UNIVERSITY OF TECHNOLOGY

(YATANARPON CYBER CITY)

FACULTY OF INFORMATION AND COMMUNICATION TECHNOLOGY

DEPARTMENT OF INFORMATION SCIENCE

DATA INTEGRITY ON BLOCKCHAIN

BY

YE MYINT OO

B.E THESIS

SEPTEMBER, 2018

UNIVERSITY OF TECHNOLOGY
(YATANARPON CYBER CITY)
FACULTY OF INFORMATION AND COMMUNICATIONTECHNOLOGY
DEPARTMENT OF INFORMATION SCIENCE

DATA INTEGRITY ON BLOCKCHAIN

BY
YE MYINT OO

A partially dissertation submitted to the University of Technology, Yatanarpon Cyber City in fulfillment of the requirement for the degree of

Bachelor of Engineering
(Information Science and Technology)

September, 2018

DECLARATION

I hereby declare that I carried out the work reported in this thesis in the Department of Information Science, Faculty of Information and Communication Technology, University of Technology (Yatanarpon Cyber City), under the supervision of Dr. Tin Win Maw. I solemnly declare that to the best of my knowledge, no part of this thesis has been submitted here or elsewhere in a previous application for award of a degree. All sources of knowledge used have been duly acknowledged.

……

26th October 2018

YE MYINT OO

6IST-152

ACKNOWLEDGEMENTS

First of all, I would like to express my special thanks to Dr. Aung Win, Rector, University of Technology (Yatanarpon Cyber City), for initiating the Master Degree Programme at the University of Technology (Yatanarpon Cyber City).

I would like to express grateful thanks to Dr. Hnin Aye Thant, Professor and Head Department of Information Science, Faculty of Information Science & Communication Technology, University of Technology (Yatanarpon Cyber City), for her kind guidance, encouragement in making my thesis to complete successfully.

I am very grateful to Dr. Phyu Phyu Tar, Professor and Course Coordinator of Information Science and Technology, Faculty of Information and Communication Technology, University of Technology (Yatanarpon Cyber City), for her arrangement, valuable suggestion, comments and advice in completion of thesis.

I am very grateful to my supervisor, Dr. Tin Win Maw, associative professor Faculty of Information and Communication Technology, University of Technology (Yatanarpon Cyber City), for her kind guidance and encouragement. She has been very supportive in this thesis, and also guides a lot, particularly, at the level of quality of presentation.

Very Special thanks to Daw Swe Zin Myint, Assistant Lecturer of English Department, University of Technology (Yatanarpon Cyber City) for her valuable supports from the language point of view in my thesis work.

I would like to thank a lot to all my teachers for their mentoring, encouragement, and recommending this dissertation.

Finally, I am grateful to my parents and friends who specially offered strong moral and physical support, care and kindness, during the year of my thesis study.

ABSTRACT

Nowadays, security system is considered to be critical for ensuring privacy in many IT industries. In these industries, data tampering becomes one of the biggest security threats as the stored data can be modified in many unauthorized ways. This system was built to prevent data tampering to the user data using blockchain technology, called Data Integrity on Blockchain. Data integrity usually means maintaining the data at the original stat over its entire life-cycle. In order to provide integrity to the data, this system used an algorithm called SHA algorithm which used the generated hash codes to validate the data. Data in this system are generally the criminal reports of the people. So, data integrity is achieved by validating each criminal report using SHA algorithm. This system is implemented in Web by using NodeJS, Express Server, JavaScript and HTML/CSS/Bootstrap programming languages.

TABLE OF CONTENTS

	Page
DECLARATION	i
APPROVAL	ii
ACKNOWLEDGEMENTS	iii
ABSTRACT	iv
TABLE OF CONTENTS	v
LIST OF FIGURES	vi
LIST OF EQUATIONS	ix
CHAPTER 1 INTRODUCTION	
1.1 Objectives	2
1.2 Field Background	2
1.2.1 SHA-256	3
1.2.2 Mathematical Background	3
1.3 Overview of the System	4
1.4 Organization of the Thesis	4
CHAPTER 2 THEORETICAL BACKGROUND	
2.1 Data Integrity	5
2.1.1 Merkle Tree and Blockchain Integrity	6
2.1.2 Data integrity Features of Blockchain	7
2.2 SHA-256 Algorithm	8
2.2.1 Pre-Processing	8
2.2.1.1 Message Padding	8
2.2.1.2 Message Parsing	8
2.2.2 Hash Computing	10
2.3 Blockchain	12
2.3.1 History of Blockchain	12
2.3.2 Data Structure of Blockchain	13
2.3.2.1 Block	13
2.3.2.2 Block Header	14
2.3.2.3 Genesis Block	15
2.3.2.4 Linking Blocks in Blockchain	15
2.3.2.5 Proof of Work	15
2.3.3 Decentralization and Consensus	16

2.3.4 Nodes 17
2.3.5 Distributed Ledger 19
2.3.6 Types of Blockchain 19
2.3.6.1 Public Blockchain 19
2.3.6.2 Private Blockchain 20
2.3.6.3 Consortium or Federated Blockchain 20
CHAPTER 3 SYSTEM DESIGN AND IMPLEMENTATION
3.1 System Design and System Flow 21
3.2 System Implementation 24
CHAPTER 4 CONCLUSION
4.1 Limitations of the System 39
4.2 Further Extension of the System 40
REFERENCES 41

LIST OF FIGURES

Figure		Page
2.1	Mekle Tree	6
2.2	Block Header	14
2.3	Proof of Work	16
2.4	Decentralized Networks	17
2.5	Mining Nodes	18
3.1	System Design for Creating New Block	21
3.2	System Design for Data Tampering	22
3.3	System Design for Validating New Block	23
3.4	System Design for Distributed Nodes 24	
3.5	Home Page	25
3.6	Input Page	25
3.7	Input Page	26
3.8	Successful Dialog	26
3.9	Blockchain Page	27
3.10	User Data	28
3.11	Block Header	29
3.12	Data in Blockchain	29
3.13	Hash linking	30
3.14	Before Refreshing the Page	30
3.15	After Refreshing the Page	31
3.16	Checking Integrity on Block Data	32
3.17	Checking Integrity on Block Data	32
3.18	Synchronizing Blockcahin Data	33
3.19	Changed Blockchain Data	33
3.20	Unchanged Blockchain Data	34
3.21	Generating a valid hash using Proof of Work	34
3.22	New Nonce for Modified Data	35
3.23	Modifying Current Blockchain	36
3.24	Records in Public Ledger	36
3.25	Using Postman to Add New Node	37

3.26 A new node is added by Postman 38
3.27 Public Ledger with connected nodes 38

LIST OF EQUATIONS

Equation		Page
2.1	Word Extension Equation	9
2.2	Compress Function	10
2.3	Final Hash Code	11

CHAPTER 1
INTRODUCTION

Security plays an important role in IT world. As data are stored online and transferred digitally, it becomes necessary to keep the data from being changed or modified by unauthorized users. There are many technologies or methods available nowadays that can strengthen the security of the user data efficiently and effectively. One of the technologies is called Blockchain.

Blockchain is a new technology that emerged in 2009 as bitcoin transaction. It is basically a distributed database or a public ledger that stores valuable data in a secured and tamper-proof way. It has many blocks connected each other and forms like a chain. Each block contains a cryptographic hash of the previous block, a timestamp and the user data. In blockchain, data are stored in many different distributed network nodes or devices, which are referred to users. The users are connected together to form a blockchain network and have the responsibility to maintain the blockchain. The security of the blockchian is relied on an algorithm called Proof-Of-Work algorithm where a nonce is increasing its values until a block hash code with leading zero bit is calculated. The process takes a lot of computational power to satisfy the blockchain network. So, in order to tamper any data in a blockchain network, the hackers need to process the computational power that is greater than the whole blockchain network. But nowadays, it is nearly impossible or very expensive to setup such powerful devices to take control over the blockchain. Thus, it makes the blockchain technology a security system to protect the crucial data.

In this system, blockchain is used as a way to store and secure the user data from being changed. Data integrity of the user data is ensured by using SHA-256 algorithm. Users can be connected each other to form a blockchain network. Users in the same network can add the data to the blockchain and the data can also be reviewed by all users. Blockchain are distributed into all connected users. Data tampering is proved by the way that if any connected user changes the data in the blockchain, the data added by that user will not be accepted by other honest users. Data integrity is ensured by the way that the honest users who don’t change the blockchain’s data will

be able to add the data to blockchain. Honest users mean the users who didn't change any data in the blockchain.

1.1 Objectives

The main objectives of the system are as follows:

- To share how blockchain technology work
- To achieve how data can be secured from tampering
- To demonstrate how SHA algorithm is applied to user data
- To implement how data is shared across distributed network nodes
- To build a demo application based on blockchain technology

1.2 Field Background

As data are stored online, people start knowing that having integrity of their data is very important. As a process, data integrity is the degree to which a collection of data is complete, consistent and accurate [8]. Technically, the most common way to provide data integrity can be achieved by using SHA algorithm.

SHA stands for Secure Hash Algorithm which is the cryptographic hash functions for computing a condensed digital representation or a hash code. It includes five algorithms which are SHA-1, SHA-224, SHA-256, SHA-384 and SHA-512. The later four algorithms are referred as SHA-2 version.

The original version of the algorithm was published in 1993 as the Secure Hash Standard by US government standards agency NIST (National Institute of Standards and Technology). This version is referred to as SHA-0. It was withdrawn by the NSA shortly after publication and was improved by the revised version, published in 1995 and commonly referred to as SHA-1. SHA-1 differs from SHA-0 only by a single bitwise rotation in the message schedule of its compression function; this was done, according to the NSA, to correct a flaw in the original algorithm which reduced its cryptographic security. Weaknesses have been reported in both SHA-0 and SHA-1. Thus, it led to the creation of the SHA-2 version. SHA-2 is considered to be the most secured and used version in the world. [2]

In this system, SHA-2 version is used to provide the integrity by validating the user data. The SHA algorithms are secured because they produce irreversible and unique hash code. It is computationally infeasible to find an original message that corresponds to a given hash code and it is also impossible to find two different messages that produce the same hash code. Minor changes to the data may result a

huge different to the hash code. Those make the SHA the most used algorithms in the online world to serve as the purpose of data integrity. [2]

1.2.1 SHA-256

SHA-256 is referred as SHA-2 version. It is one of the successor hash functions (SHA-1) and one of the strongest hash functions that are widely used nowadays. SHA-256 includes significant changes from its predecessor, SHA-1, and has not yet been compromised in any way.

The algorithm was first published in 2001 in FIPS PUB 180-2. After public review and comments were accepted, it becomes the new Secure Hash Standard in August 2002. It is widely used in many security applications and protocols, including TLS and SSL, PGP, SSH, S/MME, and IPSec.

SHA-256 is used to hash a message, M, having a length of L bits, where $0 \leq L < 2^{64}$. The algorithm uses

1. A message schedule of sixty-four 32-bit words
2. Eight working variables of 32 bits each, and
3. A hash value of eight 32-bit words.

The final result of SHA-256 is a 256-bit message digests [2]

1.2.2 Mathematical Background

SHA algorithm for calculating the hash code contains the mathematical functions and they require bitwise operations and mathematical values to generate the hash code. These values are initial constant hash values and round constant values. Bitwise operations generally contain XOR ($\oplus$) operation, NOT ($!$) operation, AND ($\wedge$) operation, Additional Modulo ($+$) operation, Right Shifting ($>>$) operation and Right Rotating ($>>>$) operation.

Initial constant hash values consist of eight hexadecimal values. These values are generated from the first eight prime numbers and applied to the first phase of the compress function.

Round constant hash values include sixty-four hexadecimal values and they are generated from first sixty-four prime numbers. Round constant hash values are required for the rest phases of the compress function.

The above operations and values are necessary to calculate the hash code of the input data. [1]

1.3 Overview of the System

This system uses SHA-256 algorithm and blockchain to provide the data integrity of the user data. The function of SHA-256 algorithm is generating hash codes of the user data and blockchain is to record the user data and hash codes. User data in the system is generally a simple criminal record of a person. To record the criminal data, user requires entering the details about the person. After successfully filling the details, user data are ready to be recorded to the blockchain. The values of hash codes and nonce are generated using SHA algorithm. The values and data are placed into the blocks.

Users can edit the stored data in their devices. if the data are modified, the blocks will show a red color indicating that the data have changed from its original stats. The blocks contain a mining button in each block that apply Proof-Of-Work algorithm to the modified data and generate the nonce and the valid hash codes. By generating a new hash code, users can understand that hacking a blockchain costs a lot of computational powers. User can also replace the modified blockchain with the original one. If the users have modified their blockchain or data, other users will not send or receive the data to those users. This system is a fully peer-to-peer system and no central authority is controlling the blockchain network.

The system has a front-end user interface and back-end system. Front-end user interface allows the users to input the data and store it on the blockchain. It includes an interface which visually shows the stored data in blockchain. The function of back-end system is to keep the network node running. If all the data are lost, they can be recovered by connecting to the blockchain network again.

1.4 Organization of the Thesis

In this book, there are four chapters to explain the system.

Chapter 1 describes introduction, objectives of the thesis, field background, and overview of system.

Chapter 2 discusses the SHA-256 algorithm, blockchain and data integrity

Chapter 3 presents the design and flow of the system and implementation of the system by using the SHA-256 algorithm and blockchain.

Chapter 4 addresses the conclusions, limitations and extensions of the thesis.

CHAPTER 2
THEORETICAL BACKGROUND

The system is about ensuring integrity to user data using blockchain technology. In ensuring the integrity, SHA algorithm is the global standard to all the systems in the world. Mostly in security systems, users are concerned about their data, So, SHA algorithms are merged into those systems to provide the integrity. On the other hand, blockchain technology is a distributed database or public ledger to store the user data.

In fact, there are many ways available to protect our data, for example, encryption and decryption methods which process user data into inaccessible form or encrypted data and then return the encrypted data into accessible form again. The drawback of these methods is that it has to process the entire data and the time efficiency is strongly based on the size of the input data and the devices' power. There is another method, called hashing which simply verifies the data through the generated hash code of that data. Hashing provides the faster and lower power consumptions because it doesn't need to take full device's power on the entire data.

In this system, hashing methods is applied to the user data. SHA algorithm is the main part of the blockchain because of data integrity. Hashing is an efficient method as the hash code is easy to be identified by the data owner. The following is the theoretical background about SHA-256 algorithm and blockchain technology.

