



**COVIBLOCK: A MERKLE DAG AND BLOCKCHAIN
IMPLEMENTATION FOR COVID-19 RECORDS**

A Capstone Project Presented to the Graduate Program
College of Engineering and Technology
Pamantasan ng Lungsod ng Maynila

In Partial Fulfillment of the Requirements for the Degree
Master in Information Technology

By
Jennifer L. Fadriquela

Dr. Khatalyn E. Mata
Adviser

August 2021



APPROVAL SHEET

The capstone project hereto titled

COVIBLOCK: A MERKLE DAG AND BLOCKCHAIN IMPLEMENTATION FOR COVID-19 RECORDS

prepared and submitted by Jennifer L. Fadriquela in partial fulfilment of the requirements for the degree of Master in Information Technology has been examined and is recommended for acceptance and approval for **ORAL EXAMINATION**.

DR. KHATALYN E. MATA

Adviser

PANEL OF EXAMINERS

Approved by the Committee on Oral Examination
with a grade of _____ on _____.

PROF. MANUEL L. OCAMPO

Panel Chair
Chairman

PROF. EDGARDO S. DAJAO

Panel Member
Member

PROF. DAN MICHAEL A. CORTEZ

Panel Member
Member

Accepted and approved in partial fulfilment of the requirements for the degree of
Master in Information Technology

DR. JOSEPH BERLIN P. JUANZON

Director, CET
Graduate Program

ENGR. JUAN C. TALLARA JR.

Dean
Graduate Program



TABLE OF CONTENTS

| | |
|-------------------------------------|----|
| TITLE PAGE | 1 |
| APPROVAL SHEET | 2 |
| TABLE OF CONTENTS | 3 |
| LIST OF FIGURES | 4 |
| LIST OF TABLES | 5 |
| INTRODUCTION | 6 |
| 1.1 Background of the Study | 6 |
| 1.2 Statement Problem | 8 |
| 1.3 Objectives of the Study | 8 |
| 1.4 Scope and Limitations | 8 |
| 1.5 Significance of the Study | 9 |
| 1.6 Definition of Terms | 10 |
| REVIEW OF RELATED LITERATURE | 11 |
| THEORETICAL FRAMEWORK | 21 |
| METHODOLOGY | 33 |
| 4.1 Requirements Modeling | 34 |
| 4.2 Quick Design | 42 |
| LIST OF REFERENCES | 50 |



LIST OF FIGURES

| | |
|---|----|
| Figure 3.0.1: Diagram of Proposed Solution | 21 |
| Figure 3.0.2: Hashing Process..... | 22 |
| Figure 3.0.3: Generic Blockchain Transactions..... | 25 |
| Figure 3.0.4: Clique PoA Block Creation Process | 26 |
| Figure 3.0.5: Merkle Tree Implementation using hashes..... | 28 |
| Figure 3.0.6: DAG Illustration | 28 |
| Figure 3.0.7: Merkle DAG implemented on a file system..... | 29 |
| Figure 3.0.8: Asymmetric Encryption and Decryption Process..... | 30 |
| Figure 3.0.9: Conceptual Framework..... | 31 |
| Figure 4.0.1: Prototype Model Phases and Process | 33 |
| Figure 4.0.2: Sample Rapid Antigen Test Result..... | 35 |
| Figure 4.0.3: Sample Real-Time Polymerase Chain Reaction (RT-PCR) Test Result.... | 36 |
| Figure 4.0.4: Sample COVID-19 Vaccination Certificate | 37 |
| Figure 4.0.5: Merkle DAG representing sample records | 40 |
| Figure 4.0.6: Context Diagram | 42 |
| Figure 4.0.7: Data Flow Diagram | 43 |
| Figure 4.0.8: Proposed Use Case Diagram | 44 |
| Figure 4.0.9: Transactional Operation Diagram | 45 |
| Figure 4.0.10: Key Generation Process Flowchart | 45 |
| Figure 4.0.11: Third party access to patient file Flowchart | 46 |
| Figure 4.0.12: File Uploading Flowchart..... | 47 |
| Figure 4.0.13: File Retrieval Flowchart | 48 |



LIST OF TABLES

| | |
|--|----|
| Table 4.0.1: Generated Hash Value for Sample Record #1 | 40 |
| Table 4.0.2: Generated Hash Value for Sample Record #2 | 40 |



Chapter One

INTRODUCTION

1.1 Background of the Study

With the advancement of computer technology, electronic documentation and the use of electronic medical records have become more feasible. Medical records on a shared computer network that are read and written electronically on a relational database using a graphic user interface are referred to as electronic medical records. In the study entitled “A comparison of electronic records to paper records in mental health centers” (Tsai and Bond, 2007), they looked at three mental health facilities that had recently switched from paper to electronic medical records. Electronic records' documentation was shown to be more thorough and retrievable than paper records. As per the study, this finding can be a factor to take in when making treatment decisions.

In the study entitled “Perceived Benefits of Implementing and Using Hospital Information Systems and Electronic Medical Records” (Khalifa, 2018), they pointed out six ways EMRs could enable data accessibility and care organization: improving access to data during patient encounters, improving processes workflow, managing information overflow to clinicians, enhancing medical decision-making process care plans, supporting operational processes and improving financial data accessibility. They also emphasized that when a computer was used to retrieve patient information, physicians earned higher overall patient satisfaction rates, and when a computer was used to enter patient information, physicians received identical satisfaction rates.

The current technological advancements in the Philippines has yet to be manifested in its healthcare system. Though there were efforts from the government to adopt various modern tools, we are still miles behind other countries. On a study entitled “Barriers to the Adoption of Electronic Medical Records in Select Philippine Hospitals: A Case Study



Approach” (Ebardo and Celis, 2019), identified barriers such as weak infrastructure, technology complexity and poor interface design of applications have made it difficult for various health organization to progress. Another study entitled “Barriers to Electronic Health Record System Implementation and Information Systems Resources: A Structured Review” (Gesulga et al., 2017), they determined another set of barriers to the adoption of EMRs in the Philippines namely: User resistance, lack of education and training, and concerns arising from data security. In a paper entitled “Identifying Healthcare Information Systems Enablers in a Developing Economy” (Ebardo and Tuazon, 2019), they discussed how the integration of existing information systems to be “paper-less” can produce potential savings. This is crucial given that the Philippines is still a developing country and has budget constraints to health systems.

Ospital ng Makati, a government-funded hospital, is one of the main facilities for COVID-19 patients in the City of Makati. Being a public hospital, it is expected to handle larger demographics compared to its private counterparts. Since the start of the pandemic, it is one of the primary health care facility in the city that facilitates COVID-19 test results and vaccination.

Presently, the hospital has no in-house molecular laboratory. They send requests to third-party laboratories in executing those tests. After successfully doing the tests, those laboratories will forward their results back to the hospital. These results are then given physically to the patient while the hospital keeps a copy in their archive. For vaccination, the City of Makati has an online web portal to assist with scheduling. This online system aims to lessen the crowd going to the vaccination site as they are guaranteed of a slot as opposed to other cities that implement a “first come first serve” basis. The proof of vaccination is a physical certificate and the patient being tagged as fully vaccinated in the system.

Other cities also had the same effort of putting up systems to cater the pandemic needs. Manila, Mandaluyong and Taguig also have their own sets of application. Though the



motivation is good, the issue of how to unify these applications have risen and calls for having a unified system on a national level are being thrown in various media outlets.

1.2 Statement Problem

At present, Ospital ng Makati is using a system for keeping COVID-19 Test Results while vaccination records are maintained on a separate system used by the entire City of Makati. The patients only receive physical copies of these records as proof of execution.

Current setup for organizing and managing these records still has shortcomings. One aspect that the researcher can improve is the strategy for storing records since this is still stored or archived physically both by the hospital and patients.

1.3 Objectives of the Study

This study aims to develop an alternative platform to store COVID-19 related records for Ospital ng Makati.

Specifically, the study seeks to address the following objectives:

1. To develop an application that will decentralize storage of COVID-19 related files.
2. To provide an alternative way to minimize record tampering of uploaded files in CoviBlock by applying concepts of Merkle DAG and Proof of Authority (PoA) blockchain.
3. To secure uploaded files in CoviBlock by using asymmetric cryptography.

1.4 Scope and Limitations

The study will be focusing on developing an application for management of COVID-related records for Ospital ng Makati. Since there are privacy regulations concerning



health information, the researcher will use dummy data and instead will probe more on the processes on how these records are archived or managed.

The study will exclude the test results generation and vaccine management. Thus, it is more focused on how the result or outputs of these processes. The study assumes that outputs are already generated in computer readable format such as images (.png, .jpg) or documents (.pdf).

The study will only be concerned on two types of records: Test Results and Vaccine Certificates. The researcher will concentrate on developing an alternative storage system and accessibility strategy for medical units, patients and other verifying party.

1.5 Significance of the Study

Results obtained from the study will benefit the following stakeholders:

Patients. Above all, patients will greatly benefit on this application. Various regulations and laws have been implemented to ensure people are not spreaders or vaccinated. Currently, there are no unified way in getting and presenting these records are proof. More so, bad actors are using this pandemic to make money out of tampering records. The application will help solve the woes of patients in terms on ease of access and portability of their records. They will also have full autonomy of said records.

Medical Personnel. The application will help medical workers to focus on their medical line of duty and alleviating various admin jobs.

Third Party Validators. As mentioned above, records tampering has become rampant. Businesses or employers requiring such records can now be protected of this illegal activity.



1.6 Definition of Terms

Cipher text - A series of randomized letters and numbers which humans cannot make any sense of.

Content addressing - A way to find data in a network using its content rather than its location.

Content Identifier (CID) - A label used to point to material in IPFS. It doesn't indicate where the content is stored, but it forms a kind of address based on the content itself. CIDs are short, regardless of the size of their underlying content.

Cryptography - Science of secret writing with the intention of keeping the data secret.

Digital Envelope - A secure electronic data container that is used to protect a message through encryption and data authentication.

Digital Signature - A cryptographic value that is calculated from the data and a secret key known only by the signer.

Distributed Hash Table (DHT) - A decentralized data store that looks up data based on key-value pairs.

Hash Digest - Output of the hash function.

Hashing - Process that calculates a fixed-size bit string value from a file.

Hash Table - A type of data structure that stores key-value pairs. The key is sent to a hash function that performs arithmetic operations on it.

InterPlanetary File System (IPFS) - A protocol and peer-to-peer network for storing and sharing data in a distributed file system.

Peer-to-Peer (P2P) Network - A group of computers are linked together with equal permissions and responsibilities for processing data.

Plain Text - Clear, basic unencrypted string of text.

Private Key - Used to decrypt cipher text to plain text and only available to its owner.

Public Key - Used to encrypt plain text to cipher text and available to anyone accessing the application.



Chapter Two

REVIEW OF RELATED LITERATURE

This chapter covers studies and other literatures carried out by foreign and domestic researchers that have a significant impact on the variables investigated in this study. These studies focus on several factors that will help with the research's development. Literatures mentioned here will be of different sources: books, journals, articles, electronic materials such as PDF or E-Book, and other existing thesis and dissertations, foreign and local. Their inclusion will be considered supplemental in developing the proposed solution of this study.

