DEVELOPMENT OF A BLOCKCHAIN-BASED ANTI-COUNTERFEITING SYSTEM WITH ENHANCED CONSENSUS ALGORITHM

By

WOSU, JEREMIAH TASHIE

B.Tech. (RSU), M.Eng. (FUTO)

Reg. No: 20194995548

A PhD DISSERTATION

SUBMITTED TO THE POSTGRADUATE SCHOOL

FEDERAL UNIVERSITY OF TECHNOLOGY, OWERRI

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE AWARD OF DOCTOR OF PHILOSOPHY (PhD) DEGREE IN COMPUTER ENGINEERING

IN THE DEPARTMENT OF ELECTRICAL AND ELECTRONIC ENGINEERING,

SCHOOL OF ELECTRICAL SYSTEM ENGINEERING AND TECHNOLOGY

MARCH 2024

CERTIFICATION

I certify that this research work "Development of a Blockchain-Based Anti-Counterfeiting System with Enhanced Consensus Algorithm" was carried out by Wosu, Jeremiah Tashie (20194995548) in partial fulfilment for the award of Doctor of Philosophy (PhD) in Electronic and Computer Engineering, in the Department of Electrical and Electronic Engineering of the Federal University of Technology Owerri.

Engr. Prof. G. A. Chukwudebe — 28/05/2024
(Supervisor) — Date

Engr. Dr. C. K. Agubor — 28/5/24
(Co-Supervisor) — Date

Engr. Dr. L. S. Ezema — 28/05/2024
(Co-Supervisor) — Date

Engr. Dr. N. Chukwuchekwa — 26/6/24
(Head, Department of Electrical and Electronic Engineering) — Date

Engr. Prof. M. C. Ndinechi — 30/7/24
(Dean, School of Electrical Systems Engineering and Technology) — Date

Prof. B. O. Esonu — Date
(Dean, Post Graduate School)

(External Examiner) — 26/6/24
Engr. Prof. Cosmas Ani — Date

DEDICATION

This work is dedicated to God Almighty, the giver of every perfect gift, for giving me wisdom and protection.

ACKNOWLEDGMENTS

I would like to express my profound gratitude to my supervisors, Engr. Prof. G. A. Chukwudebe, Engr. Dr. C. K. Agubor and Engr. Dr. L. S. Ezema for their encouragement, guidance and support.

I appreciate the invaluable contributions of Engr. Dr. N. Chukwuchekwa (Acting Head of Electrical and Electronic Engineering Department, FUTO), Engr. Prof. E. N. C. Okafor, Engr. Prof. F. K. Opara, Engr. Prof. D. O. Dike, Engr. Prof. (Mrs) G. N. Ezeh, Engr. Prof. (Mrs.) I. E. Achumba, Engr. Dr. C. C. Mbaocha, Engr. Prof. J. O. Onojo, Engr. Prof. L. O. Uzoechi, Engr. Dr. I. O. Akwukoegbu, Engr. Dr. M. Olubiwe, Engr. Dr. O. C. Nosiri and Engr. Dr. S. O. Okozi.

I also wish to thank Engr. Prof. M. C. Ndinechi (Dean, School of Electrical Systems Engineering and Technology, FUTO), Prof. B. O. Esonu (Dean, Post Graduate School, FUTO) and Prof. C. O. Nweke (Associate Dean, Post Graduate School, FUTO).

Sincerely, I appreciate the advice and support I got from my colleagues Engr. F.O.C Nwaduwa, Engr. Dr. Steve, Engr. Dr. Wali, Engr. Trust, Engr. Chidinma, Engr. Ajumo, Engr. Njumoke and Engr. Nwiido. I would also like to acknowledge my friends Obinna, Nnanyereugo, Felix, Kachi and Emmanuel for their support and contributions towards the success of this work.

My beloved wife, Cecilia and children, Jahsgift, Jahsrule, Jahstime, Jahsword and Jahswork, deserve special appreciation for their continuous support and encouragement throughout the period of this work.

Furthermore, I would like to thank my Mum, Mrs. Mercy Wosu for her love and enormous support, and also my siblings, Blessing, Abraham and James, who have encouraged me throughout my study.

My greatest appreciation goes to Jehovah God, the giver of every good gift for His love, provisions and protection.