2.1 Data Integrity

Data integrity is a fundamental component of information security. In its broadest use, "data integrity" refers to the accuracy and consistency of data stored in a database, data warehouse, data mart or other constructs. Data Integrity refers to the validity and accuracy of data rather than the act of protecting data. The term - Data Integrity - can be used to describe a process or a function. It is imposed within a database when it is designed and is authenticated through the ongoing use of error checking and validation routines. Data with "Integrity" is said to have a complete or original structure. The overall intent of any data integrity technique is the same: the ensured data is recorded exactly as the same as the recorded data is retrieved.

Data must be kept free from corruption, modification or unauthorized changes. Inaccuracies of data may occur either accidentally through programming errors or maliciously through hacking. Data are protected by

- Data encryption: data is locked using cipher.
- Data backup: a copy of data is stored in different location.
- Access controls: read/write privileges are set to the data.
- Data validation: hash is used to check the data corruption. [9]

2.1.1 Merkle Tree and Blockchain Integrity

Data validation is used in the blockchain to check the data integrity. All the transaction data in a block are hashed in the form of Merkle tree. In Merkle tree structure, hashes of child nodes are combined together into the parent node's header. This technique of combining the header of the child nodes and adding it to the header of the parent node continues iteratively till the final node which is located at top right, the root node. Thus, the root node will contain information about all of the nodes presented in the tree.

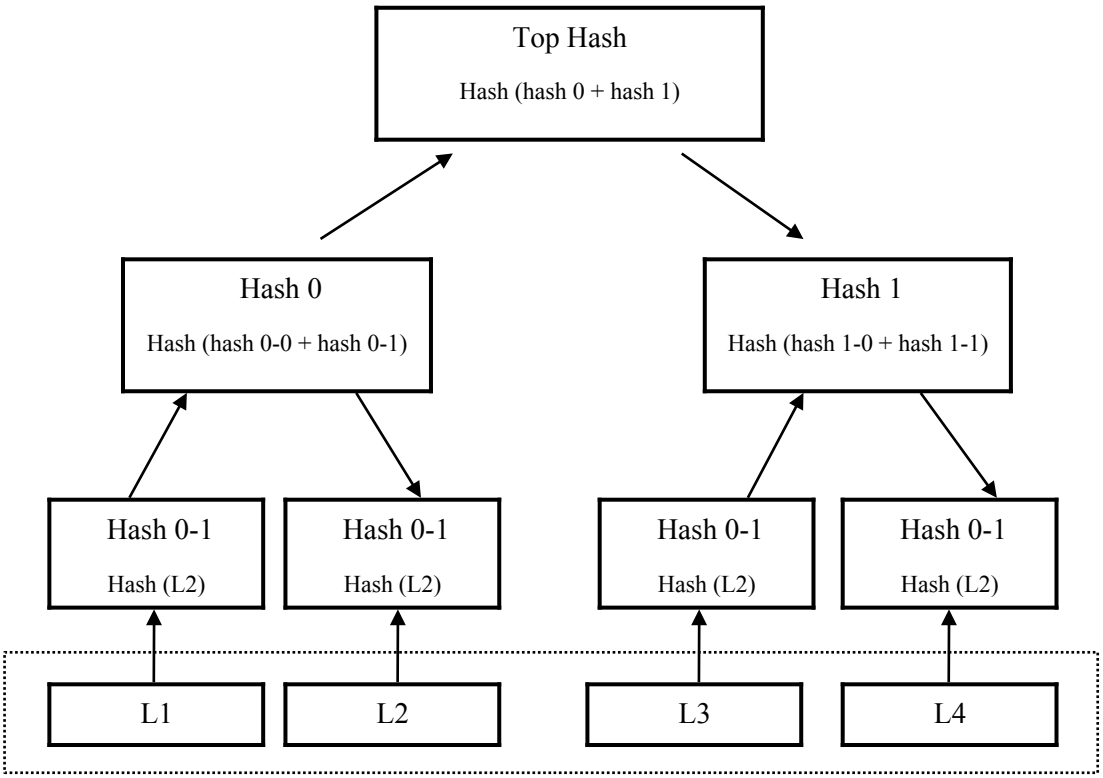

Figure 2.1 Merkle Tree

In Figure 2.1, there are four data L1, L2, L3 and L4 in a block as shown in Figure 2.5. The data are typically hashed such that the hash of L1 is 0-0, L2 is 0-1, L3 is 1-0 and L4 is 1-1. Combination of 0 and 1 is hashed and stored in its parent node. The final node of the hash is the Merkle tree. It is called the root node or simply the Merkle root. This Merkle root along with previous block header hash, a time-stamp and a nonce is used to generate the block header.

Hash of all the transactions in a block is stored in the Merkle tree. So when a node wants to verify if any transaction is changed, the node will only have to build the Merkle tree using all the transactions of block. This makes it very simple to validate a transaction. Thus, Merkle tree helps to maintain the security in Blockchain. [10]

2.1.2 Data Integrity Features of Blockchain

Data stored in the digital form can be vulnerable to tampering. This is the main security problem that are issuing the most industries concerning with IT technology. For example, some industries are regularly moving the storages of sensitive data from one location to another and the data can be exposed to hackers or any attackers and the widespread adoption of Internet of Things (IoT) to the systems have also been required by the need to ensure that data are not corrupted or tampered. With the blockchain technology, some of those big problems can probably be solved to a very low scale because it provides a lot of data integrity features and benefits to the industries and users.

The followings are the main features of the blockchain technology,

- Scalability: The service is scalable as the nodes can verify the growing data without effecting performance.
- Real-time protection: tampering to the data can be detected by monitoring.
- Long-term validity: Validation is based on hash function and will not be expired.
- Portability: The service is portable across geographies, organizational and service providers.
- Carrier-grade: The solution architecture delivers 99.999 percent availability.
- Quantum-immune: Validation does not depend on asymmetric or elliptic curve cryptography. [8]

2.2 SHA-256 Algorithm

Secure Hash Algorithm (SHA-256) is a mathematical operation that is mainly used for data validation. The operation of SHA is to calculate the hash code of the digital data. The hash code is the unique identity to the data and by comparing the computed hash code; a person can determine the data's integrity. [2]

SHA-256 algorithm is invented to generate the unique hash code of the user data. To ensure hash code is unique and secured, the algorithm uses computational power and mathematical functions to calculate the hash code. The algorithm requires two steps to calculate the hash code:

- Pre-processing
- Hash computing

2.2.1 Pre-Processing

Pre-Processing has two steps,

- Message Padding
- Message Parsing

2.2.1.1 Message Padding

The process breaks the input message as binary digit into many blocks. The length of the input message requires less than 2^256 bits long. Each block contains 512-bits of the input message. If the last block of the input message does not exceed 512-bits, the extra bits will be appended. For the case, the last block is 448-bits length. Single '1' bit is appended to the last block, and then followed by adding '0' bit. The 448-bits length requires 64-bits to be filled to become 512-bits length. Thus, 64-bits length represented as the binary digit is appended to the last bit of the total 512-bits length. After padding, the new message is then ready for message parsing. [1]

2.2.1.2 Message Parsing

The input message is padded and processed into blocks in the previous step. Each block of the new message now has 512-bits length. The 512-bits blocks in this step are converted into words. Each word has a 32-bits length. One block of 512-bits length is equal to sixteen word of 32-bits length (W [0...15]).

16 words of each block is then extended to 64 words (W [0...64]) by using the mathematical functions. The reason the words are extended because compress functions for calculating the hash code require total sixty-four times. The following equations are to extend the 16 words to remaining 48 words. [2]

For i from 16 to 63,

$$N1 = (W[i-15] >>> 7) \oplus (W[i-15] >>> 18) \oplus (W[i-15] >> 3)$$

The bitwise operations (Right-rotation and Right-shift) applied to the words produce new value and assigned to N1.

$$N2 = (W[i-2] >>> 17) \oplus (W[i-2] >>> 19) \oplus (W[i-2] >> 10)$$

The bitwise operations applied to the words result new value to N2.

$$W[i] = W[i-16] + V1 + W[i-7] + V2$$

The result is the total of 64 words and words are assigned into array, W [0...63].

There are initial values to be assigned before going to the next step. The initial values are hash values and round constants. Initial hash values are the eight values in hexadecimal format. The value is constant and derived from the first 32-bits of the fractional parts of the square roots of the first eight prime numbers (2...19). They are

- V0 = 6a09e667
- V1 = bb67ae5
- V3 = 3c6ef372
- V4 = 510e527f
- V5 = 9b05688c
- V6 = 1f83d9ab
- V7 = 5be0cd19

The round constants are also hexadecimal values that are derived from the first 32-bits of the fractional parts of the cube roots of the first 64 prime numbers. Thus, it has 64 hash values. These values are as follow, K [0...63] =
428a2f98, 71374491, b5c0fbcf, e9b5dba5, 3956c25b, 59f111f1, 923f82a4, ab1c5ed5, d807aa98, 12835b01, 243185be, 550c7dc3, 72be5d74, 80deb1fe, 9bdc06a7, c19bf174, e49b69c1, efbe4786, 0fc19dc6, 240ca1cc, 2de92c6f, 4a7484aa, 5cb0a9dc, 76f988da, 983e5152, a831c66d, b00327c8, bf597fc7, c6e00bf3, d5a79147, 06ca6351, 14292967, 27b70a85, 2e1b2138, 4d2c6dfc, 3380d13, 650a7354, 766a0abb, 81c2c92e, 92722c85, a2bfe8a1, a81a664b, c24b8b70, c76c51a3, d192e819, d6990624, f40e3585, 106aa070, 19a4c116, 2748774c, 34b0bcb5, 391c0cb3, 4ed8aa4a, 5b9cca4f, 682e6ff3, 748f82ee, 78a5636f, 84c87814, 8cc70208, 90befffa, a4506ceb, ef9a3f, c67178f2
The hash values and the round constants values are used in hash computing step. [3]

2.2.2 Hash Computing

After the input messages are pre-processed, various mathematical functions are applied to the messages to calculate the hash in this step. The mathematical functions are known as compress functions. There are generally four compress functions that are going to use. The initial hash values are assigned into working variables to be used in the functions. [2]

- A = V0
- B = V1
- C = V2
- D = V3
- E = V4
- F = V5
- G = V6
- H = V7

The working variables and round constants are used in the following compress functions. The compress functions require sixty-four looping to go through to calculate the final hash.

Looping for i from 0 to 63,

- E1 (E) = (E⋙6) ⊕(E⋙11) ⊕(E⋙25)

The first function uses Right-rotation operation to "E" working variable. The result is the new hexadecimal hash.

- Ch (E, F, G) = (E ^ F) ⊕ (! E ^! G)

The second function use AND, XOR and NOT operations to E, F, G working variables and produce the new hash.

- Temp1 = H + E1 +Ch +k [i] + W [i]

The H variable, the hash of the first and second functions, i^{th} number of the round constant and i^{th} number of the message word are compressed as a new hash.

- E0 (A) = (A⋙2) ⊕(A⋙13) ⊕(A⋙22)

The third function applies Right-rotation and XOR operations to variable "A" and a new hash is produced.

- Maj (A, B, C) = (A ^ B) ⊕ (A ^ C) ⊕ (B ^ C)

The fourth function perform AND, XOR operations to variables A, B and C. [3]

- Temp2 = E0 + Maj

The hashes from functions E0 and Maj are compressed into Temp2.
E1, Ch, E0, Maj are 4 compress functions. The functions generate new hash values and assigned to working variables in each loop as the following,

- H = G
- G = F
- F = E
- E = D + Temp1
- D = C
- C = B
- B = A
- A = Temp1 + Temp2

Each time the functions going through the loop, the new working variable and initial hash values are compressed to initial hash values. The new initial hash values are going to use in the next loop. [2]

- V0 = V0 + A
- V1 = V1 + B
- V2 = V2 + C
- V3 = V3 + D
- V4 = V4 + E
- V5 = V5 + F
- V6 = V6 + G
- V7 = V7 + H

After the finishing the looping, the value from the initial hash are joined together. The joined hash is the final hash code for the input message.

- Hash = V0 append V1 append V2 append V3 append V4 append V5 append V6 append V7

For example,

- Hash = ae03d1cd + 9de17113 + f677853e + ee02a150 + f996e2b7 + 4870e035 + 322e8b60 + 59a3cdaa [3]

2.3 Blockchain

A blockchian is a growing list of shared and immutable distributed ledgers that facilitates the process of recording transactions and tracking recorded history among blockchain network. Virtually anything of value can be tracked back. Blocks in the blockchain are linked with cryptography that each block contains a cryptographic hash of the previous block, a timestamp and data.

Security of blockchain is designed to be resistant to modification of the recorded data. Interfering with transactions on the blockchain is extremely difficult due to the complex cryptography employed and its nature of distributed ledger. Blockchain is a distributed ledger that can record transactions between senders and receivers in a permanent and verifiable way. For use as a distributed ledger, blockchains are usually managed by a peer-to-peer network that every participant in the blockchain can view any changes and transaction history. Once recorded, the data in any given block cannot be altered without altering the rest of the blocks. The altering action requires consensus of the majority of the network nodes to agree with. [7]

2.3.1 History of Blockchain

A structure which was similar to blockchain was mentioned in a research paper titled "How to Time-Stamp a Digital Document" in 1991 by Haber and Stornetta. According to that paper, a client sends a document timestamp to a time stamping server and the server would sign the document with the current timestamp. Also, the server would link the document to the previous document. Thus, document's timestamp could not be tampered or backdated. [9]

The first blockchain that applied in the real work was created by a person or a group, which is currently unidentified, called Satoshi Nakamoto who was possibly claimed to be a man living in Japan. Blockchain was published in 2008 as an online crypto currency called Bitcoin. As described in Nakamoto's paper, bitcoin is a peer-to-peer electronic cash system. The main purpose of bitcoin is to enable the users to transact the digital cash directly without relying on a third party and also to solve the double-spending problem by using peer-to-peer network. The bitcoin, where it serves as the public ledger for transactions was implemented and developed by Satoshi Nakamoto until 2010. On 3rd January 2009, the first bitcoin blockchain network came into existence with Satoshi Nakamoto mining the genesis block of bitcoin which was the first block with block number 0 was created. As the following year, the

participants around the world started using bitcoin and making bitcoin transactions. At that time, 1 bitcoin was equivalent to 0.08 USD dollars. In 2018, 1 bitcoin is worth 6000 USD dollars. The highest peak of the bitcoin was that 1 bitcoin was equal to the values of 19,000 dollars which was occurred in 2017.

