TRIBHUVAN UNIVERSITY

INSTITUTE OF ENGINEERING

PULCHOWK CAMPUS

A

PROJECT REPORT

ON

IMPLEMENTATION OF CONSORTIUM BLOCKCHAIN FOR DECENTRALIZED KYC SHARING

SUBMITTED BY:

MUKUL ATREYA (PUL075BCT051)

SANDEEP ACHARYA (PUL075BCT074)

SANGAM CHAULAGAIN (PUL075BCT078)

SAUJAN TIWARI (PUL075BCT083)

SUBMITTED TO:

DEPARTMENT OF ELECTRONICS & COMPUTER ENGINEERING

May, 2023

Page of Approval

TRIBHUVAN UNIVERSIY
INSTITUTE OF ENGINEERING
PULCHOWK CAMPUS
DEPARTMENT OF ELECTRONICS AND COMPUTER ENGINEERING

The undersigned certifies that they have read and recommended to the Institute of Engineering for acceptance of a project report entitled **"Implementation Of Consortium Blockchain For Decentralized KYC Sharing"** submitted by **Mukul Atreya**, **Sandeep Acharya**, **Sangam Chaulagain** and **Saujan Tiwari** in partial fulfillment of the requirements for the Bachelor's degree in Electronics & Computer Engineering.

....
Supervisor
Prof. Dr. Subarna Shakya
Professor
Department of Electronics and Computer
Engineering,
Pulchowk Campus, IOE, TU.

....
Internal examiner

....
External examiner

Date of approval:

Copyright

Acknowledgments

We feel immensely blessed and grateful for the invaluable guidance and assistance we received from numerous individuals throughout the completion of our Major Project. Firstly, we express our sincerest appreciation to the Institute of Engineering for incorporating the major project as an integral component of our curriculum. We extend our heartfelt thanks to the Department of Electronics and Computer Engineering for initiating and facilitating this project.

Our profound gratitude goes to our college for providing us with the necessary resources and support to work on this academic project. We are particularly indebted to our esteemed supervisor, **Prof. Dr. Subarna Shakya**, for his constructive and motivating suggestions regarding this project. His explanations of concepts such as Blockchain, Networking, Cryptography, guidelines on software usage, and analysis were immensely helpful to us throughout the project.

We are fortunate to have received constant encouragement, support, and guidance from all of our respected teachers. We would also like to thank RARA labs for the mentorship and financial resources that aided us throughout the project. Lastly, we would like to express our heartfelt appreciation to our friends, seniors, and everyone directly or indirectly involved in the project. Their invaluable contributions made it possible for us to complete the project successfully.

Sincerely,
Mukul Atreya
Sandeep Acharya
Sangam Chaulagain
Saujan Tiwari

Abstract

Know Your Customer (KYC) process that is followed by the Financial Institutions (FIs) at present is highly inefficient and inconvenient for both FIs and customers. This process requires verification of customer's identity documents independently by the businesses leading to high costs as well as wastage of resources. This project provides an efficient solution based on the Blockchain technology. The submission details of customers are collected only once for the verification process, irrespective of the number of financial institutions they register. The verified document is then shared with the organizations that require the information based on the customer's approval. It leads to an efficient KYC process reducing costs and resources. This system uses private distributed file storage to store the identity documents and consortium blockchain technology is used to record and manage the KYC transactions ensuring security and transparency. The financial institutions act as the full nodes of the blockchain and the synchronization of blocks takes place through the P2P network.

Keywords: *KYC, Consortium Blockchain,Distributed File Storage, P2P network*

Contents

List of Figures

List of Tables

List of Abbreviations

KYC	Know Your Customer
IPFS	InterPlanetary File System
SSI	Self Soverign Identity
FI	Financial Institution
PoW	Proof of Work
PoS	Proof of Stake
DPos	Delegated Proof of Stake
EVM	Ethereum Virtual Machine
ECDSA	Elliptic Curve Digital Signature Algorithm
API	Application Programming Interface
Dapp	Decentralized App
RSA	Rivest-Shamir-Adleman
ERC	Ethereum Request for Comments
ID	Identification Document
SDLC	Software Development Life Cycle
JSON	JavaScript Object Notation
SQL	Structured Query Language
DNS	Domain Name System
P2P	Peer-to-Peer
UI	User Interface
PBFT	Practical Byzantine Fault Tolerance

1. Introduction

1.1 Background

Know Your Customer(KYC) is the process or steps taken by financial institutions to establish customer identity, understand their nature of activities and assess risks involved with customers. Currently, the KYC process is carried out individually by each bank where the customer has to provide the same identification data to each bank, which is then verified independently by each bank. This is a time and cost-intensive job for FIs, as well as an irritating process for customers as they have to be physically present every time they visit a bank to verify their personal identity.

A central authority that collects and shares KYC among all financial institutions is a possible solution to above mentioned problems but it can lead to crashes, leaks, mistrusts and misuse of personal data. Also customers have no control over whom to grant or revoke KYC access because all data is managed by a centralized controller.

One solution to this problem is the use of distributed ledger where all digital data is replicated, shared, and synchronized across multiple FIs following some certain consensus protocols to ensure validity of data. Blockchain, one of the emerging and most popular distributed ledger technologies, is suited for decentralized asset management because it can act as a physically decentralized but logically centralized source of truth for information.

Many popular blockchain technologies like Bitcoin and Ethereum provide developers to build their solution upon them. For years, Ethereum has been the choice of developers to build decentralized applications. Ethereum provides smart contracts, a code that is immutable once deployed and gets executed as programmed, to build the blockchain applications. Also, bitcoin provides some of the Layer 2 solutions for building on the Bitcoin blockchain.

However, the existing blockchain technologies have their own cryptocurrencies. According to the existing laws of Nepal, all types of cryptocurrency transactions are banned. Also, For this particular use case, no such feature will be required. On the other hand, even the simple smart contracts in the ethereum blockchain are relatively expensive. Modifying these algorithms would be possible, but the effort to modify the existing source code could exceed the effort of implementing a blockchain from scratch.

So the new and improved approach to KYC filling and verification is proposed using blockchain technology implemented from scratch. The use of consortium blockchain and decentralized file storage makes this proposed system a secure and transparent medium to share KYC documents.

1.2 Problem statements

Financial institutions, government agencies, and other entities that require KYC (Know Your Customer) procedures to ensure compliance with regulations often struggle with lengthy and costly processes to verify the identity of their customers. The current KYC process involves users providing their personal information multiple times to different institutions, which can be cumbersome and risky. Also, there is a huge waste of resources by FIs in order to verify the same user who has already been verified by some other FI. Blockchain technology can be a good solution for this process, but many current blockchain platforms embed a cryptocurrency component within them. As there is a ban on cryptocurrency transactions in our country, it is impossible to leverage these platforms for this use case.

Therefore, A customized consortium blockchain is needed to facilitate the sharing of KYC information between customers and the FIs in a transparent and efficient manner, without the use of cryptocurrency.

1.3 Objectives

To implement a consortium blockchain technology for the verification and sharing of KYC documents.

1.4 Scope

Based on the concept of consortium blockchain, Decentralized KYC is the system proposed to facilitate the easy management of KYC documents by the FIs. It discourages the redundant process of filling out the KYC form by the customer and verification by the FIs every time the customer wishes to work with a certain FI. Also, the cost of verification is reduced as verification is done only once. Its main application is in banks and FIs to verify and extract the KYC documents of the customer. Furthermore, it can be modified to be used by a separate governmental institution to act as a mediator between the customer and FIs to manage KYC documents.

2. Literature Review

2.1 Related work

“Bitcoin: A Peer-to-Peer Electronic Cash System” a white paper published by Satoshi Nakamoto was one of first paper that proposes a trustless electronic payment system based on cryptography and proof of work consensus algorithm allowing any two willing parties to transact directly with each other without the need for a trusted third party through proof of work consensus mechanism over peer to peer network. Bitcoin as well as many other popular blockchains known today are based on this whitepaper.[1]

“Ethereum: A Next Generation Smart Contract and Decentralized Application Platform” a white paper published by Vitalik Buterin proposes blockchain technology with an integrated, fully-fledged Turing complete programming language that can be used to write ”contracts” that can store arbitrary state transition functions which dictate the execution of actions in the system. This provided users and developers with complete freedom to build any decentralized applications they wish for.[2]

Juan Benet demonstrated a distributed peer-to-peer file system that links all computing devices through a single file system and offers a high throughput content-addressed block storage model with content-addressed hyperlinks. This creates a generalized Merkle directed acyclic graph, a type of data structure that can be used to create distributed versioned file systems and even blockchains. IPFS, an implementation based on this paper combines all above technologies to implement a trustless peer-to-peer distributed file-storage with no single point of failure.[3]

“Implementing a blockchain from scratch: why, how, and what we learned” by Fabian Knirsch, Andreas Unterweger and Dominik Engel presents the implementation of blockchain from scratch for achieving a lightweight and simple solution for energy trading among households. The paper helps to gain an insight into the use cases of blockchain technology, where blockchain technologies are most suitable and the implementation of the blockchain from scratch. They discuss how the blockchains currently available in the market are either too complex or not suitable for their use case.[4]

The Hong Kong Monetary Authority in 2016 published the “Whitepaper On Distributed Technology” highlighting the potential benefits of blockchain technology for the FIs. It concluded that blockchain technology could be a huge benefit to digitize, update and share identity information among FIs in an effective and secure manner. The paper stated that the blockchain based identity management system would have many benefits. There would be a better customer experience as the customer would submit the KYC documents only once. Also the costs and resources needed for the KYC management would be reduced.[5]

A blockchain-based KYC optimization system was proposed in 2017 by Jose Parra Moyano and Omri Ross. The system aimed to provide a solution to decrease the cost of overall KYC management. The KYC verification process is conducted only once by the first financial institution the customer wishes to work with and the result of verification is shared in an anonymized and secure form with all financial Institutions wishing to establish a financial relationship with that customer. Furthermore, the cost is distributed among all the financial institutions working with that customer. However, they proposed a complex database architecture in which customers need to store these data privately and circulate them among the FIs with which they want to work. Such a structure is costly, and it becomes clear when one compares the customer journey that emerges from this structure with the existing customer journey that the self-storage aspect would be a disadvantage.[6]

A distributed ledger based KYC which can be shared with multiple financial institutions was proposed by Syed Azhar Hussain,Zeeshan-ul-hassan Usman. It has suggested a Proof Of Importance consensus algorithm to establish a scoring mechanism for details provided in the KYC process . For example, supply of basic phone number,email addresses can be graded as 50 while supply of biometrics and fingerprints can be graded as 100 and similarly supply of citizenship cards, national identities are graded as 200. So users can control the amount of KYC data to be shared with financial institutions according to the grade of information needed.[7]

Prince Sinha and Ayush Kaul proposed a system in their paper "Decentralized KYC System" that is more cost-efficient due to offchain data storage in which KYC data is stored in a decentralized database and only hash and username is stored as on-chain data on the blockchain reducing cost for the deployment of the smart contract.[8]

Table 2.1: Literature Review Matrix

S.N	Title	Author	Comments
1	Bitcoin: A Peer-to-Peer Electronic Cash System	Satoshi Nakamoto	This paper proposes a trustless electronic payment system based on cryptography allowing any two willing parties to transact directly with each other without the need for a trusted third party through proof of work consensus mechanism over peer to peer network.
2	Ethereum: A Next-Generation Smart Contract and Decentralized Application Platform	Vitalik Buterin	The blockchain technology described in this paper has an integrated, fully-fledged Turing-complete programming language that can be used to write ”contracts” that can store arbitrary state transition functions and enable users to build any kind of decentralized system.
3	IPFS - Content Addressed, Versioned, P2P File System	Juan Benet	This paper presents the InterPlanetary File System(IPFS), a peer-to-peer distributed file system.
4	Implementing a blockchain from scratch: why, how, and what we learned	Fabian Knirsch, Andreas Unterweger and Dominik Engel	This paper presents the application of the blockchain technology for use cases other than financial transactions and the implementation of blockchain from scratch for achieving a lightweight and simple solution based upon a particular use case.

5	A Taxonomic Approach to Understanding Emerging Blockchain Identity Management Systems	Loïc Lesavre, Priam Varin, Peter Mell, Michael Davidson, and James Shook	This paper presents about different blockchain-based Identity management systems. It categorizes these systems into a taxonomy based on differences in blockchain architectures, governance models, and other salient features.
6	Whitepaper On Distributed Ledger Technology	The Hong Kong Monetary Authority	This paper highlighted the potential benefits of blockchain technology for the banks and financial institutions and concluded that blockchain technology could be a huge benefit to digitize, update and share identity information among banks and FI's in an effective and secure manner.
7	KYC Optimization Using Distributed Ledger Technology	Jos´e Parra-Moyano and Omri Ross	First paper to suggest that the KYC verification process should be conducted only by the first financial Institution which wishes to work with a customer, and that the result of verification and validation should be shared in an anonymized and secure form with all financial Institutions which will wish to establish a financial relationship with that customer.

8	Decentralized KYC System	Prince Sinha and Ayush Kaul	This paper proposed system is more cost-efficient due to offchain data storage in which KYC data is stored in a decentralized database and only hash and username is stored as on-chain data on the blockchain.This reduces cost for deployment of the smart contract.

