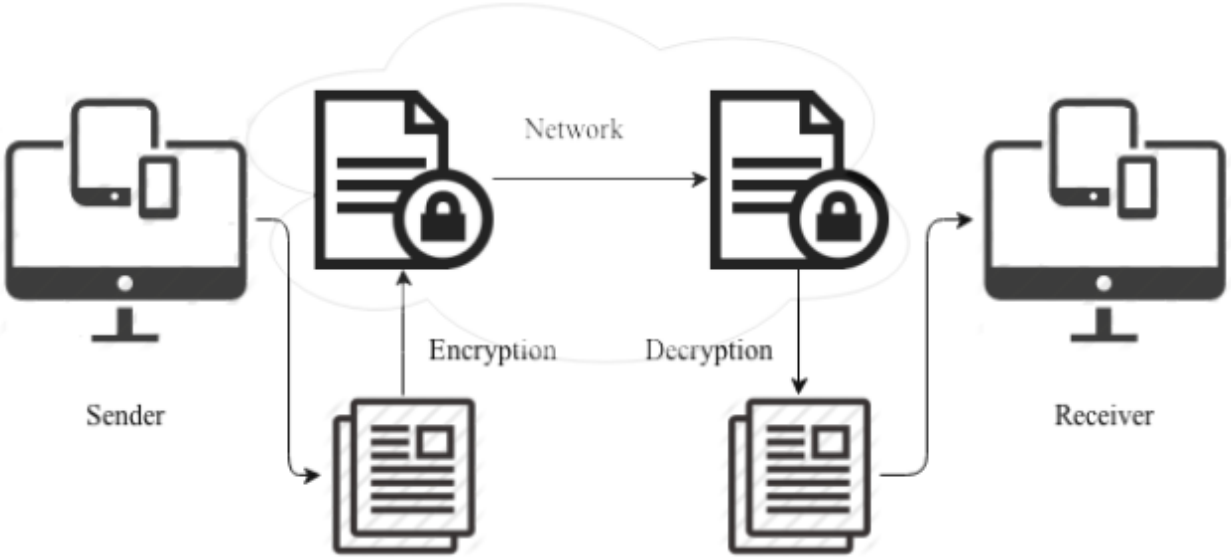

Figure 12. Cryptography [16]

Terms that are related to cryptography are:

- Plain Text: These are the data that can be read and are understandable by anyone accessing it.
- Encryption: It is the process of converting the plain text into some sequence of bits that are not accessible and understandable by the readers unless they have the decryption key. Encryption is done by the help of some mathematical function.
- Decryption: It is the process of converting back the random sequence of bits to original plain text. It is the inverse process of encryption.
- Cipher text: Cipher text is the random sequence of bits which are generated when plain text is encrypted with the help of mathematical function.
- Key: Key is the information which helps in conversion between plain text and cipher text. Basically there are 2 types of keys in cryptography and they are:
    - Public key: This key can be shared with the person you want to.
    - Private key: This key cannot be shared and should be kept private and is only accessible by the owner.

- Hash: It is a value which is generated by the cryptographic hash function. It is used to map data of arbitrary size to fixed-size values.

Cryptography can be classified into two types:

- **Symmetric-key Cryptography:**

  It is a type of cryptography algorithm in which encryption and decryption can be done by the use of the same cryptography key. This kind of encryption algorithm relies on a single key that has been shared between the sender and receiver. The same key is used to encrypt the plain text into cipher text and the same key is used to decrypt the cipher text into original plain text.

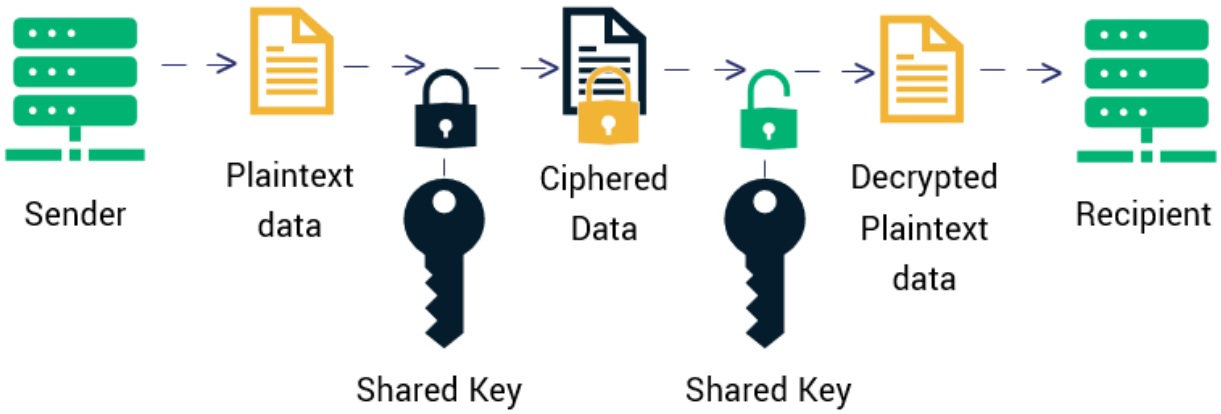

Figure 13. Symmetric-key Cryptography [16]

- **Asymmetric-key Cryptography:**

  It is also known as the public key encryption algorithm. It is a type of encryption algorithm in which the plaintext is encrypted with the help of a public key and the ciphertext is decrypted back to the original plaintext with the use of a private key.

  Here the sender encrypts the plaintext with the help of the receiver's public key and the encrypted ciphertext is transmitted over the network. The receiver can only decrypt the ciphertext back to the original plain text with his private key which should be kept secret and should not be shared with anyone.

Figure 14. Asymmetric-key Cryptography [16]

Hashing:

Hashing is the process of scrambling or converting the raw data to the extent that it cannot be reproduced back to the original form. Hashing is performed by the hash function which takes the data as value and converts it into some other encrypted form by performing some mathematical calculations in data using hashing function.The hash functions are those functions which map data of arbitrary size to fixed value sizes. The result obtained after hashing of data is known as hash value/digest or hash. The hash values/digest is a small value that is used to represent a large piece of data in a hash function. The good hash functions are those which use a one-way hashing algorithm which means the hash cannot be converted back to original value. Hashing algorithms must possess an avalanche effect which means if a single word from the original data is changed then the hash value changes significantly or completely and the algorithm should be complex enough so that the same hash value is not generated for two different data. Hashing differs from encryption in that hashing is a one way process and encryption is a two way process.

Digital Signature:

Digital signature is a technique which is based on a public-key cryptography algorithm. However there is a difference between public key cryptography and digital signature. In public key cryptography, public key is used for the encryption of data and private key is used to decrypt the data but in case of Digital signature, private key is used for encrypting the message and public key is used for decrypting the message. Only the person who has the private key can

encrypt the data or sign the data. Senders can share their public key with the person with whom they want to communicate the data and the receiver can decrypt the data or validate the data.

The private key is considered as a signing key while the public key is considered as verification key. The signer feeds the data into a hash function which in return generates the hash value of data and the hash value along with the private key is processed with a signature algorithm which produces a digital signature of the given hash. Thus the produced signature is appended with the data and is sent to the verifier. Now the verifier feeds the digital signature along with the verifier key(public key) to the verification algorithm and it generates some values as output. The verifier uses a hash function on received data to generate the hash and compares it with the hash values of the output from the verification algorithm. If both the hash values match then the signature is valid otherwise not valid. In digital signature, instead of signing data directly by algorithm usually a hash of data is created and this hash is signed in place of data. RSA (Rivest-Shamir-Adleman) , DSA (Digital Signature Algorithm), ECDSA(Elliptic Curve DSA) etc are some widely used digital signature algorithms.

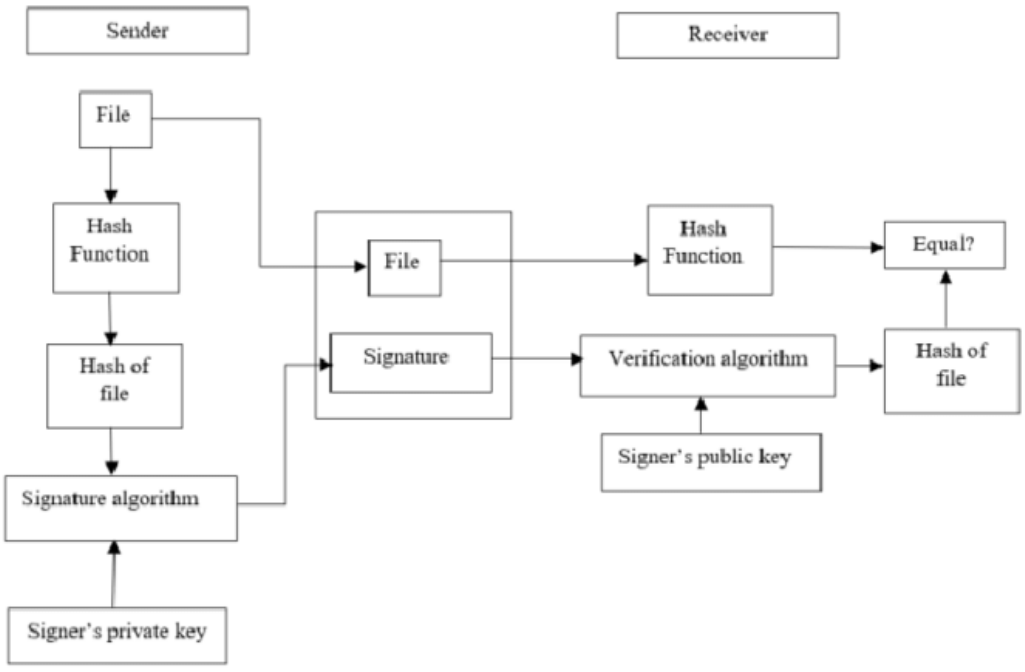

Figure 15. Working mechanism of Digital signature [17]

ECDSA:

Elliptical Curve Digital Signature Algorithm is an algorithm which is used for the creation of digital signatures based on the concept of mathematical elliptical curve. This algorithm presents the proof for the ownership of a private key without revealing the private key. It uses Elliptical-curve cryptography (ECC). ECC is a type of asymmetric-key cryptography based on the logarithmic problems as expressed by addition and multiplication on the points of an elliptical curve.

Data when feeded into the hashing algorithm, generates the hash of the data. Then the hash of the data is encrypted with the signer's private key and digital signature is generated. Thus generated digital signature is appended to the original data and is sent to the verifier.

The verifier again generates the hash of the original data using the same hashing algorithm as used by the signer. The digital signature of the sent data is decrypted using ECDSA decryption algorithm along with the use of signer's public key which gives a hash value of the digital signature as output. Now if the hash value of the data and digital signature after decryption matches then the verifier validates the data and if not the data is not validated.