As the bitcoin was continuously running, its size grew. In August 2014, the bitcoin blockchain file size, containing the record of all transactions reached 20 GB. Following the year 2015 January, the size had continually grown up to 30 GB and later, it grew from 50 GB to 100 GB in size.

Since the bitcoin had been popularly used, people started focusing on the technology using in it, Blockchain. From 2014, people around the world started shifting attention from bitcoin to blockchain. The world realized that Blockchain can be separated from the crypto currency and applied to various other use-cases such as smart contract, trade surveillance, collateral management, insurance, banking, marketing, settlement and capturing historical ownership of high-value items. [8]

2.3.2 Data Structure of Blockchain

A blockchain is a decentralized, distributed and public digital ledger that is used to store the record across many computers. It is a back-linked list of blocks of records. The records cannot be altered without altering all subsequent blocks and consensus of the network. Record in blockchain can be stored as a raw file or in a simple database. Each block is identifiable by a hash, generated using SHA-256 cryptographic hash algorithm on the header of the block. Each block references a previous block, also known as the parent block. Since blockchain database is managed by a peer-to-peer network, all the participants have a benefit of cost-effectiveness and time-efficiency. Blockchain-based system can be completely quicker, safer and cheaper than the traditional systems. [10]

2.3.2.1 Block

A block is a container data structure, which brings the valid transactions that are hashed and encoded for inclusion in the public ledger, known as the blockchain. The block is made up of a header, containing metadata, followed by a list of transaction data. The block header consists of three sets of block metadata. Firstly, there is a reference to a previous block hash, which connects current block to the previous block, lying in the blockchain. The second set relates to the miner who minded the block that is timestamp and nonce. The third set is the hash used to

summarize the transaction in the blockchain. Each block includes the cryptographic hash of the previous block that link back to the genesis block and form a chain. [9]

2.3.2.2 Block Header

The block header consists of three sets of block metadata as shown in Figure 2.2. Metadata is data that provides information about other data. Firstly, there is a reference to a previous block hash, which connects this block to the previous block, lying in the blockchain. The second set of metadata relates to the mining competition; namely the difficulty, timestamp and nonce. Lastly, the third piece of metadata is the Merkle Tree root; a data structure used to summarize all the transactions in the block in an efficient manner. [10]

Size	Field	Description
4 bytes	Version	The Bitcoin Version Number
32 bytes	Previous Block Hash	The previous block header hash
32 bytes	Merkel Root	A hash of the root the Merkle tree of this block's transactions
4 bytes	Timestamp	The timestamp of the block in UNIX
4 bytes	Difficulty Target	The difficulty target for the block
4 bytes	Nonce	The counter used by miners to generate a correct hash

Figure 2.2 Block Header

Block headers can be regarded as an example of a dynamic membership multi-party signature (DMSS). DMSS is a digital signature formed by a set of signers which has no fixed size (Back, Corallo, Dashjr, & Friedenbach, 2014). Bitcoin's block headers are DMSS because their proof of work has the property that anyone can contribute without undergoing an enrolment process. Furthermore, contribution is weighted by proportional computational power rather than one threshold signature contribution per party (Back, Corallo, Dashjr, & Friedenbach, 2014). This allows anonymous membership without risk of a Sybil attack. A Sybil attack is when one party joins many times and has an uneven, disproportionate input into the signature.

Since the blocks are chained together, Bitcoin's DMSS is cumulative. A chain of block headers is also a DMSS on its first block, with computational strength equivalent to the sum of the computational strengths of the composing DMSS. Therefore, the key innovation in Blockchain is a signature of computational power, rather than the typical signature of knowledge. [9]

2.3.2.3 Genesis Block

A genesis block is the first block of a blockchain. It has a block number of 0. The genesis block is almost hardcoded in the blockchain as it is a special block which does not contain its previous block.

The genesis block is the backbone of the entire blockchain. Whenever a new block is created, the previous hash of the new block will eventually be related to the previous block one by one back to the genesis block. Every node can identify the hash of the genesis block and its structure, the fixed time of creation and any data within it. Thus, every node has a secure "root" from which it is possible to build a trusted blockchain on. [8]

2.3.2.4 Linking Blocks in Blockchain

Nodes maintain a copy of the blockchain locally, starting from the genesis block. The local copy of the blockchain constantly updates as new blocks are discovered and subsequently built on the chain. As a node receives information of incoming blocks from the network, it will validate these blocks first, and then link them to the existing blockchain.

The process to establish a link is as follows; a node will examine the incoming block header and look for the "previous block hash". Looking at this incoming block, the node finds the "previous block hash" field, which contains the hash of its parent block. This hash is known to the node previously. Therefore, the node reasons that this new block is a child of the last block on the chain, and is the legitimate extension of the chain. The node adds this new block to the end of the chain, making the blockchain longer with a new height of the incoming block, now validated. [8]

2.3.2.5 Proof of Work

A Proof-of-Work (PoW) system, to deter the denial of service attack and other service abuses, is widely known as mining algorithm in blockchain. The algorithm is originally derived from another algorithm called "Hash cash". Mining algorithm typically uses a piece of data which is difficult (power and time consuming) to

produce but easy for other to verify. It works like computational puzzle to give CPU a massive workload to solve the puzzle. In bitcoin blockchain, the node which successfully solved the puzzle will be rewarded bitcoin cash.

The algorithm is to find a satisfied value or nonce that when hashed with SHA-256, the hash value begins with required number of zero bits. Number of zero bits required for a hash is referred as the difficulty target for that hash. To do that, the nonce is increasing its value from zero until the block's hash start with prefix zero bits is found. After the result satisfies the proof-of-work, a new block is created and added to the blockchain. The new block cannot be changed or altered without finding the value again. The process requires immense amount of energy and computational usage but verifying whether the block belongs to the chain or not is a simple process.

The required hash for the block should be begun with '0000' as shown in Figure 2.3. The number of difficulty target for the hash is four. Each time the nonce is increased, a new hash is generated. Finally, the valid hash for "Hello, world!" is found after 4250 tries. [8]

Message	Count	Hash Code
Hello World!	0	1312af178c253f8402…..9ec81976192e2ec934c64
Hello World!	1	e9afc424b79e4f6av42d99…..9be78e948a9332a7d8
Hello World!	2	ae37343a357a8297…..28ve8ca2a32aa475cf05fddf8
…		
Hello World!	4248	ab5e6224a6521e…..e65f3646b366a6566b466e6f566
Hello World!	4249	b5af5e8a45f69b……55a9f8b5fe65f54a6ab32fae56fb5a6
Hello World!	4250	0000afb5e5af5b5fe5a4f1b…..a45f5b2f74ea5fb45e56fab5f

Figure 2.3 Proof of Work

2.3.3 Decentralization and Consensus

One of the most exciting aspects of blockchain technology is that it is entirely decentralized. The decentralized nature of blockchain technology means that it doesn't rely on a central point of control. Elimination to a single authority makes the system considerably more secure and safety. To securely transact the data without relying on

central authority, blockchain utilizes an innovative protocol, called consensus protocol, across a network of nodes to validate and record blockchain data in an incorruptible way. [6]

A consensus protocol is fault-tolerant and has a set of rules that describes how the communication and transmission of data between network nodes works and achieves agreement on a single data value among distributed processes. The protocol's purpose is to decide whether to commit a distributed transaction to the public ledger, synchronize the replicas of the ledger among the network, ensure consistency and guarantee the blockchain network to function without being corrupted. [6]

Furthermore, the information recorded on the blockchain can be ensured that it is nearly impossible to manipulate due to there being multiple copies that require a complex consensus to be handled. Being entirely decentralized, attacking one point of storage would result in no loss of data because the information stored on multiple devices around the world.

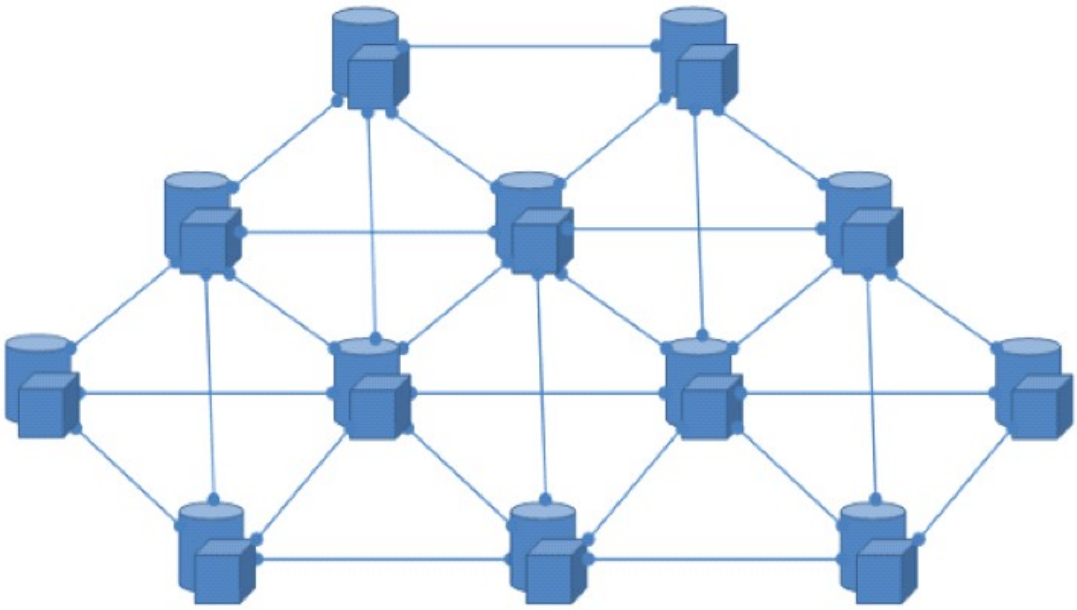

Figure 2.4 Decentralized Networks

2.3.4 Nodes

A node is a device on a blockchain network that allows the blockchain to function and survive. Nodes are distributed across over the networks and have the responsibility to maintain the blockchain.

A node is a device on a blockchain network that allows the blockchain to function and survive as shown in Figure 2.5. Nodes are distributed across over the networks and have the responsibility to maintain the blockchain.

A node can be any electronic device, including a computer, phone as long as it is connected to the internet. The role of a node is to support the network by maintaining a copy of a blockchain. The nodes are also to capable of processing the transactions, synchronizing the blockchain. Each crypto currency has its own nodes, maintaining the transaction records. Nodes are the individual part of the large data structure, blockchain. As the nodes' owners contribute their computing resources to keep blockchain running. They have a chance of earning the extra fees through mining or PoW. [10]

Processing the transactions require large amount of computing power, meaning that normal computers are not capable of doing it efficiently. Generally, the mining nodes in the blockchain need to invest extremely powerful computing devices such as CPUs (central processing units) or GPUs (graphics processing units). However, the user's node can be installed the reasonable computing devices that have the enough power to run the blockchan, but not participating in mining the new block.

Figure 2.5 Mining Nodes

2.3.5 Distributed Ledger

A distributed ledger is a database that is consensually shared and synchronized across the network and spread across multiple nodes, institutions or geographies. It allows transactions to have public “witnesses”, making a cyber attack more difficult. The participant at each node of the network can access the recordings shared across that network and can own an identical copy of it. The distributed ledgers are in a dynamic form that hold and update the transaction records independently by each node in a large network. Any changes made to the ledgers are reflected and copied to all participants in the matter of second or minutes.

A distributed ledger can be described as a ledger of any transactions or data that are maintained in decentralized forms across different locations and eliminating the need of a centr5al authority to keep a check against manipulation. All the information on it is securely and accurately stored using cryptographic signatures. Once the information is stored, it becomes an immutable database and is governed by the rules of the network. While centralized ledgers are prone to cyber-attack, distributed ledgers are harder to get attacked because all the distributed copies need to be taken down simultaneously for an attack to be successful. [8]

2.3.6 Types of Blockchain

There are three types of Blockchains that are mainly used in the industries. They are

- Public Blockchain
- Private Blockchain
- Consortium or Federated Blockchain

2.3.6.1 Public Blockchain

A public blockchain is designed to securely exchange asset by setting up a peer-to-peer transaction. Each transaction is verified and synchronized with every node affiliated with the blockchain before it is written to the system. It means that no one is in charge and any participant can read, write and auditing the blockchain. The public blockchains are open and transparent hence anyone can review the entire history. The power needed to run blockchain increases with each additional node. The benefit is that blockchain is more secured when the node's count is increased. Compare to private blockchain, public blockchain costs more and slower speed. But less expensive and faster than accounting systems that used today. Some well-known

crypto currencies using public blockchain are Bitcoin, Ethereum, Litecoin, Monero and Dash. [10]

2.3.6.2 Private Blockchain

A private blockchain is the blockchain that is used for the private individual or organizational purpose. Unlike public blockchain, private blockchain lets the middleman look after the important things such as granting read/write to particular participants. In order to be a new participant, an invitation or permission has to be granted to the participant by an owner of the blockchain, existing participants or a certain regulatory authority. It might seem that a blockchain is no longer the blockchain as it lacks the transparency and decentralization. But within it, the rules remain the same like public blockchian and it is still transparent for all the participants.

This kind of blockchains is commonly used in business companies. The companies get much greater efficiency and significantly faster transaction speed. But it allows for greater privacy as the read/write is controlled by the middleman. There are many big companies that are relying on private blockchain such as Ripple, Monax, Multichain and Chain Inc. [10]

2.3.6.3 Consortium or Federated Blockchain

Consortium blockchain is partly private and semi-decentralized. Instead of allowing one person to participate in the verification and transaction, a group of persons or companies coming together and making decision for the best benefit of the whole network. The groups are called consortiums or federation. It has the same privacy as private blockchain. This kind of blockchain is great for collaboration between organizations or companies.

Consortium blockchains are often associated with enterprise use, with a group of companies collaborating together to leverage blockchain technology for improved business processes. Examples of consortium blockchins are Quorum, hyper ledger by IBM and Corda. [10]

CHAPTER 3
SYSTEM DESIGN AND IMPLEMENTATION

This chapter describes the system design and system flow in detail. The implementation of the system includes the guides on how to run the system and explains its usage clearly.

3.1 System Design and System Flow

The system design is implemented as data integrity on blockchain. The system designs show that how a new block is created after the users added the data, how data tampering is proved by using data validation, and how to validate a new block that is broadcasted by other users in the network.