Table of Contents

LIST OF TABLES

LIST OF FIGURES

LIST OF PLATES

LIST OF ABBREVIATIONS

3D - Three Dimensions

AI - Artificial Intelligence

AMPQ – Advanced Message Queuing Protocol

ANFIS - Adaptive Neuro-Fuzzy Inference System

ANN – Artificial Neural Networks

APIs – Application Program Interface

BBACS - Blockchain-Based Anti-Counterfeiting System

BRISK - Binary Robust Invariant Scalable Keypoints

CSGC - Copy Sensitive Graphical Codes

CSS – Cascading Style Sheets

DAOs - Decentralized Autonomous Organizations

dApp - Distributed Application

DPoS - Delegated Proof-Of-Stake

EPCs - Electronic Product Codes

EUIPO - European Union Intellectual Property Office

GA-ORB - Geometric Algebra-based Oriented-fast and Rotated Brief

GPS - Global Positioning System

HTML – Hypertext Mark-up Language

HTTP – HyperText Transfer Protocol

IDE – Integrated Development Environment

IoT – Internet of Things

JSON – JavaScript Object Notation

OECD - Organization for Economic Cooperation and Development

P2P - Peer-to-Peer

PBFT - Practical Byzantine Fault Tolerance

PoA - Proof of Authority

PoB - Proof of Burn

PoS - Proof of Stake

PoW - Proof-of Work

QR - Quick Response

RANSAC - Random Sample Consensus

RFID - Radio Frequency Identification

SDPs - Software Design Patterns

SIFT - Scale Invariant Feature Transform

SLCML- Smart-Legal-Contract Markup Language

SQL - Structured Query Language

SURF - Speeded-Up Robust Features

TCP/IP - Transmission Control Protocol/Internet Protocol

UI – User Interface

UML - Unified Model Language

URL – Uniform Resource Locator

USB – Universal Serial Bus

USD – United State Dollar

WHO – World Health Organization

WiFi - Wireless Fidelity

ABSTRACT

This thesis presents the development of a blockchain-based anti-counterfeiting system with enhanced consensus algorithm. Market surveys reveal that counterfeit trading activities are increasing rapidly, and the rise of counterfeit products has adverse effect on economic growth as well as public health and safety. Existing anti-counterfeiting solutions do not employ blockchain technology with enhanced consensus algorithm in combination with product inherent features, copy-sensitive Quick Response (QR) codes and location information technologies. Thus, fully functional and affordable product anti-counterfeiting solutions with traceability, immutability and transparency functionalities are widely and urgently demanded. In this research, object-oriented software analysis and design technique in combination with Rapid Application Development (RAD) methodology were adopted for the modelling of a prototype system used in demonstrating the work. Ganache, a private Ethereum blockchain network was setup to serve as the backend platform. Open-source software such as Truffe suite and Solidity compiler were utilized in setting up the Ganache network as well as in compiling and deploying smart contracts written in Solidity language. This work designed and applied an enhanced consensus algorithm named Proof of Product Contribution (PoPC) that is fully decentralized, and balances between efficiency and security. It also developed special QR code generator and scanner using Kotlin. The developed system is unique as it combines blockchain technology, product texture (which is an inherent feature), copy-sensitive QR Code, location information i.e. GPS coordinates as well as Track and Trace technologies in proffering reliable solution to counterfeit trading. Test results prove that PoPC is very fast with average execution time of 5 seconds as against 56 seconds for Proof of Stake (PoS) algorithm, 62 seconds for Practical Byzantine Fault Tolerance (PBFT) algorithm, 75 seconds for Delegated Proof of Stake (DPoS) algorithm, and 600 seconds for Proof of Work (PoW) algorithm.

Keywords: Blockchain, consensus algorithm, counterfeiting, distributed applications, Ganache, Ethereum, security, product texture, copy-sensitive QR Code.

CHAPTER ONE
INTRODUCTION

1.1 Background Information

Counterfeiting remains a significant challenge in both local and global commercial systems, and can be noxious when it has to do with food and pharmaceutical products. Trading activities in counterfeit goods do not only have negative impact on sales and profits of industries, but also generate profits for organized crime at the expense of the affected companies and governments (Yiu, 2021a). Hence, counterfeit trading activities have adverse effects on the economy and public health, as well as the safety and security of the wider community. For instance, in case of food and medical products, the implications are more severe than monetary: counterfeit products do not contain the right active ingredients, and therefore can be nocuous. This is more predominant in developing countries as people die after being treated with fake medicine (Sayyad, Adil, Vaishnavi, Mukesh & Devisha, 2021).