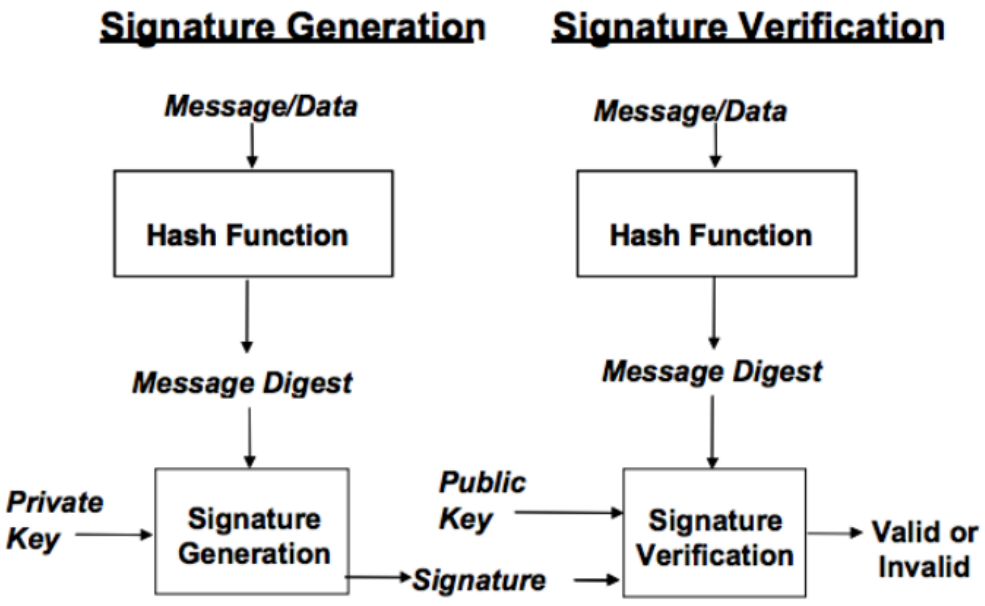

Figure 16. Working mechanism of ECDSA [14]

SHA-256:

SHA stands for secure hash algorithm and SHA-256 is part of SHA-2 algorithm's family. The 256 in SHA means final hash digest value which is of 256bits long irrespective of size of data which may be plaintext or ciphertext as well. It is called a cryptographic hash function which runs on sequence of mathematical operational functions on digital data which is to be hashed.

It has some standout features and they are:

- Message Length: The message length of the plaintext should be less than 264 bits and the size should be in comparison to keep digest values as random as possible.
- Digest Length: In SHA-256 algorithm, the length of the hash digest should be of 256-bits.
- Irreversible: The hash function SHA-256 algorithm is irreversible in nature. It means we neither get plaintext when we have the digest value beforehand nor the digest provides its original value when we pass it through a hashing function again.

The steps involved in SHA-256 algorithm are:

- Padding Bits: The total length of data to be hashed should must be multiple of 512 and can be made by adding 64 bits of data. In this process, we add bits to one end of the data so that the final length of data must be 64-bits less than multiple of 512. The padding bits can be calculated by formula:

  D + P = (n*512) - 64

  Where, D = data length,

  P = Padding length,

  n is a constant.

- Padding Length: It is obtained by taking modulus of original data with 2^{23} to get 64-bits of data. Appending this one to our data makes our data exact multiple of 512. The final data after appending consists of original data, padding bits and padding length bits.
- Buffer initialization: First some buffers are initialized which by default 8 buffers values are initialized which are hard coded constant values representing the hash values. After initialization, computations are carried out on the data.

- Compress function: Here the final data obtained after appending bits are broken down into multiple blocks of 512-bits in each block. Each block has to go through 64 rounds of operation with the output of each block which serve as the input for the following blocks.

Firebase Authentication:

Firebase authentication is a service that is being provided by firebase which helps us to build user authentication systems for web as well as mobile applications. It creates identity for the users which helps them to authenticate themselves by using email/ password or by using some social application. It is important and essential for the system to only allow the authenticated user to access the system to avoid private data stealing and firebase authentication provides all the server side things for authenticating the users. Firebase authentication allows users to authenticate using email, phone number and social sites like Google, Facebook, Twitter etc.

Whenever the user try to sign in or log in into the system, we first get authentication credentials from the users which are then sent to the firebase authentication software development kit, Then after backend services verify those credentials and return a possible response to the client.

Software Development Process

The project is developed based on the Software Development Life Cycle. The SDLC for the project development is based on the Incremental Model which is used to develop the system by developing multiple working prototypes throughout the project. Testing and Debugging is done in every prototype to develop a better product.

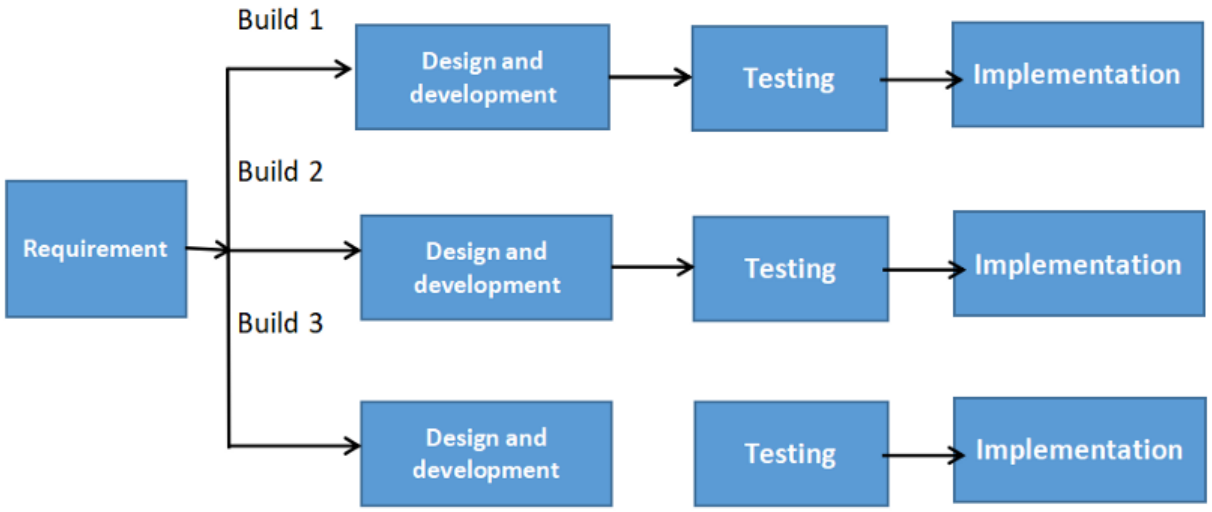

Figure 17. Incremental Model

Scrum(Agile framework) was used for development and daily meetings to discuss the problems occurring while developing a project and solve those issues. Daily scrum meetings were conducted through proper planning and reviewing. Backlogs were the major focus. As scrum provides opportunities for inspection and adaptation, this framework has been beneficial. The reasons behind using this model are as follows:

- Since the product required changes throughout the development process and agility supports changes
- A functioning early production of product can be developed
- Major function of the project can be seen early
- Initial development costs (resources and efforts) were very less for the development

- Each release is a product increment, so that there is a working product at hand all the time
- Feedback were needed for the improvement in every build and this SDLC supports feedbacks
- Requirements changes can be easily accommodated

Prior to the Build 1, research and analysis on Blockchain Technology and implementation on Banking Systems, study of previous works on related topics and literature study were done. The results of different builds of our SDLC were :

- Build 1 : Creation Of Block and Blockchain
- Build 2 : Backend Services
- Build 3 : Decentralization, Frontend, Backend optimisation, Integration

Each individual component went through analysis, design, implementation, testing, deployment and maintenance.

3.5 Dev Tools and Dependencies

Table 5. Tools and dependencies and their version

Tools and Dependencies	Version	Used For
body-parser	1.20.0	• npm library used to process the data that is sent through an HTTP request body
cors	2.8.5	• supports secure cross-origin requests and supports the data transfers between browsers and servers.
crypto-js	4.1.1	• CryptoJS is a growing collection of secure cryptographic algorithms in JavaScript. It contains best practices and usage.
elliptic	6.5.4	• Elliptic contains cryptographic algorithms that are used by cryptocurrencies like Bitcoin to ensure that funds can only be spent by their owner
etcd3	1.1.0	• Used to interact with etcd endpoints, • Etcd is used to manage data over a distributed network. Etcd is based on raft consensus
express	4.18.1	• a layer built over the top of the Node js that helps to manage the servers and routes

jest	29.0.1	• Jest is a testing framework of javascript to ensure Javascript code is correct
mongodb	4.10.0	• It is a non relational database used to store email and public key of users in this project
mongoose	6.6.1	• Mongoose is a popular ODM (Object Data Modeling) library for node.js and MongoDB
nodemon	2.0.19	• Nodemon automatically restarts the node application whenever it observes any changes in any file present in the current working directory
uuid	8.3.2	• Uuid was used for identifying information that needs to be unique within our system or network
react	18.2.0	• Popular front-end JavaScript library for building user interfaces
axios	0.27.2	• Axios is a promise based HTTP client for web browsers and Node. js
firebase	9.10.0	• A BaaS app development platform for developing apps that offers hosted backend services such as a real-time database, cloud storage, authentication, crash reporting, machine learning, remote setup, and hosting

		is known as a backend-as-a-service (BaaS).
material-ui/core	4.12.4	• It allows us to import and use different components and templates to create a UI in our React applications
Hashing algorithm	SHA-256	• It is a Cryptographic hash algorithm that produces irreversible and unique hashes

3.6 Project Task And Time Schedule

The project schedule was designed as per the project requirements and constraints involved. This project was scheduled to be completed within about 3 months. Requirement analysis was given more emphasis and research on BLOCKCHAIN and implementation on Banking Systems was done first. Debugging and Testing was done during each iteration and prior to the completion of the project.

Table 6. : Project Task And Time Schedule

Task	***Approx time (days)***
Project Planning	4
Literature Study	9
Requirement Analysis and Specification	12
Analysis of the System	7
Design and Development of System	49
Testing and Debugging	11

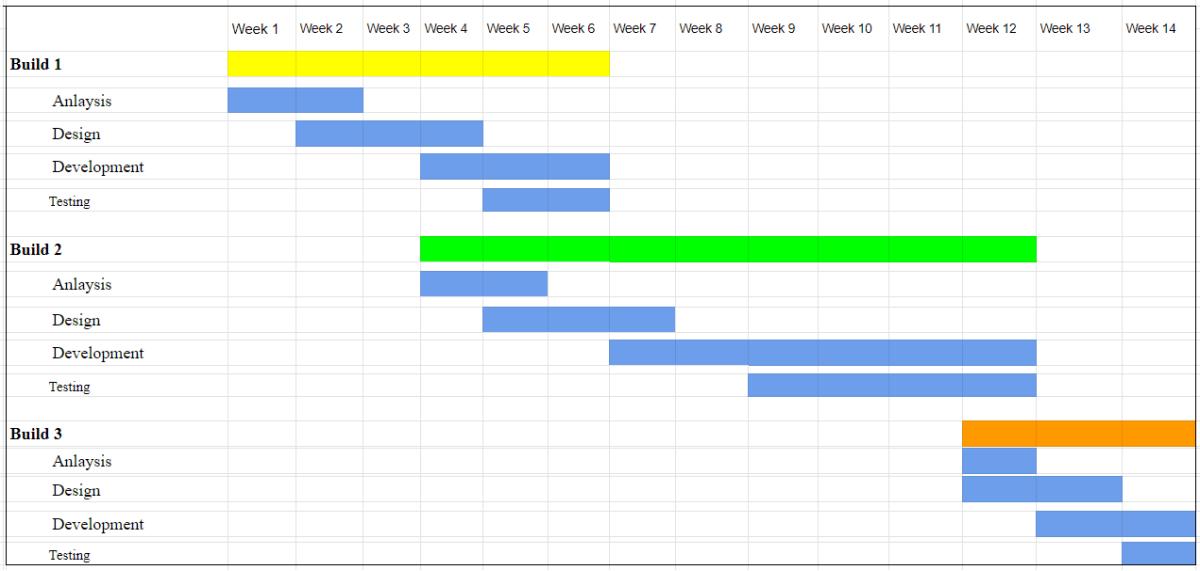

Figure 18. Gantt Chart

4. Application Areas

‘Centralized Blockchain Based Banking Transactions using Digital Currency’ is a proof of concept project revolving around the usability of blockchain in banking transactions using digital currency. One of our major objectives in this project was to represent Blockchain and digital currency as a future potential to upgrade existing banking systems. Major application areas of this project are:

1. A digital currency representation to a fiat currency system
2. Monitoring and traceability of transactions
3. Secured System design backed by blockchain technology
4. Scalable Banking System
5. Enhanced transparency and security

5. Limitations and Future Works

This system has tried to eliminate the limitations as much as possible and tried to develop this application as a suitable solution to the modern problem of using physical fiat currency. However there are some limitations that still prevail and will be working in the future. Some limitations that are considerable of this project are:

1. Difficult to attain widespread adoption.
2. Digital currencies can also increase the risks of system-wide bank runs. Such types of bank runs could increase faster in times of financial crisis without any dependence on time and proximity.
3. Digital currencies are acceptable only in the country that issues them.
4. The system services cannot be used in internet connection unavailability.