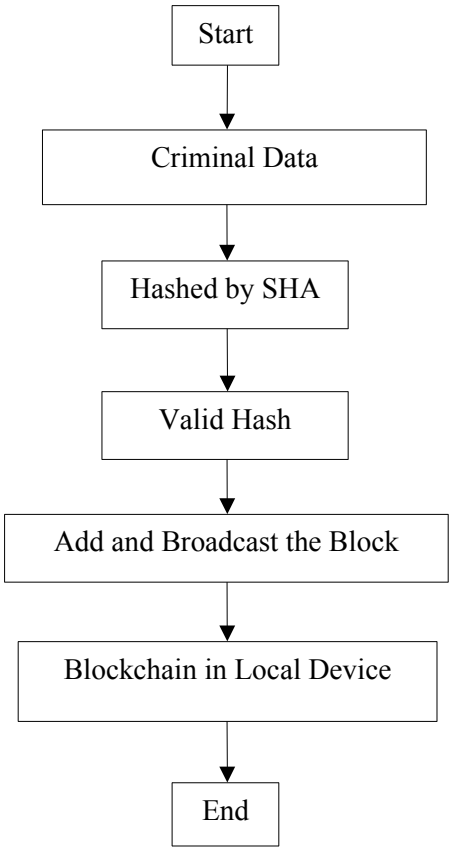

Figure 3.1 System Design for Creating New Block

In Figure 3.1, the criminal data is filled by the user. After the data is ready, SHA-256 algorithm is applied to the hash code of the previous block and the data. The process is looking for a nonce value that when hashed with SHA-256, it results a satisfied hash code of the current block, starting with four zeros. After finding the satisfied hash code, a new block is created with the current timestamp. The data, nonce value, hash code of the previous block and hash code of the current block are then added to the new block. The new block is created with resources and considered to be a valid block. The block is also broadcasted and can be validated by other users.

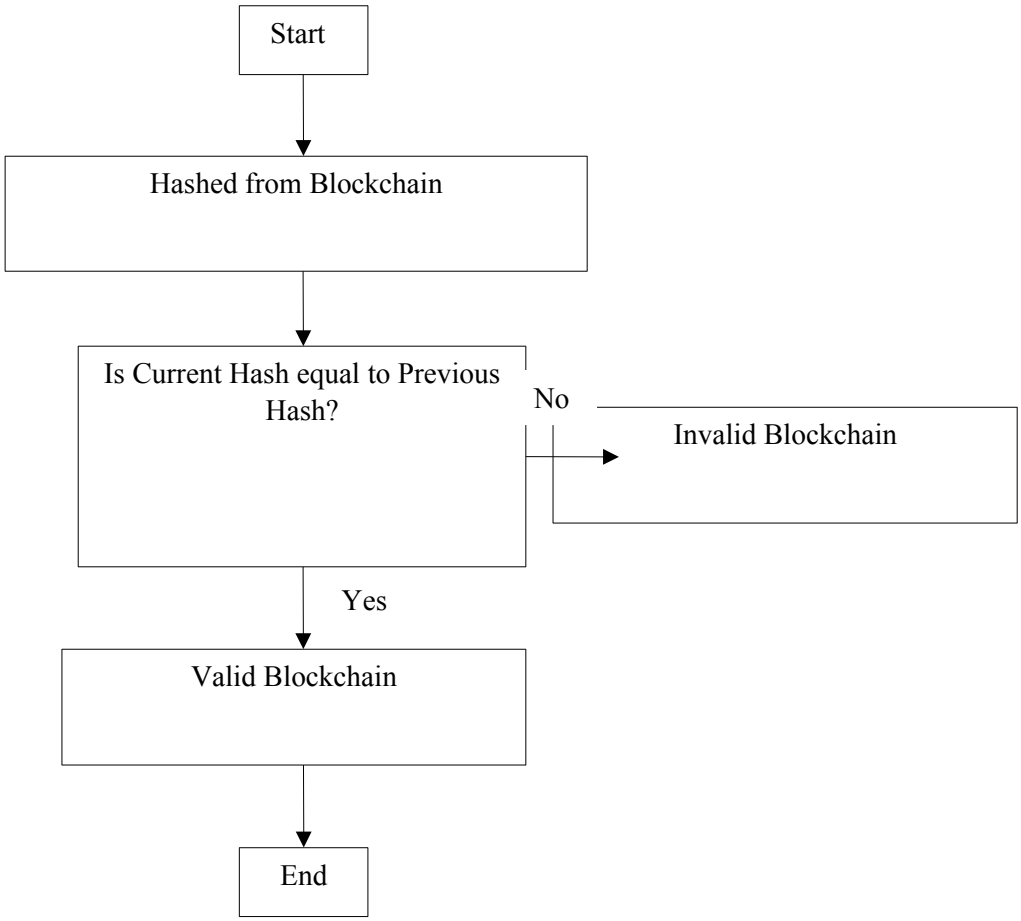

Figure 3.2 System Design for Data Tampering

The system detects data tampering through data validation as shown in Figure 3.2. Blocks in the blockchain are linking each other using cryptographic hash functions. As each block contains a hash code of the previous block and that of the current block, data validation is achieved by comparing the hash codes of the two blocks. Since changing the data in one block will alter the hashes in the subsequent

blocks, any changes made to the data in one block will result an invalid blockchain. Thus, data tampering can be detected by checking the validation status of the blockchain.

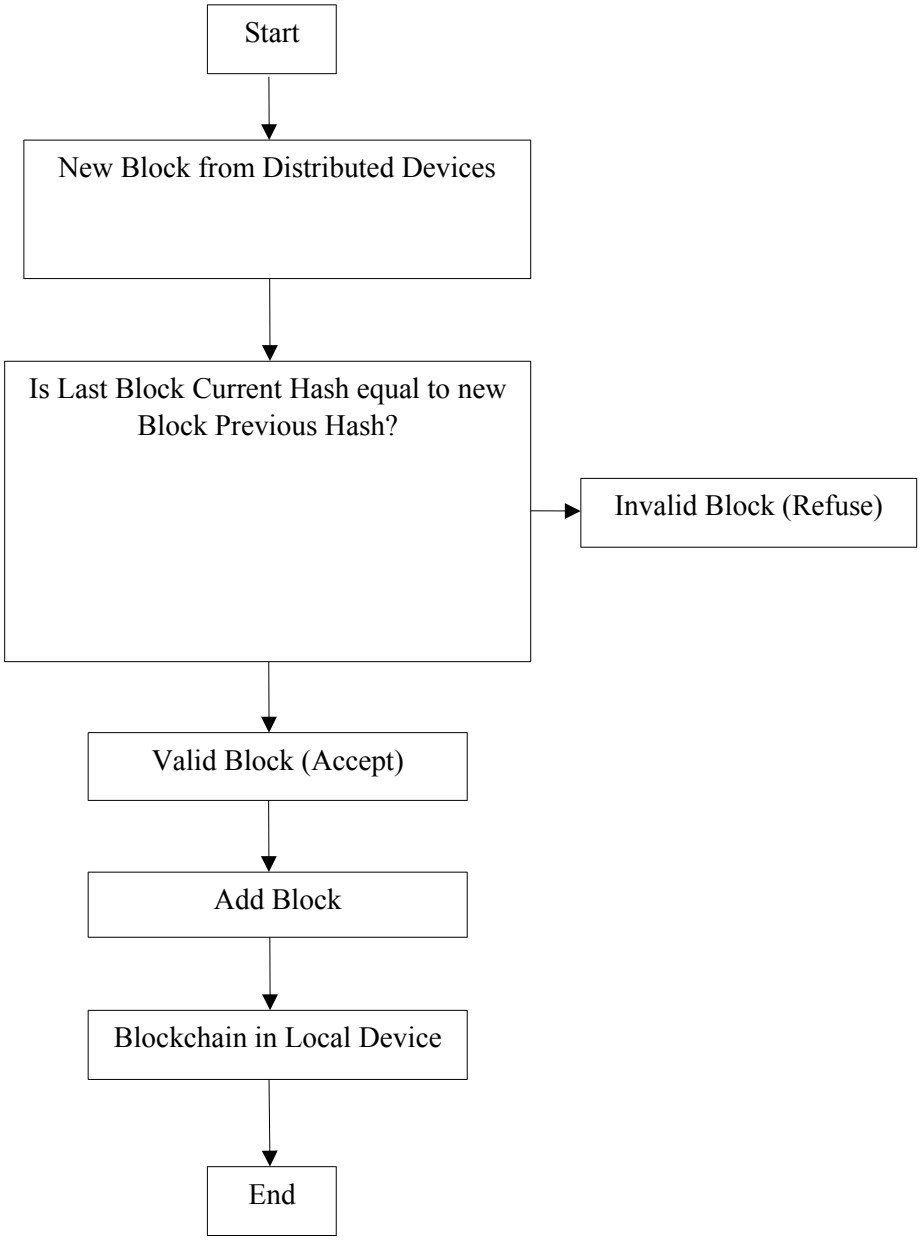

Figure 3.3 System Design for Validating New Block

A new block containing the data, the nonce value, the previous hash code and the current hash code of the block is broadcasted by the network devices as shown in Figure 3.3. The devices or nodes are connected to the blockchain network and participating in broadcasting the new blocks. Validation for the blocks required

because only the valid blocks will be added to the blockchain. Data validation to the block is done by comparing the current hash code of the block and the previous hash code of the new block. If the incoming block's blockchain is modified, the hash code will not be the same. If the hash codes are not the same, the new block will not be accepted and added to the blockchain.

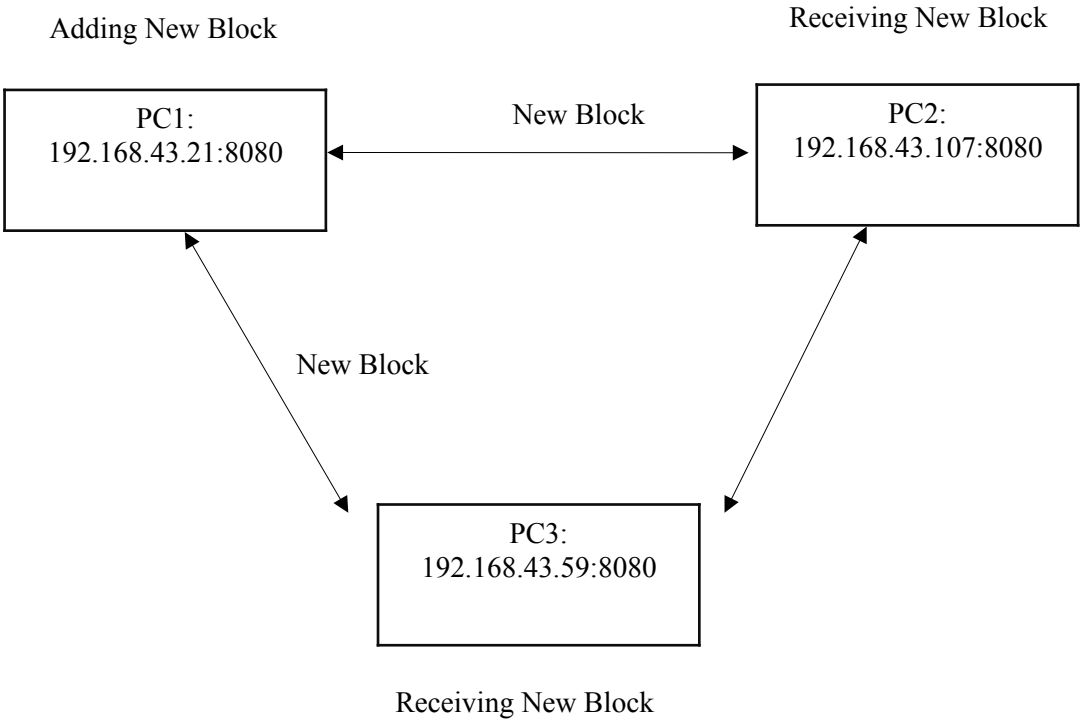

Figure 3.4 System Design for Distributed Nodes

There are three nodes currently connected in the same network and no central authority exists between them. Each PC contains a blockchain and the blockchains in the PCs are storing the same block and data. If any PC in the network creates a new block, the block will be broadcasted to the other two nodes. If the nodes more than two are, the procedures will be the same.

A new node can join and participate in the network. After joining the network, the node will store the same blockchain as in other nodes and can start adding the data to the blockchain.

3.2 System Implementation

This system is to implement "Data Integrity on Blockchain" that data recorded in this system are tamper-proof.

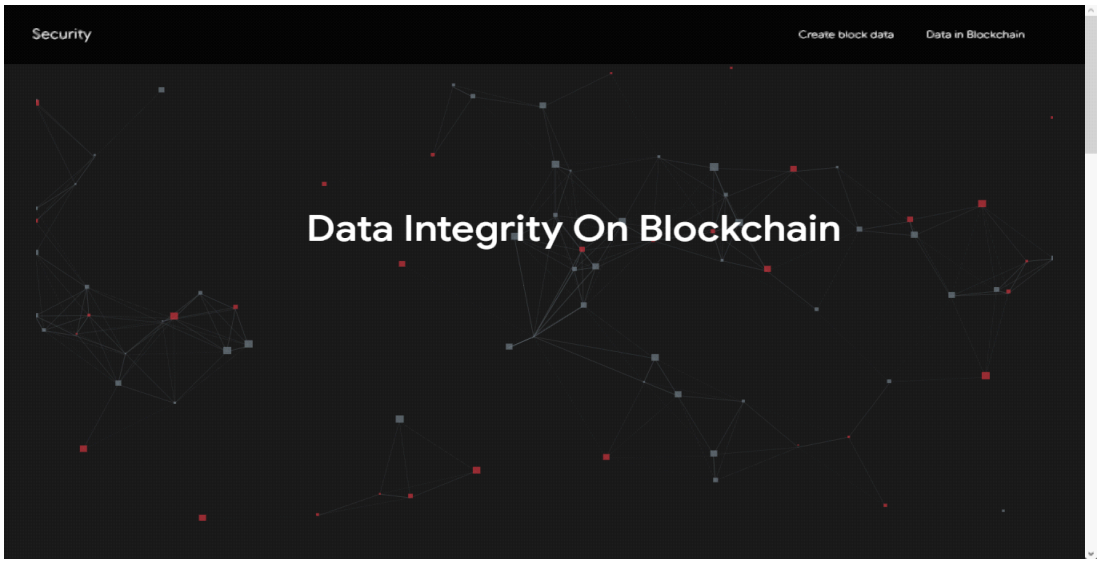

Figure 3.5 Home Page

When the users start running the system, this is the first page that will be seen as shown in Figure 3.5. The system is greeting with the text "Data Integrity On Blockchain" to the user. On the top bar, there are three buttons that is named Security, Create Block Data and Data in Blockchain. When the Security button is clicked, the home page will be shown to the users. If the "Create Block Data" button is clicked, users will be in the input page where user can enter the criminal data. By clicking the "Data in Blockchain" button, users can see the data that have been entered by all the users in the blockchain network.