Merkle Tree

In 1989, Ralph Merkle introduced the Merkle tree in his paper “A Certified Digital Signature”. The Merkle tree is a tree constructed bottom-up. More precisely, the tree discussed in this paper is a full binary tree and constructed from the bottom-up. Assume that the height of the tree is h_m , and the tree owns 2^{h_m} data blocks x_i and $y_i = \text{hash}(x_i), i \in [0, 2^{h_m} - 1]$, where y_i is a leaf node value of the Merkle tree. Each value of the parent node is the hash of the concatenation of its children, $y_{\text{parent}} = \text{hash}(y_{\text{left}} | y_{\text{right}})$, where $|$ refers to concatenation. Below is a pseudocode format of the Classic Merkle Tree Traversal algorithm:

1. Set $\text{leaf} = 0$.
2. Output:
 - Compute and output leaf with $\text{LEAFCALC}(\text{leaf})$
 - For each $h \in [0, H - 1]$ output $\{\text{auth}_h\}$.
3. Refresh Auth Nodes:
For h such that 2^h divides $\text{leaf} + 1$:
 - Set auth_h be the sole node value in stack_h .
 - Set $\text{startnode} = (\text{leaf} + 1 + 2^h) \oplus 2^h$.



- $stack_h.initialize(startnode, h)$.

4. Build Stacks:

For all $h \in [0, H - 1]$:

- $stack_h.update(2)$.

5. Loop

- Set $leaf = leaf + 1$.
- If $leaf < 2^H$ go to Step 2

A Logarithmic Merkle Tree Traversal was proposed by M. Szydlo (2003). The main idea of the improved algorithm is, to reduce the memory requirements, by reducing the number of active treehash instances during the signature generation.. Here is the pseudocode:

1. Set $leaf = 0$.

2. Output:

- Compute and output leaf with $LEAFCALC(leaf)$
- For each $h \in [0, H - 1]$ output $\{auth_h\}$.

3. Refresh Auth Nodes:

For h such that 2^h divides $leaf + 1$:

- Set $auth_h$ be the sole node value in $stack_h$.
- Set $startnode = (leaf + 1 + 2^h) \oplus 2^h$.
- $stack_h.initialize(startnode, h)$.

4. Build Stacks:

Repeat the following $2H - 1$ times:

- Let l_{min} be the minimum of $\{stack_h.low\}$ for all $h = 0, \dots, H - 1$.
- Let focus be the least h so that $stack_h.low = l_{min}$.
- $Stack_{focus}.update(1)$.

5. Loop

- Set $leaf = leaf + 1$.
- If $leaf < 2^H$ go to Step 2.



In Fractal merkle tree representation (Micali et al., 2003) and traversal, the goal is to divide the merkle tree in subtrees and to preserve and compute these subtrees, instead of single nodes. Below is the pseudocode:

1. Set $leaf = 0$.
2. Output:
 - Compute and output leaf with $LEAFCALC(leaf)$
 - For each $j \in [0, H - 1]$ output $\{auth_j\}$.
3. Next Subtree:

For each i for which $Exist_i$ is no longer needed, i.e., for $i \in \{1, 2, \dots, L\}$ with $leaf = 1(mod 2^{h_i})$:

 - Set $Exist_i = Desire_i$.
 - Create new empty $Desire_i$ (if $leaf + 2^{ih} < 2^H$).
4. Grow Subtrees

For each $i \in \{1, 2, \dots, h\}$: Grow tree $Desire_i$ by applying 2 units to modified treehash (unless $Desire_i$ is completed)
5. Increase $leaf$ and return back to step 2 (while $leaf < 2^H$).

Distributed Hash Tables

Distributed Hash Tables (DHTs) are widely utilized to manage metadata for peer-to-peer systems. For example, the BitTorrent MainlineDHT monitors sets of peers' part of a torrent swarm. Kademlia was introduced in a paper titled "Kademlia: A peer-to-peer information system based on the xor metric" (Maymounkov and Mazieres, 2002). It is a DHT which provides:

1. Efficient lookup through massive networks: queries on average contact $d \log_2(n)$ nodes.
2. Low coordination overhead: it optimizes the number of control messages it sends to other nodes.
3. Resistance to various attacks by preferring long-lived nodes.



4. Wide usage in peer-to-peer applications, including Gnutella and BitTorrent, forming networks of over 20 million nodes.

In “Democratizing content publication with coral” (Freedman et al., 2004), it examined Coral DSHT as an extension of Kademlia in three particularly important ways:

1. Kademlia stores values in nodes whose ids are “nearest” (using XOR-distance) to the key.
2. Coral relaxes the DHT API from `get_value(key)` to `get_any_values(key)` (the “sloppy” in DSHT).
3. Additionally, Coral organizes a hierarchy of separate DSHTs called clusters depending on region and size.

Another approach, S/Kademlia DHT (Baumgart and Mies. 2007) extends Kademlia to protect against malicious attacks in two particularly important ways:

1. S/Kademlia provides schemes to secure NodeId generation, and prevent Sybill attacks
2. S/Kademlia nodes lookup values over disjoint paths, in order to ensure honest nodes can connect to each other in the presence of a large fraction of adversaries in the network.

Xie (2003) discussed how DHTs are implemented in P2P systems in his paper “P2P Systems based on Distributed Hash Table”. Files are connected with keys (which are generated by hashing the file name); each node in the system is responsible for storing a specific range of keys and handles a fraction of the hash space. The system will return the identity (e.g., the IP address) of the node storing the object with that key after a lookup for that key. The DHT capability allows nodes to put and get files based on their key and has shown to be a viable substrate for large distributed systems, with a number of projects proposing to overlay Internet-scale services on top of DHTs. Each node in a DHT is in charge of a specific key range and a portion of the hash space. Routing is a distributed lookup that is location-deterministic. Deterministic locating and load balance are the most significant improvements.

- No global knowledge
- Absence of single point of failures



Blockchain

Blockchains are a sort of decentralized distributed ledger and usually anonymous groups of agents rather than known centralized parties. This novel method of recordkeeping has introduced two economic innovations that overcome the two limitations of competition among centralized ledgers. The entry of record-keepers is unrestricted: any agent may write on the ledger as long as they follow a set of regulations. Furthermore, information on an existing blockchain is portable to a competing one. A software developer can propose to “fork off” an existing blockchain to establish one with different policies while retaining all the information contained in the original blockchain. Fork competition eliminates the inefficiencies arising from switching costs in centralized record-keeping systems (Abadi and Brunnermeier, 2018).

On an article “Blockchain Technology Overview” (Yaga et al. 2018), they mentioned four key characteristics of this technology:

- Ledger – the technology uses an append only ledger to provide full transactional history. A blockchain, unlike traditional databases, does not allow transactions and values to be overwritten.
- Secure – blockchains are cryptographically secure, ensuring that the data in the ledger has not been changed with and that the data is attestable.
- Shared – multiple participants will share the ledger. This provides transparency across the node participants in the blockchain network.
- Distributed – the blockchain can be distributed. This lets a blockchain network's number of nodes to be scaled up to make it more resilient to bad actors' attacks. By expanding the number of nodes, a bad actor's capacity to influence the blockchain's consensus procedure is lessened.

Like a traditional public ledger, blockchain is a series of blocks that carry a comprehensive list of transaction data. A block has just one parent block if the block header contains a preceding block hash. It's worth mentioning that hashes for uncle blocks



(children of the block's ancestors) would be saved as well. The first block of a blockchain is called genesis block which has no parent block (Zheng et al., 2017).

In the article of Monrat et al. (2019) titled “A Survey of Blockchain From the Perspectives of Applications, Challenges, and Opportunities”, they identified six comparison perspectives when comparing blockchain networks:

1. Consensus Determination - All the nodes can participate in the consensus process in the public blockchain such as Bitcoin, while only a few selected set of nodes are being responsible for confirming a block in the consortium blockchain. In the private blockchain, a central authority will decide the delegates who could determine the validated block.
2. Read Permission - Public blockchain allows read permission to the users, where the private and consortium can make restricted access to the distributed ledger. Therefore, the organization or consortium can decide whether the stored information needs to be kept public for all or not.
- 3) Immutability - In the decentralized blockchain network, transactions are stored in a distributed ledger and validated by all the peers, which makes it nearly impossible to modify in the public Blockchain. In contrast, the consortium and private Blockchain ledger can be tampered by the desire of the dominant authority.
- 4) Efficiency - In the public blockchain, any node can join or leave the network which makes it highly scalable. However, with the increasing complexity for the mining process and the flexible access of new nodes to the network, it results in limited throughput and higher latency. However, with fewer validators and elective consensus protocols, private and consortium blockchain can facilitate better performance and energy efficiency.
- 5) Centralized - The significant difference among these three types of Blockchain is that the public blockchain is decentralized, while the consortium is partially centralized and private blockchain is controlled by a centralized authority.



Proof-of-Authority

Proof of Authority (PoA) is a group of permissioned blockchain consensus algorithms that have gained popularity due to improved performance over traditional BFT algorithms due to fewer message exchanges. PoA was first proposed as part of the Ethereum ecosystem for private networks, and it was implemented in the Aura and Clique clients. The authorities are a group of N trusted nodes that PoA algorithms rely on. Each authority is identifiable by a unique id, and a majority of them, precisely at least $N/2 + 1$, is believed to be trustworthy. To execute the transactions issued by clients, the authorities run a consensus. The mining rotation schema, a commonly used way to fairly spread the burden of block creation across authority, is used to achieve consensus in PoA algorithms. Time is split into steps, each of which has a mining leader elected by the nodes. (Bitfury Group and Garzik, 2015).

There are two main PoA algorithms currently: AuRa and Clique. Aura (Authority Round) is the PoA algorithm implemented in Parity, the Rust-based Ethereum client. It is expected that the network is synchronous and all authorities to be synchronized within the same UNIX time t . The index s of each step is deterministically computed by each authority as $s = t/step_duration$, where $step_duration$ is a constant determining the duration of a *step*. The leader of a *step* s is the authority identified by the id $l = s \bmod N$. Clique is the PoA algorithm implemented in Geth, the GoLang-based Ethereum client. The algorithm proceeds in epochs which are identified by a prefixed sequence of committed blocks. When a new epoch starts, a special transition block is broadcasted. It specifies the set of authorities (i.e., their ids) and can be used as snapshot of the current blockchain by new authorities needing to synchronize (De Angelis et al., 2018).



Asymmetric Encryption

Asymmetric encryption systems are typically employed for discreetly delivering a symmetric encryption scheme's session key for message encryption. In fact, asymmetric and symmetric encryption techniques are frequently used in practice. (Fujisaki and Okamoto, 2011).

Goldwasser and Micali (1984) discussed the symmetric (aka private-key) encryption scheme as follows. Given by a pair of algorithms, $\Pi = (E, D)$, where for every sufficiently large $k \in \mathbb{N}$,

- E , the encryption algorithm, is a probabilistic polynomial-time (in k) algorithm that takes secret key $a \in \text{KSP}$ and message $x \in \text{MSP}$, draws coins r uniformly from coin space COIN , and produces ciphertext $y := E_a(x; r)$. This experiment is written as $y \leftarrow E_a(x)$. The key, message, and coin spaces, KSP , MSP and COIN , are uniquely determined by k .
- D , the decryption algorithm, is a deterministic polynomial-time (in k) algorithm that takes secret key $a \in \text{KSP}$ and ciphertext $y \in \{0, 1\}^*$, and outputs message $x := D_a(y)$.