Future Works:

1. Study and implementation of other consensus algorithms
2. Distribution

6. Conclusion

In the modern global context everything is digitizing and there is a huge concern and debate on adoption of blockchain technology in the banking system. However the characteristics of blockchain are suitable for requirements of banking design with digital currency. In this project, we make a literature review on different approaches and study implementations of blockchain. However, this concept is still not adopted by central banks because there are some challenges to be solved in blockchain based banking digital currency. There are a lot of challenges to be solved like cross-chain interoperability, scalability, security and performance. There are some countries who have initiated their steps towards the CBDC concept and are working on it to completely convert their cash flow system to digital currency. There are certain things that should be considered whether the network should be kept private or public or both which are only decidable by the central bank in the monetary policy which can be defined when CBDC are initiated.

However, our work on Banking Transactions with blockchain can be a Proof Of Concept and can provide some guidelines for blockchain based digital transactions.

References

[1] S. Haber and W. S. Stornetta, “How to time-stamp a digital document,” J. Cryptology, vol. 3, no. 2, pp. 99–111, 1991.

[2] S. Nakamoto, “Bitcoin: A peer-to-peer electronic cash system,” 2008.

[3] “OpenCBDC,” MIT Digital Currency Initiative. [Online]. Available: https://dci.mit.edu/opencbdc?fbclid=IwAR37qY9JKlnS_TNI_gVBJd_jh_wvXJiZPvygwdBFR1f2GrOJrQUSl6O1sfE. [Accessed: 10-May-2022].

[4] C. Briefing, “The digital yuan app - all you need to know about the new E-CNY tool,” China Briefing News, 13-Apr-2022. [Online]. Available: https://www.china-briefing.com/news/china-launches-digital-yuan-app-what-you-need-to-know/. [Accessed: 20-May-2022].

[5] Matthieu Bouchaud, Tom Lyons, Matthieu Saint Olive, Ken Timsit, “Central banks and the future of digital money,” Consensys.net, Jan-2022. [Online]. Available: https://pages.consensys.net/central-banks-and-the-future-of-digital-money. [Accessed: 20-Sep-2022].

[6] “Central Bank Digital Currency (CBDC) Definition,” Investopedia.com, 19-Mar-2022. [Online]. Available: https://www.investopedia.com/terms/c/central-bank-digital-currency-cbdc.asp. [Accessed: 20-May-2022].

[7] D.N. Bank, “DNBCoin/Dukaton,” Dnb.nl. [Online]. Available: https://www.dnb.nl/binaries/Jaarverslag%202015_tcm46-339389.pdf.[Accessed: 15-Jul-2022].

[8] G. Danezis and S. Meiklejohn, “Centrally Banked Cryptocurrencies,” Ucl.ac.uk, 26-May-2015.[Online].Available:http://www0.cs.ucl.ac.uk/staff/G.Danezis/papers/ndss16currencies.pdf. [Accessed: 25-Jun-2022].

[9] “Digital currencies and fintech: projects,” Bankofcanada.ca. [Online]. Available: https://www.bankofcanada.ca/research/digital-currencies-and-fintech/projects/.[Accessed:22-Jun-2022].

[10] “E-Krona,”Riksbank.se.[Online].Available: https://www.riksbank.se/globalassets/media/rapporter/e-krona/2019/the-riksbanks-e-krona-pilot.pdf. [Accessed: 25-Jul-2022].

[11] “Project-Inthanon,” Bot.or.th. [Online]. Available: https://www.bot.or.th/English/FinancialMarkets/ProjectInthanon/Documents/Inthanon_Phase2_Report.pdf. [Accessed: 20-Jul-2022].

[12] “Inthanon-LionRock,” Gov.hk. [Online]. Available: https://www.hkma.gov.hk/media/eng/doc/key-functions/financial-infrastructure/Report_on_Project_Inthanon-LionRock.pdf. [Accessed: 07-Jul-2022].

[13] Simplilearn, “Merkle Tree in Blockchain: What is it and How does it work,” Simplilearn.com,03-Nov-2021.[Online].Available:https://www.simplilearn.com/tutorials/blockchain-tutorial/merkle-tree-in-blockchain. [Accessed: 10-Jul-2022].

[14] Nicolascourtois.com, 2022. [Online]. Available: http://www.nicolascourtois.com/bitcoin/thesis_Di_Wang.pdf. [Accessed: 22- Sep- 2022].

[15] D. Ongaro and J. Ousterhout, “In search of an understandable consensus algorithm (extended version),” *Github.io*. [Online]. Available: https://raft.github.io/raft.pdf. [Accessed: 22-Sep-2022].

[16] J. Thakkar, “Types of encryption: What to know about symmetric vs asymmetric encryption,” InfoSec Insights, 25-Apr-2020. [Online]. Available: https://sectigostore.com/blog/types-of-encryption-what-to-know-about-symmetric-vs-asymmetric-encryption/. [Accessed: 01-Sep-2022].

[17] “Digital signature and digital certificate,” Exam Nights Live, 01-May-2017. [Online]. Available:https://examnightslive.wordpress.com/2017/05/01/digital-signature-and-digital-certificate/. [Accessed: 01-Sep-2022].

[18] *Researchgate.net*. [Online]. Available: https://www.researchgate.net/figure/Signing-and-verification-phases-of-ECDSA_fig1_316174537. [Accessed: 01-Sep-2022]..

[19] N. Dashkevich, S. Counsell, and G. Destefanis, “Blockchain application for central banks: A systematic mapping study,” IEEE Access, vol. 8, pp. 139918–139952, 2020.

[20] J. Lundqvist, “Private blockchains in financial services,” PaymentsJournal, 29-Jun-2021. [Online].Available:https://www.paymentsjournal.com/private-blockchains-in-financial-services/. [Accessed: 21-May-2022].

[21] The Harvard Law School Forum on Corporate Governance, “Blockchain in the banking sector: A review of the landscape and opportunities,” Blockchain in the Banking Sector: A Review of the Landscape and Opportunities, Jan. 2022.

[22] T. Zhang and Z. Huang, “Blockchain and central bank digital currency,” ICT Express, vol. 8, no. 2, pp. 264–270, 2022.

[23] The Blockchain Show, “A private key in the context of Bitcoin is a secret number that allows bitcoins to be spent.#Cryptocurency #bitcoin #Ethereum #nftcollector pic.twitter.com/pk9d9eyojw,” Twitter, 20-Aug-2021. [Online]. Available: https://twitter.com/. [Accessed: 01-Sep-2022]

[24] Euromoney, “Blockchain Explained: The rise of private blockchains,” Euromoney. [Online].Available:https://www.euromoney.com/learning/blockchain-explained/the-rise-of-private-blockchains. [Accessed: 20-May-2022].

[25] “Central bank digital currencies: foundational principles and core features,” Bis.org. [Online]. Available: https://www.bis.org/publ/othp33.pdf. [Accessed: 10-Jul-2022].

Appendix A

Block Chain:

Chain in block is the list of blocks starting from very first block called ***genesisBlock,*** followed by numbers of other blocks with hash of previous blocks included in it.

[genesisBlock, block1 block2, …]

Block:

Block stores transactions, accounts with other useful informations such as ***blockHeight, timeStamp*** - time of the block creation, ***size*** of block, ***merkle_root, block_hash, parent_hash*** etc. The block structure in this system is stored as,


```
{
  "blockHeight": 132,
  "transactions": [
   {
    "id": "d4d3d8f0337f11ed877aed1cea93021c",
    "input": {
     "timestamp": 1663085964287,
     "amount": 50000,
     "sender":
"04eea9237e54cd2a0ede29a37a3865dc0685ccad6d04415b720b4ee2bed74577118b86223d2365ba9f5c5745d27566
8e5c836e17f48a12a5ccd951889fa46e813a",
     "signature": {
      "r": "5525a074ecb017f1c09ed3ffe479769b4c9fdb9aeb7a630ae358a1bb363c1709",
      "s": "a827a02c26316dfa1ee5fc11dca265d8d7b0504dbe2b1df70c9bca2c424bc42f",
      "recoveryParam": 1
     }
    },
    "outputs": [
     {
      "amount": 20000,
      "address":
"04eea9237e54cd2a0ede29a37a3865dc0685ccad6d04415b720b4ee2bed74577118b86223d2365ba9f5c5745d27566
8e5c836e17f48a12a5ccd951889fa46e813a"
     },
     {
      "amount": 30000,
      "address":

"049028f154b124f8f6122229358173c231210b4920584f8c5cb7b72d086fe9cd7e0e906e55c8e71714676a0245c184be
8174664083a4c01d3096b6188fc845bc00"
     }
    ],
    "remark": "This is transaction by Bank",
    "hash": "82ba233aac14648af2fa1ac560b0f058913b4164779fafffc905a556ac6d3053",
    "blockNumber": 132
   }
  ],
```

```
  "size": "1589 Bytes",
  "hash": "a8c579e3b6e98f7840178f9280800c673e7f4962e0040da0f2dc0743c8a37582",
  "parent_hash": "b9c539e3c6e68f7740118f2280800c673e7f4962e0040da0f2dc0743c8a3b3h2",
  "merkle_root": "82ba233aac14648af2fa1ac560b0f058913b4164779fafffc905a556ac6d3053",
  "timestamp": 1663969907296,
  "blockData": "This is CBBDT server mining this block.",
  "accounts": [
   {
     "name": "Ram Bahadur Karki",
     "citizenshipNumber": "197-123123",
     "accountAddress":
"04205ba9287444a8f93eb96e838f75ee92d618eac1b39764e982565c2b7edfdc72c047b7786169ce583e5c88c88d46ef
d765d68d6a27f7135126968721c423cffc",
     "role": "USER"
   }
  ]
}
```


Accounts:

Accounts holds list of all accounts associated with the system and remains embedded inside respective block,

[account1, account2, account3, …]

Account:

Account is the data of every verified account users of this blockchain system,


```
{
    "name": "Ram Bahadur Karki",
    "citizenshipNumber": "197-1223",
    "accountAddress":
"04205ba9287444a8f93eb96e838f75ee92d618eac1b39764e982565c2b7edfdc72c047b7786169ce583e5c88c88d46ef
d765d68d6a27f7135126968721c423cffc",
   "role": "USER"
}
```


Transactions :

It is the list of all transactions, placed in blockchain. These are verified and successful transactions only.