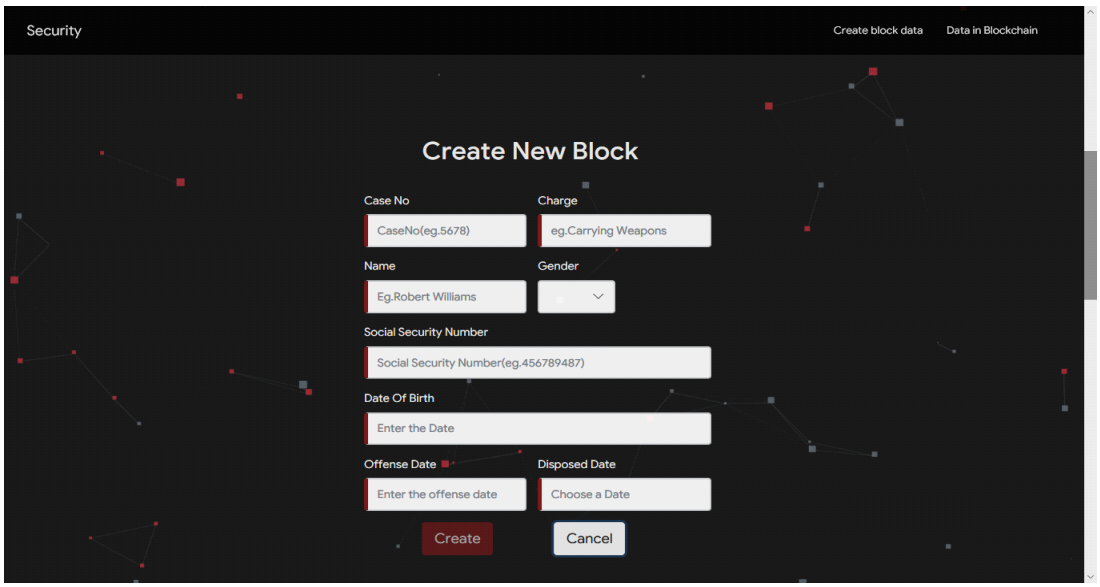

Figure 3.6 Input Page

This is the input page after the users have clicked the second button on the top bar. As shown in Figure 3.6, there are eight input fields that are the case number, the reason of charge, the name of the person, his/her gender, his/her social security number, date of birth, the date that the person offended a criminal and the date that the case is disposed. The input fields and the create button are initially shown red because no data is filled in the fields.

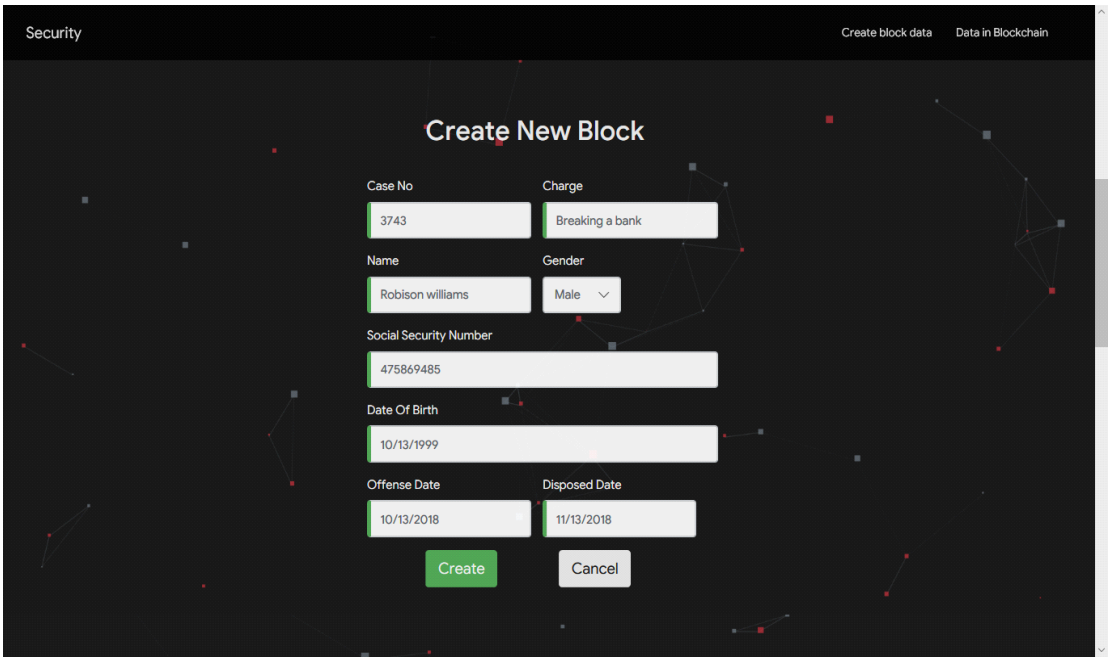

Figure 3.7 Input Page

After all the data are entered in the input fields, the fields and button are turned green indicating that the entered data are correct and it is ready to create a new block as shown in Figure 3.7.

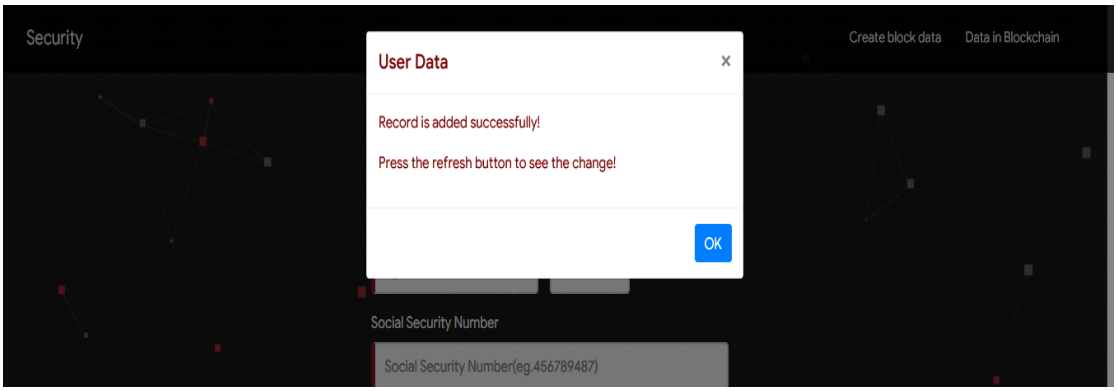

Figure 3.8 Successful Dialog

As shown in Figure 3.8, it is a dialog showing that the block is created. The created block is broadcasted to other users in the network and successfully added to the blockchain. After adding the block to the blockchain, users in the network have to click the refresh button or refresh the page to see the latest the blockchain in their devices.

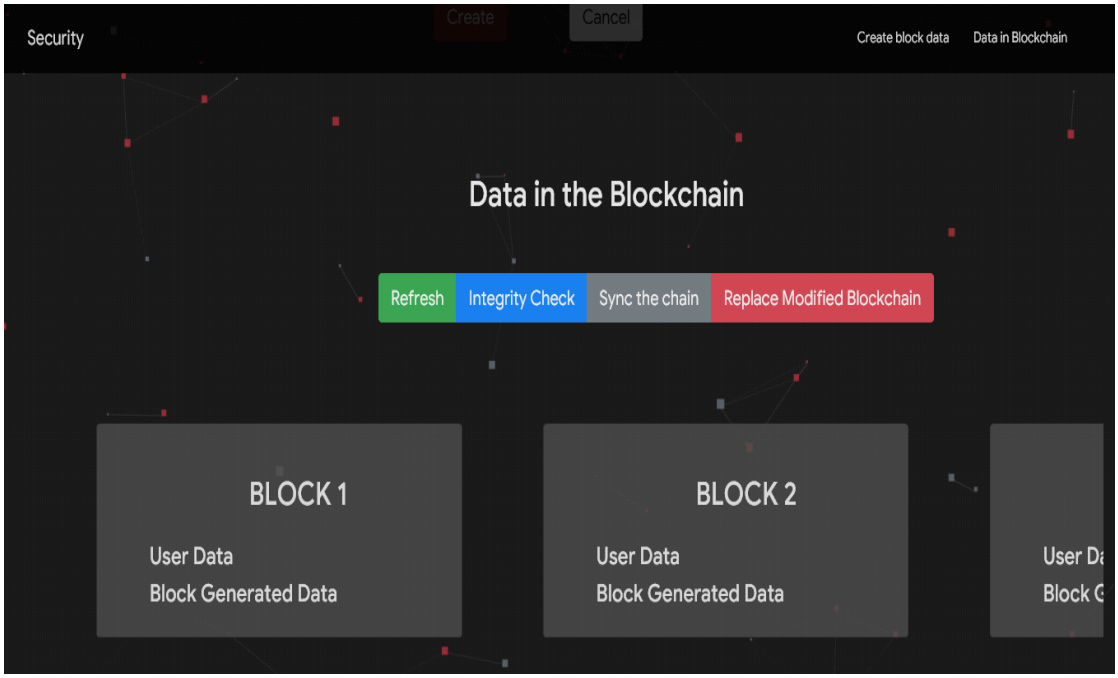

Figure 3.9 Blockchain Page

This is the page that shows all the blockchain data added by all the network devices. As shown in Figure 3.9, there is a text that shows “Data in the Blockchain” and four colorful buttons which are to refresh the blockchain data, to check the blockchain integrity, to synchronize the blockchain data if a new node is connected for the first time and to insert the modified data to the blockchain. Below the buttons, there are blocks that show the blockchain data visually. The block will increase itself dynamically and the number of blocks depends on the data added by the users to blockchain. Users can slide from left to right to see the successive block lists.

The blocks can either be expanded or collapsed. As shown in Figure 3.10, the block has two sections; first section is to display the user data and second is for block header. In this figure, there are two blocks; block 1 is the genesis block of the blockchain. As this is the genesis block, all the data are assumed as zero at the beginning. The genesis block is created when the users run their systems for the first time and was specifically coded into the system.

Figure 3.10 User Data

The first block created by the users starts at the block number two and is adjacent to the genesis block. The new blocks created by the users will be added dynamically from the right side. In block 2, the user data are actually the values stored in the blockchain.

In Figure 3.11, block header is expanded and there are five metadata inside it. They are the nonce value, the index of the block, the hash code of the current block, the hash code of the previous block and IP address of the node. As block 1 is the genesis of the blockchain, the metadata in the block are already assigned with values at the beginning.

In block 2, the values in the metadata are not entered by the users, but generated by the system. Nonce value is 25713 which is the count that system has tried for finding the satisfied hash code. The hash code is found and the value is0000d53ed9788a00b9f9321e26c06552b14846bd2e82cf41a9c854ea98f37aff.

Genesis block is important in the blockchain because the hash code of the previous block is a necessary part in finding the satisfied hash code of the next block. So, blockchain cannot be built without the genesis block.

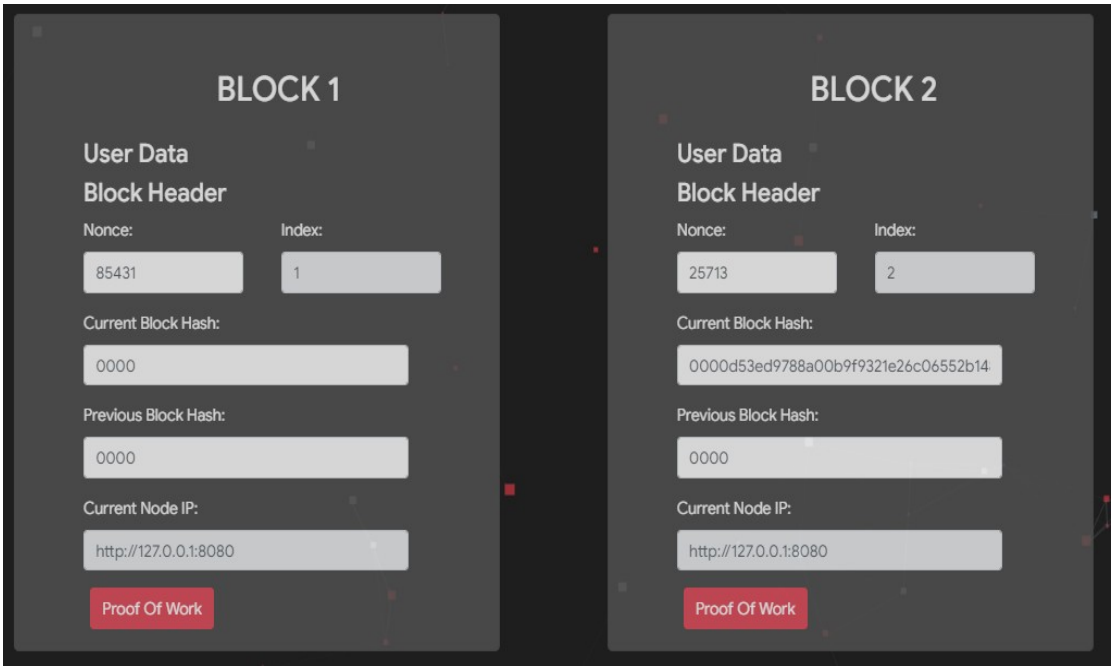

Figure 3.11 Block Header

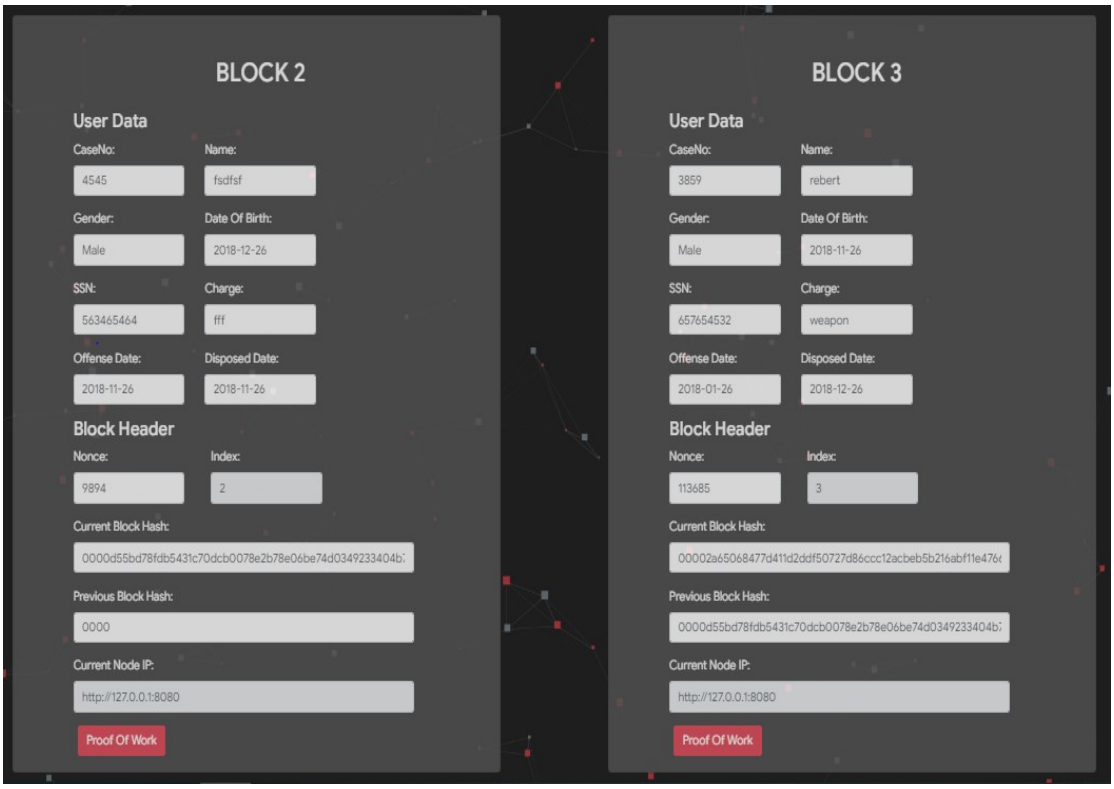

Figure 3.12 Data in Blockchain

In Figure 3.12, one more block, block three, is added to the blockchain. The data in the block are different from the previous one including the metadata such as

nonce value, index of the block, hash code of the current block and hash code of the previous block.