Worse yet, market surveys carried out in recent times reveal that counterfeit trading activities are increasing at exponential rate. For example, the counterfeit medicine industry is estimated to be growing at twice the rate of legitimate pharmaceuticals, thereby accounting for 2.5% of total pharmaceutical market. In fact, International Chamber of Commerce predicts that the negative impacts of counterfeiting and piracy will drain $4.2 trillion (USD) from the global economy by 2025, and this will put 5.4 million legitimate jobs at risk (Keith & Jith, 2020; Pacific Research Institute, 2020).

Counterfeit product trading activities are operated swiftly in the global economy. This results to the misuse of free trade zones, thereby taking undue advantage of many legitimate trade facilitation mechanisms. In most cases, these illicit activities thrive in economies with weak governance standard and limited innovative technologies to combat product counterfeits. Furthermore, counterfeiters are becoming more and more professional and sophisticated. They are always developing approaches to better package

counterfeit products and bring them to the market undetected (Tavares, 2020). Thus, detection of counterfeit products in today's market is a great challenge for consumers.

Many individuals, firms and organizations are working tirelessly to stop this process which is harmful to everyone in the entire world. Various branded or reputed companies are leveraging modern technologies to develop anti-counterfeit solutions that can identify counterfeited products. Such anti-counterfeiting solutions will not only protect organizations and individuals from financial losses, but will also provide customer safety, especially in the case of food and pharmaceutical products. According to Sayyad et al (2021), good anti-counterfeiting techniques should generally be simple to apply, but difficult to duplicate and reuse. It should be possible to utilize them without special equipment, and it should be obvious if they were tampered with.

Presently there are three common categories of anti-counterfeiting solutions: overt, covert, and track and trace. Overt technologies contain features that are expected to assist the consumers to confirm the genuineness of a product pack. Such features will be significantly visible, yet complex and expensive to reproduce (Keith & Jith, 2020). Color-shifting inks, security threads, watermarks and holograms technologies belong to this category. End consumers need to be briefed in advance so that they interpret these technologies correctly to verify fake products.

Covert technologies are applied to the product itself but are not identifiable without special equipment. The rationale of a covert feature is to aid the brand owner to recognize a counterfeit product. The general public will not be aware of its presence nor will they have the resources to confirm it. These resources include ultraviolet electromagnetic radiation, bi-fluorescent and pen-reactive ink, as well as digital watermarks and hidden printed messages.

The final category is track and trace. Radio Frequency Identification (RFID) tags, Product Inherent Features, Electronic Product Codes (EPCs), Barcodes and Quick

Response (QR) codes are the main technologies in this category. Track and trace technologies allow for simpler tracing of products, by making the history of a product available to consumers. The tag or QR Code is included by the manufacturer. Then, distributors will scan the identification, enabling them to check the authenticity of the product and update the status. Finally, retailers can also scan the product, to check the history and authenticity of the product. This approach does not only tackle the counterfeit problem, but also enables track and trace throughout the product lifecycle (Daoud, 2019).

All the three categories of anti-counterfeiting technologies mentioned above have disadvantages and limitations. Overt technologies can be imitated. Hence, it is difficult for the average end-consumer to distinguish between a convincing imitation and the real product. Covert technologies require special devices to identify counterfeited products so that customers are neither able to detect counterfeits nor verify the genuineness of products. The track and trace technologies with its encoding and security features can be used in combination to improve this situation but it leads to another major issue – overhead cost. Consequently, the price of a product continuously increases, which in certain cases eventually encourages the end user to seek out counterfeit products (Daoud, 2019). Moreover, track and trace technologies rely on a centralized authority that is neither immutable nor fully transparent.

In response to the growing concern in product counterfeiting as well as the limitations of existing systems; innovative, fully functional and affordable product anti-counterfeiting solutions with traceability, immutability and transparency functionalities are widely and urgently demanded. These solutions are expected to utilize cutting-edge technologies so as to ensure the provenance, as well as the identification and traceability of genuine products. Unfortunately, current anti-counterfeiting technologies do not meet these requirements as every one of them relies on a centralized authority to combat counterfeit products. Such centralized architecture results in issues such as single point processing,

storage, and failure. Interestingly, blockchain technology has emerged to provide a promising solution for such issues.