[transaction1, transaction2, transaction3, ……]

Transaction:

Transaction in this system is stored as,


```
{
  "id": "d4d3d8f0337f11ed877aed1cea93021c",
  "input": {
    "timestamp": 1663085964287,
    "amount": 50000,
    "sender":
"04eea9237e54cd2a0ede29a37a3865dc0685ccad6d04415b720b4ee2bed74577118b86223d2365ba9f5c5745d275668e5c836e17f48a12a5ccd951889fa46e813a",
    "signature": {
      "r": "5525a074ecb017f1c09ed3ffe479769b4c9fdb9aeb7a630ae358a1bb363c1709",
      "s": "a827a02c26316dfa1ee5fc11dca265d8d7b0504dbe2b1df70c9bca2c424bc42f",
      "recoveryParam": 1
    }
  },
  "outputs": [
    {
      "amount": 20000,
      "address":
"04eea9237e54cd2a0ede29a37a3865dc0685ccad6d04415b720b4ee2bed74577118b86223d2365ba9f5c5745d275668e5c836e17f48a12a5ccd951889fa46e813a"
    },
    {
      "amount": 30000,
      "address":
"049028f154b124f8f6122229358173c231210b4920584f8c5cb7b72d086fe9cd7e0e906e55c8e71714676a0245c184be8174664083a4c01d3096b6188fc845bc00"
    }
  ],
  "remark": "This is transaction by Bank",
  "hash": "82ba233aac14648af2fa1ac560b0f058913b4164779fafffc905a556ac6d3053",
  "blockNumber": 132
}
```

Appendix B

MATHEMATICS OF ECDSA (ELLIPTIC CURVE DIGITAL SIGNATURE ALGORITHM)

ECDSA is a cryptographically secure digital signature algorithm. It is based on elliptic curve cryptography (ECC).

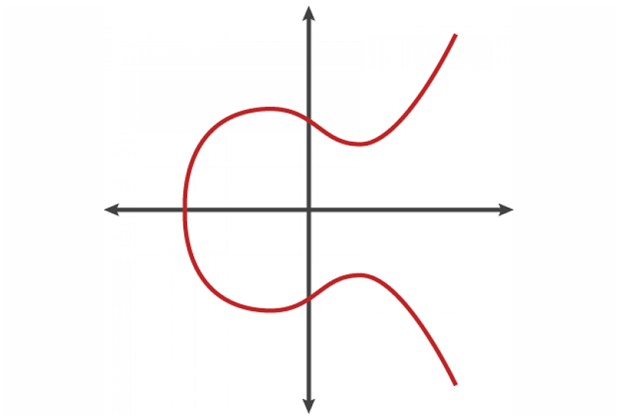

Figure 19. Elliptic Curve

Elliptic curves, used in cryptography define,

- **Generator point G**, used for scalar multiplication on the curve (multiply integer by EC point)
- **Order *n*** of the subgroup of EC points, generated by **G**, which defines the length of the private keys (e.g. 256 bits)

- **Key Generation**

  ECDSA has key pairs as,

  > private key (integer) : **Pk**, a randomly generated integer in the range **[0 … n-1]**

  > public key (EC point) : ***Pu = Pk * G*** , **G** is the generator point of a subgroup of large prime order **n**.

- **ECDSA Sign**

  Signing algorithm takes input messages (**msg**) and **Pk** to produce **signatures**. Signature

consists of a pair of integers **{r, s}**. ECDSA signing algorithm works as,

i) Calculate message hash using hash functions like SHA-256,

$$h = hash(msg)$$

ii) Generate a random number **k** in range **[1 …...n-1]**

ii) Calculate signature proof:

$$s = k^{-1} * (h + r * Pk) \pmod n,$$

$\mathbf{k^{-1}}$ **(mod n)** is an integer satisfying $k * k^{-1} \equiv 1 \pmod n$

iii) Return signature **{r, s}.**

Calculated signature **{r, s}** encodes random point **R = k*G,** with proof **s,** confirming the signer knows the message and private key. Proof **s,** is verifiable using corresponding **Pu.**

- **ECDSA Verification**

Verification algorithm takes signed message **msg** , signature **{r, s}** produced from signing algorithm and **Pu** corresponding to signer's **Pk**.

i) Calculate message hash, with hash function used during signing,

$$h - hash(msg)$$

ii) Calculate modular inverses of signature proof,

$$s^1 = s^{-1} \pmod n$$

iii) Recover random point used during signing:

$$R' = (h * s^1) * \mathbf{G} + (r * s^1) * Pu$$

iv) Take from R' its x-coordinate: $r' = R'.\mathbf{x}$

v) Calculate the signature validation result by comparing as, $r' == r$

In general, verification recovers R' using **Pu** and check with **R**

- **Public Key Recovery**

With message **m** and signature **r, s** public key can be recovered.

i) Verify **r** and **a** are integers in **[1, n-1]**

ii) Calculate curve point, ***R = (x1, y1)*** where **x1** is one of **r, r+n, r+2n ……**

iii) Calculate ***h = hash(msg)***

iv) Calculate ***u1 = -z r^{-1} (mod n)*** and ***u2 =sr^{-1} (mod n) ,*** where z = leftmost bit of h

v) Calculate curve point

Pu = (xA, yA) = u1 * G + u2 * R

Pu is the recovered public key.

This way ECDSA works to provide double key encryption.

Appendix C

SHA-256 , HASH GENERATING ALGORITHM

Hashing algorithm takes an input to return hashes or message digest. Let's consider the message '**msg**' being inputted to the hash algorithm. The hash generation steps for **msg** can be summarized as,

- **Pre-Processing,**

i) Message imputed is converted to binary using ASCII**.**

'DCVS' = 01000100 01000011 01010110 01010011

ii) **1** is appended to resultant data at the end.

phrase = 01000100 01000011 01010110 01010011 1

iii) Phrase size is calculated and converted to binary, **41 = 101001**

iv) Phrase is the padded up with '**0**' and phrase size is appended at the end to make size **512 bit** long,

010001000100001101010110010100111000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000
00000000101001

v) Block is then divided to **16** words of **32-bits** each,

0) 01000100010000110101011001010011

1) 10000000000000000000000000000000

.

.

.

15) 00000000000000000000000000101001

Let's represent this block as,

block 1 = M(1)

and, word as,

M(i)(b) where, **b** is block number and **i** refers to the specific in the block and **i** refers to word.

- **Initial Hash Values**

8 words of length **32-bit** Initial hash values **(H^0)** are defined after generating from the **first 8 primes.** These hashes should remain the same. Primes are firstly **square rooted** and **modulus 1** is taken and the result is then multiplied by 16^8 and rounded down to the nearest integer.

p = 2,3,5,7,11,13,17,19

initial hash value = int((√p mod 1) * 16^8

Results from above is converted to hexadecimal form, giving **8** words assigned from **a-h**

$a = H_0^{(0)} = 6a09e667$

$b = H_1^{(0)} = bb67ae85$

$c = H_2^{(0)} = 3c6ef372$

$d = H_3^{(0)} = a54ff53a$

$e = H_4^{(0)} = 510e527f$

$f = H_5^{(0)} = 9b05688c$

$g = H_6^{(0)} = 1f83d9ab$

$h = H_7^{(0)} = 5be0cd19$

The process continues for every block.

- **Hash Computation**

64 words are required thus, 48 more words have to be computed resulting in 4 blocks. The words can be generated using the equation,

$$W_t = RotL\ (\sigma_1(W_{t-2}) + W_{t-7} + \sigma_0(W_{t-15}) + W_{t-16}) \quad where,$$

$$\sigma_0(x) = RotR^7(x) \oplus RotR^{18}(x) \oplus SHR^{10}(x)$$

$$\sigma_0(x) = RotR^{17}(x) \oplus RotR^{19}(x) \oplus SHR^{10}(x)$$

$$t=16 \text{ for } 17th \text{ word}$$

Equation finds word **15** places back **(W(1))**, and **2** copies made. First copy is Right Rotated by **7** places **(RotR^7(x))**, i.e. each digit is moved to the right one place **7** times and moved front after number falls to end.

Now, original is right shifted by **3** **($SHR^3(x)$).** If the number falls off the end it is replaced by zeros at the beginning of the word. Then these **3** words and $\sigma_0(W_1)$ computed as,

$$\sigma_0(x) = RotR^7(x) \oplus RotR^{18}(x) \oplus SHR^3(x) \text{ where,}$$

$$x = W_{t-15} = W_1 \text{ and } \oplus = \text{bitwise XOR}$$

Result is then converted to hexadecimal form. The process is repeated until all **64** words are computed.

Final Hash is then generated by the equation below:

For t = 0 to 63:

{

$$T_1 = h + \Sigma_1^{\{256\}}(e) + Ch(e,f,g) + K_t^{\{256\}} + W_t$$

$$T_2 = \Sigma_1^{\{256\}}(a) + Maj(a,b,c)$$

$$h = g$$

$$g = f$$

$$f = e$$

$$e = d + T_1$$

$$d = c$$

$$c = b$$

$$b = a$$

$$a = T_1 + T_2$$

}

where,

$$Ch(e, f, g) = (e \wedge f) \oplus (\neg e \wedge g)$$

$$Maj(a,b,c) = (a \wedge b) \oplus (a \wedge c) \oplus (b \wedge c)$$

$$\Sigma_1^{\{256\}}(e) = RotR^6(e) \oplus RotR^{11}(e) \oplus RotR^{25}(e)$$

$$\Sigma_1^{\{256\}}(a) = RotR^2(a) \oplus RotR^{13}(e) \oplus RotR^{22}(a)$$

$$K_t^{\{256\}} = A\ set\ constant\ in\ the\ Bitcoin\ code$$

$\neg$ ***, Bitwise NOT***

$\wedge$ ***, Bitwise AND***

Values generated after every **63** iterations, intermediate hash value **H(i) , ith** intermediate hash can be generated as,

$$H_0^i = a + H_0^{(0)}$$
$$H_1^i = b + H_1^{(0)}$$

$$H_2^i = c + H_2^{(0)}$$

$$H_3^i = d + H_3^{(0)}$$

$$H_4^i = e + H_4^{(0)}$$

$$H_5^i = f + H_5^{(0)}$$

$$H_6^i = g + H_6^{(0)}$$

$$H_7^i = h + H_7^{(0)}$$

a-h are values from final iteration **(t(63))** and **H(0)(0)** to **H(7)(0)** are initial hash values.

These obtained intermediate hash values intare converted into hexadecimal form, giving the final message digest. Appending values together with the final message digest is obtained.

$$M = H_0^N \,||\, H_1^N \,||\, H_2^N \,||\, H_3^N \,||\, H_4^N \,||\, H_5^N \,||\, H_6^N \,||\, H_7^N$$

This way data is cryptographically hashed using SHA-256 hashing algorithm.

Appendix D

Mathematical Solution to Time Stamping Problem

Haber and Stornetta intended to introduce a mathematically sound and computationally practical solution to the time-stamping problem. A TSS can be made trusted using Collision Free Hash and Signature.

- **Hash** : Hash functions compress bit-strings of arbitrary length to bit strings of a fixed length *l*, and it is easy to pick a member of the family at random. It is computationally infeasible, given one of these functions,

  $$h : \{0,1\}^* \rightarrow \{0,1\}^l$$

  to find a pair of distinct strings x, x' satisfying $h(x) = h(x')$. Such a pair is called collision for h. The hash functions are one way functions and it is infeasible to compute another string x' given string x with $x \neq x'$ satisfying,

  $$h(x) \neq h(x'), \quad \text{for a randomly chosen } h.$$

  Hash function in the paper has been used to send hash value, $h(x) = y$ of

  document to TSS with y as time-stamping equivalent to time-stamping x .

- **Signature** : Signature scheme is an algorithm for a party, the signer, to tag messages in a way that uniquely identifies the signer. One way functions can be used to design a signature scheme satisfying the very strong notion of security that was first defined by Goldwasser, Micali, and Rivest. TSS receives a hash value, appends data and time, then signs the document and sends it to the client using signature. TSS assures client checking the signature, ensures hash was correctly received and correct time has been included.