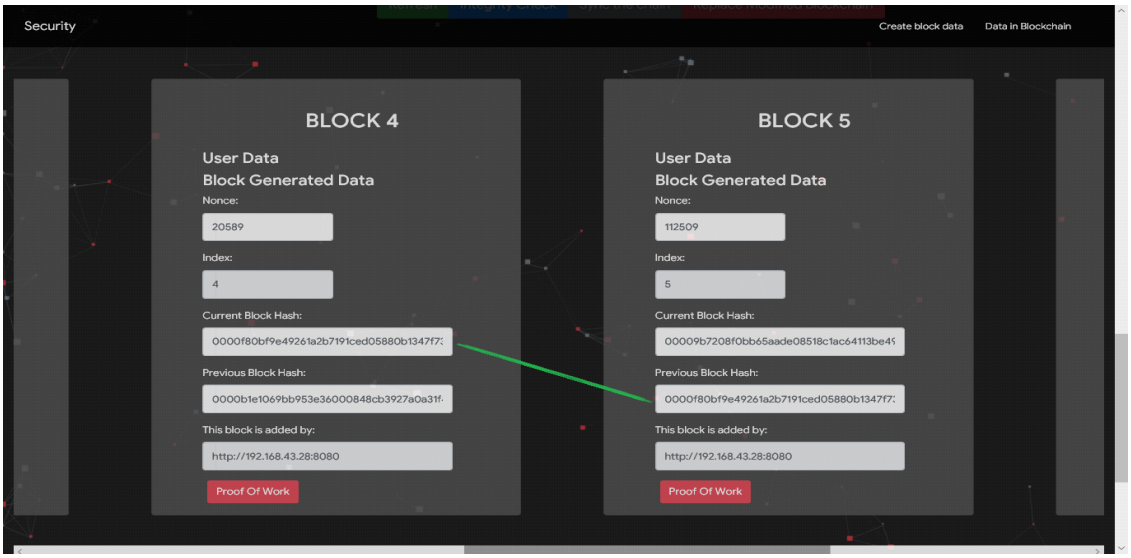

Figure 3.13 Hash linking

Blocks are linking each other with cryptographic hash in blockchain as shown in Figure 3.13. The current hash code of block 4 is the same as the previous block of block 5. By that way, the hash codes are linking from genesis block to block N.

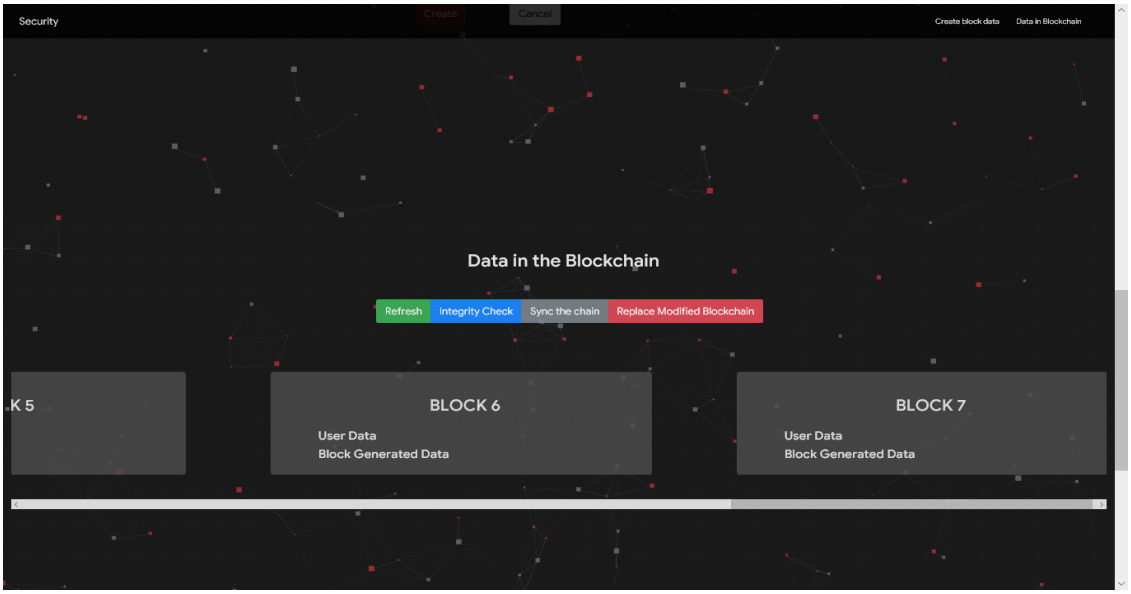

Figure 3.14 Before Refreshing the Page

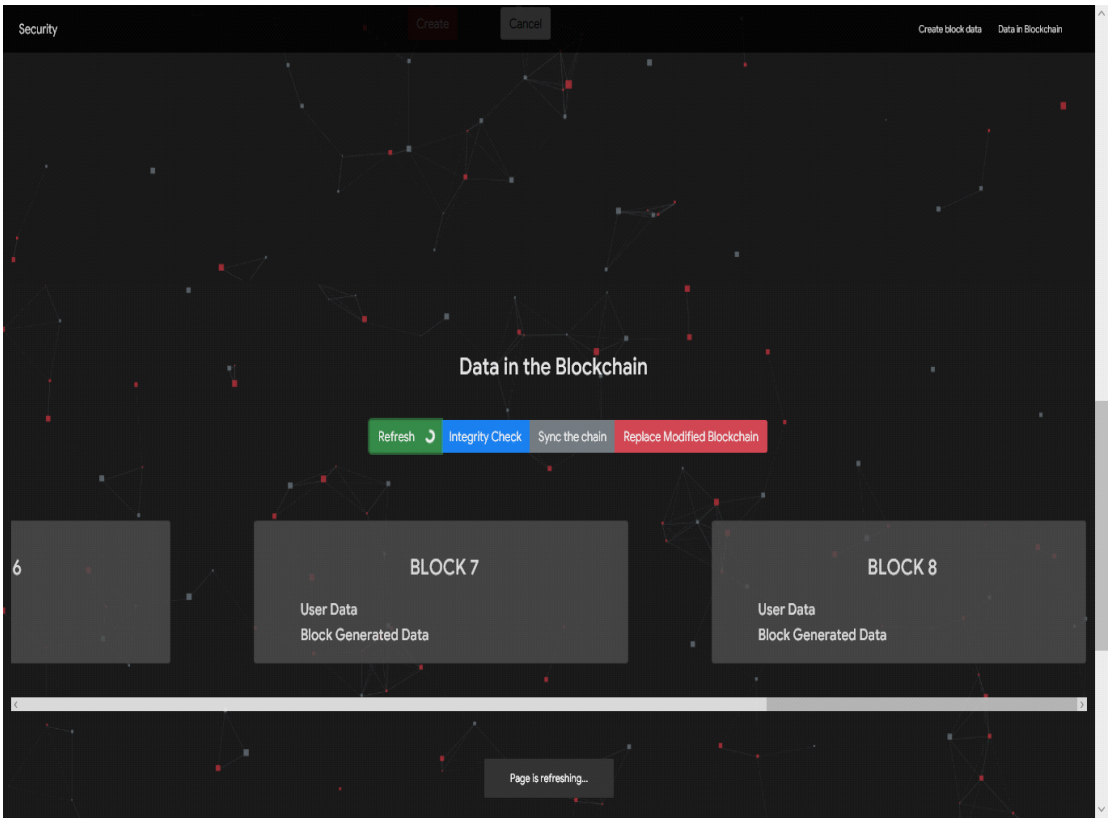

Figure 3.15 After Refreshing the Page

When a new block is added by the users in the blockchain network, the new block is not shown automatically. It needs to press the refresh button to load the latest blockchain data. Refresh button is shown in green color. The refreshing process can also be done by refreshing the webpage. But this makes graphic flicker and looking bad. In Figure 3.14, there are just seven blocks in the blockchain although a new block is added. After the page is refreshed, the total eight blocks are loaded in the blockchain as shown in Figure 3.15.

The blockchain data can be loaded by pressing the refresh button. After all the data is loaded into the blocks, the blocks are showing in grey color or after refreshing the blocks, it will show the same color. Integrity check button is to notify the users about data in the blocks are correct. This is done by comparing the current hash code of the previous block and the previous hash code of the current block. The blocks will show in green color if the hash codes are the same. The comparing process take place from genesis block to block N. The blocks with green color means that the data in the blocks have integrity.