We require that a symmetric encryption scheme should satisfy the correctness condition: For every sufficiently large $k \in \mathbb{N}$, every $a \in \text{KSP}$ and every $x \in \text{MSP}$, we always have $D_a(E_a(x)) = x$.

Bellare et al. (1998) detailed the asymmetric (aka public-key) encryption scheme. Given by a triple of algorithms, $\Pi = (K, E, D)$, where for every sufficiently large $k \in \mathbb{N}$:

- K , the key-generation algorithm, is a probabilistic polynomial-time (in k) algorithm which on input 1^k outputs a pair of strings, (pk, sk) , called the public and secret keys, respectively. This experiment is written as $(pk, sk) \leftarrow K(1^k)$.
- E , the encryption algorithm, is a probabilistic polynomial-time (in k) algorithm that takes public key pk and message $x \in \text{MSP}$, draws coins r uniformly from coin space COIN , and produces ciphertext $y := E_{pk}(x; r)$. This experiment is written as $y \leftarrow E_{pk}(x)$. The message and coin spaces, MSP and COIN , are uniquely determined by pk .
- D , the decryption algorithm, is a deterministic polynomial-time (in k) algorithm that takes secret key sk and ciphertext $y \in \{0, 1\}^*$, and returns message $x := D_{sk}(y)$.



We require that an asymmetric encryption scheme should satisfy the following correctness condition: For every sufficiently large $k \in N$, every (pk, sk) generated by $K(1^k)$ and every $x \in MSP$, we always have $D_{sk}(E_{pk}(x)) = x$.

Decentralized Storage, Blockchain and Medical Records

MedRec, a system proposed by Azaria et al. (2016) shows how principles of decentralization might be applied to largescale data management in an EMR system by using blockchain technology. It utilized Proof-of-Work consensus in mining transaction blocks. Patient data are stored in centralized SQL server while transaction logs of updating patient records are in the Ethereum blockchain. A study by Sharma et al. (2020) did a similar EMR model but introduced cloud storage as an alternative to a centralized on-premise server. These two studies posed limitations on storing files. Though Sharma attempted to solve this by putting a cloud application layer, Cloud providers will have autonomy to data stored in their servers.

Kumar and Tripathi (2020) presented a distributed framework handling COVID-19 patient reports. It utilized Proof-of-Work blockchain and IPFS to decentralize data storage. However, the system has no patient access interface and only shares data for provider use only. Wu and Du (2019) also added IPFS on their Delegated Proof-of-Stake blockchain implementation of EMR. They also used data-masking to protect patient data once uploaded on the network and specified Digital Imaging and Communications in Medicine (.dcm) image format of files to be uploaded. Like Kumar and Tripathi, system did not provide data access to patients.

Sun et al. (2020) proposed attribute-based encryption for EMRs with IPFS and blockchain implementation. The scheme provides good access control for the electronic medical records using attribute-based encryption technology so that people who are not related to the patient cannot see the private data of the patient without authorized. Khubrani (2021) proposed a proposed a theoretical blockchain-based framework via



blockchain, IPFS and asymmetric encryption but did not mention technical specifications on how these technologies will integrate with one another.

At this point, related studies mentioned above either used Proof-of-Work (PoW) or Proof-of-Stake (PoS) as their consensus scheme for EMRs. A comparative study of existing literature for EMR system based from blockchain and IPFS was presented by Kumar et al. (2021). It compared different metrics such as Technology used, Cost-effectiveness, Complexity and Shortcomings. Most of the shortcomings were implementation-related such as lack of data formatting and workflow for data sharing, but the authors gave emphasis on the need of a cost-effective way to deploy blockchain as an immutable ledger since most of the studies were using Proof-of-Work as a consensus scheme.

On a paper by Al Asad et al. (2021), they proposed a theoretical blockchain-based framework with Proof-of-Authority (PoA) as the consensus scheme. It cited comparisons among other consensus (Proof-of-Work and Proof-of-Stake) and shown why PoA is a better alternative for EMRs. However, this paper only examined the feasibility of PoA consensus implementation and did not dwell on strategies for decentralized file storage and encryption. Reen (2019) on an earlier study, also mentioned PoA as an excellent choice for medical records. He made a conceptual model on IPFS as a decentralized file storage but did not provide technical specification about PoA and how it will be integrated in the system.



Chapter Three

THEORETICAL FRAMEWORK

Present State of COVID-19 Tests and Vaccine Certificate Storage

Documents and certificates given out by various units (private and public) for COVID-19 related tests such as antigen and Real-Time Polymerase Chain Reaction (RT-PCR) are still on paper-form. There are some units that store the results in their server and can be accessed online thru their website. Same is true with giving out vaccine certificates. Primary providers of vaccines are Local Government Units (LGUs) and they vary in implementation. Some only give out physical copies (certificates, cards) and others have virtual copies on their websites stored on their servers. There is a disconnect on a unified tracking of all these documents and might result to issues when these documents will be used on different areas of the Philippines. The usual proposition to solve this is to create a unified website that will be hosted in a central server.

Proposed Documents Storage Structure

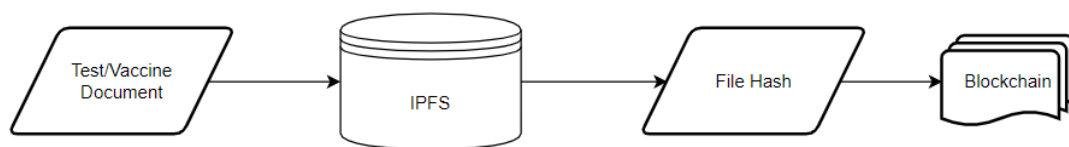


Figure 3.0.1: Diagram of Proposed Solution

Above is a summarized approach in solving the problem in document storage. The main components of this application will be the IPFS for file storage and blockchain to record the logs of transaction being done in the system.

The next sections will discuss the different algorithms and frameworks to be used in order to achieve the proposed solution.



Cryptographic Hash Functions

A cryptographic hash function is a process that converts data of arbitrary size (commonly referred to as the "message") into a fixed-size bit array ("hash value", "hash", or "message digest"). A one-way function, which means that inverting or reversing the computation is almost impossible. The only way to identify a message that generates a particular hash is to try a brute-force search of all potential inputs to see whether any of them create a match, or to use a rainbow table of matched hashes. Cryptographic hash functions are a primary instrument of modern cryptography.

The following are the major characteristics of an ideal cryptographic hash function:

- it is deterministic, meaning that the same message always results in the same hash
- it is quick to compute the hash value for any given message
- it is impossible to generate a message that produces a given hash value
- it is infeasible to find two different messages with the same hash value
- a small change to a message should alter the hash value in such a way that a new hash value appears to be unrelated to the old hash value

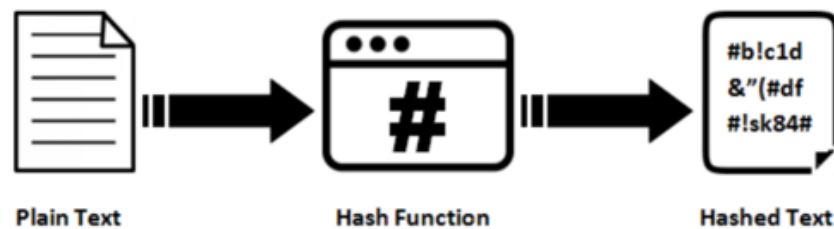


Figure 3.0.2: Hashing Process

The majority of cryptographic hash functions accept any length string as input and return a fixed-length hash value.

A cryptographic hash function must be cryptanalytically resistant to all known types of attacks. The security level of a cryptographic hash function has been determined using the following properties in theoretical cryptography:



- Pre-image resistance

Given a hash value h , it should be hard to determine any message m such that $h = \text{hash}(m)$. This concept is connected to that of a one-way function. Functions that do not have this property are susceptible to preimage attacks.

- Second pre-image resistance

Given an input m_1 , it should be hard to determine a different input m_2 such that $\text{hash}(m_1) = \text{hash}(m_2)$. This property is occasionally stated to as weak collision resistance. Functions that do not have this attribute are susceptible to second-preimage attacks.

- Collision resistance

It should be hard to determine two different messages m_1 and m_2 such that $\text{hash}(m_1) = \text{hash}(m_2)$. Such a pair is referred as cryptographic hash collision. This attribute is occasionally called as strong collision resistance. It needs a hash value at least twice as long as that required for pre-image resistance; or else collisions may be identified by a birthday attack.



Blockchain

In 2008, Satoshi Nakamoto released a whitepaper titled “Bitcoin: A peer-to-peer electronic cash system”. This paper proposed a system for electronic transactions which uses a peer-to-peer network. Participating nodes in the network utilize Proof-of-Work to record public history of transactions.

At its most basic level, blockchain technology permits a network of computers to have a consensus on the true status of a distributed ledger at regular intervals. Blockchain network users submit potential transactions to participating nodes. The network will then choose a publishing node to update the pending transaction. Once this is done, transaction will be propagated to non-publishing nodes. Transactions are logged chronologically – with information being passed from the first transaction (or blocks) up to the last. This repetitive process forms an immutable chain on which all blocks are interconnected with each other.

Transactions are inserted to the blockchain when a publishing node creates a block. A block may represent various types of data from simple texts to complicated ones such as digital rights or intellectual property. It is divided into two parts, header and body. Header contains metadata and body is for the actual data being persisted in the blockchain. Below is a typical specification of these 2 parts:

1. Block Header

- Previous block header’s hash value
- Hash representation of block data
- Timestamp
- Size of the block
- Nonce value. In Bitcoin and other Proof-of-Work blockchains, this is a number manipulated by the publishing node to solve the hash puzzle.



2. Block Data

- Actual data (text, files)

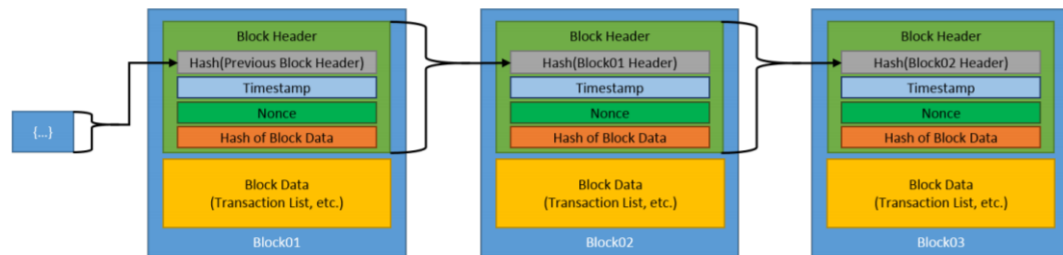


Figure 3.0.3: Generic Blockchain Transactions

Figure 3.0.3 shows how blockchain works given we have a simple data of text. The initial block is referred to the genesis block and is automatically generated upon the chain's creation. This genesis block will be the seed and considered as reference of all blocks going forward. Blocks are linked through each block containing the hash value of the previous block's header, thus creating the chain. In case a previously published block was changed, it will have a different hash. This will create a domino effect on all subsequent blocks to also have a different hash because they contain the hash of the altered block.