2.2 Related theory

2.2.1 Blockchain

A blockchain is a digital ledger of transactions that is duplicated and distributed across the entire network of computer systems (nodes). Each block in the chain contains a number of transactions, and every time a new transaction occurs on the blockchain, a record of that transaction is added to every participant's ledger. Blockchain is a type of DLT (Distributed Ledger Technology) in which transactions are recorded with an immutable cryptographic signature called a hash.

There are mainly three types of blockchain:

- **Public blockchain:** A public blockchain is the permission-less distributed ledger technology where anyone can join and do transactions. It is a non-restrictive version where each peer has a copy of the ledger and anyone can access the public blockchain if they have an internet connection.

- **Private blockchain:** A private blockchain can be best defined as the blockchain that works in a restrictive environment, i.e., a closed network. It is also a permissioned blockchain that is under the control of an entity.

- **Consortium blockchain:** A consortium blockchain is an approach to solving organizations needs where there is a need for both public and private blockchain features. In a consortium blockchain, some aspects of the organizations are made public, while others remain private. A consortium blockchain is managed by more than one organization.

2.2.2 Peer to peer network

A peer-to-peer (P2P) network is a sort of computer network where each user functions as both a client and a server. This type of network eliminates the need for a centralized server and allows for the sharing of resources and data. All nodes in a P2P network are equal and have the same ability to make requests, reply to requests, and share resources.

A central server is in charge of overseeing and delivering resources and data to clients in a classic client-server arrangement. In contrast, P2P networks do not require a central server because any node or participant can share and receive data and resources directly from other nodes. These networks are often used for file sharing, instant messaging, and other communication applications.

The architecture of a P2P network can be classified into unstructured and structured. Unstructured P2P networks are more common and simpler in design. They are characterized by a lack of a formal structure, which means that nodes can connect to any other nodes in the network without any predefined rules or requirements. In an unstructured P2P network, nodes are typically connected to a limited number of other nodes, forming a loosely connected mesh network. This design is often used for file sharing applications.

Structured P2P networks, on the other hand, have a more formal structure that organizes the nodes into a hierarchical or distributed structure. This allows for more efficient routing and searching of resources in the network. Structured P2P networks are often used for content distribution and communication applications.

P2P network provides scalability. Because there is no central server, the network can handle large amounts of traffic and can easily scale up as more nodes join the network. Additionally, P2P networks are often more robust and resilient than traditional client-server networks because they do not have a single point of failure. However, P2P networks can also be used for illegal activities, such as piracy or malware distribution, and can be difficult to monitor and regulate. Additionally, the decentralized nature of P2P networks can make them slower and less efficient than centralized networks for certain types of applications.

A P2P network is essential to the distributed and decentralized character of blockchain systems in the context of blockchain technology. Each node in a blockchain network keeps a copy of the blockchain ledger, and all nodes communicate with one another to agree on the ledger's current state.

A new transaction on the blockchain is broadcasted to all network nodes. The transaction is subsequently verified by each node using a set of network-agreed consensus standards. If the transaction is legitimate, it is included in a fresh block that is subsequently broadcasted to all network nodes.

2.2.3 Transactions

In blockchain network transactions are any kind of data that is stored on the blockchain. It could be anything from the transfer of ownership of a digital asset to a simple message that needs to be stored permanently. A transaction is initiated by a participant on the network who wants to create a new record on the blockchain.The transaction contains information such as the Sender, Receiver, Amount, Timestamp, Signature and other relevant data. When a user initiates a transaction, it is broadcasted to all the nodes on the network, which validates the transaction using a consensus algorithm. Once the transaction is validated, it is added to a block, which is then added to the existing blockchain.

This information is then verified and validated by the nodes on the blockchain network using consensus algorithms to ensure the transaction is valid and meets the rules of the network and finally added to blocks.

2.2.4 Block

A block is a container data structure that aggregates transactions for inclusion in the public ledger, the blockchain.

Block can be generally divided into two major sections:

- **Header:** Header section contains metadata about the block, such as a timestamp, a nonce (a random number used in the mining process to create a new block), and the hash of the previous block in the chain. The header also contains a hash of the data section, which is used to ensure the integrity of the block's transactions.
- **Data:** Data section contains a set of transactions that have been validated by the network's consensus mechanism followed by a long list of transactions that make up the bulk of its size.

2.2.5 Consensus

Consensus is the process of achieving agreement among multiple nodes or participants in the system regarding a particular state or decision. In a distributed system, where there is no central authority, achieving consensus is essential to ensure that all participants agree on the current state of the system and any changes made to it.

Consensus is achieved through a consensus algorithm, which is a set of rules and protocols that allow nodes to agree on a specific state or decision. In the blockchain network, the consensus algorithm ensures that all nodes in the system have the same copy of the distributed ledger. Consensus algorithms are used to validate transactions to confirm their authenticity, and ensure that they are added to the blockchain in a secure and decentralized manner. It is necessary for maintaining the integrity of blockchain.

There are different types of consensus algorithms used in blockchain technology. The popular consensus algorithms include:

- **Proof of Work (PoW):** This is the most commonly used consensus algorithm, used by cryptocurrencies like Bitcoin and Ethereum. It involves miners competing to solve complex mathematical problems to validate transactions and add new blocks to the blockchain.

- **Proof of Stake (PoS):** This consensus algorithm selects validators based on their stake in the network, and validators are chosen to add new blocks based on the amount of cryptocurrency they hold.

- **Delegated Proof of Stake (DPoS):** This consensus algorithm is similar to PoS, but instead of selecting validators based on their stake, it selects a group of delegates to validate transactions and add new blocks to the blockchain.

- **Practical Byzantine Fault Tolerance (PBFT):** This consensus algorithm is used in permissioned blockchains and relies on a small group of nodes, known as validators, to reach consensus on transactions and blocks.

2.2.6 Public Key Cryptography

Public key cryptography, also known as asymmetric cryptography, is a cryptographic system that uses two keys, a public key and a private key, to encrypt and decrypt data. This system was developed to overcome the limitations of traditional symmetric cryptography, where the same key is used to encrypt and decrypt data, making it vulnerable to attacks if the key is compromised.

In public key cryptography, the public key is freely available to anyone who wants to send encrypted data to the owner of the private key, while the private key is kept secret and only known to the owner. The public key can be distributed widely without any security concerns, as it is only used to encrypt data and not to decrypt it.

The use of public key cryptography provides several advantages over traditional symmetric cryptography. It eliminates the need for a secure channel to transmit the encryption key, as the public key can be freely distributed. It also enables digital signatures, where the sender can sign a message using their private key, and the receiver can verify the signature using the sender's public key. This provides a way to authenticate the sender and ensure that the message has not been tampered with in transit.

In a blockchain network, each participant has a public key and a private key. The public key is a unique identifier that is used to receive transactions and to validate digital signatures, while the private key is a secret that is used to sign transactions and prove ownership of the associated public key.

Wallet is used to generate and manage these public and private keys. Each participant generates a key pair, and the public key is used to create a unique digital identity on the blockchain network. The private key is kept secure and used to sign transactions and prove ownership of the public key. When a transaction is submitted to the blockchain network, it is digitally signed using the private key of the sender. The digital signature is then verified by the network using the public key of the sender. This ensures that the transaction was sent by the rightful owner of the public key and that it has not been tampered with in transit.

2.2.7 Hashing

Hashing is a process of converting any input data of arbitrary length into a fixed-size output, typically a string of digits and letters of a specific length, using a mathematical algorithm called a hashing function. The output is commonly referred to as a hash or message digest.

The hashing function takes the input data and applies a series of mathematical operations to it to produce the output hash.

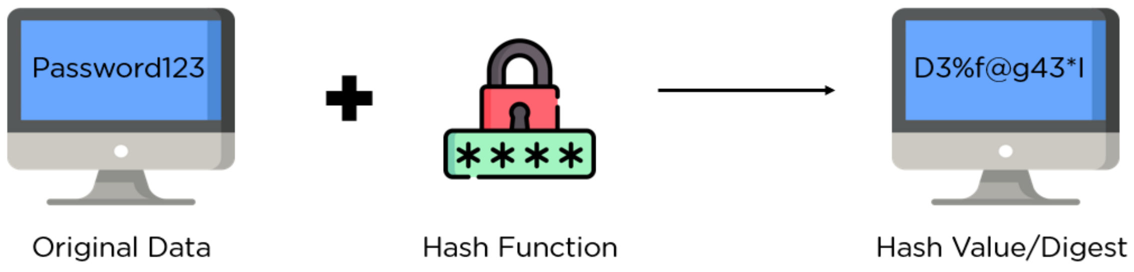


Figure 2.1: Hashing procedure
[9]

The key properties of a hashing function are:

- **Deterministic:** For a given input, the output of the hashing function will always be the same.
- **Irreversible:** It is computationally infeasible to reconstruct the original input data from the hash.
- **Uniformity:** A small change in the input data should produce a significant change in the output hash.
- **Collision-resistant:** It is computationally infeasible to find two different inputs that produce the same output hash.

Hashing is used in blockchain technology to ensure the integrity and security of the data stored in blocks. Each block in a blockchain contains a hash of the previous block, forming a chain of blocks that is resistant to tampering. Any attempt to modify the data in a block would result in a different hash value, breaking the chain and making the modification immediately detectable.

2.2.8 Wallets

Crypto wallets are digital wallets that are used to securely store private and public keys for transactions on blockchain networks. Private keys are used to sign transactions, while public keys are used to prove ownership of transactions through digital signatures.When a user initiates a transaction, the crypto wallet creates a digital signature using the private key to authenticate the transaction. The digital signature is a mathematical algorithm that verifies the ownership of the private key and ensures that the transaction cannot be altered after it has been signed.

The private key must be kept secret and secure at all times. If the private key is lost or stolen, the funds in the wallet are lost forever. Therefore, it is important to backup the private key and store it in a secure location. Many wallets offer features such as multi-signature and hardware wallets to provide additional security for private keys.

Crypto wallets come in two forms: hot wallets and cold wallets. Hot wallets are connected to the internet and allow for easy access to funds, while cold wallets are offline and offer greater security.

Most crypto wallets use a combination of software and hardware to store private keys. Hardware wallets, for example, store private keys on a physical device, such as a USB drive, that is disconnected from the internet when not in use. Software wallets, on the other hand, store private keys on a computer or mobile device and can be protected with passwords or other security measures.

In addition to storing private and public keys, crypto wallets also allow users to view their cryptocurrency balances, track transactions, send and receive funds which depend on the type of blockchain network we are working with.

2.2.9 Javascript and Go packages for cryptography

Ethers.js is a JavaScript library for interacting with blockchain networks. One of the main features of Ethers.js is the ability to create and manage wallets that can be used to sign transactions and messages on the blockchain. When creating a new wallet, Ethers.js uses the Elliptic Curve Digital Signature Algorithm (ECDSA) to generate a new private key that is associated with the wallet. This private key is then used to sign transaction

Wallet.signMessage() method, takes transaction data as input and generates a signature using the private key associated with the wallet. This signature can be included with the transaction data and forwarded to the blockchain network.

go-ethereum/crypto is a Go implementation for interacting with blockchain networks. crypto.SigToPub() method reconstructs the public key from the signature and transaction data and compares it to the expected public key to determine if the signature is valid.

Both Ethers.js and go-ethereum/crypto rely on the Elliptic Curve Digital Signature Algorithm (ECDSA) for signing and verification of transactions and messages on the blockchain.

2.2.10 Networking

In the context of blockchain, Networking refers to the process of communication and data exchange between the nodes in the network. To facilitate this communication, blockchain networks use a variety of networking protocols and technologies among which P2P plays a crucial role. In a peer-to-peer network, each node acts both as a client and a server, and is responsible for transmitting and receiving data to and from other nodes. Nodes can join or leave the network at any time, and new nodes can be added without the need for a central authority to manage the process. P2P networking is implemented along with other networking protocols which specifies how nodes communicate and share and broadcast data with each other. The protocol defines the format of messages and blocks, as well as rules for validation of received data. One of the important aspect of networking is block broadcasting.

Block Broadcasting

Block broadcasting is the process of distributing a newly mined block to the entire network of nodes in a blockchain system. When a new block is created, it needs to be broadcasted to the network in order to be verified and added to the blockchain.

In a peer-to-peer network, the block is initially broadcasted by the node that mined it, known as the "originating node". The node will then send the block to a number of its connected peers, who in turn send the block to their connected peers, and so on. This process continues until the block has been distributed to all nodes in the network.

There are two main methods for block broadcasting in a blockchain network. They are:

- **Flooding:** In this method, the originating node sends the block to all of its connected peers, who in turn send the block to all of their connected peers, and so on. This process continues until the block has been received by all nodes in the network. Flooding is simple and effective, but it can result in a large amount of network traffic and can potentially overload nodes with too many incoming blocks.

- **Gossiping:** In this method, the originating node sends the block to a small number of its connected peers, who in turn send the block to a small number of their connected peers, and so on. This process continues until the block has been received by all nodes in the network. Gossiping is more efficient than flooding, as it reduces the amount of network traffic and can help prevent network congestion. Additionally, it allows nodes to prioritize blocks based on their importance and relevance to the network.

3. Requirement Analysis

3.1 Functional Requirements

3.1.1 Registration of Customers and Financial Institutions

Customers as well as financial institutions should be able register into the platform.

3.1.2 Uploading KYC details

Customers should be able to upload KYC details on the platform as well as approach financial institutions for verification.