A blockchain is a software that allows a network of computers to connect directly to one another without a trusted third party as a transaction intermediary (Ma, Lin, Chen, Sun, Chen, & Wang, 2020). It establishes a distributed or decentralized network of computers through which values can be exchanged instantly, stored securely, and at a lower cost. The data is copied to multiple nodes and each of these nodes runs the copy of the blockchain. Due to this, and the fact that the data is stored immutably in chains, blockchain eliminates the chances of the digital records being lost. It also reduces the chances of having the documents tampered, and a situation where they become unavailable in case one user's node or computer is inaccessible. Blockchain system is further based on distributed consensus algorithm, smart contracts and encryption algorithms (Sayyad et al, 2021).

Following the descriptions of Ma et al (2020), the characteristics of Blockchain technology are briefly discussed below.

(a) Decentralization: Through decentralized operations and storage, each node of the blockchain implements the verification, delivery, and management of information at the local side. Blockchain technology does not rely on an additional third-party control; it has no centralized control.

(b) Immutability: After a block has been approved and added to the blockchain, it cannot be tampered with. Accordingly, once a block in the blockchain is altered, it will be detected immediately and rejected by other nodes.

(c) Transparency: The data in blockchain is completely public and anyone can access it. Within the information flow, one can clearly see who is passing data to who as blockchain maintains a continuous transaction log file.

(d) Flexibility: The technology of blockchain is open source and anyone can modify it into his own version. There are already numerous flexible blockchain platforms

available, and users can also redevelop a new blockchain platform if they desire to do so. Blockchain is an unlimited technology meaning that users can create multiple applications based on blockchain.

(e) Privacy: In blockchain, each user is anonymous, and each user can have multiple addresses. When the system is operating, only one address is used as the identity, and the anonymous address can hardly be mapped to the real person, thereby protecting the user's privacy.

(f) Security: Public key encryption is used in blockchain to protect data security. Users can generate their own key pairs, including a private key and a public key. A private key is used to sign data, and a public key is used to verify the authenticity of the signed data. As long as the user prevents the private key from leaking, the data will remain secured.

Basically, there are three common types of blockchain networks, namely: public, private, and hybrid or consortium blockchain. A public blockchain, also known as permissionless networks, does not have any central authority controlling or directing its operations. All users participate in governance. Hence it is censorship-resistant since anyone can participate in the network regardless of location and nationality. Transactions on a public blockchain are public and visible to anyone through explorers. **(**Tushar, Saha, Yuen, Smith & Poor, 2020**)**

A private blockchain network, also known as permissioned network, on the other hand, is run by a private organization. The organization acts as a means of centralization because it limits the participants based on given criteria, and defines who connects to and transacts on the network.

A consortium based blockchain, also known as hybrid blockchain, is a blockchain available to an industry, group of organizations or individuals. It is a shared ledger, but unlike a public blockchain, access can be limited to members of the consortium. Though

write access could be limited, but all users are granted access to read. It is also possible to use hybrid routes, so that some information is publicly available, while others are not. **(**Sayyad et al, 2021**)**.

Blockchain architecture is divided into three layers which are Applications, Decentralized Ledger and Peer-to-Peer Network **(**Tushar et al, 2020**)**. Application layer provides a human readable interface where users can keep track of their transactions. Decentralized Ledger is the middle layer in a blockchain architecture that confirms a consistent and temper-proof global ledger. In this layer, transactions can be grouped into blocks which are cryptographically linked to one another. The bottom layer in the blockchain architecture is the Peer-to-Peer Network where node types play different roles and various messages are exchanged to the main Decentralized Ledger.

Blockchain employs a consensus algorithm to determine how the system is governed, how users formulate and agree on the rules, and how transactions occur. Some blockchain technologies use hybrid algorithms to leverage the benefits of more than one algorithm. Common blockchain algorithms are proof-of-work (PoW), Proof of Stake (PoS), Delegated Proof-of-Stake (DPoS), Proof of Burn (PoB) and Proof of Authority (PoA).

In view of the foregoing, it is pellucid that blockchain technology has the potential to redefine the battle against counterfeiting as it allows immutable transactions, which are publicly available and distributed globally. Moreover, it is chronologically updated and cryptographically sealed.