Since, neither the signature nor hash functions could prevent issuing false time-stamping, a further mechanism was suggested. The legitimate time-stamp of a Algorithm A with document x, and timing information T in bit string can be written as,

$$c = A(x,T)$$

The above c is to be prevented from being forged to produce same time information T , and same certificate c later. For this two approach was proposed,

i) Linking : This approach is to constrain a centralized but possibly untrustworthy TSS to produce genuine time-stamps, in such a way that fake ones are difficult to produce. If bits from the previous sequence of client requests in signed certificates are included in present then we know time-stamp occurred after these requests. Linking leaves no room to insert a new document with forged timing information to the service, making it ineligible.

Let $\sigma(.)$ be procedure to sign document by TSS where, it issues signed and sequentially numbered time-stamp certificates. H, be a hash function, l-bit string y be time-stamping request and ID as client id number. Also, client request be, $(\gamma n, IDn)$, in nth request. TSS

- *Sends signed certificate*, $S = \sigma(Cn)$ *where,* $Cn = (n, tn, IDn, yn; Ln)$
  Here, Ln is the linking information, coming from certificates previously issued.

  $Lo = (t(n-1), ID(n-1), \gamma(n-1), H(n-1))$.
- *TSS sends* $ID(n+1)$ *for the next request*.

Client checks signature S, and $ID(n+1)$ on arrival. Timestamp of the document can be verified as $(S, ID(n+1))$ for any collision between client and TSS. The challenger of time-stamp can also call $ID(n+1)$ to produce their time-stamp, $(S', ID(n+2))$ and check copy of yn in $L(n+1)$ and authenticate by including $H(Ln)$ of linking information to verify with $L(n+1)$.

The paper also suggests another variation of linking, in which each request is linked to next k requests.

- *Linking information,* Ln *is* , $Ln = [(t(n-k), ID(n-k), \gamma(n-k), H(L(n-k))) \ldots (t(n-1), \gamma(n-1), H(L(n-1))]$.
- *Tss sends a client list* $(ID(n+1), \ldots, ID(n+k))$ *after next* k *requests have been processed.*

ii) Distributed Trust : The distributed trust approach is to distribute the required trust among the users of service. When a client needs to time-stamp, hash value y can be used as seed for a pseudorandom generator, whose output can be interpreted in a standard as a ***k-tuple*** of client ID numbers, $G(y) = (ID1, ID2, ..., ID(k))$. Client then sends a request (y, ID) to each client. Sending client receives a signed message, $\sigma j = (t, ID, y)$. The final time-stamp of the client now consists, $[(y, ID), (S1, \ldots, S(k)]$. This scheme suggests the number of dishonest clients is always less than the majority of clients and suggests choosing seeds to be impossible to find.

This paper concludes by giving the idea that, if both the schemes/approaches be used together can be a secure practical solution to time-stamping of digital documents. This way it gave rise to block chain technology.