3.1.3 Verification of KYC

Financial institutions should be able to view and verify documents provided by the users.

3.1.4 Request for permission

Financial institutions should be able to request the customers for their KYC documents which are already verified by some other institution in the network.

3.1.5 Grant permission

Customers should be able to give permission for financial institutions in order to view their KYC documents.

3.1.6 Record transactions

All the transactions in the system should be recorded and broadcasted in the blockchain.

3.2 Non Functional Requirements

3.2.1 Reliability

System should be free from faults, bugs and be available all the time to the users to upload and share the KYC details and FIs to verify or access the documents at any time upon request.

3.2.2 Performance

The KYC details encryption, upload, decryption, viewing, verification and sharing process all should occur within proper time.

3.2.3 Maintainability

The system should be easily maintainable and the errors and bugs should be found and debugged easily.

3.2.4 Security

The KYC details should be encrypted and should be viewed only by authorized persons.

3.3 Tools and Technologies

3.3.1 Hardware Requirements

Banks will serve as full nodes for blockchain networks, therefore hardware for the bank side will consist of a computer with adequate processing power, storage, and internet connectivity, with proper power backups.Users act as half node and only engages through front end interface hence the necessary hardware is an internet-connected laptop or mobile device.

3.3.2 Software Requirements

Table 3.1: Software Requirements Table

S.N	Languages/Frameworks	Function
1	Golang	Used for developing blockchain servers.
2	NodeJS	Used for connecting blockchain, frontend and databases.
3	ReactJS	Used for making User Interfaces.
4	MongoDB	Used for storing off-chain data.
5	IPFS Swarm	Used for storing KYC documents.
6	BadgerDB	Used for storing on-chain data.

4. Feasibility Study

4.1 Economic Feasibility

The project is economically feasible as very high computation power is not required for the development process. Also, a custom consortium blockchain is created where cryptocurrencies aren't required.

4.2 Technical Feasibility

While implementing a blockchain solution from scratch is a complex and time-consuming process, it is technically feasible with the right expertise and resources. Since blockchain technology has been around for decades, there's a lot of resources to study about them. As long as the development team has the required technical skills, knowledge, and tools, they can design and implement a blockchain solution like this.

4.3 Operational Feasibility

The project is operational feasible as it can be operational just after configuring nodes. The complexities of blockchain technologies are abstracted away from the users. So users can use the system easily.

4.4 Schedule Feasibility

The schedule of the project as mentioned in the Gantt chart of the project proposal was met. It was completed in a total of 28 weeks. Thus, this project is schedule feasible.

5. Methodology

5.1 System Architecture

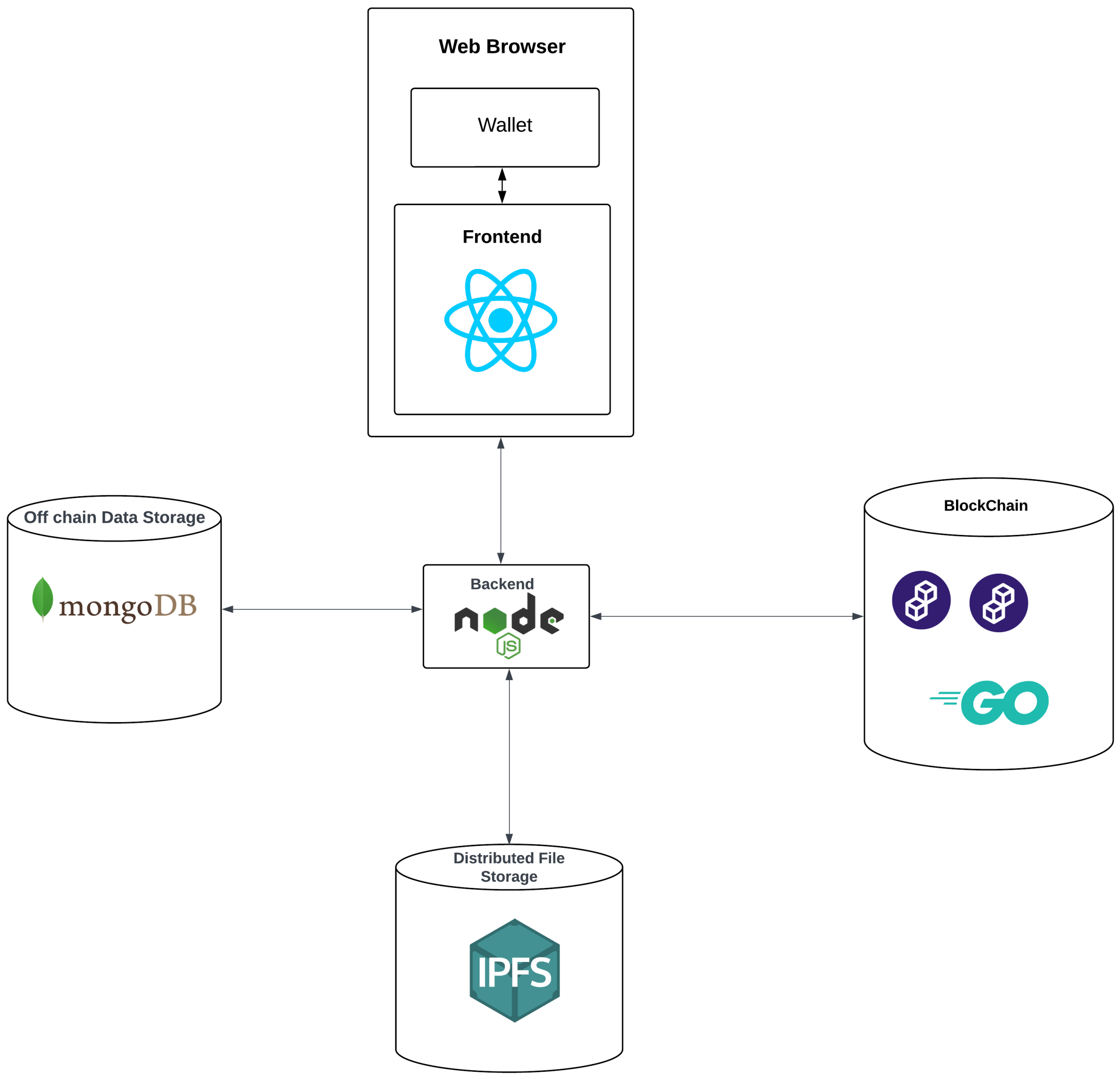


Figure 5.1: System Architecture Diagram

The major components of the system are:

5.1.1 Blockchain

Blockchain is the core part of the proposed system. A consortium blockchain is created where all the transactions between the customers and FIs are stored. The FIs themselves are the distributed nodes to run the programs to process blocks. FIs participate as full nodes by mining and processing the transactions whereas the users participate as the half nodes by only participating in transactions and not concerned with the mining.

5.1.2 Private Distributed File Storage

In order to store the KYC documents of the user, a private distributed file storage is implemented where the documents of the users are stored safely and reliably. The documents are encrypted cryptographically and stored in the distributed file storage. The FIs themselves are the nodes of the distributed file storage and are responsible for storing the KYC documents of the users.

5.1.3 Centralized Off-Chain Database

There is some information that is too extensive to be stored in the blockchain. Blockchain is mostly utilized to maintain immutable events with user KYC (verification,access banks,IPFS address etc.). The information that is extensive to be stored in the blockchain is stored in the centralized database. This database helps to keep track of offchain events like user requests for verification,banks requests for KYC access etc.

5.1.4 Wallet

Public and Private keys are username and password for blockchain which are unique to each user. So a simple wallet program is implemented in order to generate,store and manage key value pairs of public and private keys associated with users and FIs.

5.1.5 Backend Server

Backend server is used as an interface between the Blockchain Server and the User Interface. Also, it is responsible for the communication with the distributed file storage and databases. Thus, the backend server acts as an intermediary responsible for the communication with all other components of the system.

5.1.6 User Interface

User Interfaces are used by customers and FIs to interact with the system. An easy to use User interface is created hiding all the blockchain complexities from the end users.

5.2 System Development Workflow

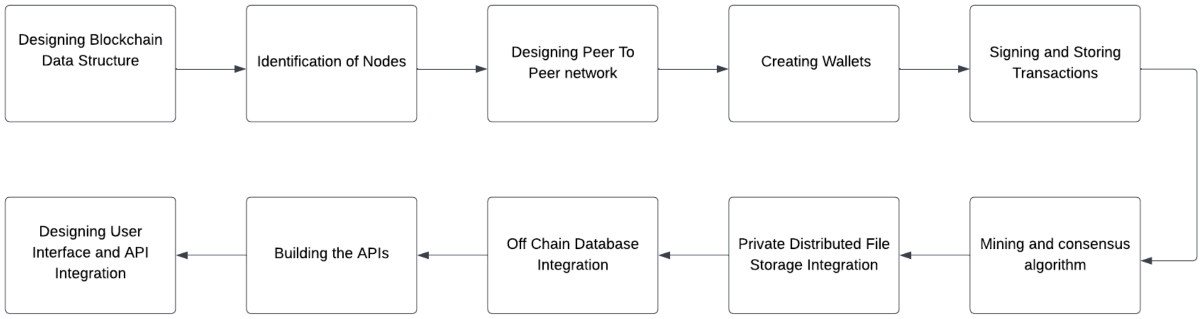

Designing Blockchain Data Structure
Identification of Nodes
Designing Peer To Peer network
Creating Wallets
Signing and Storing Transactions
Mining and consensus algorithm
Private Distributed File Storage Integration
Off Chain Database Integration
Building the APIs
Designing User Interface and API Integration

Figure 5.2: System Development Workflow

5.2.1 Designing Blockchain Data Structure

Blockchain is a linear chain of blocks. Each chain contains a set of transactions and other details. Blocks are linearly connected and are cryptographically secured. Each block header contains the previous block hash, nonce, and other details. All blocks are connected linearly by carrying the hash of the previous block. The first block with no previous block hash is called Genesis Block. Transaction is a variable-sized field that includes the list of all transactions contained in the block.

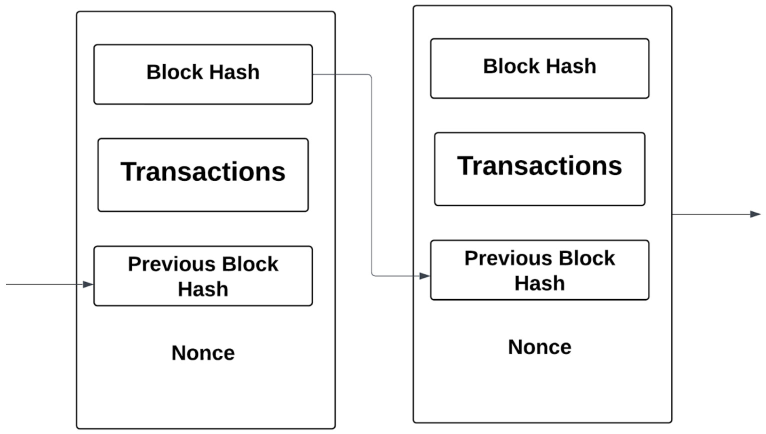


Figure 5.3: Block structure in blockchain

Different parts contained in the block are:

- **Block Hash:** A block hash is a unique identifier for a block in a blockchain network. It is a fixed-length cryptographic hash function that is used to secure and validate the integrity of the data within the block. The hash is generated by applying a complex mathematical algorithm (SHA256) to the block's contents.

- **Previous Block Hash:** The previous block hash is the hash of the previous block in the blockchain. Each block in the chain contains the hash of the previous block, which creates a chain of blocks that are linked together. This field is used to ensure the immutability of the blockchain because any changes to a previous block will cause all subsequent blocks to become invalid.

- **Nonce:** A nonce is a number that is added to a block's header during the mining process in a Proof of Work (PoW) blockchain network. The nonce is a random number that is added to the block's data to create a hash that meets the network's difficulty target. The difficulty target is a value that determines how hard it is to mine a block, and the nonce is used to adjust the hash until it meets the target.

- **Block Height:** Block height refers to the number of blocks in a blockchain network. Each block in the chain is assigned a unique height, starting with the genesis block, which has a height of 0. As new blocks are added to the network, the block height increases, and each new block is added to the end of the blockchain. The block height is used to determine the current state of the network and to synchronize the blocks with other nodes.