An essential part of the blockchain is identifying which user will publish the next block or become the next publishing node. This is solved by implementing a consensus model. The common model used is to compete on who will publish it and winning an incentive in doing so.

Once a user joins a blockchain network, they agree to the preliminary state of the system. This is documented in the only pre-configured block or the genesis block. Each blockchain network have a genesis block on to which all subsequent blocks would reference to. Each block must be valid and can be validated independently by each blockchain network user.



Proof of Authority (POA) - Clique

In a Proof of Authority (PoA) consensus algorithm, a set of trusted nodes called Authorities, each recognized by their unique identifier, are responsible for mining and validating the blocks in the blockchain. Clique is a PoA protocol implemented in Geth.

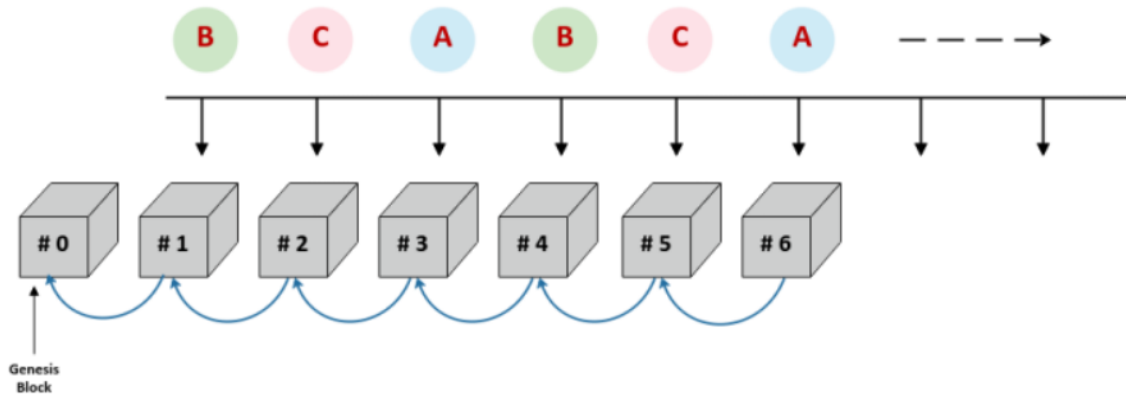


Figure 3.0.4: Clique PoA Block Creation Process

The Clique consensus protocol adheres to the following rules:

- Set of trusted authorities are referred to as the "Signers"
- Process of mining a block is referred to as "Sealing a block"
- WHEN the next block is identified by BLOCK_NUMBER and the number of signers is identified by SIGNER_COUNT

AND the signers are lexicographically sorted by their unique identifiers in a list
THEN the next block is sealed by the signer located at the index
 $\text{BLOCK_NUMBER} \% \text{SIGNER_COUNT}$, where % is the modulus operator

The signers compile and execute network transactions into a block, updating the world state. At the fixed interval referred to as the BLOCK_PERIOD, the next signer in the list (identified by $\text{BLOCK_NUMBER} \% \text{SIGNER_COUNT}$) calculates the hash of the block and then signs the block using its private key (sealing the block). The sealed block is then broadcast to all nodes in the network.



InterPlanetary File Storage (IPFS)

IPFS is a distributed platform for storing and retrieving files, websites, applications and data. It has rules that regulate in what manner data and content move around on the network. These rules are similar to Kademia, the peer-to-peer distributed hash table (DHT) popularized by its use in the BitTorrent protocol.

IPFS is essentially a peer-to-peer system for getting and sharing IPFS objects. An IPFS object is a data structure have two fields:

- Data: a blob of unstructured binary data of size < 256 kB.
- Links: an array of Link structures. These are links to other IPFS objects. Links have 3 sub-parts:
 - o Name: the name of the Link.
 - o Hash: the hash of the linked IPFS object.
 - o Size: the cumulative size of the linked IPFS object, including following its links.

IPFS builds a Merkle DAG, a blend of a Merkle Tree and a Directed Acyclic Graph (DAG).

A Merkle tree summarizes all of the transactions in a block by generating a digital fingerprint of the complete collection of transactions, allowing a user to check whether or not a transaction is included in the block. Merkle trees are made by hashing pairs of nodes repeatedly until only one hash remains (this hash is called the Root Hash, or the Merkle Root). They are built from the ground up, utilizing individual transaction hashes (known as Transaction IDs). Each non-leaf node is a hash of its previous hashes, while each leaf node is a hash of transactional data. Merkle trees are binary, hence an even number of leaf nodes is required. The last hash will be repeated once to establish an even number of leaf nodes if the number of transactions is odd.

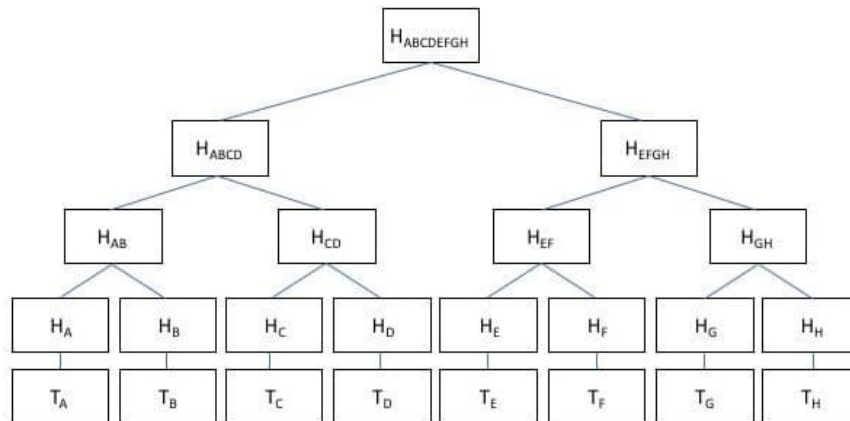


Figure 3.0.5: Merkle Tree Implementation using hashes

A directed acyclic graph (DAG) is a visual representation of a sequence of events. A graph depicting the order of the activities is visually portrayed as a group of circles, each representing an activity, some of which are connected by lines, which represent the flow from one action to the next. Each circle is referred to as a "vertex," and each line is referred to as a "edge". "Directed" signifies that each edge has a specific direction, implying that each edge reflects a single directional flow from one vertex to the next. The term "acyclic" refers to a network that contains no loops (or "cycles"), meaning that if you follow an edge connecting one vertex to another, there is no way to return to the original vertex.

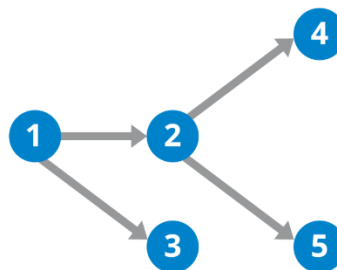


Figure 3.0.6: DAG Illustration

A Merkle DAG is a DAG in which each node has an identification that is generated by hashing the content of the node — any opaque payload carried by the node, as well as a list of its children's identifiers — by utilizing a cryptographic hash function like SHA256. This brings some important considerations:



- Merkle DAGs can only be built from the leaves, or nodes that have no offspring. Parents come after children because the identifiers for the children must be computed ahead of time in order to link them. Every node in a Merkle DAG is the root of a (sub)Merkle DAG, and the parent DAG contains this subgraph.
- Merkle DAG nodes cannot be changed. Any change to a node's identity would affect all ascendants in the DAG, effectively resulting in the creation of a new DAG.
- Merkle DAGs are like Merkle trees, but they don't have to be balanced, and each node can have a payload. Many branches can re-converge in DAGs, or, to put it another way, a node can have multiple parents.

Content addressing is the process of identifying a data object (such as a Merkle DAG node) based on the value of its hash. As a result, the node identifier is referred to as the Content Identifier, or CID.

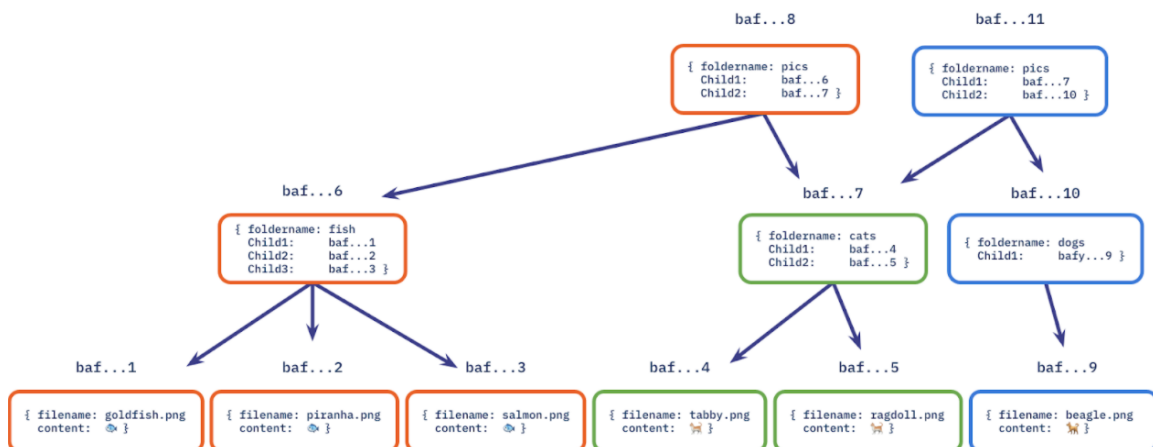


Figure 3.0.7: Merkle DAG implemented on a file system



Asymmetric Cryptography

Asymmetric cryptography encrypts plain text messages using mathematical permutations as well, but it employs two separate permutations, still known as keys, to encrypt and decrypt messages. A public key that may be shared with everyone is used to encrypt messages in asymmetric cryptography, whereas a private key known only by the receiver is used to decrypt messages. With this scheme:

- Each user has two keys: a public key and a private key.
- Both keys have a mathematical relationship (both keys together are called the key pair).
- The public key is available to anyone, but the private key is kept secret by the intended owner.
- To complete an operation, you'll need both keys. The public key, for example, decrypts data encrypted with the private key. With the private key, data encrypted with the public key becomes unencrypted.
- A digital signature is created by encrypting data with the private key. This confirms that the message was from the specified sender (since only the sender had access to the private key to create the signature).
- A digital envelope encrypts a message using the public key of the recipient. A digital envelope that ensures that only the intended recipient can open the message as a sort of access control (since only the receiver will have the private key required to open the envelope; this is also referred to as receiver authentication).
- A fresh key pair must be generated if the private key is ever discovered.

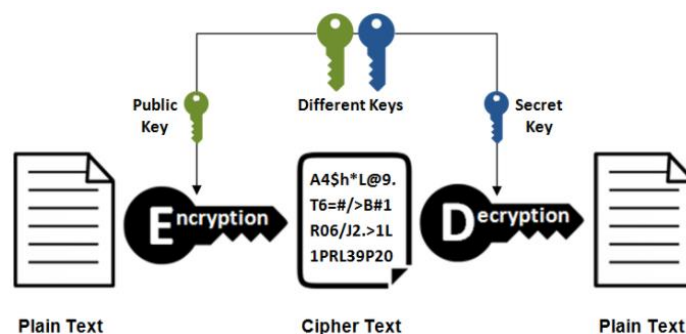


Figure 3.0.8: Asymmetric Encryption and Decryption Process



This section aims to demonstrate the overview of the final product of this research. It identifies relevant variables, inputs, mappings and other components and how they will interact with each other. This includes all the underlying concepts and their associated mappings based on the system's use.