Appendix E

Figure 20. Landing Page

User Home Page

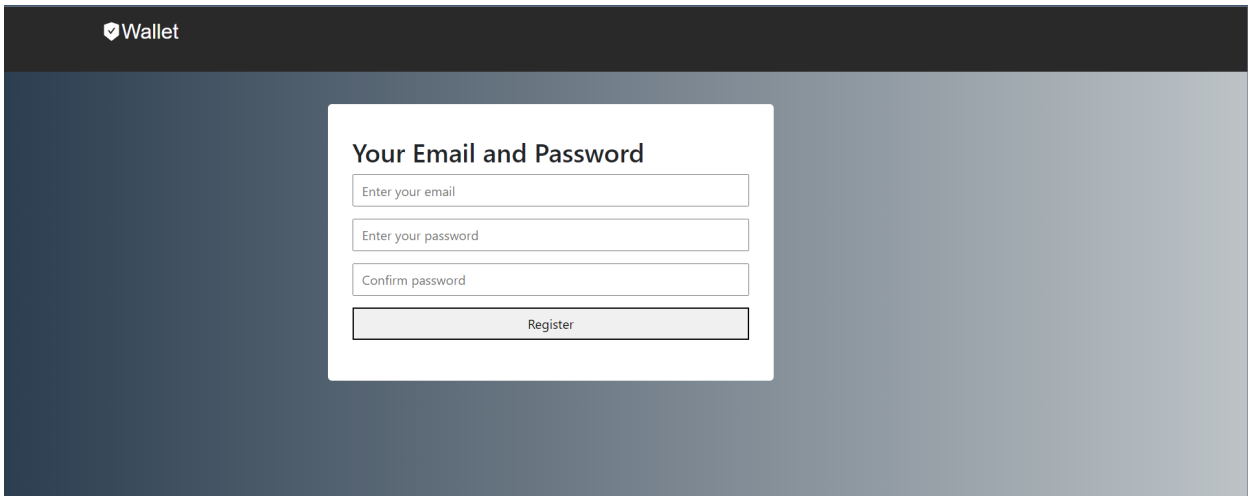

Figure 21. Email Verification

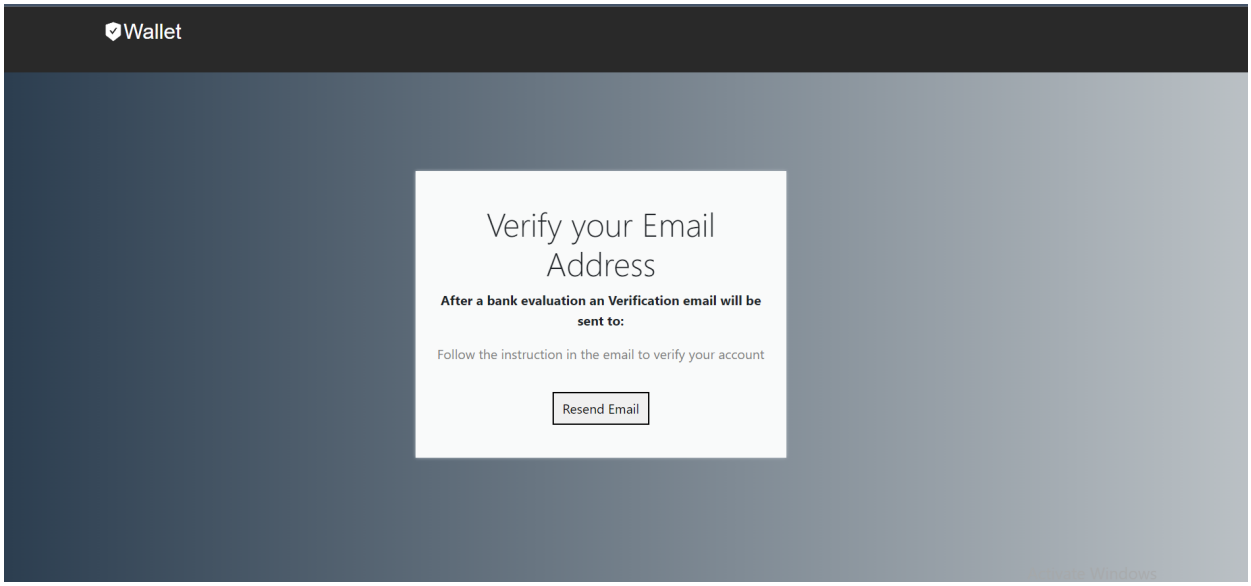

Figure 22. Email validation

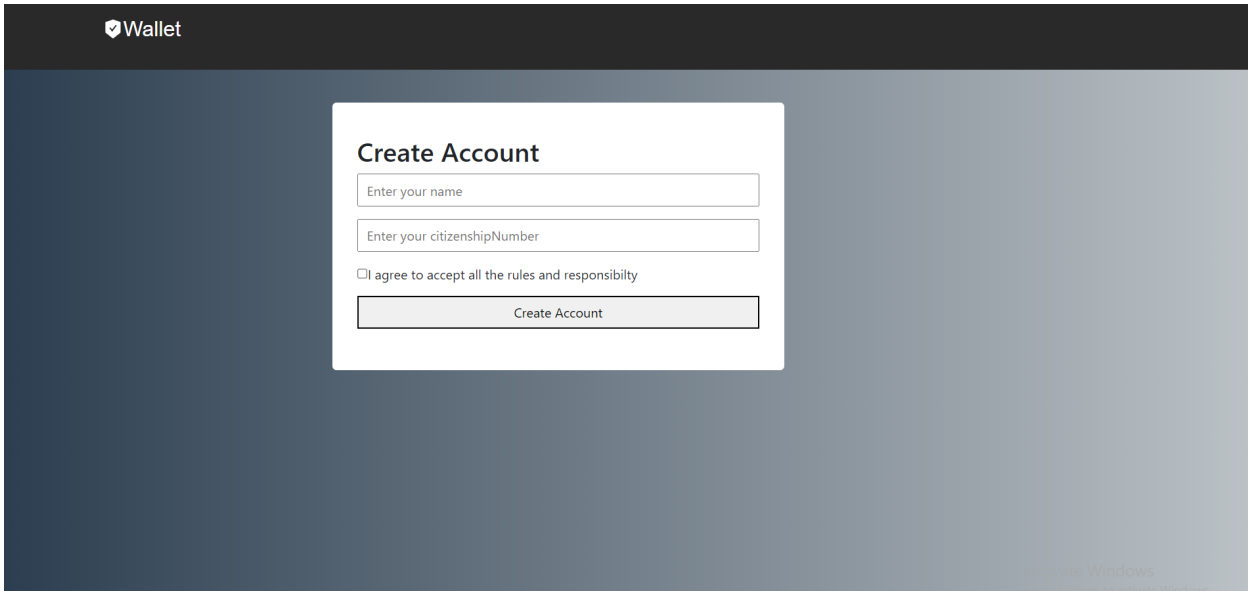

Figure 23. Creating account

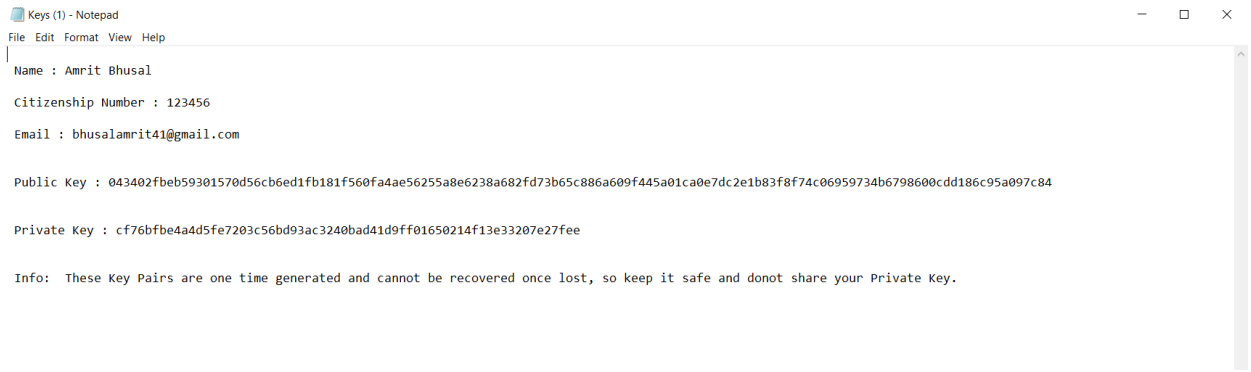

Figure 24. User Keys

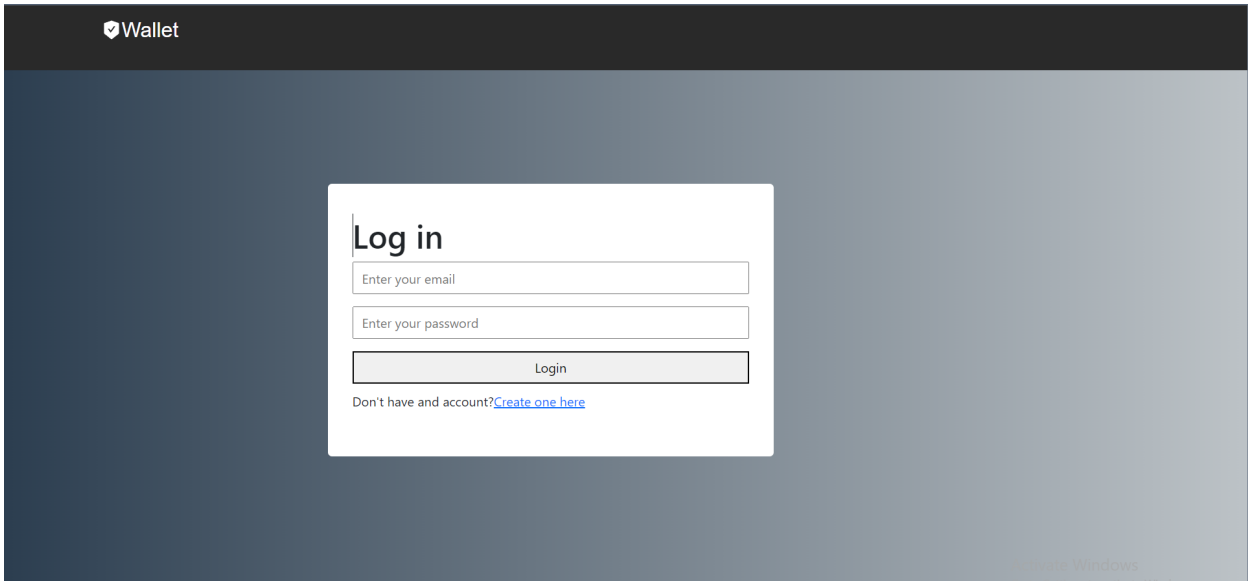

Figure 25. User Login

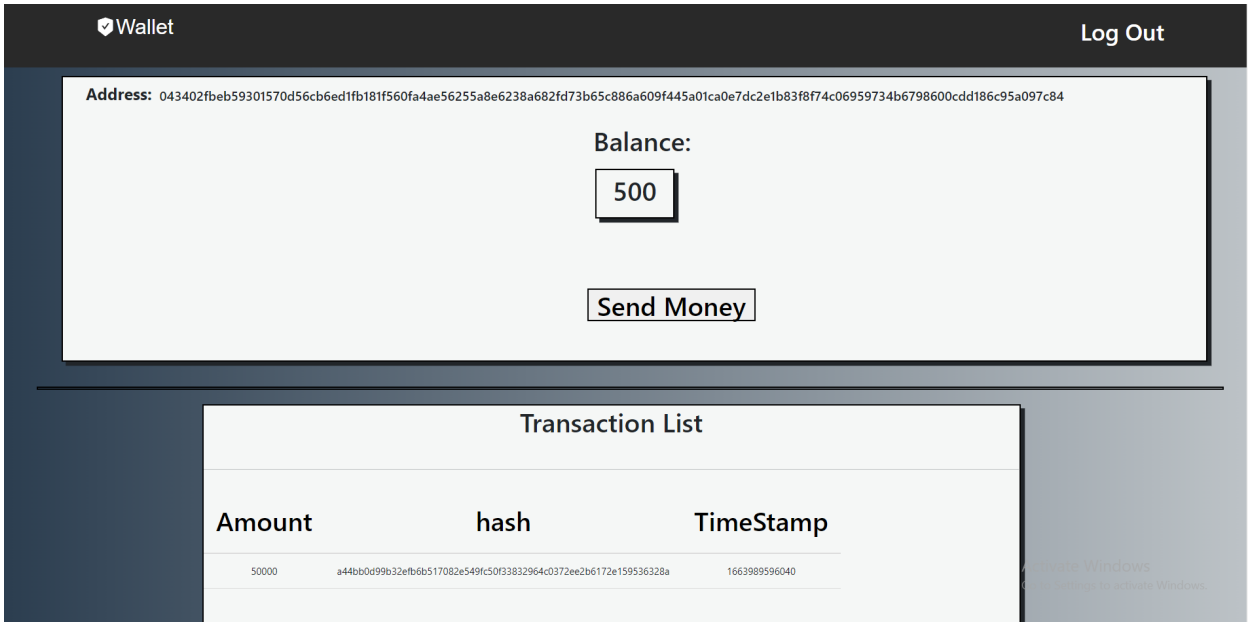

Figure 26. User Acount Dashboard