- **Transactions:** Transaction is a variable-sized field that includes the list of all transactions contained in the block. The blockchain implemented in the project contains only one transaction per block. Transactions are initiated in three different situations. They are:

  1. **KYC Upload**

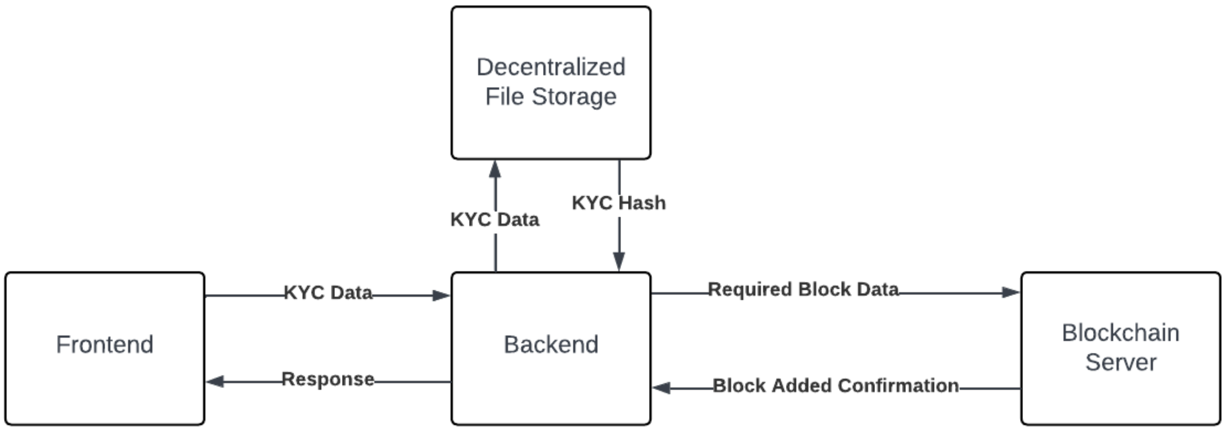


Figure 5.4: KYC Upload Transaction

  2. **KYC Verify**

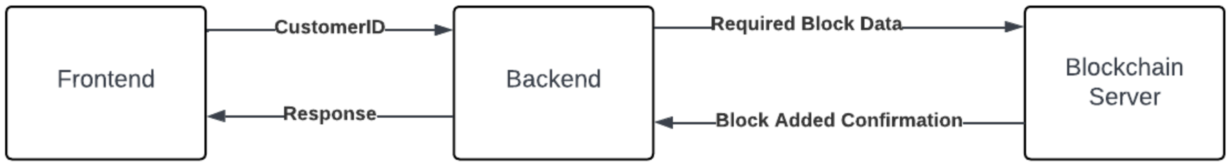


Figure 5.5: KYC Verify Transaction

  3. **Grant Access**

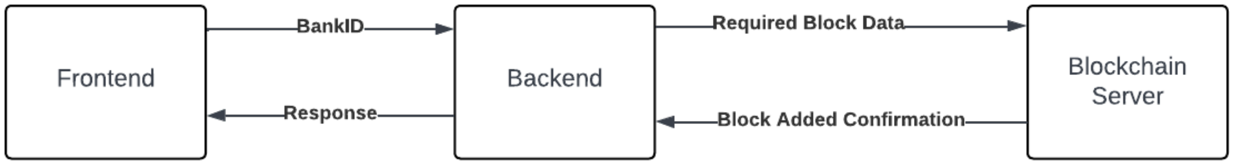


Figure 5.6: KYC Grant Access Transaction

There are different fields in a transaction. They are:

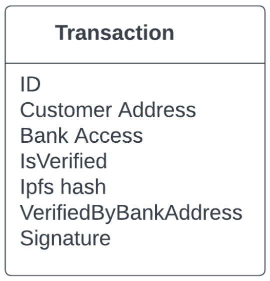


Figure 5.7: Transaction structure

- **ID:** A transaction ID, also known as a TXID, is a unique identifier assigned to each transaction in a blockchain network. The transaction ID is a string of characters that is generated through a hashing algorithm, which converts the transaction data into a fixed-length, alphanumeric string.
- **Customer Address:** This field contains the wallet address of the customer whose transaction is stored. This field is important because it is used to differentiate one user from another in the blockchain.
- **Bank Access:** This field contains the wallet address of the FI. This field means that the FI has access to the KYC details of the customer with the address present in Customer Address Field.
- **Is Verified:** It is a simple boolean field which identifies whether the user is verified or not.
- **Ipfs Hash:** This field contains the hash of the KYC documents of the users.
- **Verified By Bank Address:** Once the user is verified by the FI, this field contains the address of the FI who verified the user.
- **Signature:** This is one of the important fields of the transaction as this field contains the cryptographically signed data. The signature is done by the transaction initiator. It can be used to verify if the transaction is really initiated by the entity or not.

5.2.2 Identification of Nodes

The FIs themselves are the blockchain nodes of the proposed system. They are the devices which are authorized to keep record of the distributed ledger. Their responsibility is to validate the accuracy of each transaction block adding it to the blockchain. Once the blocks are mined by the miner node, the blocks are broadcasted to all other participating nodes in order to synchronize the blockchain across all the nodes.

5.2.3 Designing Peer To Peer Network

Nodes in the blockchain communicate with each other via a Peer To Peer network. Every Financial Institution acts as a full node in our blockchain. They maintain their own copy of the chain. A random node acts as a main node in our implementation to provide a list of connected nodes and to provide all the blocks for the new node. Whenever a new node enters the platform, they download all the blocks from the main node. After the block transfer is complete, the Node can validate and mine the blocks. The miner node mines the incoming block and then adds it to its own chain. After that, the block is sent to all other nodes.

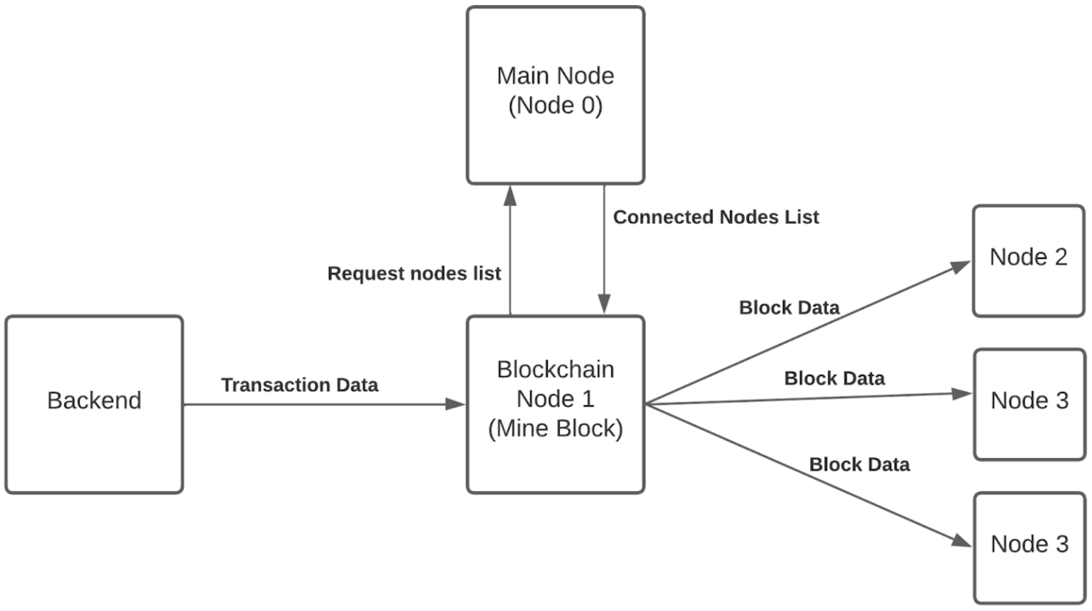


Figure 5.8: Block broadcast from miner node to other nodes.

5.2.4 Creating Wallets

A new wallet is created for each financial institution and customer during registration. For added security each wallet is encrypted with the user's password and stored securely on off-chain data storage, along with other registration details.

Wallet holders can access their wallets and use them to sign and broadcast transactions within the system. This allows them to securely interact with the blockchain network in a user-friendly way.

5.2.5 Signing and Storing Transactions

When signing transactions in the system, the wallet must be decrypted with a valid password. Once decrypted, the private key from the wallet is used to sign the transaction, resulting in a unique signature of the transaction data.This signature, along with the signer's public key and the transaction data, is then sent to the blockchain network. The blockchain network can verify the credibility of the signature using the signature, public key, and transaction data. If the signature matches, the transaction is added to the block. Otherwise, the transaction is discarded.

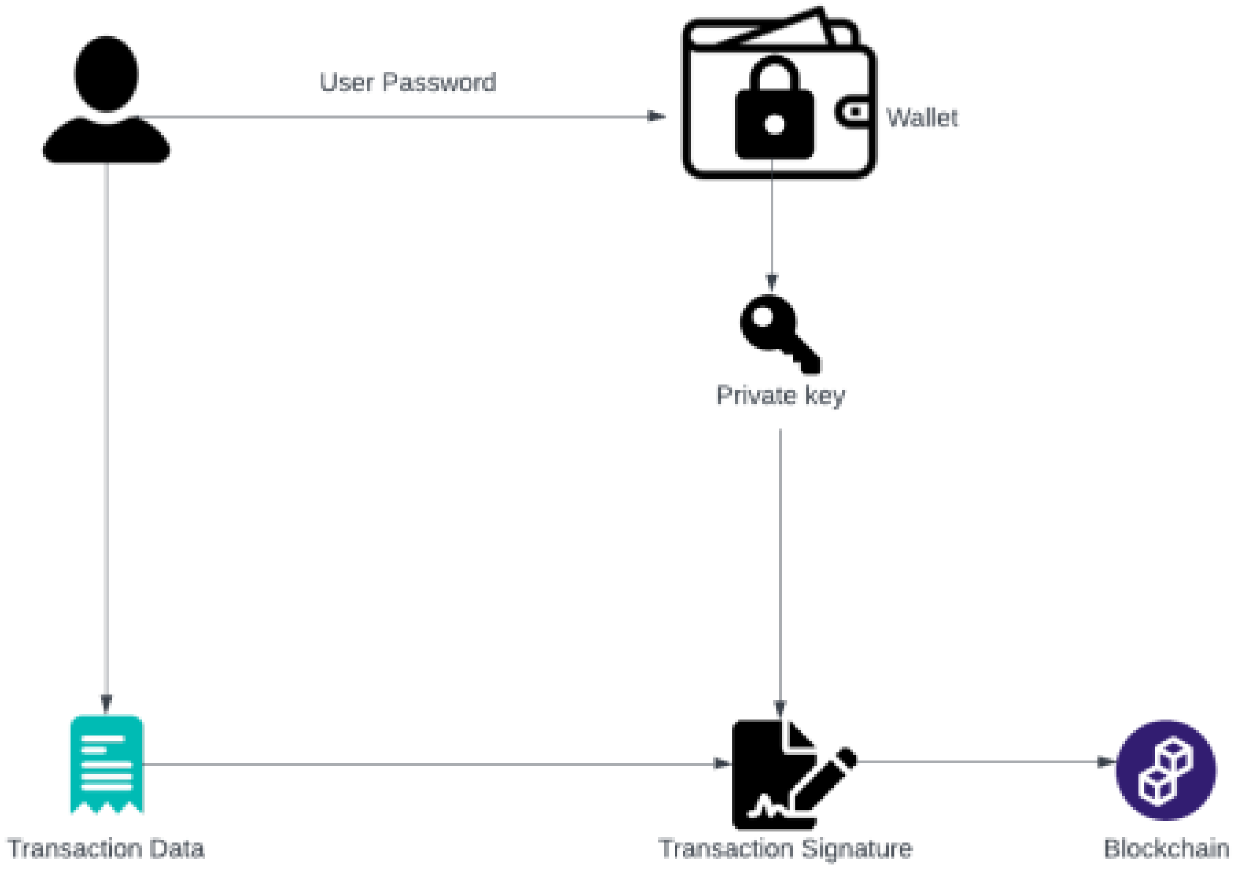


Figure 5.9: Signing of Transaction

This process ensures that only authorized parties with access to the correct private key can sign transactions and interact with the blockchain network. By verifying the signature and public key, the blockchain network can ensure the authenticity and integrity of transactions added to the block. This helps to maintain the security and trustworthiness of the blockchain network.

5.2.6 Mining and consensus algorithm

Block mining is the process by which new transactions are validated and added to the blockchain. Proof of work consensus algorithm is used to validate the transaction. Banks as full nodes use their resources to mine the blocks and add them in the chain. It is designed to prevent fraud and ensure the security and integrity of the blockchain.

Independent verification of the transaction is done by a full node, based on a comprehensive list of criteria. The verifying full node includes the transaction into a new block by mining, a demonstrated computation through a Proof-of-Work algorithm. The newly added block is broadcasted to other nodes in the peer to peer network. The nodes do independent verification of the new block and assemble the block into their own chain.

5.2.7 Private Distributed File Storage Integration

It is not cost-effective and efficient to store all KYC information in blockchain. Therefore, a third-party private distributed file storage system is used to store KYC information in encrypted format. Address or cryptographic hash of the file returned by the storage system is stored on the blockchain. The content of the file can only be accessed by connected banks with the user's consent.

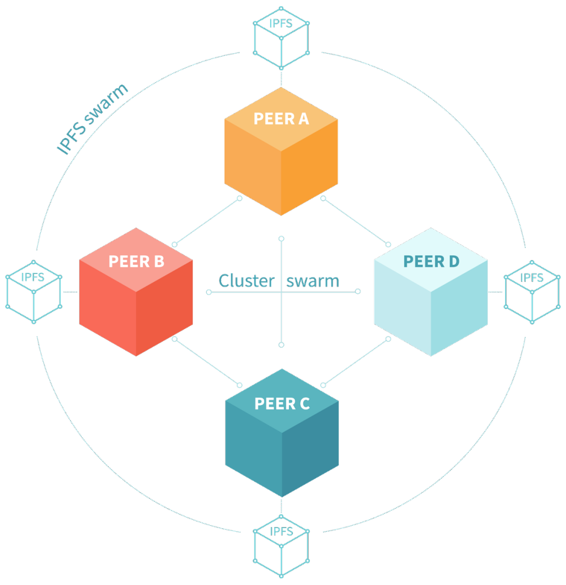


Figure 5.10: IPFS Swarm Cluster

IPFS Swarm Cluster is used as a Private distributed file storage. It is a group of nodes that are connected and share content with each other.