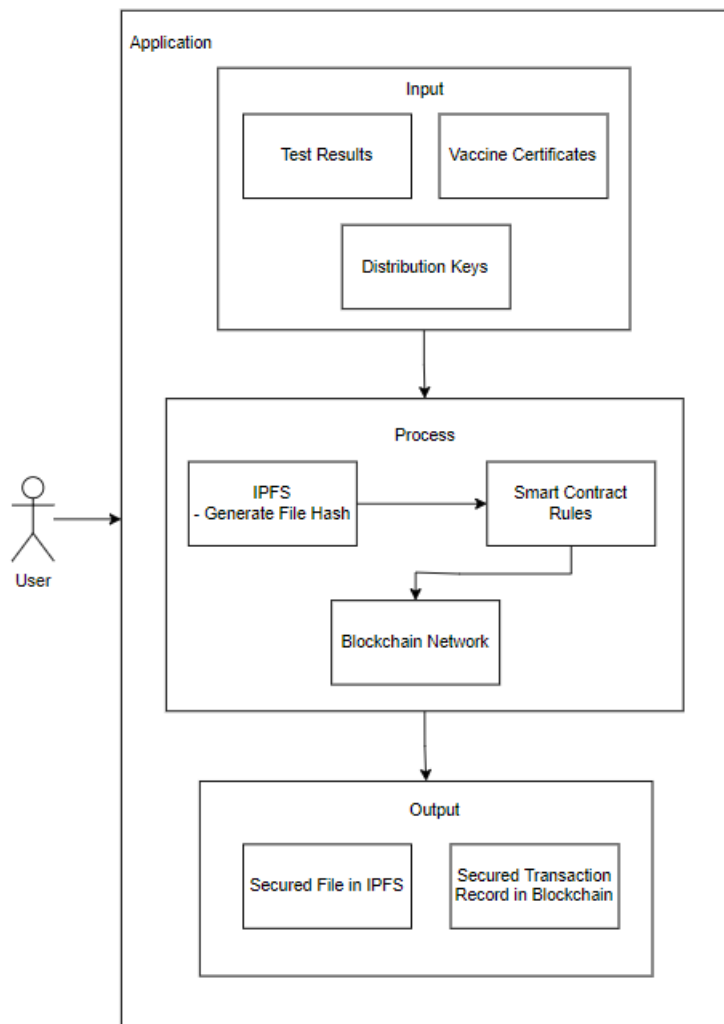


Figure 3.0.9: Conceptual Framework



The users of the proposed application will be patients, medical workers or other third-party requiring the patient to present a COVID-19 Test Result or Vaccine Certificate. The users will access the same application but with different levels of access depending on their role.

The input are the medical documents and distribution key. There will be different types of keys which will be discussed on Chapter 4. These keys will be used to authenticate and unlock or lock the files.

Once all required inputs are provided, the file will now go thru the necessary steps to access it. Depending on the type of transaction (insert a new file or retrieval), the keys provided should have enough privilege for it to succeed. The file hash will be then stored in the blockchain after going thru smart contracts. Once the blockchain successfully updated the network, provided file will now become an immutable component of both IPFS and blockchain network.



Chapter Four

METHODOLOGY

This chapter provides an overview of the strategies used to attain the study's goals. It describes the study's respondents as well as the research instruments that were used. It then goes on to explain the data collection strategies that contributed in the completion of the study endeavor.

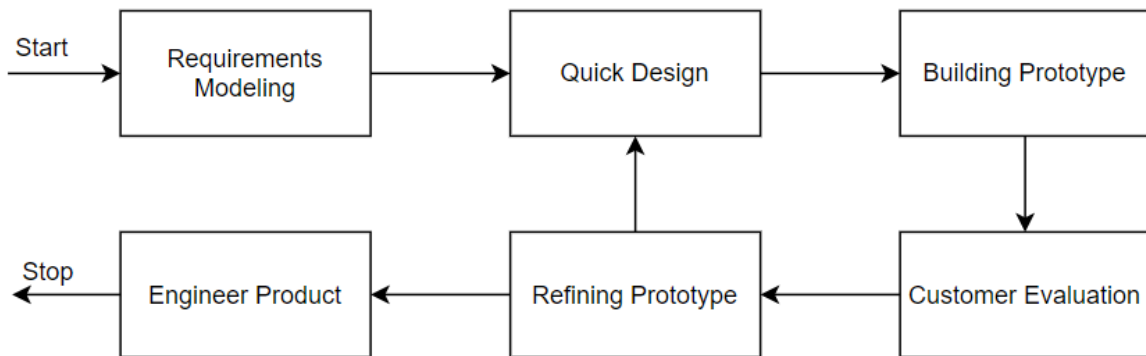


Figure 4.0.1: Prototype Model Phases and Process

Figure 4.0.1 illustrates Prototype Model used by the researcher in developing the proposed study entitled “CoviBlock: A Blockchain and Peer-to-peer Platform for COVID-19 Test Results and Vaccine Records” which is under the family of System Development Life Cycle (SDLC). Prototyping was used to ensure faster turnaround time on each phase while addressing client’s requirements and feedbacks. This model also enables the researcher and client to have discussions in between development cycles.

The next sections of this chapter will discuss the phases of the used model.



4.1 Requirements Modeling

The Prototype Model starts with outlining the requirements. The researcher will conduct an initial investigation to determine the purpose and utilization of the application coupled with the nature and scope of the study. It is also in this stage that the researcher requested permission from medical unit authorities and other parties to conduct the study and all relevant data and information were examined.

Fact-finding was used via interviews and probing of processes to build a logical model of the application. With these interviews, the researcher was able to piece out a picture of transactions involved and analyzed them against the proposed solution. This information will also enable the researcher to identify critical decisions geared toward implementing the application.

At present time, Ospital ng Makati does not have its own molecular laboratory to conduct independent COVID-19 tests. Instead, they forward test requests to their partner third party laboratories. Once requested by the patient from the hospital, they will either execute the tests in-house or hand-over the execution to the third-party lab. After results comes out, the hospital will furnish a physical copy to the patient whilst maintaining a softcopy in their archive.

For vaccination records, Makati citizens are encouraged to register online via the web portal. This will ensure a scheduled slot on a specified date. On the day of vaccination, patient will be checked up by a physician to ensure he is fit for vaccination. The physician's findings are logged on the system. Upon issuing a go signal, patient can now be vaccinated. After vaccination, vaccination site will sign a vaccination card while tagging the patient in their system as fully vaccinated.

The study will be focused on two COVID-19 related records: Test Results and Vaccine Certificate. Mocked test data will be used and will only be for the purpose of this research. This is due to various privacy regulation such as Health Insurance Portability



and Accountability Act (HIPAA). This is a United States created health law adopted by medical facilities in the Philippines.

WILMER LABORATORY AND DIAGNOSTIC CLINIC
19 SALCEDO ST. SALCEDO VILLAGE MAKATI CITY
DOH LIC No. 4A-0445-20-JL-2
323-1234

NAME: JUAN DELA CRUZ **AGE:** 67 **GENDER:** MALE
REQUESTING PHYSICIAN: OPD **DATE:** August 4, 2021
CONTROL NO.: CWB 123 **TIME COLLECTED:** 4:30 PM

| TEST | RESULT |
|---------------------------|-----------------|
| SARS-CoV-2 Antigen | POSITIVE |

Specimen Type: NASAL

Note:
*This is a screening test only using a rapid antigen test with FDA-approved brand. Clinical and ancillary test correlation should be done only by a qualified physician.

Test Kit Used: GENRLH
Lot No.: 20201210
Expiration Date: 5/31/2022


Daniel A. Silva, MD
Pathologist
Lic No.: 00345222

Figure 4.0.2: Sample Rapid Antigen Test Result



Pamantasan ng Lungsod ng Maynila



| | | | |
|---|--|--|---|
|  | RESEARCH INSTITUTE FOR TROPICAL MEDICINE Filinvest Corporate City Compound Alabang, Muntinlupa City | | LRD-DIV-SPE-FM-005 Revision No.: 2 Page No.: Page 1 of 1 |
| | HEALTH CARE DELIVERY LABORATORY RESULT FORM | | |

| | | | |
|--|--|--|------------------------------------|
| Name of Patient: MARQUEZ, MICHAEL JARO <small>(Full Last Name, First Name, Middle Name)</small> | | Disease Reporting Unit: MAKATI CITY HEALTH OFFICE | |
| Age/Sex: 30 / MALE | | Date of Birth: 06/05/1990 <small>(DD/MM/YYYY)</small> | |
| Patient Location: <input type="checkbox"/> OPD <input checked="" type="checkbox"/> REFERRAL <input type="checkbox"/> INPATIENT | | RITM Hospital No: No Information <small>RITM patients</small> | |
| Requisitioner: DR MARIE ALDO | | Date Admission: --- <small>(DD/MM/YYYY) RITM patients</small> | |
| Patient Address: 89 JACINTO ST. GUADALUPE NUEVO, CITY OF MAKATI, NCR, FOURTH DISTRICT (NOT A PROVINCE) | | Contact No.: 09154556332 | |
| Specimen Type: NPS/OPS | | Accession No: --- | Laboratory No: COV20-292398 |
| Date and Time of Specimen Collection: 02/10/2020 11:25 <small>(DD/MM/YYYY HR:MIN)</small> | | Date and Time of Specimen Receipt: 03/10/2020 09:40 <small>(DD/MM/YYYY HR:MIN)</small> | |
| Date and Time of Release of Result: 04/10/2020 21:52 <small>(DD/MM/YYYY HR:MIN)</small> | | | |

| LABORATORY TEST RESULT | |
|--|---|
| LABORATORY TEST PERFORMED: SARS-CoV-2 (causative agent of COVID-19) virus detection by Real-Time Polymerase Chain Reaction | |
| TEST RESULT: SARS-CoV-2 (causative agent of COVID-19) viral RNA detected | |
| RESULT AND UNITS OR MEASUREMENTS: NONE BIOLOGICAL REFERENCE INTERVALS: NONE | |
| INTERPRETATION OF RESULTS WHEN APPROPRIATE | |
| Final Result | Interpretation |
| SARS-CoV-2 (causative agent of COVID-19) viral RNA detected | Positive for SARS-CoV-2 (causative agent of COVID-19) |
| SARS-CoV-2 (causative agent of COVID-19) viral RNA not detected | Negative for SARS-CoV-2 (causative agent of COVID-19) |
| Invalid due to specimen quality | Negative for test internal control (most likely due to poor specimen quality) |

This laboratory result should be interpreted together with the available clinical and epidemiological information.

| | | |
|---|--|--|
| COMMENTS: | | |
| Performed by: Lisa I. Arbado, RMT Medical Technologist | Verified by: Jose Q. Chua Molecular Biologist | Noted by: Eden T. Yap, PhD Head, RITM Molecular Biology Laboratory Selena R. Tajo, MD Pathologist |

Figure 4.0.3: Sample Real-Time Polymerase Chain Reaction (RT-PCR) Test Result



CERTIFICATE OF VACCINATION

This is to certify that JUAN DELA CRUZ, 25 years of age, residing at 123 J.P. Rizal St., Barangay Poblacion, Makati City, was fully vaccinated by the Makati Health Department.

| | 1 st Dose | 2 nd Dose |
|-------------------|----------------------|----------------------|
| Date Administered | 05/01/2021 | 05/25/2021 |
| Brand | Pfizer | Pfizer |
| Injection Site | Deltoid, Left | Deltoid, Left |
| Lot Number | FD100 | FD100 |
| Route | IM | IM |
| Expiration Date | 09/30/2021 | 09/30/2021 |
| Vaccinator | Nita Lim | Andy Cruz |
| Site | Ospital ng Makati | Ospital ng Makati |

This certificate is issued this 05th day of August 2021 for whatever legal purpose this may serve best.