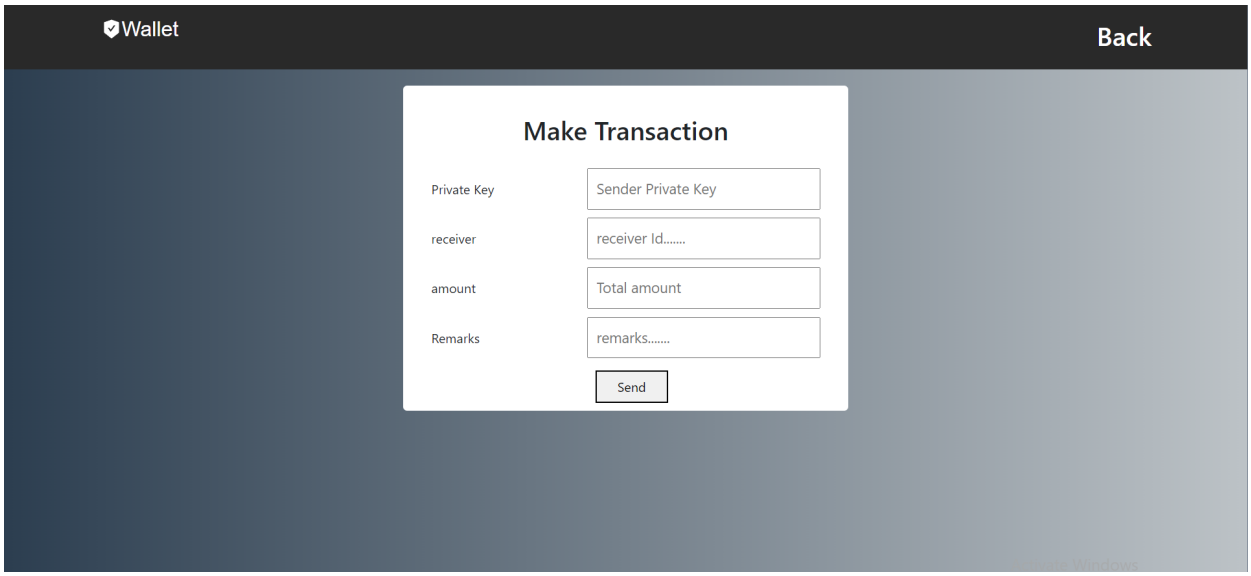

Figure 27. Make Transactions

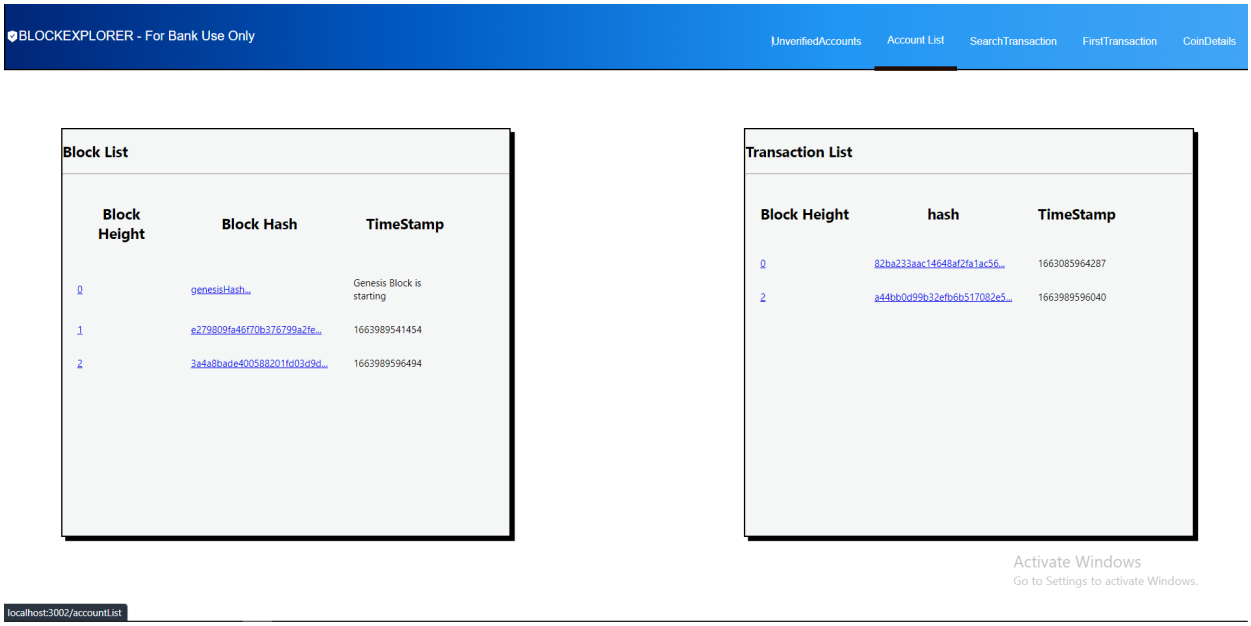

Figure 28. Blockchain Explorer

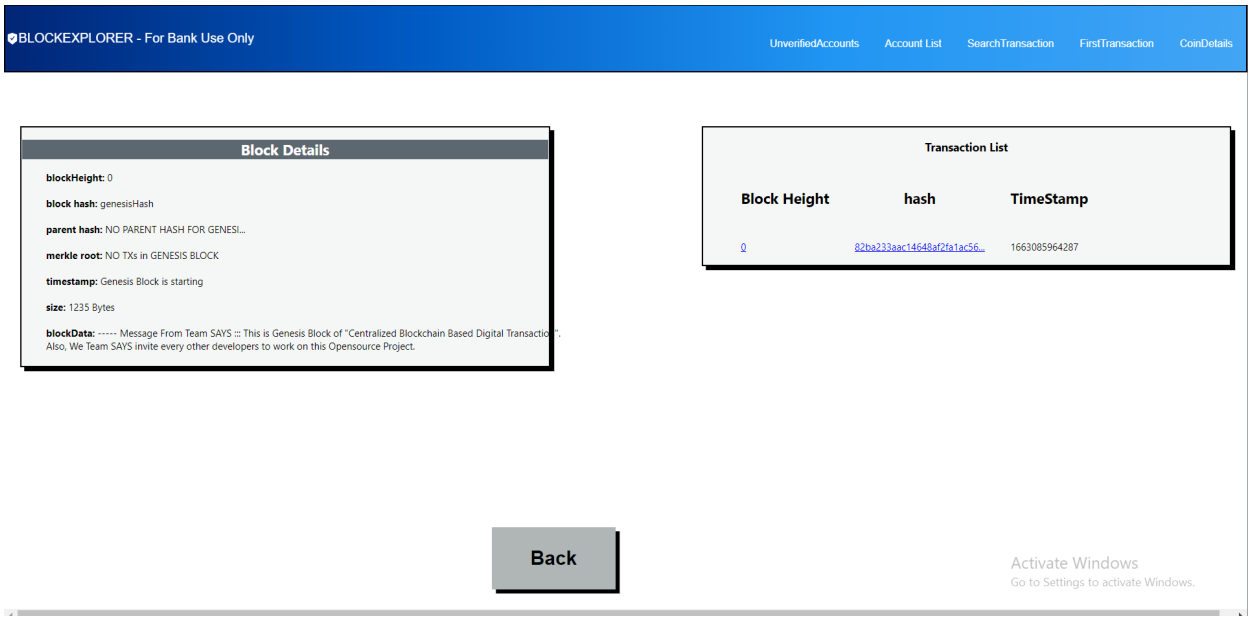

Figure 29. Block Explorer

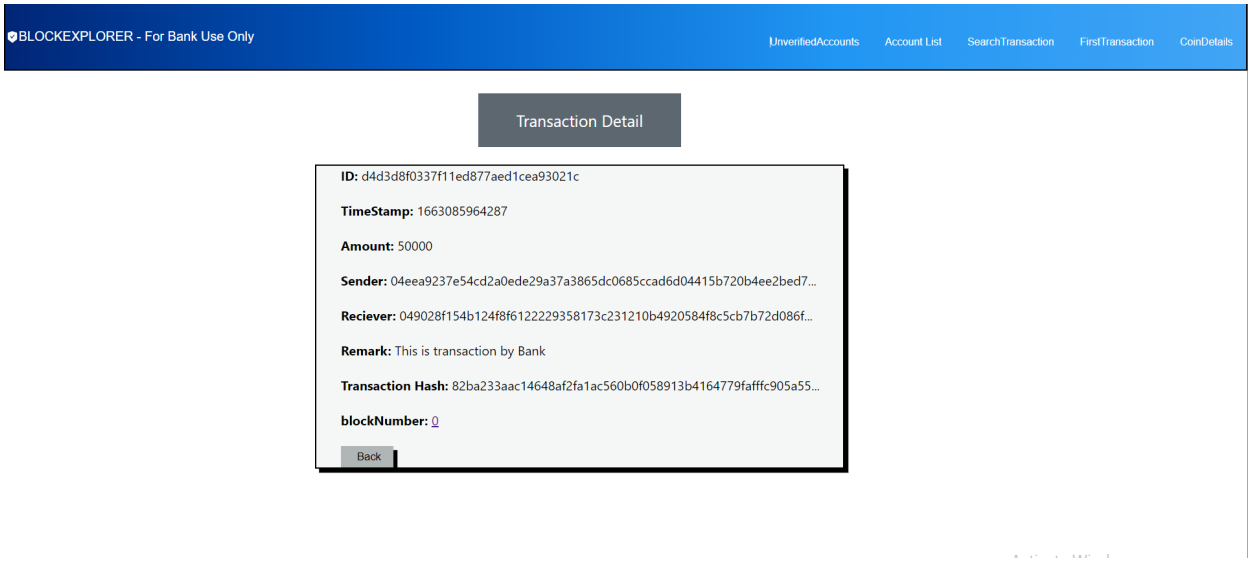

Figure 30. Transaction Explorer

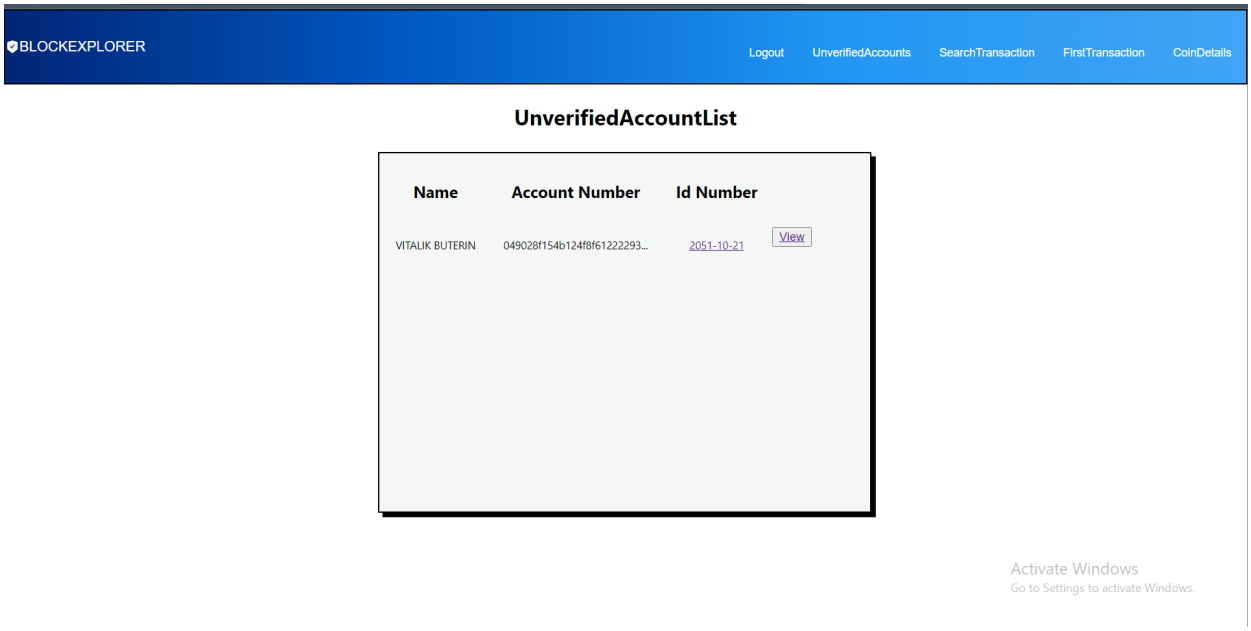

Figure 31. Unverified Accounts

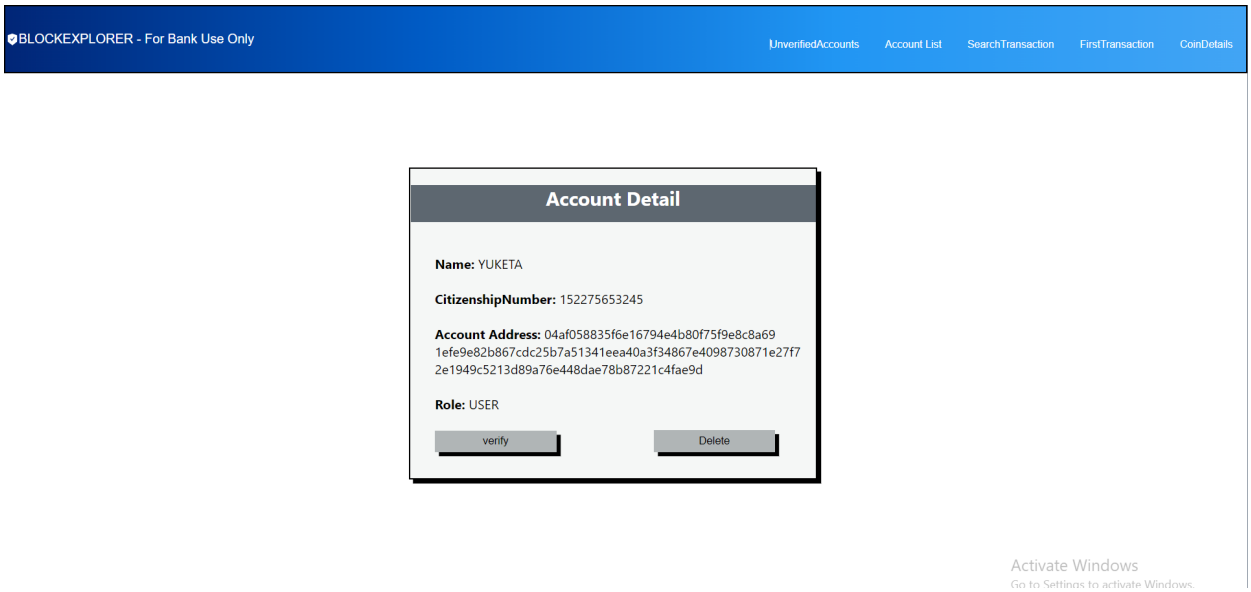

Figure 32. Account Details

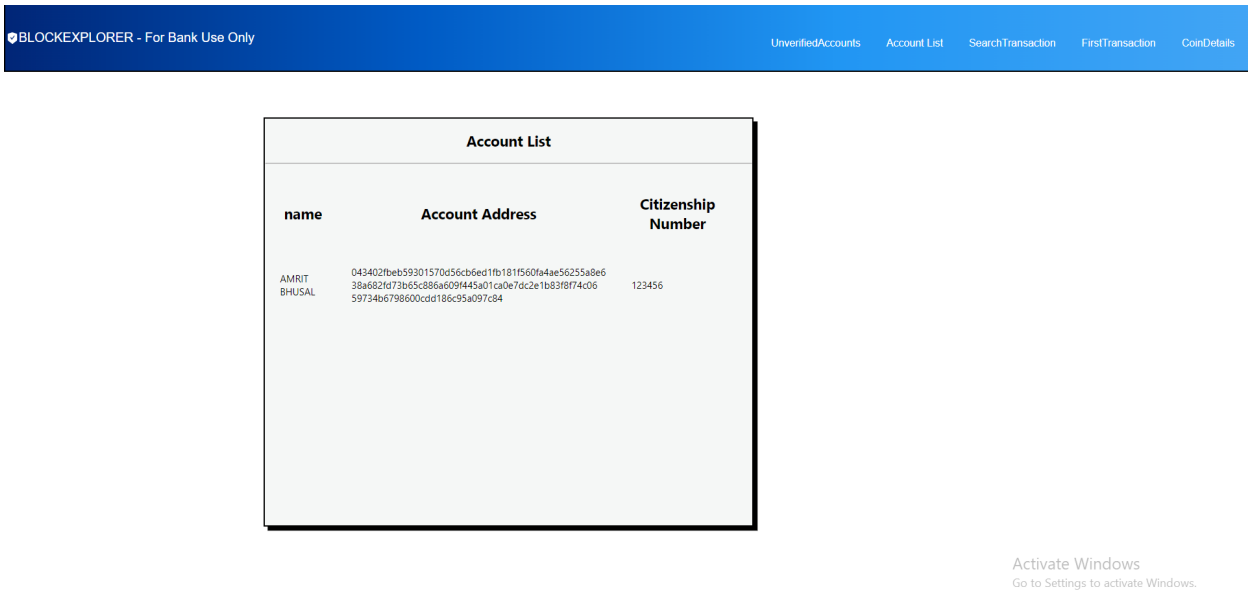