There are two different types of nodes:

- **Bootstrap Node:** The backend server acts as the bootstrap node and is responsible for managing the cluster and client nodes.
- **Client Node:** The FIs themselves are the client nodes of the IPFS swarm cluster. They store the encrypted KYC documents and are used to download and retrieve content from the network.

5.2.8 Off Chain Database Integration

Off-chain Database is used to store the information associated with the users, financial institutions (FIs) and KYC.

User details such as the email, hashed password, KYC status, encrypted wallet, wallet address, etc are stored. Bank details such as bank name, email, hashed password, registration number, address, phone number, encrypted wallet, wallet address, etc. are stored. KYC details associated with users are also stored.

5.2.9 Building the APIs

APIs are used to connect the blockchain server, backend and frontend part of our project. For sending the transaction data from Nodejs server to Go blockchain, a TCP socket is used. The data is sent as a stream of bytes delimited by a newline character. The data is passed in the following format:

```
header {
    from:     "nodejs",
    command:  "commandToPerform",
};

transactionData {
    customerAddress:  "0xAddr355",
    signature:        "0x51gnAtuR3",
    bankAddress:      "0xAddr355",
};
```

For the communication between central backend server and frontend, API is based on HTTP request/response. HTTP requests are sent by the frontend at the certain URL end-points and response is sent in the JSON format.

5.2.10 Designing User Interface and API Integration

A web application with a user-friendly interface has been developed to facilitate transactions for both users and financial institutions (FIs). To simplify the blockchain's intricacies, the application employs a password-based system for initiating transactions, which abstracts away the complexities of the blockchain wallet. During transactions, the password provided is used to decrypt the wallet, thereby retrieving the private key required for signing the transactions.

The application backend utilizes APIs exposed by the blockchain nodes to communicate with them, providing the necessary transaction data to initiate transactions within our application. Additionally, the backend fetches chain data from the blockchain nodes using the APIs they provide.

The frontend of the application uses APIs exposed by the backend for user registration, authentication, transaction initiation, and fetching data from distributed storage, off-chain storage, and the blockchain.

5.3 System Usage Workflow

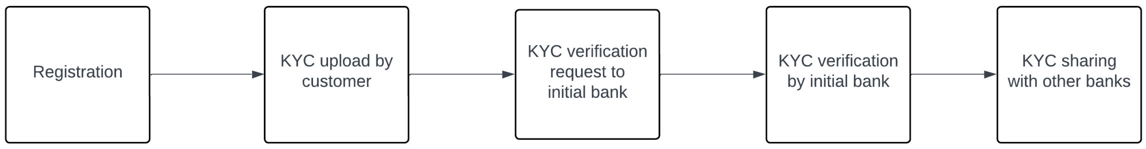

Registration
KYC upload by customer
KYC verification request to initial bank
KYC verification by initial bank
KYC sharing with other banks

Figure 5.11: System Usage Workflow

5.3.1 Registration

The users and FIs are registered to the platform using their email address and password. A unique wallet address is generated for each entity and is linked with the account. The wallet is encrypted by using the user's password and is stored in the database.

5.3.2 KYC upload

After registration on the platform, KYC details and documents are uploaded by the users. The documents are encrypted, stored in private distributed file storage and the returned hash is then stored in the blockchain.

5.3.3 KYC verification

The user can request one of the registered FI in order to verify the KYC. The requested FI can view the details shared by the user and can verify the user. Now, the user is verified and the verification status for the user is added in the form of transaction on the chain together with the identity of the verifier.

5.3.4 KYC sharing

Once the user is verified by a FI, the KYC details can be easily shared with other FIs. The user can grant access to the FI in order to view the KYC details if FI requests the user for the details. After the access is granted, a new transaction is initiated and stored in the blockchain.

6. Evaluation, Result and Analysis

6.1 Verification and Validation

The verification and validation process is an essential part of software development, ensuring that the product meets the requirements and performs as expected. Various techniques for verification and validation to ensure the quality and functionality of the blockchain-based KYC sharing system was employed.

6.1.1 Verification

Verification is the process of ensuring that the system meets the specified requirements. Following techniques were used for verification:

6.1.1.1 Code Review

The blockchain code was reviewed by experienced developers who have experience in blockchain development. The code review was focused on ensuring that the code followed best practices and coding standards for Golang and blockchain development. The review covered security, performance, and consistency. The review process included code walkthroughs and peer reviews to ensure that all issues were identified and resolved.

6.1.1.2 Unit Testing

Unit tests were written for the blockchain and other component's code to ensure that it worked as expected and met the project requirements. The unit tests were designed to cover all possible use cases and edge cases and were executed with every code change.

6.1.1.3 Integration Testing

Integration tests were performed to ensure that the blockchain, back-end and front-end components interact correctly with each other and with external dependencies. The tests were designed to cover all possible interactions between the components and external dependencies.

6.1.1.4 Code Coverage

Code coverage was measured to ensure that all important parts of the code were tested and to identify any parts of the code that were not covered by unit tests. It was used to guide further testing and identify any areas for improvement.

6.1.2 Validation

Validation is the process of ensuring that the system meets the user's needs and is fit for purpose. In our project, we used the following techniques for validation:

6.1.2.1 System Testing

Overall system and features were tested by creating multiple nodes simulating multiple banks and customers and sharing KYC data between them. This was done in a simulated production environment to ensure that the system would work correctly in the real world.

The tests covered the following scenarios:

1. Creating a new KYC record.
2. Sharing a KYC record between nodes.
3. Verification of record.
4. Recording of transactions on chain.

Finally results were recorded and analyzed for any issues or bugs.

6.1.2.2 User acceptance testing

This test ensured that the system was designed to meet users requirements. Based on the feedback from the user system was refined.

6.1.2.3 Performance testing

Load testing was performed by simulating multiple banks and users creating high network load.Stress testing on blockchain was also performed by creating a large number of blocks and transactions from different nodes.This test ensured that the overall system could handle large number nodes and high traffic condition without impacting the system's performance..

6.1.2.4 Persistence testing

To ensure that the KYC data, off-chain data, and blockchain data were persistently and reliably stored, persistence testing was performed. Test focused on the IPFS storage system to ensure that it could accurately store KYC data over time.Original uploaded data on IPFS was tested with the hash present on the blockchain to verify that the data was reliably stored and could be retrieved when needed. Additionally, tested the persistence of blockchain data during a server crash to ensure that the system could recover without any data loss.Overall, persistence testing on the different storage systems ensured reliability,accuracy,durability of data over time.

6.1.2.5 Security testing

Conducted security testing to identify any vulnerabilities in the system and ensure that it is secure and resilient against attacks such that unverified and invalid signature transactions were not added to the blockchain.

Overall, the verification and validation process was critical to ensuring the quality and functionality of our blockchain-based KYC sharing system. By using a combination of testing, peer review, and user acceptance testing, we were able to verify that the system met the specified requirements and validate that it was fit for purpose.

6.2 Results

6.2.1 Easy and Secure KYC Sharing

Decentralized KYC sharing was successfully implemented using ReactJS, NodeJs, and IPFs on top of the consortium blockchain. The system provides secure and transparent sharing of KYC data between different parties, with data stored on IPFs and hashes stored as transactions on the blockchain. Decentralized KYC sharing reduces the time and cost associated with KYC verification and increases privacy and security.

6.2.2 Intuitive User Interface

User feedback was collected to assess the usability and user experience of the implementation. Easy and intuitive user interface was developed to allow users to interact with the application without any advance knowledge of blockchain and web technologies.

6.2.3 Blockchain Development

The blockchain was designed and implemented using Golang, with a focus on simplicity and efficiency. The architecture of the blockchain supports the creation and validation of transactions, and includes features such as peer-to-peer communication and consensus mechanisms. The development process presented several challenges, such as ensuring consistency and synchronizing data across the network. These challenges were addressed through the implementation of robust consensus algorithms and data storage mechanisms.

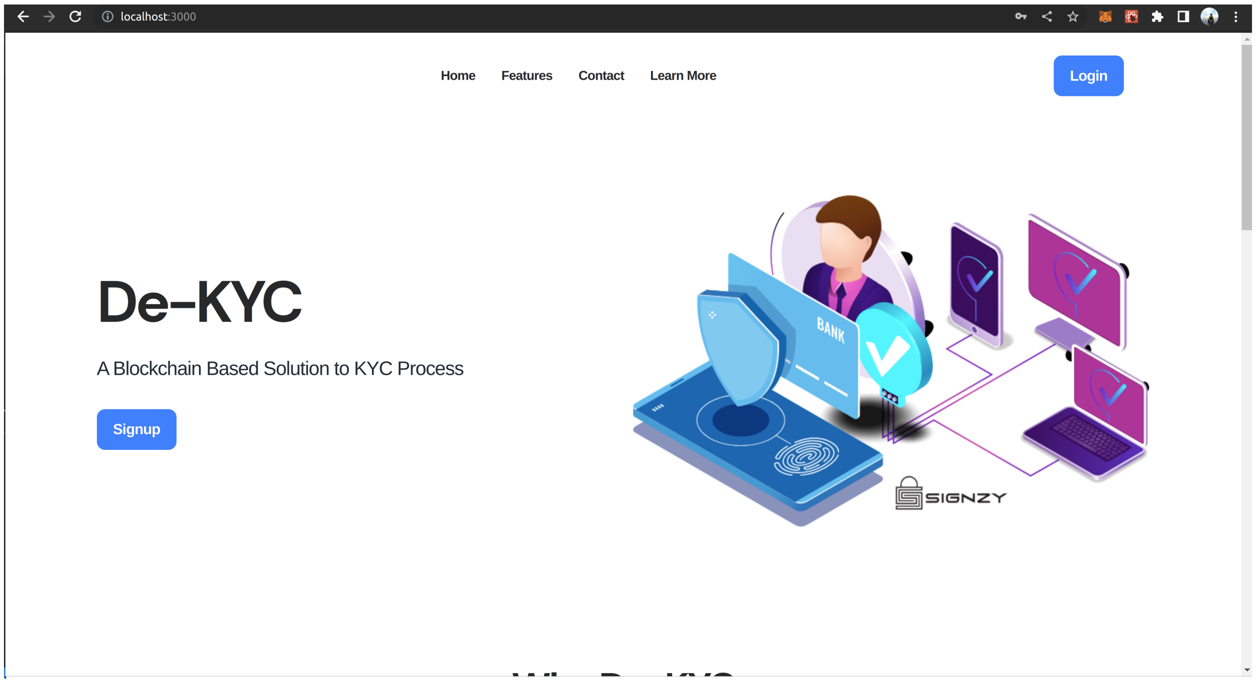
6.2.4 Efficient Storage

Mongo DB was used as an off-chain database to store data that was too extensive to store on the blockchain. The off-chain database stores user profiles, transaction history, and other metadata using cryptographic techniques such as hashing. This approach improves scalability, reduces the size of the blockchain, and reduces storage costs while ensuring the consistency and security of the data.

6.2.5 Crypto-less Blockchain

The blockchain was developed without involving any cryptocurrency. Instead, it was designed to support the decentralized sharing of KYC data. This approach provides a secure and efficient alternative to centralized KYC verification systems, while improving privacy and reducing costs.