Jaime Santos, MD
City Health Officer

Figure 4.0.4: Sample COVID-19 Vaccination Certificate



Once above files are generated from existing system or printed by medical volunteers or workers, it is now ready to be consumed by the application.

Below are requirements grouped by specific role:

Patient

- Register and Login – register to gain access to the system
- Grant Access – Grant access to file requestors
- View Own Record – Get access to own results/certificate

Verifying Third Party

- Register and Login – register to gain access to the system
- View Patient Record – retrieve and view patient record if was given access

Physician/Medical Unit

- Register and Login – register to gain access to the system
- Upload record – upload a record for the patient



4.1.1 Data Storage Scheme

Since the study is primarily concerned on how medical records will be stored, this section will discuss the different schemes that will be used in the application. This will involve simulation, graphical visualizations and detailed discussions.

4.1.1.1 IPFS – Merkle DAG

The algorithm used in IPFS to manage content and assets is Merkle DAG. Suppose we want to upload 2 vaccine certificates. For brevity, we will use a small size text file to better illustrate the process. The default chunk size of IPFS is 256Kb but in this example we will reduce it to 32Kb to have appropriate representation using small sample files.

File 1

Name: cert_allen_smith.txt

Size: 86 bytes

Content:

```
this is a sample certificate given to Allen Smith  
and only used for research purposes
```

File 2

Name: cert_john_doe.txt.txt

Size: 83 bytes

Content:

```
this is a sample certificate given to John Doe  
and only used for research purposes
```



Generated Details for cert_allen_smith.txt:

Table 4.0.1: Generated Hash Value for Sample Record #1

| Node Type | Size (Bytes) | Hash |
|-----------|--------------|--|
| Root | 0 | QmZkJLp7PJGMc3mMSxTeLtyQCRqZ5CudGdjPB3jjTSFaoX |
| Links | 32 | QmdsyzBk5nWmC7a92gaAuRHxWTQu6e4wwyv2bVmZtF7mcq |
| Links | 32 | QmPsFk9hcP4WmN96r8mXYjV5rKCNNb94c95jfqLBNZvigT |
| Links | 22 | QmVVrfBPAnF5DC1DXDZH2yftW6MEoCSKXEQEbY5LKfFzAt |

Generated Details for cert_john_doe.txt:

Table 4.0.2: Generated Hash Value for Sample Record #2

| Node Type | Size (Bytes) | Hash |
|-----------|--------------|--|
| Root | 0 | QmanmTVLostTHeeLiz8vr99QDWmVbmbd53rSA2iFoDcmXu |
| Links | 32 | QmdsyzBk5nWmC7a92gaAuRHxWTQu6e4wwyv2bVmZtF7mcq |
| Links | 32 | QmTfDsTDe3nVu7b3hij43R3mBzyhJZgVm9eFBewVb5FfKV |
| Links | 32 | QmRF3DNTkA43a7AG26uva4n7pgR22ctz6PjZW4KMuN5Cvu |

We can now map out the links with their respective roots. Notice that link “Qmdsy” is referenced by both root objects.

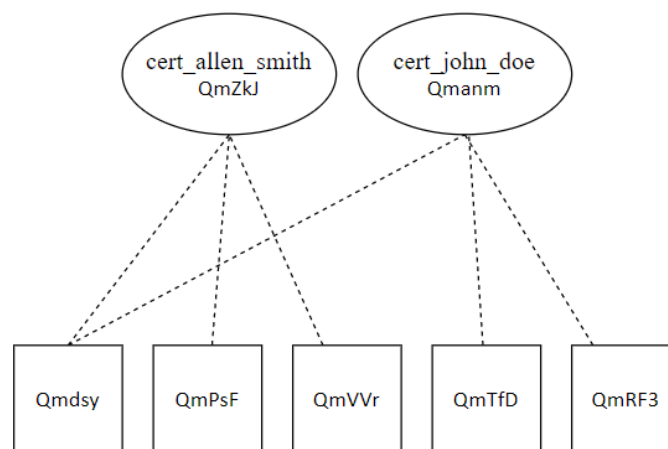


Figure 4.0.5: Merkle DAG representing sample records



4.1.1.2 Blockchain

The main purpose of using a blockchain is to validate whether a given or requested CID is authentic in the context of the system. After a doctor uploads a record in IPFS, the generated IPFS CID will then be logged to the blockchain. Blockchain validation will then be used as a proof that a CID exists in the context of CoviBlock. This will prevent illegal tampering or modification of records.

Taking our sample files from 4.1.1.1, we will create a blockchain of transactions given the files were already uploaded to IPFS and CIDs are generated. JSON Objects will be used as format of the payload.

Object for cert_allen_smith.txt:

```
{  
  "CID": "QmZkJLp7PJGMc3mMSxTeLtyQCRqZ5CudGdjPB3jjTSFaoX",  
  "MedPerson": "Dr. Ramon Cruz",  
  "LicenseNum": "123-456",  
  "DateTime": "23/07/2021 14:00:00"  
}
```

Object for cert_john_doe.txt:

```
{  
  "CID": "QmanmTVLostTHeeLiz8vr99QDWmVbmbd53rSA2iFoDcmXu",  
  "MedPerson": "Dr. Erik Lim",  
  "LicenseNum": "122-322",  
  "DateTime": "23/07/2021 09:00:00"  
}
```



4.2 Quick Design

After identifying the requirements, a design of the proposed application is created. This is not a detailed design with complete technical specifications but a simplified one with critical aspects of the solution. This phase will give a bird's eye view to the client of the application.

4.2.1 Context Diagram

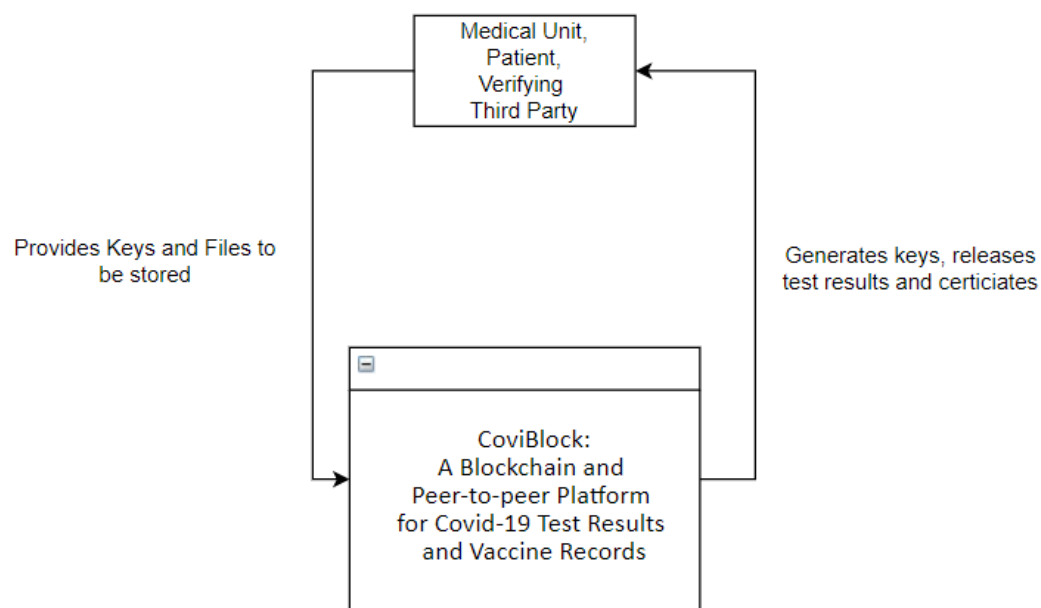


Figure 4.0.6: Context Diagram

The context diagram shown in Figure 4.0.6 summarizes the application on inputs and outputs of the system and targeted users. On general, users of the application will be required to provide public/private keys and raw files to be stored. It is now the application will trigger and execute various processes to upload, encrypt/decrypt, or release files. Note that this is a general illustration of inputs and outputs. Next sections of this chapter will discuss the mentioned processes on this diagram.



4.2.2 Data Flow Diagram

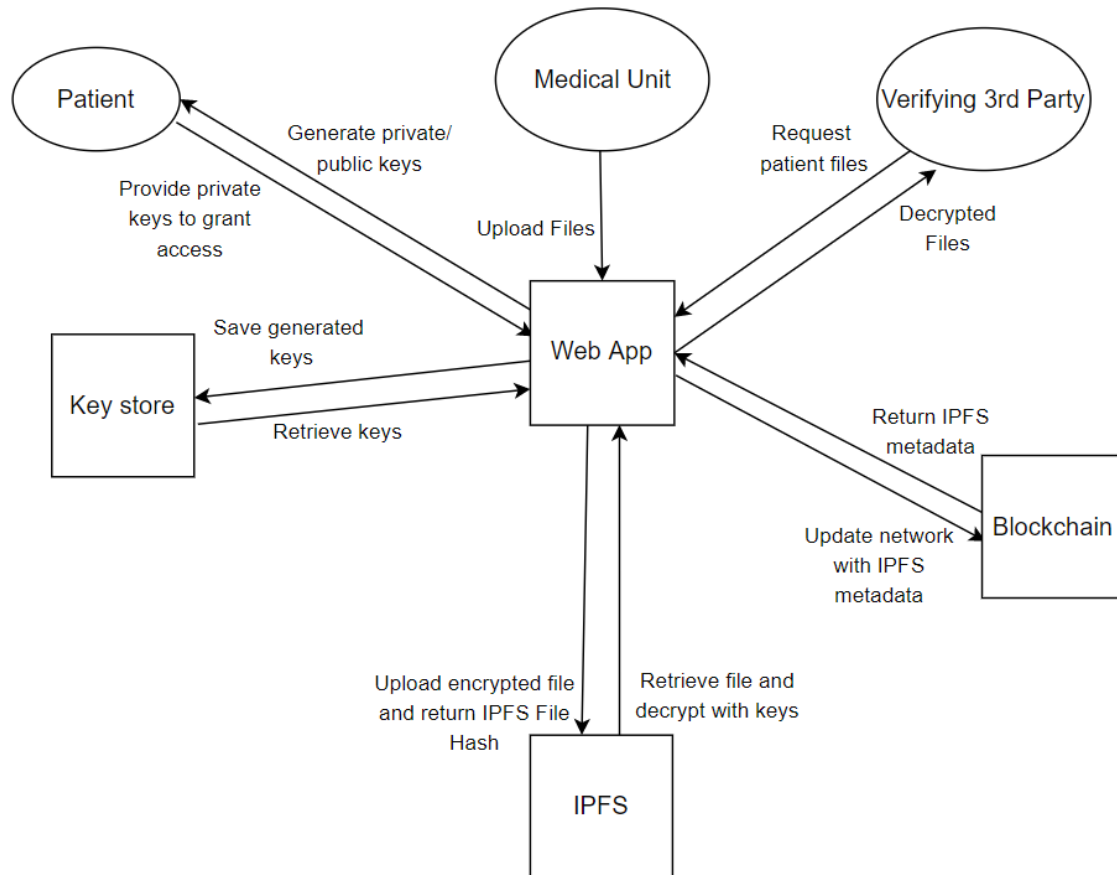


Figure 4.0.7: Data Flow Diagram

Figure 4.0.7 illustrates how various types of users receives and provides information to the application and how the application provides and receives data from users. This also mentions the executing process to generate the data.