Figure 33. Account Details

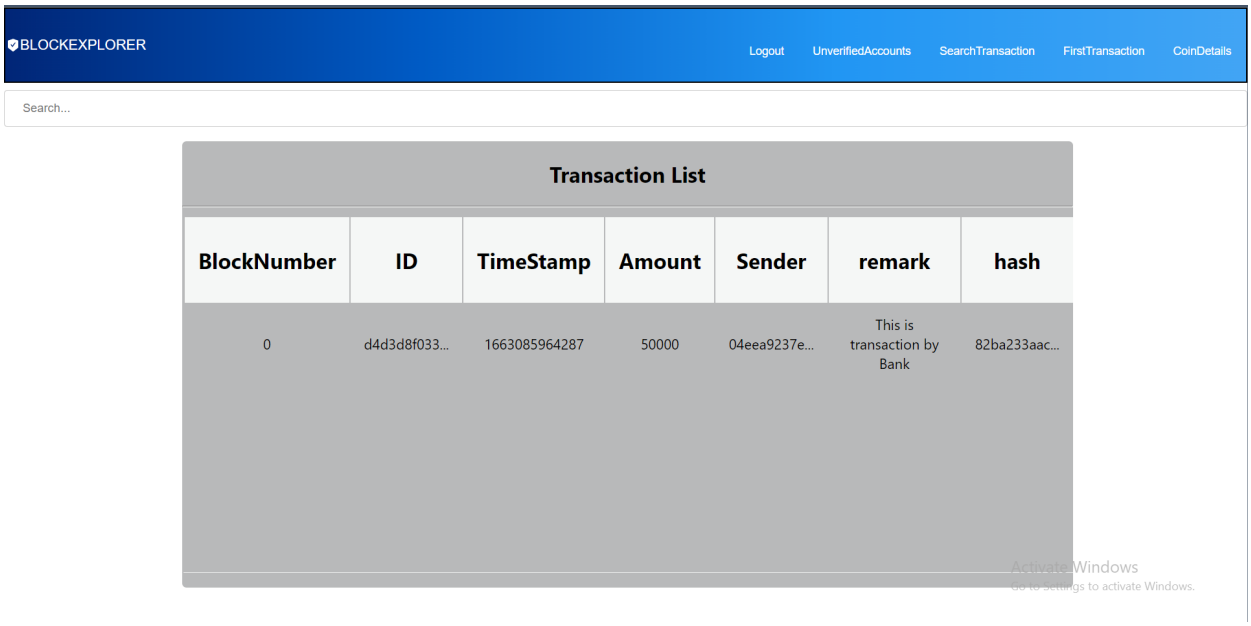

Figure 34. Transaction List

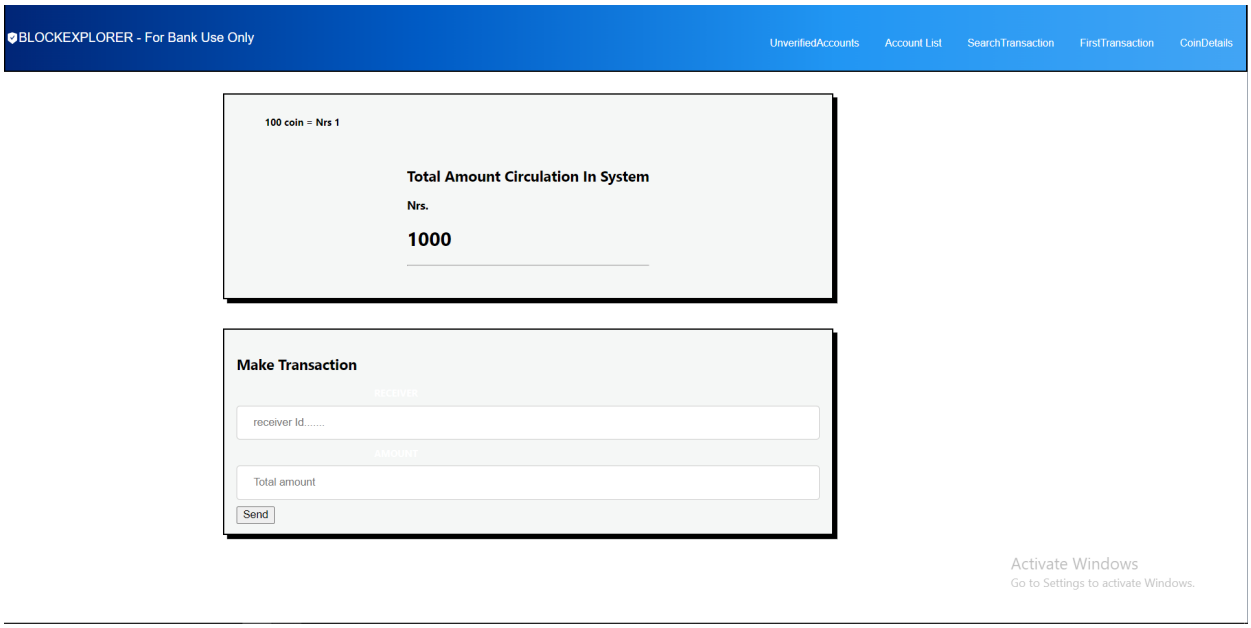

Figure 35. Make Transactions

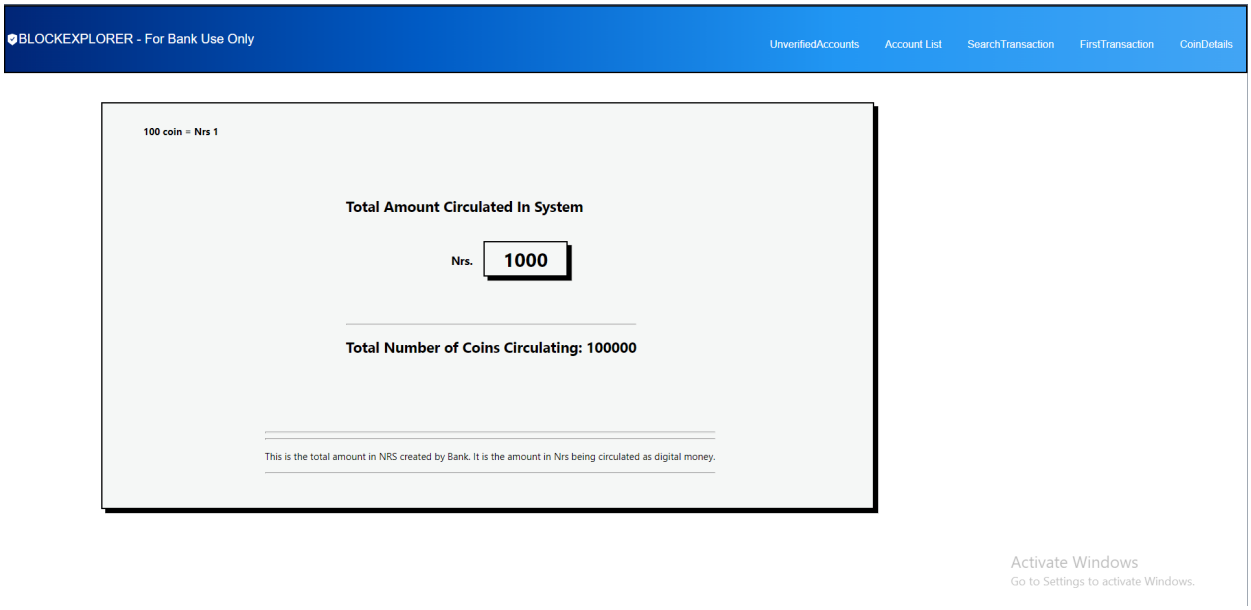

Figure 36. Coin Details

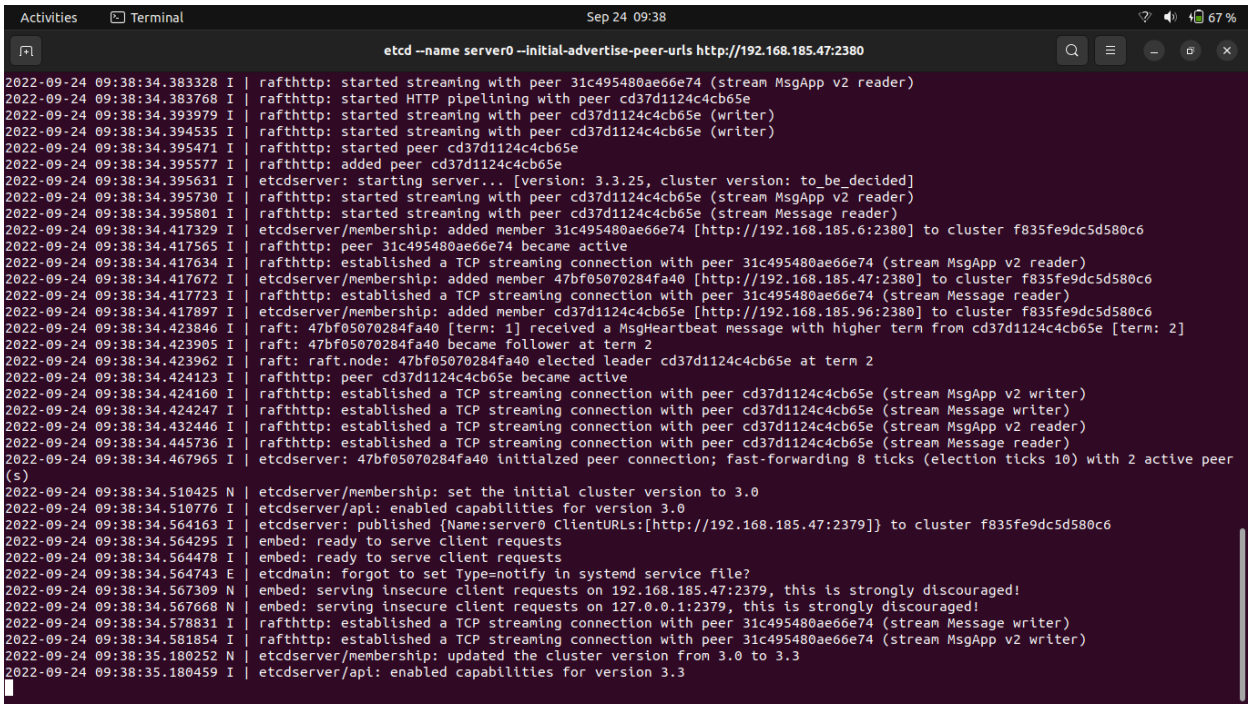

Figure 37. Etcd distributed network

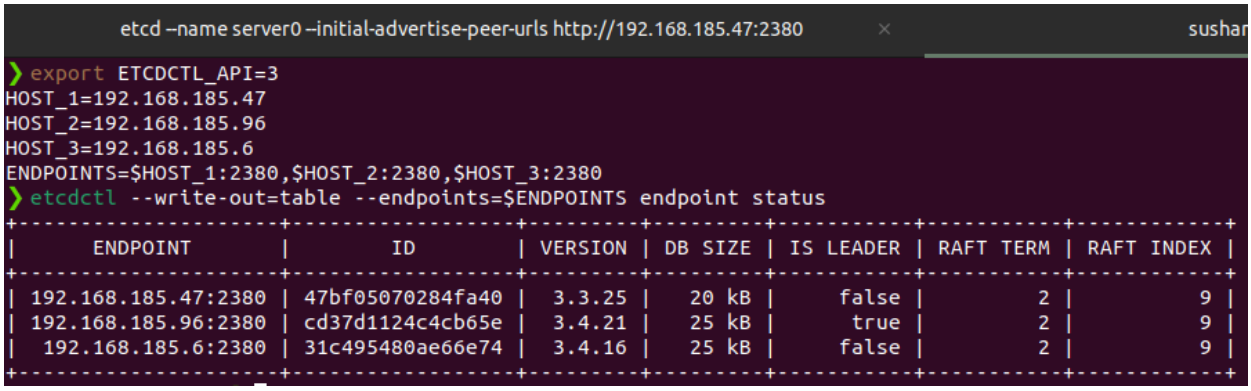

Figure 38. Status of Distributed Nodes

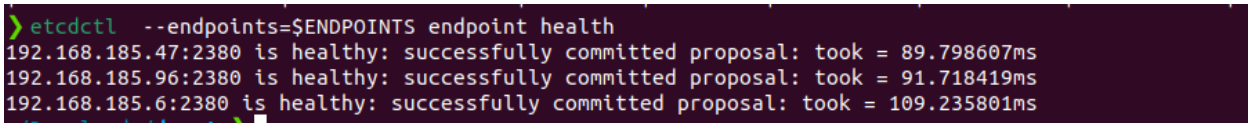

Figure 39. Health of Distributed Nodes