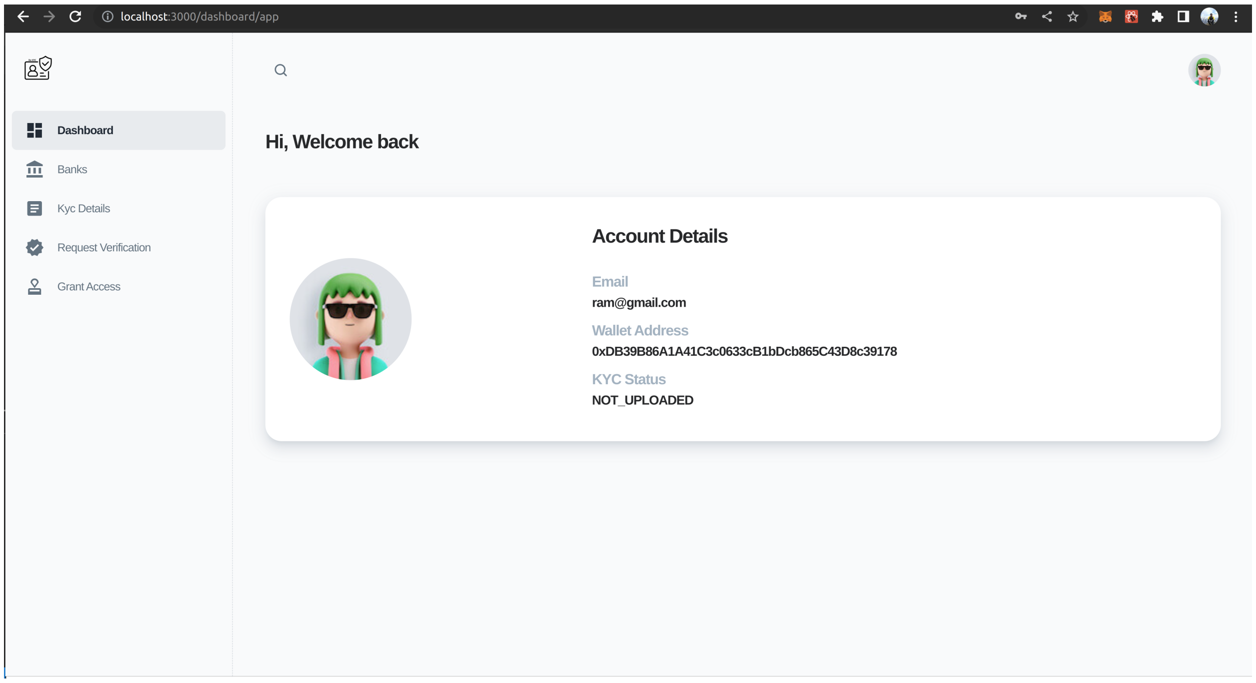
6.3 Outputs



De–KYC

A Blockchain Based Solution to KYC Process

Figure 6.1: De KYC Landing Page



Hi, Welcome back

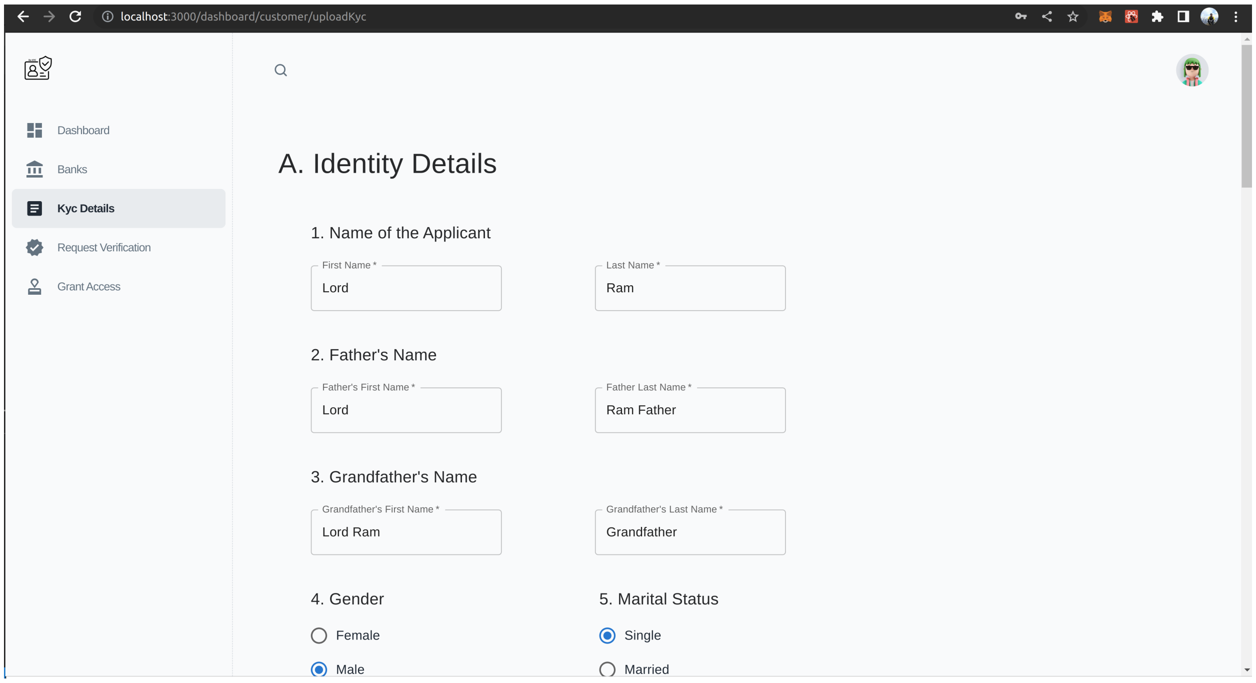
Account Details

Email
ram@gmail.com

Wallet Address
0xDB39B86A1A41C3c0633cB1bDcb865C43D8c39178

KYC Status
NOT_UPLOADED

Figure 6.2: Customer Profile



localhost:3000/dashboard/customer/uploadKyc

A. Identity Details

1. Name of the Applicant

First Name *: Lord
Last Name *: Ram

2. Father's Name

Father's First Name *: Lord
Father Last Name *: Ram Father

3. Grandfather's Name

Grandfather's First Name *: Lord Ram
Grandfather's Last Name *: Grandfather

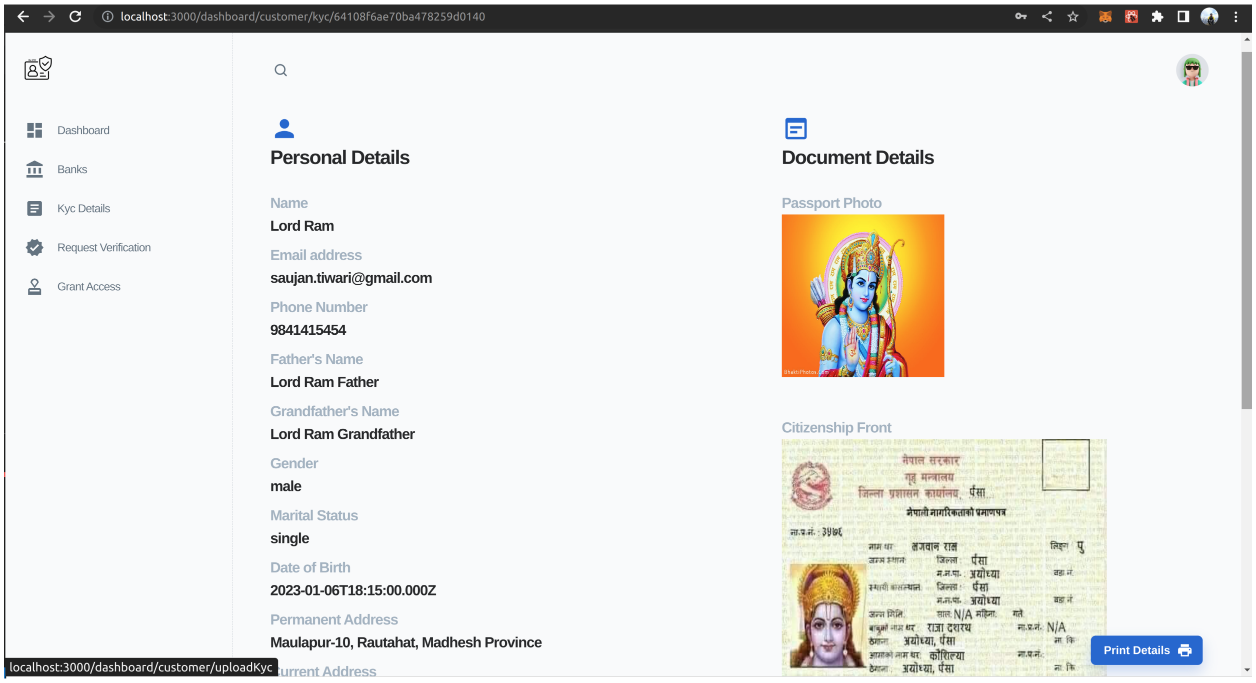
4. Gender

Female [✓ Male]