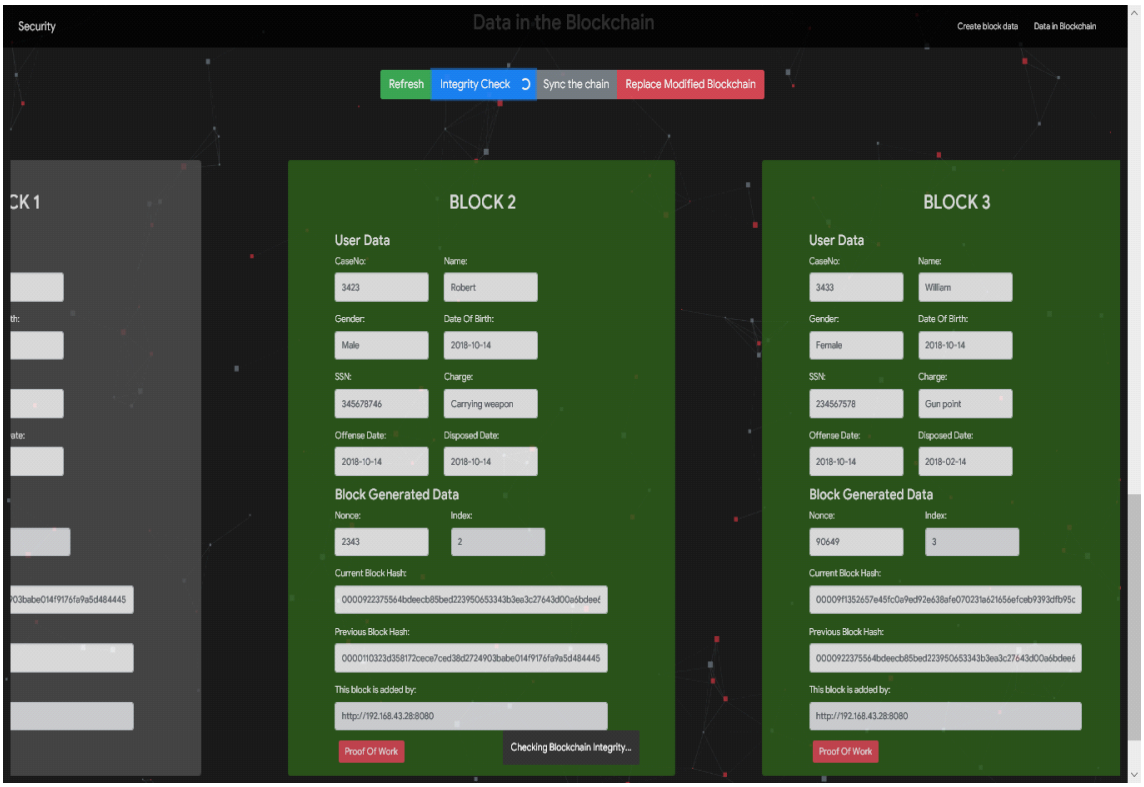

Figure 3.16 Checking Integrity on Block Data

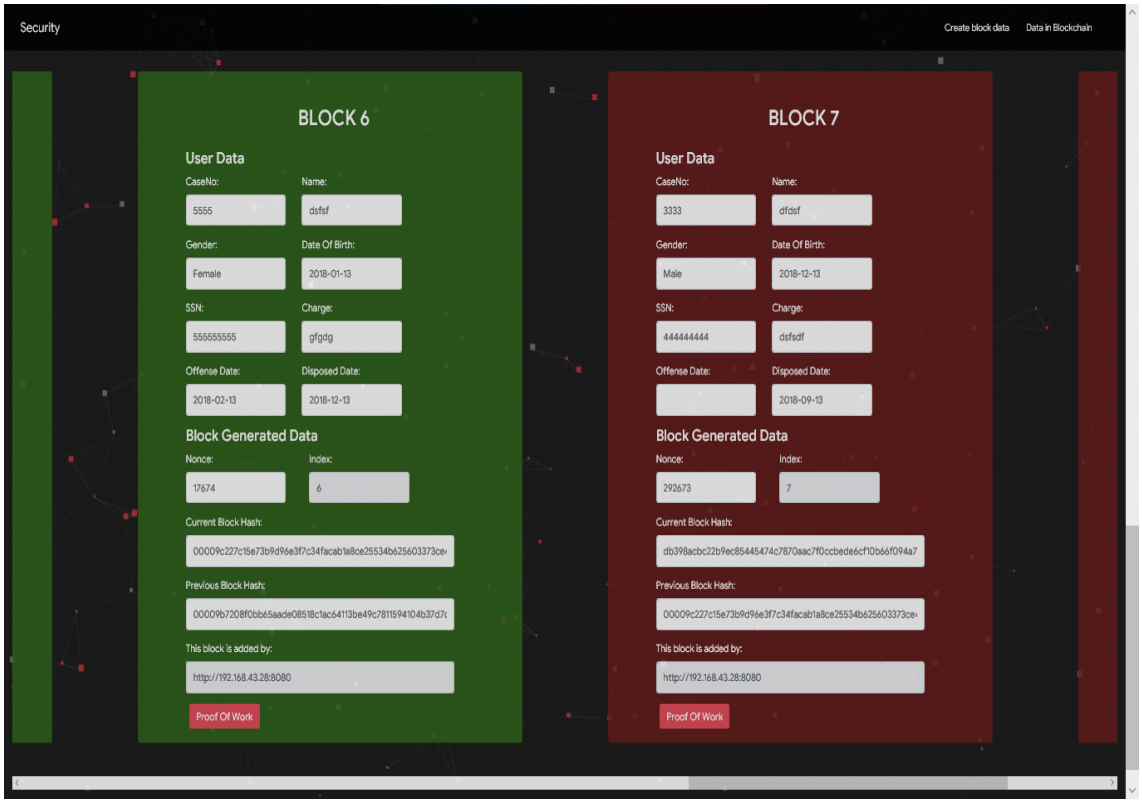

Figure 3.17 Checking Integrity on Block Data

The data in the blocks are loaded form the blockchain and placed into the text field. As the blocks are changed dynamically, the data are editable. If users change any data in a block, the hash codes in that block will also be changed. The modified data will lead to an unsatisfied hash code which means the hash codes do not start with four zeros. If the hash code is invalid, the block will turn into red color as shown in Figure 3.17. The block with red color means that the data in that block are modified.

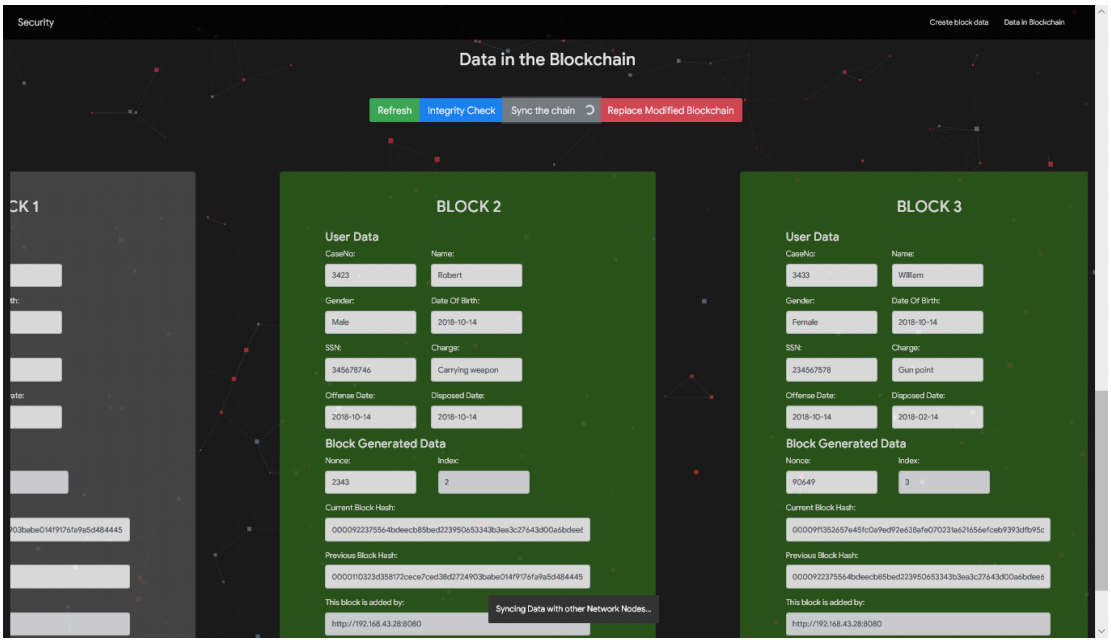

Figure 3.18 Synchronizing Blockchain Data

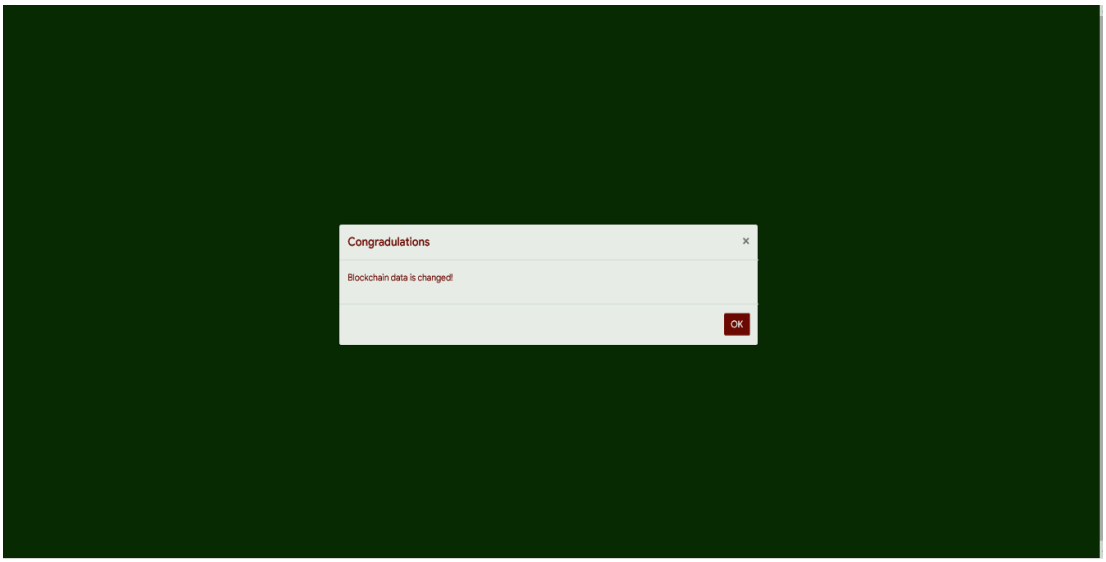

Figure 3.19 Changed Blockchain Data

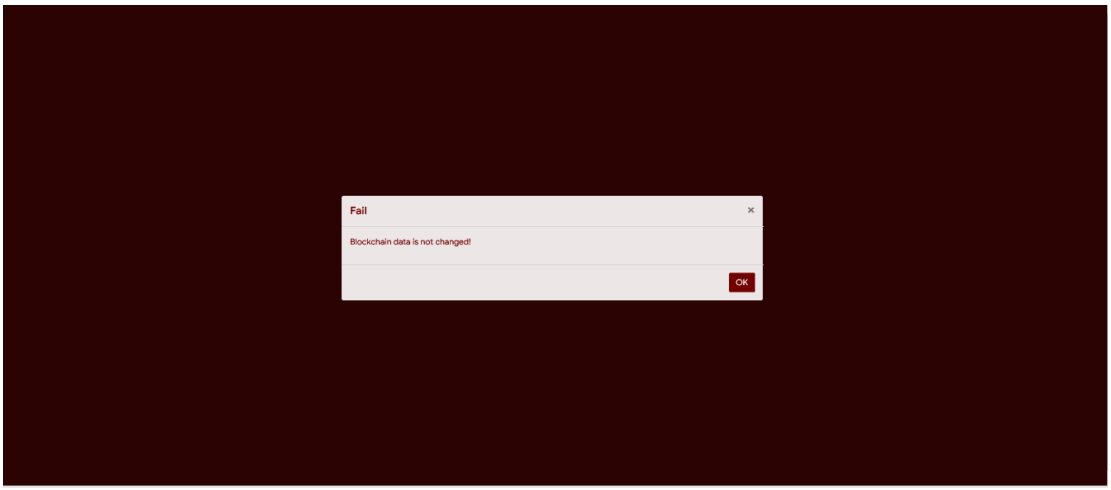

Figure 3.20 Unchanged Blockchain Data

When a new node is connected to the network, the first thing that the node has to do is to synchronize the blockchain. After synchronizing the blockchain, the node can start to participate in the blockchain network. Synchronizing is about fetching the data from the other users. As shown in Figure 3.18, the blocks are show up in the nodes after it synchronizes with other nodes. If the synchronized process is successful, a dialog will show with a message “Blockchain data is changed” as shown in Figure 3.19. If the synchronized process is failed, a message “Blockchain data is not changed” will show up.

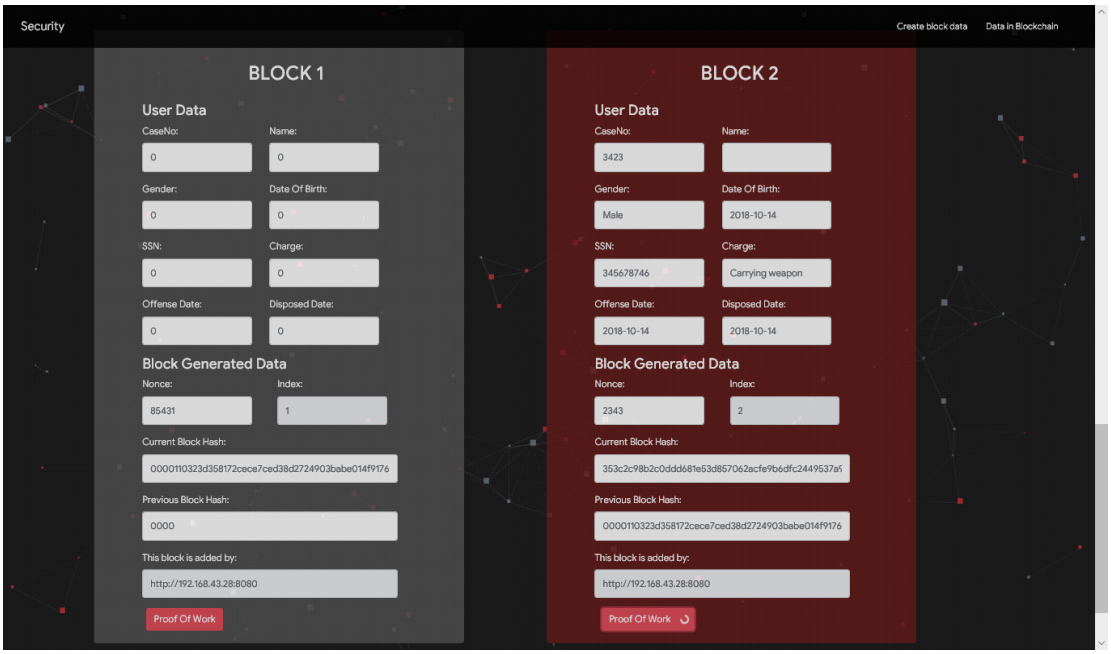

Figure 3.21 Generating a Valid Hash using Proof of Works

Data in the block 2 has been modified in Figure 3.21 because name value in the input field is missing, the current hash code of the block is not starting with four zeros and the block is showing that it does not have data integrity. At the bottom of the block, there is a button named “Proof Of Work”. This button is to apply the proof-of-work algorithm to the modified data in the block. The process will consume computational power to find a new satisfied hash code for the block.

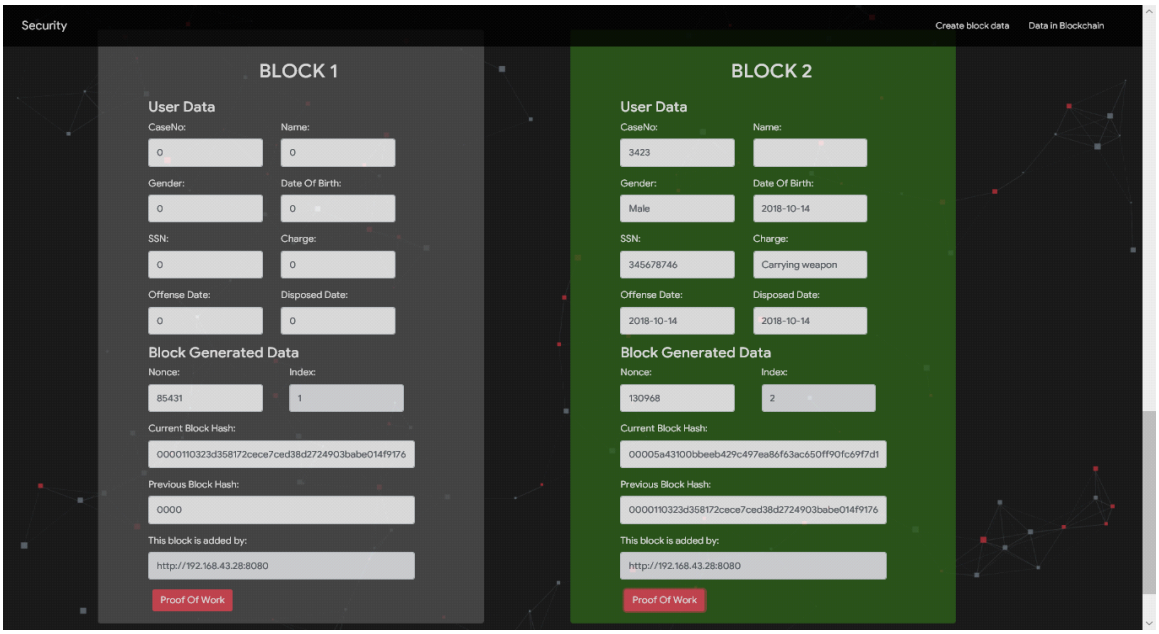

Figure 3.22 New Nonce for Modified Data

After the “Proof of Work” button is clicked, the system will start finding a valid hash that is accepted by the system. As in the Figure 3.22, the value in text field is still missing, but the block is showing green color and the hash code is starting with four zeros. This is because Block 2 has successfully found a new satisfied hash code with the new nonce. The nonce value is 130968.