It is important to note that authorized medical personnel are the only ones allowed to upload files. Patients will have to generate private and public keys for their files to be uploaded or requested. These keys are crucial for a patient file to be encrypted or decrypted. Third parties can request for patient files and will be granted access to view decrypted files.



4.2.3 Use Case Diagram

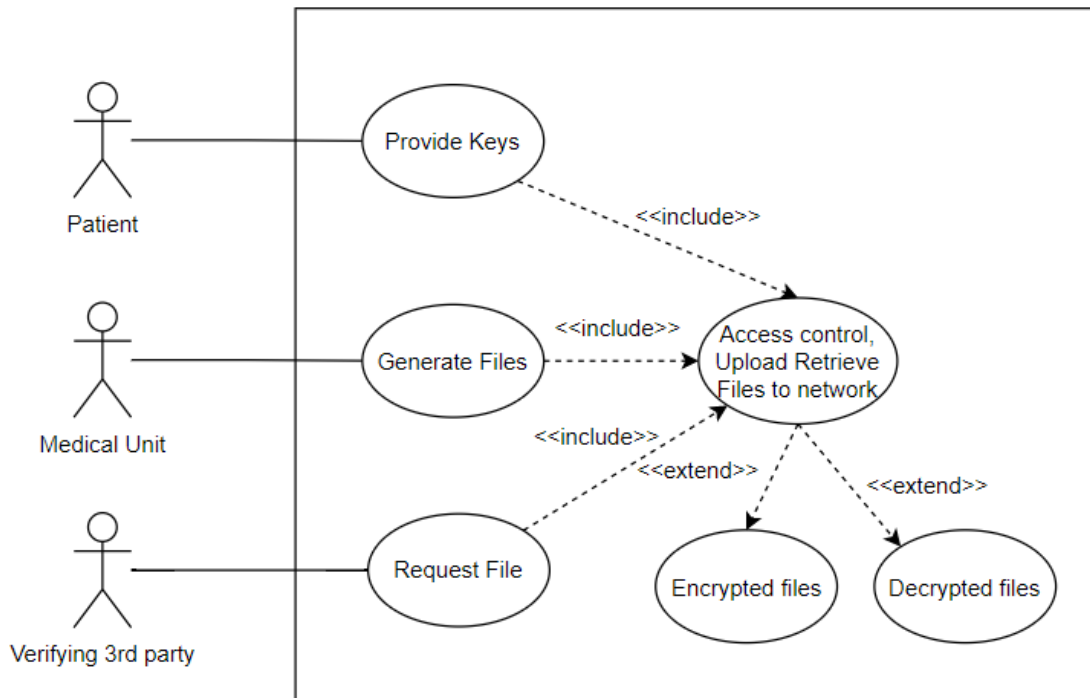


Figure 4.0.8: Proposed Use Case Diagram

The suggested application's development is not solely dependent on the system's functionality. It also depends on the workflow procedure that needs to be identified, implemented, and followed. The components of the proposed application “CoviBlock: A Merkle Dag and Blockchain Implementation for COVID-19 records”, is demonstrated in Figure 4.0.8 and utilized a Use Case Diagram. The patient, being the central user of this system will provide appropriate keys with reference to the executing process. These in turn can trigger uploading or granting of view access to either medical unit or a third party.



4.2.4 Transactional Operation Diagram

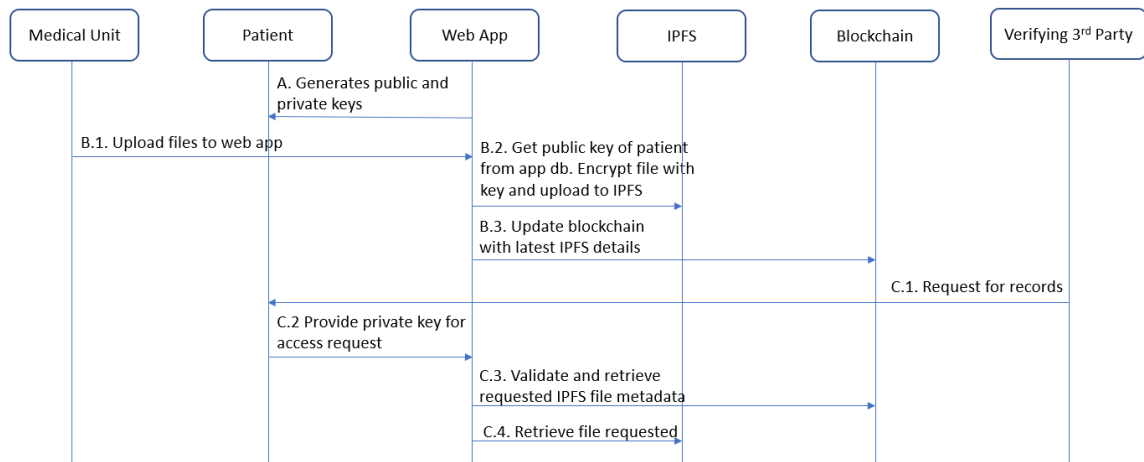


Figure 4.0.9: Transactional Operation Diagram

Figure 4.0.9 illustrates the operations that exist in the proposed application. It is divided according to the users triggering the process (A. B. C). The crucial process of generating the private and public keys will be prompted by the patient. Without these keys, medical personnel cannot upload files which in turn, the third parties will not be able to request any files.