5. Marital Status

[✓ Single] Married

Figure 6.3: KYC Upload Form



localhost:3000/dashboard/customer/kyc/64108f6ae70ba478259d0140

Personal Details

Name
Lord Ram

Email address
saujan.tiwari@gmail.com

Phone Number
9841415454

Father's Name
Lord Ram Father

Grandfather's Name
Lord Ram Grandfather

Gender
male

Marital Status
single

Date of Birth
2023-01-06T18:15:00.000Z

Permanent Address
Maulapur-10, Rautahat, Madhesh Province

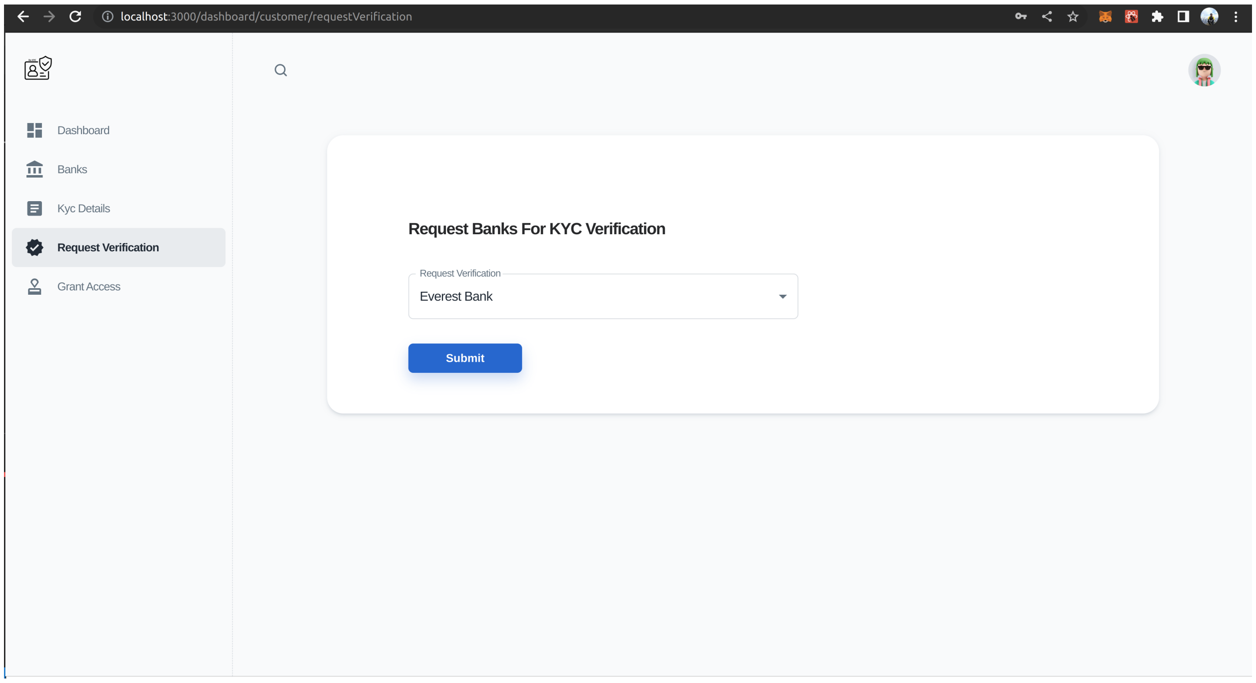
Current Address

Document Details

Passport Photo

Citizenship Front

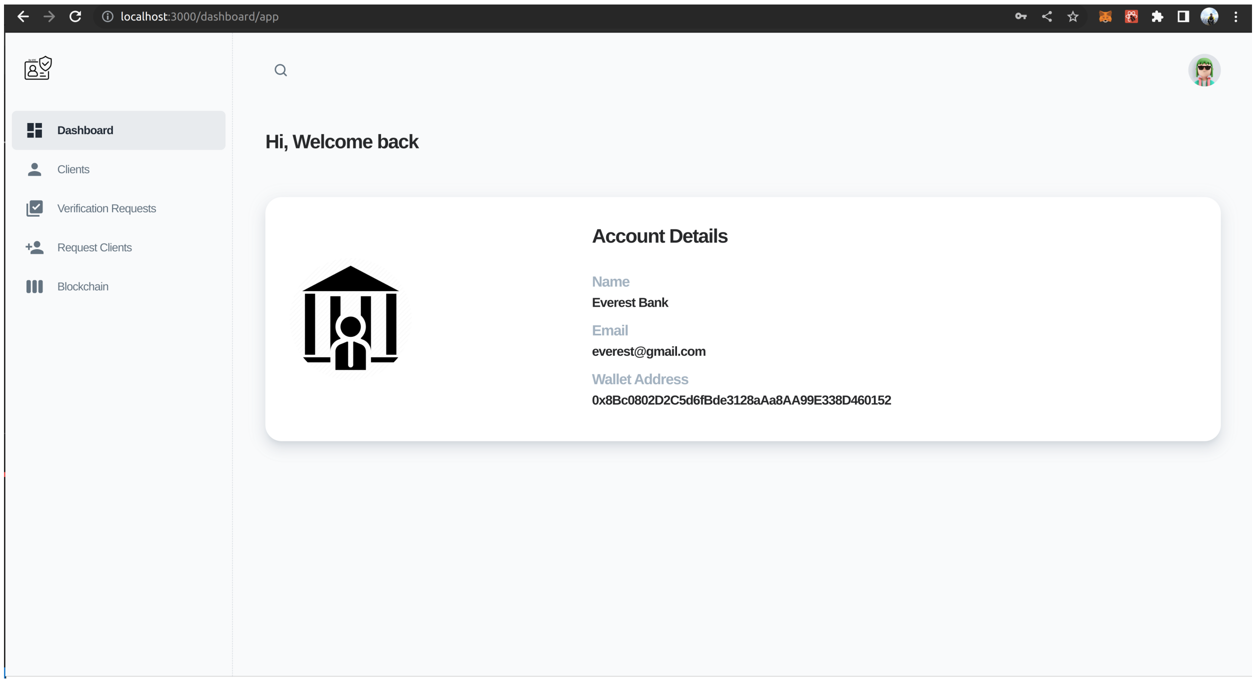
Figure 6.4: KYC Details Page



Request Banks For KYC Verification

Request Verification: Everest Bank

Figure 6.5: Request Verification



Hi, Welcome back

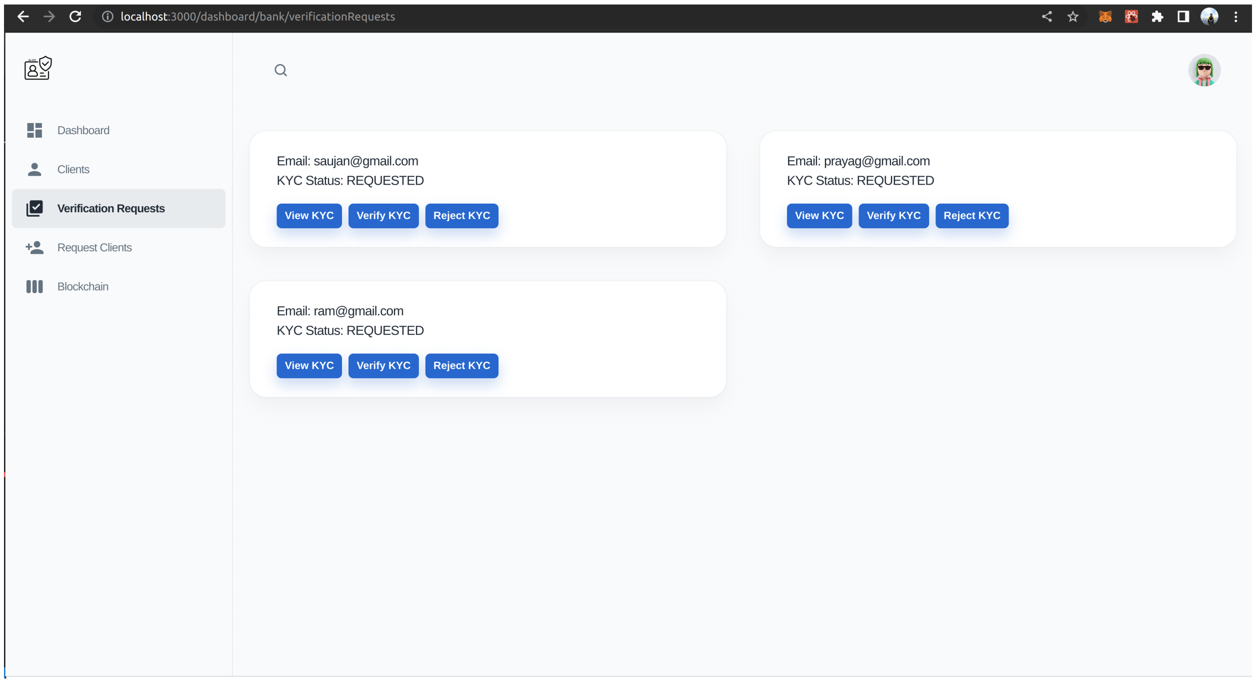
Account Details

Name
Everest Bank

Email
everest@gmail.com

Wallet Address
0x8Bc0802D2C5d6fBde3128aAa8AA99E338D460152

Figure 6.6: Bank Profile

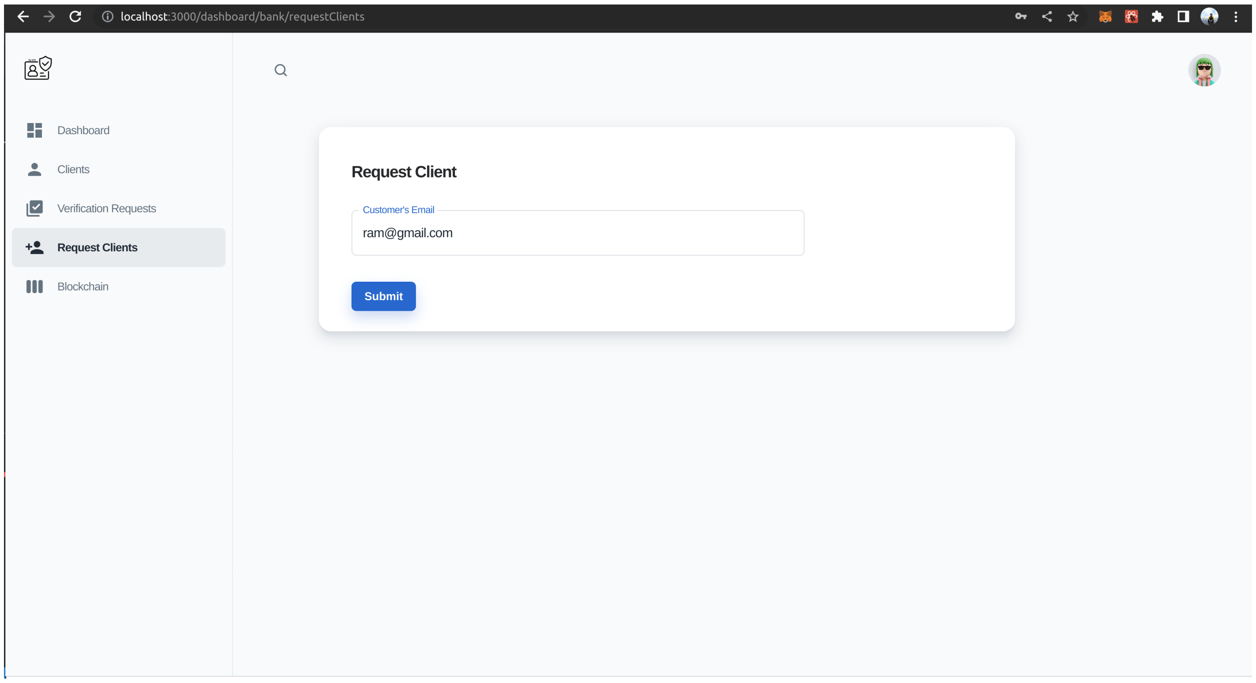


Email: saujan@gmail.com
KYC Status: REQUESTED

Email: prayag@gmail.com
KYC Status: REQUESTED

Email: ram@gmail.com
KYC Status: REQUESTED

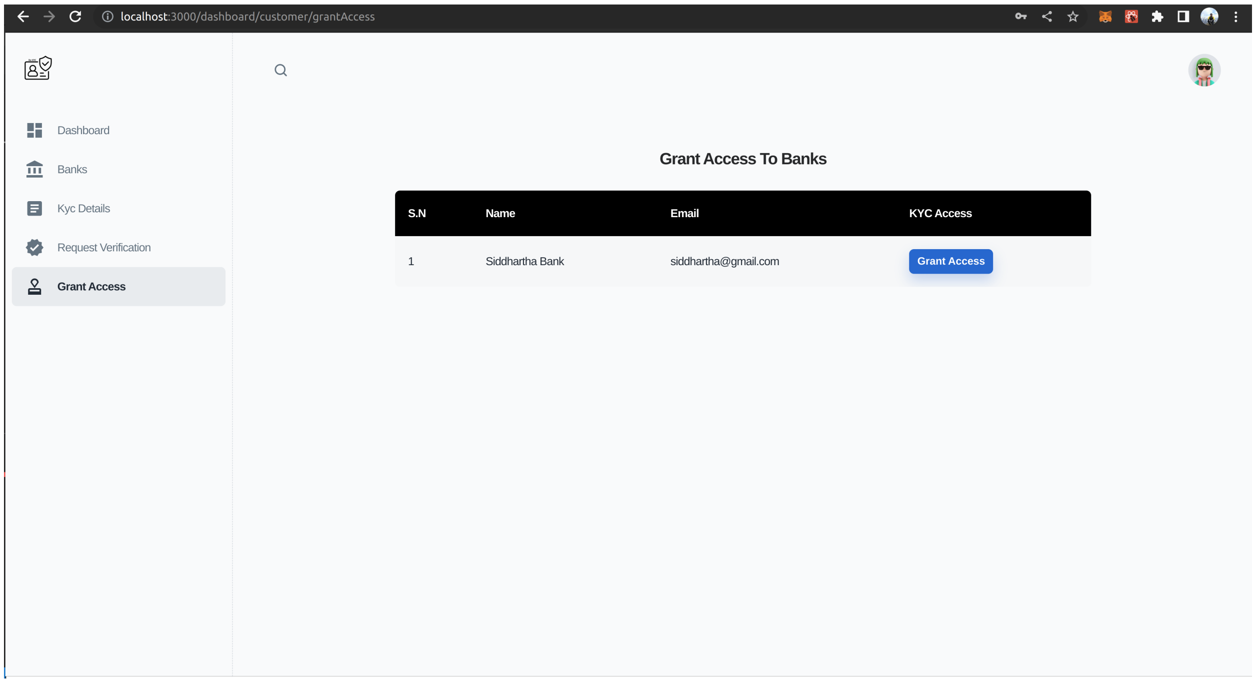
Figure 6.7: Verification Requests



Request Client

Customer's Email: ram@gmail.com

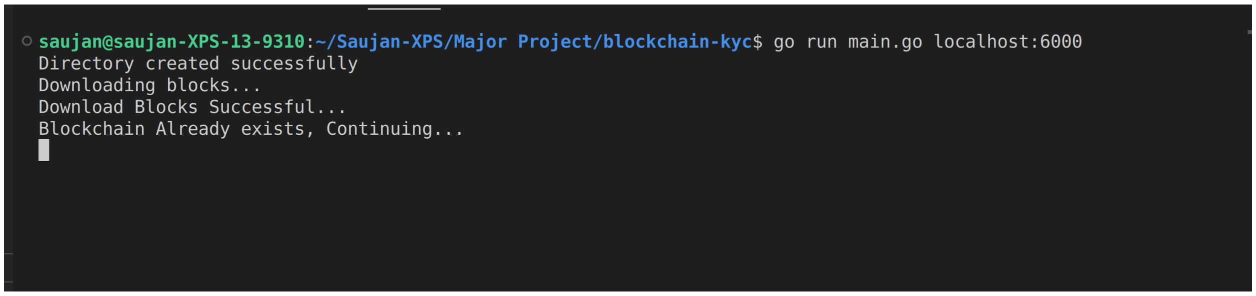
Figure 6.8: Request client for KYC



Grant Access To Banks

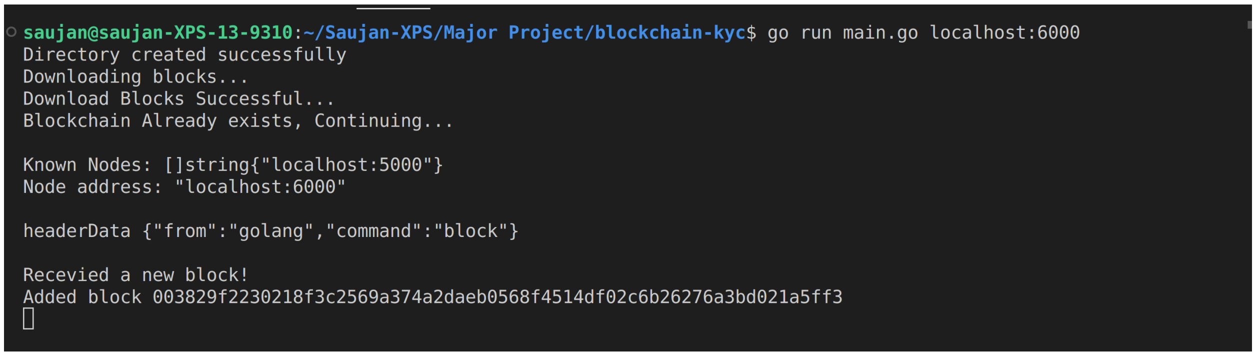
S.N	Name	Email	KYC Access
1	Siddhartha Bank	siddhartha@gmail.com	Grant Access

Figure 6.9: Grant Access to KYC



```
saujan@saujan-XPS-13-9310:~/Saujan-XPS/Major Project/blockchain-kyc$ go run main.go localhost:6000
Directory created successfully
Downloading blocks...
Download Blocks Successful...
Blockchain Already exists, Continuing...
```