In this system, users can attempt to tamper the data in the blockchain. As shown in Figure 3.23, data in the block 3 is modified by user and it lead to the modification to the rest of the block. “Replace Modified Blockchain” button will replace the current modified blockchain with the original one. After doing it, original blockchain cannot be recovered without restarting the node again. It will lead the node to be an invalid one in the network. The node cannot synchronize the latest blockchain with other users.

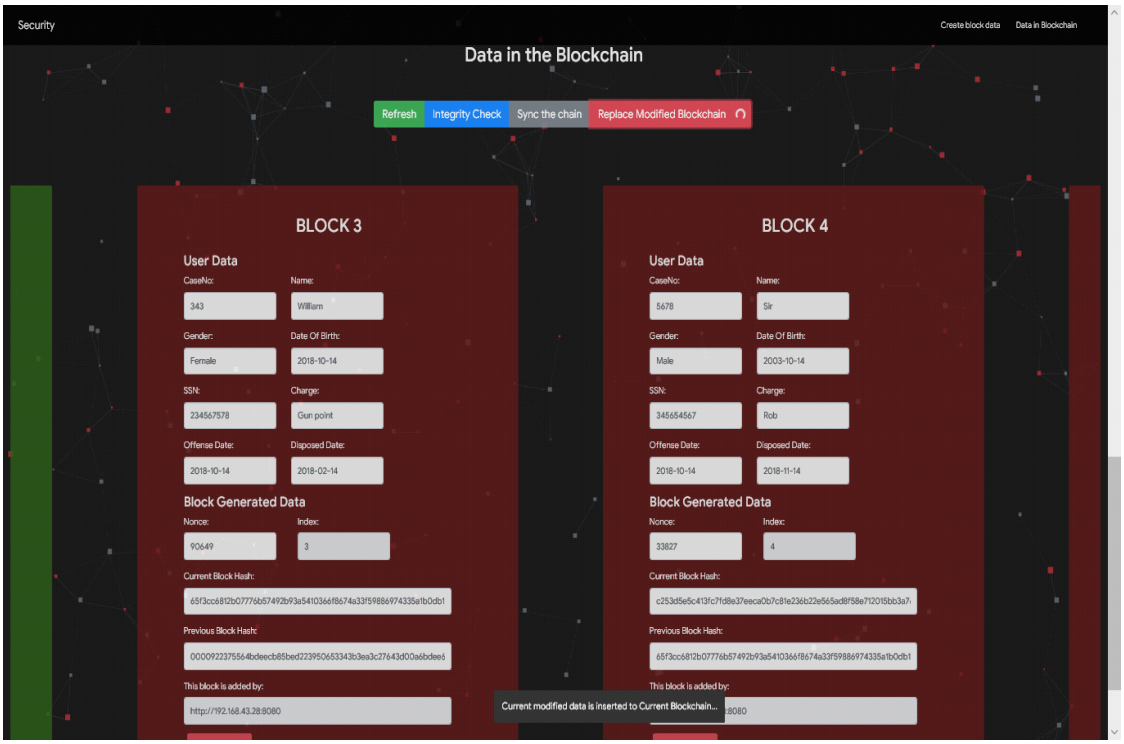

Figure 3.23Modifying Current Blockchain


```
{
    "Index": 2,
    "Nonce": 121026,
    "Timestamp": "Mon Nov 26 2018 21:18:34 GMT+0630 (Myanmar Standard Time)",
    "Name": "Robert",
    "Gender": "Male",
    "DateOfBirth": "2018-11-07",
    "SSN": "765468097",
    "Charge": "Weapon",
    "CaseNo": "8394",
    "OffenseDate": "2018-11-15",
    "DisposedDate": "2018-11-07",
    "CurrentBlockHash": "00005371f4de5dbad3e02aa1eb585f00d23cecc3df94476ca4ee7324d2199bda",
    "PreviousBlockHash": "0000"
},
{
    "Index": 3,
    "Nonce": 35065,
    "Timestamp": "Mon Nov 26 2018 21:19:04 GMT+0630 (Myanmar Standard Time)",
    "Name": "FFR",
    "Gender": "Male",
    "DateOfBirth": "2018-11-08",
    "SSN": "657894566",
    "Charge": "Weapon",
    "CaseNo": "6578",
    "OffenseDate": "2018-11-22",
    "DisposedDate": "2018-11-30",
    "CurrentBlockHash": "0000867aaf34d648d63124dc5226da7bb9a1324dbecd2f31e1d63d6258428736",
    "PreviousBlockHash": "00005371f4de5dbad3e02aa1eb585f00d23cecc3df94476ca4ee7324d2199bda"
}
```


Figure 3.24 Records in Public Ledger

Since the blockchain is a distributed database or public ledger that the data can be stored in a raw file or simple digital form. The public ledger of the blockchain in this system is shown in Figure 3.24. The ledger is in JSON format. All the data added by the users will be stored in this ledger. Every user has the same public ledger in their device.

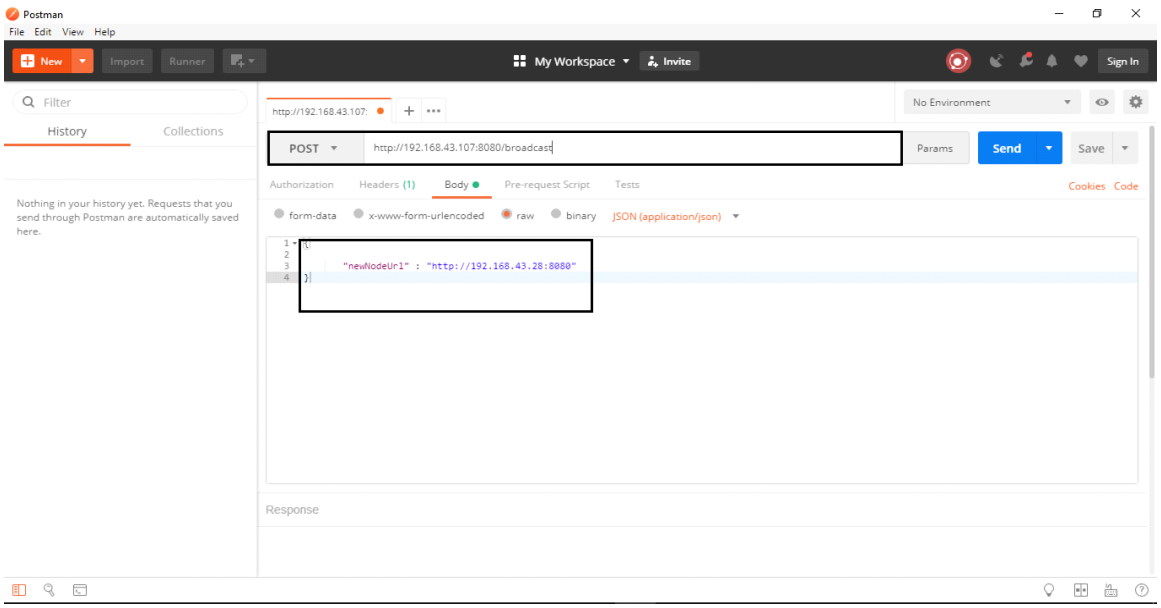

Figure 3.25 Using Postman to Add New Node

In this system, the nodes have to be added by a third-party application, Postman. Postman is a development and environment tool which can interact with the system. In Figure 3.25, it shows that a new node is being added by using "Postman". In order to join a node, user needs to enter the URL ("http://destination-node's ip:8080/broadcast") in POST method. The destination node has to be one of the nodes in the blockchain network. In the body tab, there is a JSON message, {"newNodeUrl": "http://user-node's ip:8080"}. The message type, raw and JSON formation must be selected. If everything is ready, a new node can be added by pressing the send button.

If a new node is added successfully as shown in Figure 3.26, a message in the body tab will say "New node registered with network successfully". If the user accepts the message, it means that the node is connected to the blockchain network and the user can start synchronizing the blockchain with other users in the same network.

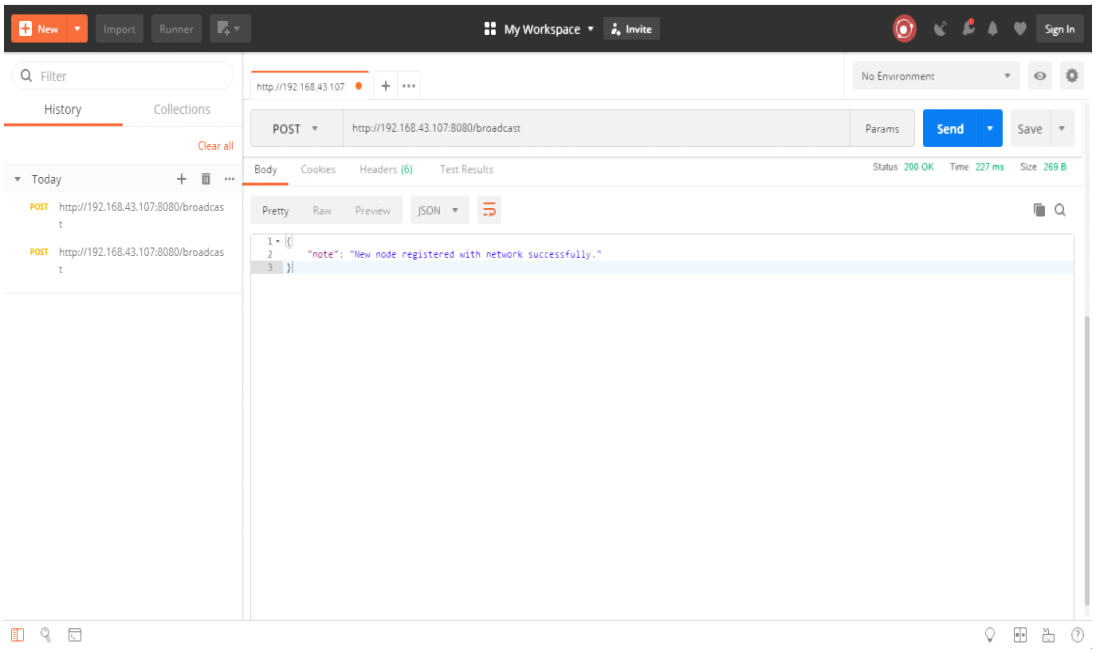

Figure 3.26 A new node is added by Postman


```
            "CaseNo": "5678",
            "OffenseDate": "2018-08-14",
            "DisposedDate": "2018-02-14",
            "CurrentBlockHash": "00004af7fb18b164967a9852cd2ee85d9a17e8ee8719afcdc8fb8dd5b2e2f03f",
            "PreviousBlockHash": "0000244affaca40d62898f7891440de2e1136d9d1e7c8559441c9dde8951e2ff"
        },
        {
            "Index": 4,
            "Nonce": 118718,
            "Timestamp": "Sun Oct 14 2018 13:59:00 GMT+0630 (Myanmar Standard Time)",
            "Name": "William",
            "Gender": "Male",
            "DateOfBirth": "2018-10-14",
            "SSN": "343234567",
            "Charge": "Carrying weapon",
            "CaseNo": "3432",
            "OffenseDate": "2018-10-14",
            "DisposedDate": "2018-10-14",
            "CurrentBlockHash": "0000ecf046ff812aab851e1ddb2424af2d226bcff7e1a7e641849a21cb0812a7",
            "PreviousBlockHash": "00004af7fb18b164967a9852cd2ee85d9a17e8ee8719afcdc8fb8dd5b2e2f03f"
        }
    ],
    "CurrentNetworkNode": "http://192.168.43.28:8080",
    "AllNetworkNodes": [
        "http://192.168.43.107:8080"
    ]
}
```


Figure 3.27 Public Ledger with connected nodes

The network address of user's node and that of all the connected nodes in the blockchain network are recorded in the public ledger. As shown in Figure 3.27, there are two nodes connected each other and form a blockchain network. If more nodes are connected to the blockchain network, the public ledger will update itself and add the new node address to it.

CHAPTER 4
CONCLUSION

Securing the information becomes very crucial in the world. There are many technologies that are being innovated, created and developed in order to protect the data from hacking. One of the trended technologies is the blockchain. This technology is widely used in many companies because it secures the information effectively and efficiently.

This system is implemented as a way to demonstrate how a blockchain works. It is useful for the beginners who are curious about the blockchain technology. They can understand it clearly by using the system. It contains all the minimal functionalities of a blockchain and coded in easy way. The system provides an easy-to-use user interface that allows the users to perform data validation, data storing and also data tampering. The user interface contains the visual effects that help users understand more quickly. In this system, users can be connected each other and form a blockchain network. By using the system, user can have the knowledge of understanding proof-of-work algorithm and more and more.

In short term, this system can record the data and stored in the blockchain. Data tampering is prevented by data validation using SHA algorithm. The nodes can be connected to form a blockchain network. The node with modified blockchain will not be considered as a valid node. The data added by that node will not be accepted by other nodes. The only nodes with the valid blockchain can function in the blockchain network. Thus, data integrity is ensured.

4.1 Limitations of the System

In this system, data tampering can only occur when the users replace the modified blockchain with the original one and there is no middle man who attempts to change the data in the blockchain. A new node can only be added using third-party soft wares. In the input page, error checking between dates is not provided in this system. Webpage has

problem when it is displaying on mobile browser. It is because the technical problems issuing with the decoration. Some messaging and dialogs to notify the users about the events are lacking. Overall system is not suit for applying in the real world.

4.2 Further Extension of the System

The further improvement is required for the consensus algorithm in the system. The better input validations are required to be improved in the system.

REFERENCES

[1] FIPS PUB, Descriptions of SHA-256, SHA-384, and SHA-512, 2018

[2] FIPS PUB 18-4, Federal Information Processing Standards Publication, Secure Hash Standard (SHS), August 2015

[3] Igor Zikratov, Alexander Kuzmin, VladislavAkimenko, Viktor Niculichev, Lucas Yalansky ITMO University Saint-Petersburg, Russia

[4] Ittay Eyal, Distributed currency and consensus algorithm, July 2017

[5] Invenstopedia, www.investopedia.com/terms/d/distributed-ledgers.asp, 2018

[6] Mr. Vijay Divecha, Data Integrity in view of the Manufacturer,2018

[7] Ronald Chan, www.linkedin.com/pulse/blockchain-data-structure-ronald-chan, February 2, 2018

[8] Satoshi Nakamoto, Bitcoin (a peer-to-peer cash system), 2008

[9] Ambisafe, https://ambisafe.com/blog/public-vs-private-blockchain/, 2018

[10] Ericsson,www.ericsson.com/en/security/data-centric-security/blockchain-data-integrity, 2018