4.2.5 System Flowchart of the Proposed Application

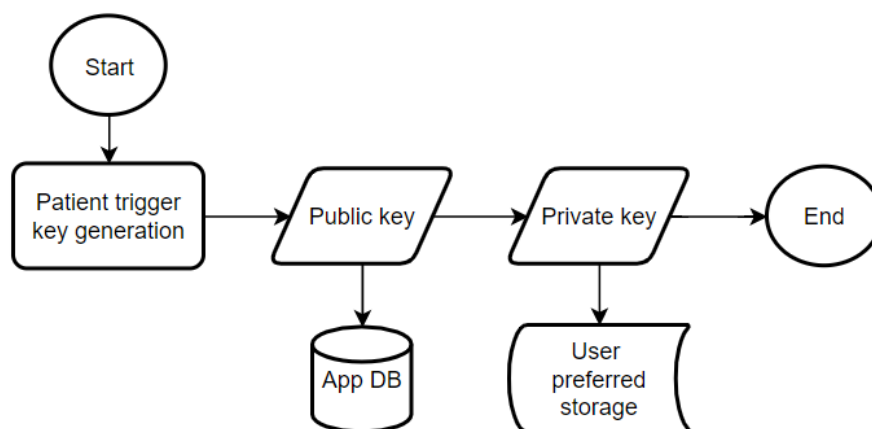


Figure 4.0.10: Key Generation Process Flowchart

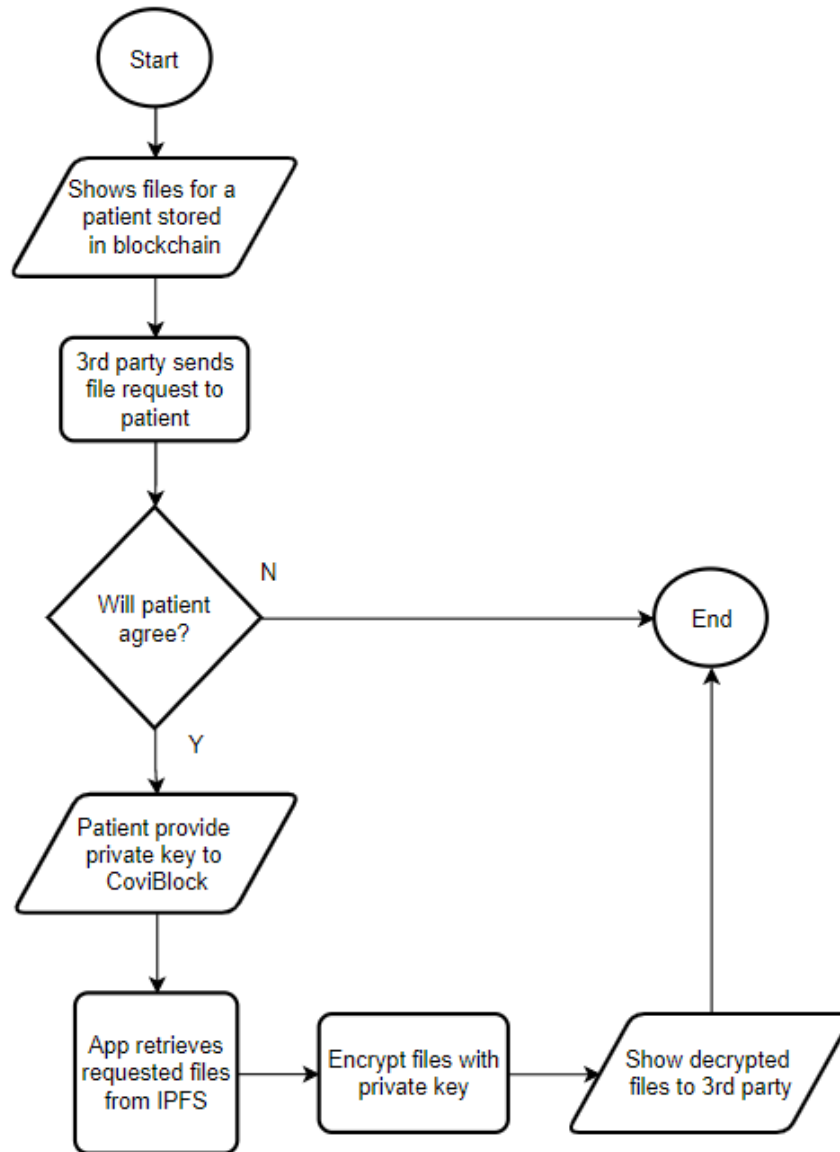


Figure 4.0.11: Third party access to patient file Flowchart

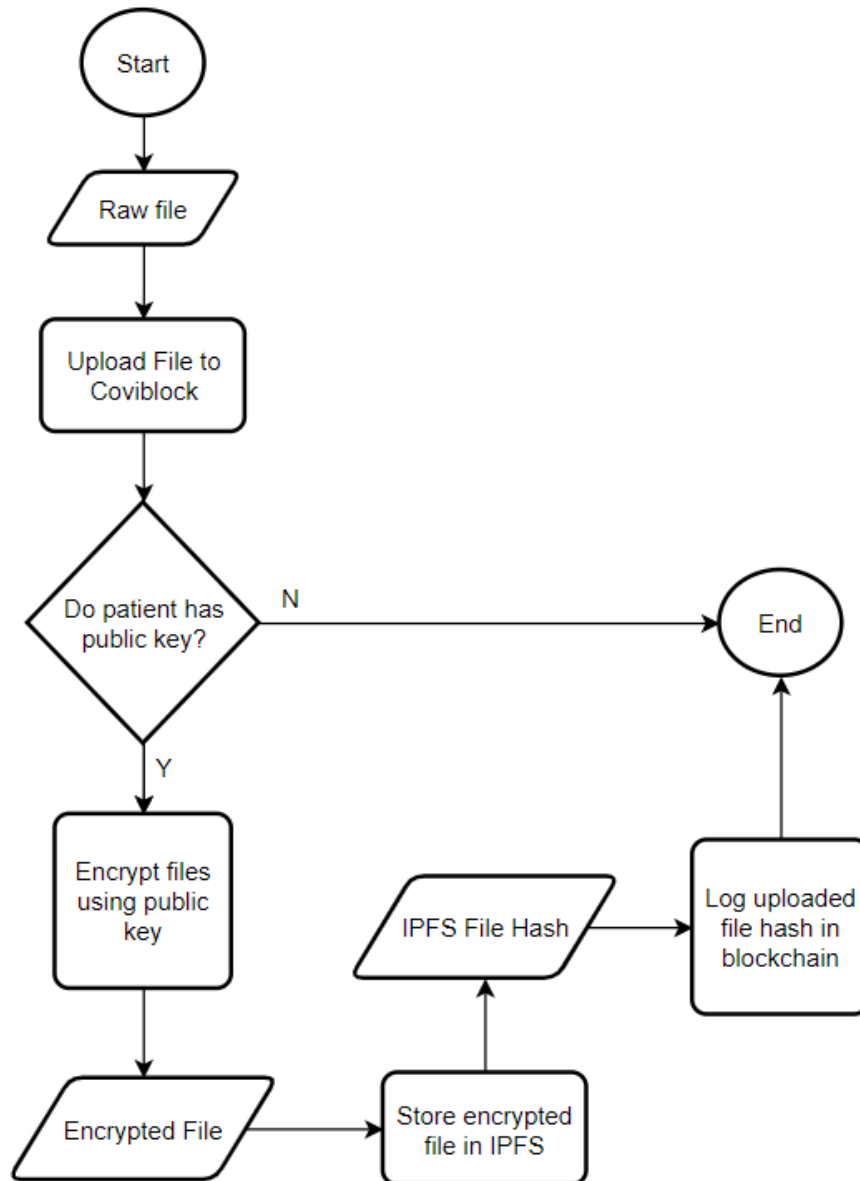


Figure 4.0.12: File Uploading Flowchart

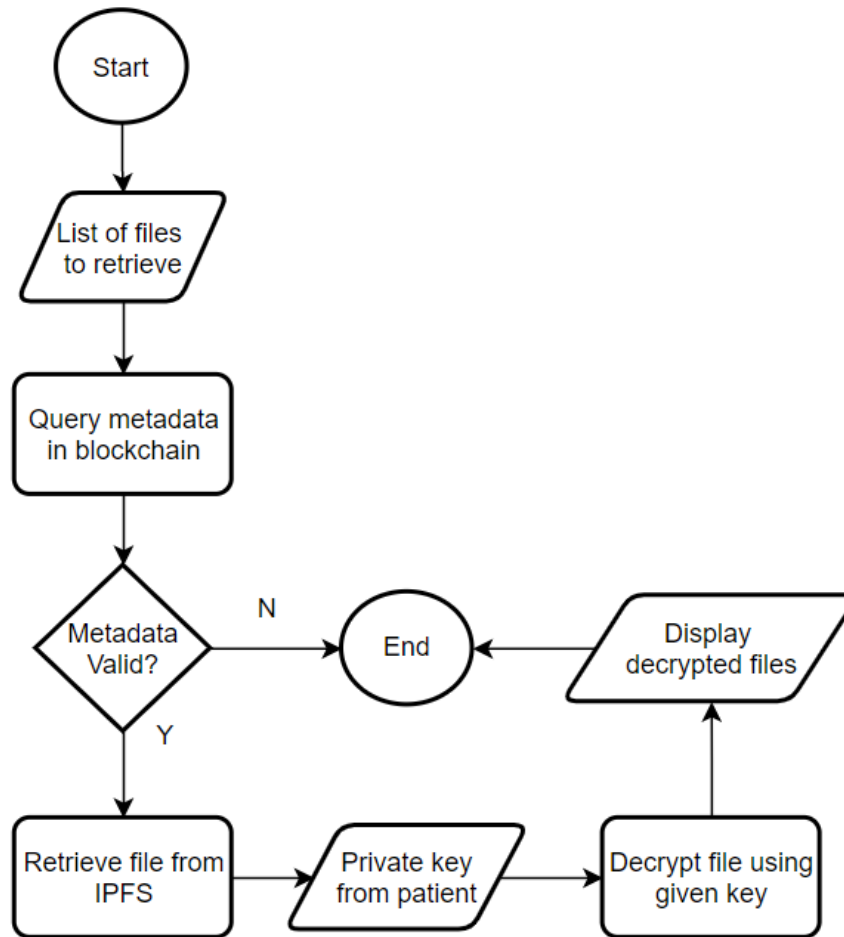


Figure 4.0.13: File Retrieval Flowchart

For the key generation process illustrated in figure 4.0.10, once the user generates keys it will then go to separate storages. Public key will be saved to the application database while private key will be the user's responsibility to store securely. Figure 4.0.11 details how a third party can request for files. Logs from the blockchain will be displayed to the third party that has information about files available for viewing. They will choose the file they want to access, and a request will be sent to the patient. The application will notify the patient that a request is sent to access their files and will be asked to provide keys. It will be patient's discretion if they will grant the request. If patient agrees, they will provide the private key to be used in decrypting the file requested.



Figure 4.0.12 illustrates how the application will handle uploading of files. Authorized medical personnel will trigger the upload. The application will check if the patient being referenced by the record has an existing public key. If yes, it will proceed on encrypting it using the key and uploading the encrypted file to IPFS. IPFS will generate a hash of the uploaded file. This hash will then be stored as a transaction in the blockchain. Figure 4.0.13 shows how the application will handle retrieval of files. The patient will provide the private key to enable decryption of files retrieved from IPFS.



LIST OF REFERENCES

Tsai, Jack and Bond, Gary (2007). *A comparison of electronic records to paper records in mental health centers*

Khalifa, Mohamed (2018). *Perceived Benefits of Implementing and Using Hospital Information Systems and Electronic Medical Records*

Ebardo, Ryan and Celis, Nelson (2019). *Barriers to the Adoption of Electronic Medical Records in Select Philippine Hospitals: A Case Study Approach*

Ebardo, Ryan and Tuazon, John Byron (2019). *Identifying Healthcare Information Systems Enablers in a Developing Economy*

Gesulga, Jaillah Mae; Berjame, Almarie; Moquiala, Kristelle Sheen; Galido, Adrian (2017). *Barriers to Electronic Health Record System Implementation and Information Systems Resources: A Structured Review*

Merkle, Ralph (1989). *A Certified Digital Signature*

Benet, Juan (2014). *IPFS - content addressed, versioned, P2P file system (draft 3)*

Maymounkov, Petar and Mazieres, David (2002). *Kademlia: A peer-to-peer information system based on the xor metric*

Freedman, Michael J.; Freudenthal, Eric; Mazieres, David (2004). *Democratizing content publication with Coral*



Baumgart, Ingmar and Mies, Sebastian (2007). *S/kademlia: A practicable approach towards secure key-based routing.*

National Institute of Standards and Technology (2002). FIPS-180-2: *Secure Hash Standard (SHS)*

Yoshida Hirotaka and Biryukov, Alex (2005). *Analysis of a SHA-256 Variant*

Gilbert Henri and Handschuh, Helena (2004). *Security Analysis of SHA-256 and Sisters*

Menezes, Alfred; van Oorschot, Paul; Vanstone, Scott (1997). *Handbook of Applied Cryptography*

Sklavos, Nicolas and Koufopavlou, Odysseas G. (2003). *On the hardware implementation of the SHA-2 (256, 384, 512) Hash functions*

Szydlo, Michael (2003). *Merkle tree traversal in log space and time*

Micali, Silvio; Jakobsson, Markus; Leighton, Tom; Szydlo, Michael (2003). *Fractal merkle tree representation and traversal.*

Xie, Ming (2003). *P2P Systems based on Distributed Hash Table*

Szabo, Nick (1997). *Smart contracts: formalizing and securing relationships on public networks.*

Yaga, Dylan; Mell, Peter; Roby, Nik; Scarfone, Karen (2018). *Blockchain Technology Overview*

Abadi, Joseph and Brunnermeier, Markus (2018). *Blockchain Economics*



Zheng, Zibin; Xie, Shaoan; Dai, Hong-Ning; Chen, Xiangping (2017). *An Overview of Blockchain Technology: Architecture, Consensus, and Future Trends*

Monrat, Ahmed Afif; Schelén, Olov; Andersson, Karl (2019). *A Survey of Blockchain From the Perspectives of Applications, Challenges, and Opportunities*

Fujisaki, Eiichiro and Okamoto, Tatsuaki (2011). *Secure Integration of Asymmetric and Symmetric Encryption Schemes*

Bellare, Mihir; Desai Anand; Pointcheval, David; Rogaway, Phillip (1998). *Relations among notions of security for public-key encryption schemes*

Goldwasser, Shafi and Micali, Silvio (1984). *Probabilistic encryption*

Azaria, Asaph; Ekblaw, Ariel; Vieira, Thiago; Lippman, Andrew (2016). *MedRec: Using Blockchain for Medical Data Access and Permission Management*

Sharma, Saurabh; Mishra, Ashish; Lala, Ajay; Singhai, Deeksha (2020). *Secure Cloud Storage Architecture for Digital Medical Record in Cloud Environment using Blockchain*

Kumar, Randhir and Tripathi, Rakesh (2020). *A Secure and Distributed Framework for sharing COVID-19 patient Reports using Consortium Blockchain and IPFS*

Wu, Sihua and Du, Jiang (2019). *Electronic medical record security sharing model based on blockchain*

Sun, Jin; Yao, Xiaomin; Wang, Shangping; Wu, Ying (2020). *Blockchain-based secure storage and access scheme for electronic medical records in IPFS*

Khubrani, Mousa Mohammed (2021). *A Framework for Blockchain-based Smart Health System*



Kumar, Shivansh; Bharti, Aman Kumar; Amin, Ruhul (2021). *Decentralized secure storage of medical records using Blockchain and IPFS: A comparative analysis with future directions*

Al Asad, Nafiz; Elahi, Md. Tausif; Al Hasan, Abdullah; Yousuf, Mohammad Abu (2020). *Permission-Based Blockchain with Proof of Authority for Secured Healthcare Data Sharing*

Reen, Gaganjeet (2019). *Decentralized Patient Centric e-Health Record Management System using Blockchain and IPFS*