Figure 6.10: Downloading blocks on first connection



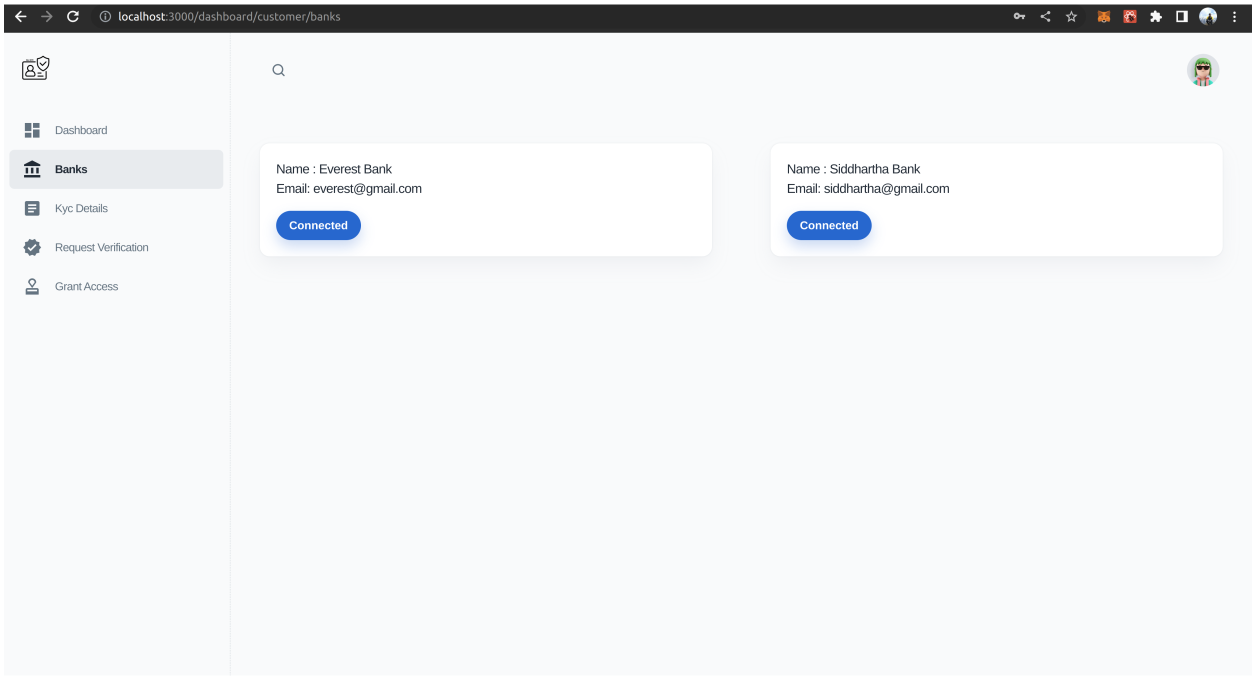
```
saujan@saujan-XPS-13-9310:~/Saujan-XPS/Major Project/blockchain-kyc$ go run main.go localhost:6000
Directory created successfully
Downloading blocks...
Download Blocks Successful...
Blockchain Already exists, Continuing...

Known Nodes: []string{"localhost:5000"}
Node address: "localhost:6000"

headerData {"from":"golang","command":"block"}

Recevied a new block!
Added block 003829f2230218f3c2569a374a2daeb0568f4514df02c6b26276a3bd021a5ff3
```

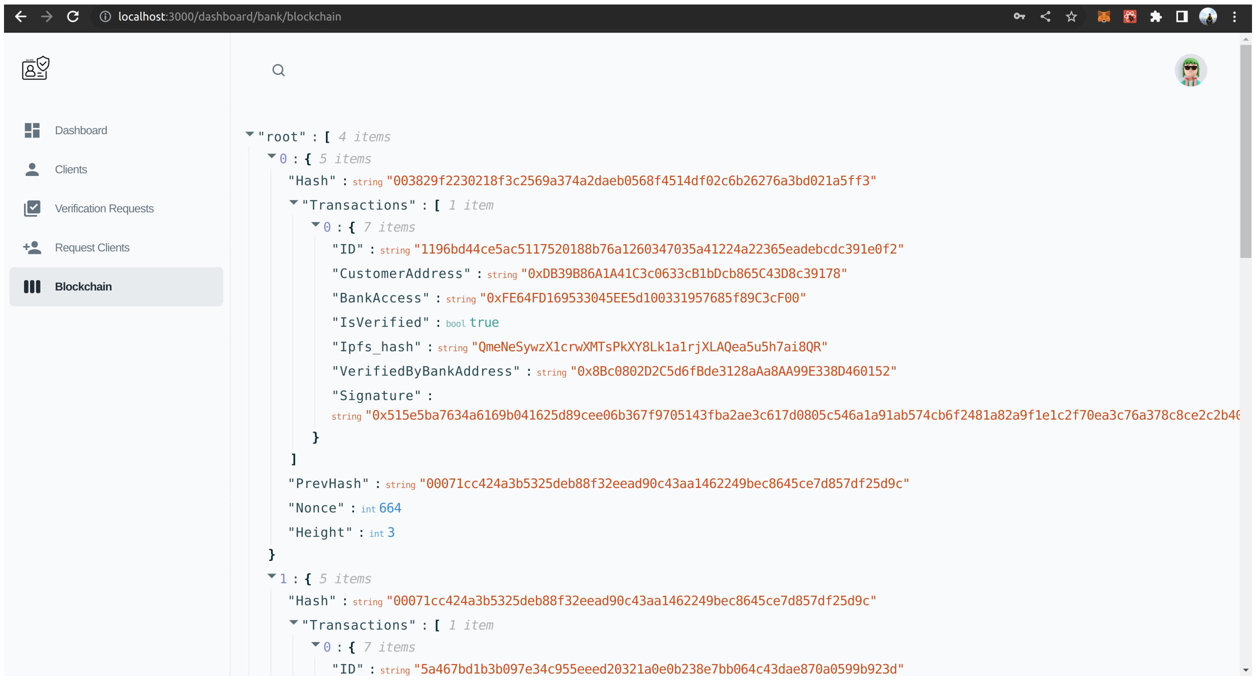
Figure 6.11: Receive a sent block



Name : Everest Bank
Email: everest@gmail.com
Connected

Name : Siddhartha Bank
Email: siddhartha@gmail.com
Connected

Figure 6.12: Bank users



```
"root" : [ 4 items
  0 : { 5 items
    "Hash" : string "003829f2230218f3c2569a374a2daeb0568f4514df02c6b26276a3bd021a5ff3"
    "Transactions" : [ 1 item
      0 : { 7 items
        "ID" : string "1196bd44ce5ac5117520188b76a1260347035a41224a22365eadebcdc391e0f2"
        "CustomerAddress" : string "0xDB39B86A1A41C3c0633cB1bDcb865C43D8c39178"
        "BankAccess" : string "0xFE64FD169533045EE5d100331957685f89C3cF00"
        "IsVerified" : bool true
        "Ipfs_hash" : string "QmeNeSywzX1crwXMTsPkXY8Lk1a1rjXLAQea5u5h7ai8QR"
        "VerifiedByBankAddress" : string "0x8Bc0802D2C5d6fBde3128aAa8AA99E338D460152"
        "Signature" :
        string "0x515e5ba7634a6169b041625d89cee06b367f9705143fba2ae3c617d0805c546a1a91ab574cb6f2481a82a9f1e1c2f70ea3c76a378c8ce2c2b4
      }
    ]
    "PrevHash" : string "00071cc424a3b5325deb88f32eead90c43aa1462249bec8645ce7d857df25d9c"
    "Nonce" : int 664
    "Height" : int 3
  }
  1 : { 5 items
    "Hash" : string "00071cc424a3b5325deb88f32eead90c43aa1462249bec8645ce7d857df25d9c"
    "Transactions" : [ 1 item
      0 : { 7 items
        "ID" : string "5a467bd1b3b097e34c955eeed20321a0e0b238e7bb064c43dae870a0599b923d"
```

Figure 6.13: Blockchain Explorer

6.4 Analysis

6.4.1 Benefits of decentralized KYC sharing

Decentralized KYC sharing allows for the secure sharing of KYC data among different organizations without a centralized intermediary. This reduces the risk of data breaches and improves data privacy. It also enables faster onboarding processes for customers and reduces costs for businesses by eliminating the need for redundant KYC checks. By leveraging decentralized technologies, businesses can improve their KYC processes and provide a better experience for their customers.

6.4.2 Advantages of IPFS storage mechanism

Use of IPFS storage mechanism ensures data privacy as data is not stored on a centralized server and can only be accessed by authorized nodes. Additionally, IPFS enables efficient and scalable storage of data, which is essential in a decentralized system.

6.4.3 Limitations of a crypto-less blockchain

A cryptocurrency-less blockchain may be limited in terms of incentivization as it lacks a native cryptocurrency to reward users for their participation. This could make it challenging to maintain the blockchain. Additionally, it is not suitable for blockchain applications that rely on a cryptocurrency component to function properly.

6.4.4 Potential of blockchain technology

Blockchain technology's secure and efficient data sharing capabilities can be applied to other industries and use cases where data privacy and security are paramount. By leveraging blockchain, organizations can store and share data in a secure, transparent, and decentralized manner.

Overall, our project showcases the potential of blockchain technology to transform the way we handle sensitive data, specifically in the area of KYC sharing. By using a decentralized system that is secure and efficient, we have created a practical solution to the challenge of securely sharing KYC data among different organizations.

7. Conclusion and Future Enhancements

7.1 Conclusion

We have successfully implemented a decentralized KYC sharing system using blockchain technology that does not involve cryptocurrency. The blockchain was designed and implemented with a focus on simplicity and efficiency, and the off-chain database and IPFS storage mechanism used provides scalability and reduces storage costs while ensuring the consistency and security of the data. The implementation provides a secure and efficient alternative to centralized KYC verification systems, reduces the time and cost associated with KYC verification, and increases privacy and security. The project demonstrates the potential of blockchain technology in developing decentralized and efficient systems that can improve existing processes in various industries, and contributes to the advancement of blockchain technology.

7.2 Future Enhancements

While we have put in our best efforts to complete our project, there is always room for improvement and future enhancements. Some potential areas for future improvement and development include:

- **Implementation of KYC update mechanism:** The implementation could be extended to update the existing KYC of the customers.
- **Improved user interface and user experience:** The user interface and user experience of the implementation could be improved to make it more intuitive and user-friendly, thus increasing adoption and usage.
- **Integration with AI and machine learning:** The implementation could be extended to integrate with AI and machine learning algorithms, enabling advanced data analysis and decision-making.
- **Adoption of more efficient consensus mechanisms:** The implementation could be further optimized by adopting more efficient consensus mechanisms, such as proof-of-stake, to reduce energy consumption and increase scalability.

References

[1] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system.

[2] Vitalik Buterin. Ethereum: A next- generation smart contract and decentralized application platform.

[3] Juan Benet. Ipfs - content addressed, versioned, p2p file system.

[4] Andreas Unterweger Fabian Knirsch and Dominik Engel. Implementing a blockchain from scratch: why, how, and what we learned.

[5] Hong Kong Monetary Authority. Whitepaper on distributed ledger technology.

[6] José Parra Moyano and Omri Ross. Kyc optimization using distributed ledger technology. *Business and Information Systems Engineering*, 59(6), 2017.

[7] Syed Azhar Hussain and Zeeshan-ul-hassan Usmani. Blockchain-based decentralized kyc (know-your-customer). In *ICSNC 2019: The Fourteenth International Conference on Systems and Networks Communications*. IARIA, 2019.

[8] Prince Sinha and Ayush Kaul. Decentralized kyc system. *International Research Journal of Engineering and Technology (IRJET)*, 2018.

[9] https://www.simplilearn.com. A definitive guide to learn the sha-256 (secure hash algorithms), 2023.

Appendices

ECDSA

ECDSA (Elliptic Curve Digital Signature Algorithm) is a public key cryptosystem that is widely used for secure communication over the internet. It is based on the mathematics of elliptic curves and is an efficient alternative to traditional digital signature algorithms such as RSA.

The algorithm is used to generate digital signatures for electronic documents and is based on the use of elliptic curves in finite fields. The security of the algorithm is based on the complexity of the discrete logarithm problem in these finite fields.ECDSA does not encrypt or prevent someone from seeing or accessing data, what it protects against though is making sure that the data was not tampered with.

The process of generating a digital signature using ECDSA involves two steps: key generation and signature generation.

Key Generation

In the first step, generate a public-private key pair using an elliptic curve in a finite field. The public key is shared with others, while the private key is kept secret. The key pair is generated using the following steps:

1. Select an elliptic curve over a finite field and a point G on the curve, which is used as the generator point.

2. Choose a private key k randomly from the set 1, 2, . . . , n-1, where n is the order of the generator point G.

3. Compute the public key point Q = k * G, where * represents the elliptic curve point multiplication operation.

Signature Generation

In the second step, generate a digital signature for a document using private key and the ECDSA algorithm. The signature is verified by corresponding public key. The signature generation process involves the following steps:

1. Hash the document to be signed using a cryptographic hash function to generate a message digest.

2. Generate a random number r from the set (1, 2, . . . , n-1), where n is the order of the generator point G.

3. Compute a point R = r * G on the elliptic curve.

4. Compute the value s = (hash + k * r) / d mod n, where hash is the message digest, d is the user's private key, and mod n means that the result is reduced modulo n.

5. The digital signature is the pair (R, s).

Signature Verification

In ECDSA, a digital signature is a pair of numbers (r, s) that are derived from the message being signed and the sender's private key. The receiver verifies the signature by using the sender's public key, which is a point on an elliptic curve, along with the signature and the hash of the message.
The verification process involves the following mathematical steps:

1. The receiver computes the value of the hash of the message using a secure hash function.

2. The receiver then computes a value called the inverse of s. This inverse value is computed by finding a value s^{-1} such that $s * s^{-1}$ mod n = 1, where n is the order of the curve. This modular inverse of s is used in the next step of the verification process.

3. Next the receiver computes two points on the elliptic curve. The first point P is computed as s multiplied by the base point G of the elliptic curve plus the hash value of the message multiplied by the public key point Q of the sender. The second point is the point on the elliptic curve represented by the pair of numbers (r, s), which is the signature obtained from the sender.The base point G is a well-defined point on the elliptic curve that is known to both the sender and the receiver. In ECDSA, the base point G is a fixed point on the elliptic curve that is chosen at the time of selecting the elliptic curve parameters.

4. The receiver then checks whether the first point P obtained is the same as the second point obtained from the signature (r, s). If the points match, then the signature is verified to be valid. This is because the signature is generated by computing the scalar multiplication of the base point G with a random scalar k and then decomposing it into two numbers r and s. In the verification process, the receiver computes a point

P using the scalar multiplication of s with G and adds the hash value of the message multiplied by the public key point Q. If the signature is valid, the two points P and (r, s) will be the same.