Thus, in this work, we propose a low-cost and user-friendly blockchain-based solution for detecting and reporting fake products. The proposed system combines product inherent features, copy-sensitive QR Code, location information as well as Track and Trace technologies. It utilizes a method for detecting counterfeit by capture of inherent features indissolubly linked with the product induced by the production process itself.

The proposed system will utilize a new enhanced consensus algorithm that is, unlike existing protocols, fully decentralized and balances between efficiency and security. The system prototype will be a distributed application (dApp) with a supporting blockchain network. We chose Ethereum as our backend platform due to the fact that our system would require multi-state and flexibility in block storage to record data as well as short block time.

1.2 Problem Statement

Counterfeiting can be noxious when it has to do with food and pharmaceutical products. Counterfeit drugs do not contain the right active ingredients, and therefore can compromise the treatment of chronic and infectious diseases, causing disease progression, drug resistance, and death (Sayyad et al, 2021). This is more predominant in Nigeria as people die after being treated with fake medicine. Secondly, trading activities in counterfeit goods have negative impact on sales and profits of industries, and also generate profits for organized crime at the expense of the affected companies and governments (Sayyad et al, 2021; Yiu, 2021a). Unfortunately, Nigeria is among the countries most affected by counterfeits. In 2020, the World Health Association reported that about 70% of all drugs present in Nigeria were second generation goods (Tavares, 2020; Odumade, 2020).

Furthermore, counterfeiters are becoming more and more professional and sophisticated. They are always developing approaches to better package counterfeit products and bring them to the market undetected (Tavares, 2020). Thus, counterfeit trading activities are increasing at exponential rate. Unfortunately, most current anti-counterfeiting technologies rely on a centralized authority to combat counterfeit products. Such centralized architecture results in issues such as single point processing, storage, and failure. Moreover, even the few currently available blockchain-based anti-counterfeiting solutions do not use efficient consensus algorithms in combination with product inherent features, copy sensitive QR code and location information.

1.3 Objectives of the Study

The primary objective of this work is to develop a blockchain-based anti-counterfeiting system with enhanced consensus algorithm. The specific objectives are to:

(i) Model the proposed blockchain anti-counterfeiting system by adopting object-oriented system analysis and design techniques utilizing Unified Model Language (UML) tools.

(ii) Setup a private blockchain network that uses Ethereum as the backend operating system.

(iii) Design an efficient blockchain consensus algorithm that balances between efficiency and security.

(iv) Develop and code smart contracts for automatic execution of system transactions using Solidity.

(v) Develop the user interface component of a distributed application (dApp) that utilizes location information

(vi) Develop applications for generating and scanning product inherent feature-based special QR code.

(vii) Integrate developed applications into the blockchain network, and carryout comprehensive tests on the developed system, as well as validate results.

1.4 Justification of the Study

In response to the growing concern in product counterfeiting as well as the inefficiencies of already developed systems; innovative, efficient and low-cost product anti-counterfeiting solutions with traceability, immutability and transparency functionalities are widely and urgently demanded. A blockchain-based anti-counterfeiting system with enhanced consensus algorithm leveraging product inherent features, copy-sensitive QR codes and location information will provide a permanent solution. The combination of product inherent features, copy sensitive QR code and location information will ensure the provenance, as well as the identification and traceability of genuine products.

The implementation of this research work will have a far-reaching impact on the society (manufacturers, distributors and consumers) as it will combine innovative technologies to combat product counterfeiting. This will result in a very fast, easy, affordable and, most importantly, secured means of detecting counterfeit products. Consequently, this will drastically reduce counterfeiting and its adverse effects on economy and public health.

1.5 Scope of the Study

The area of subject matter in this work investigates the use of blockchain network with an efficient consensus algorithm in combination with product inherent features (such as texture, shape and colour), location information i.e. GPS coordinates and copy-sensitive QR Codes in combatting counterfeit product trading.

The network will use Ethereum as the backend blockchain operating system as well as Solidity for writing and compiling smart contracts. The proposed system will use an improved blockchain consensus algorithm, and will be modelled by adopting object-oriented system analysis and design techniques utilizing Unified Model Language (UML) tools. TestRpc will be utilized in testing the performance as well as the efficiency of the developed prototype system.

On the other hand, this work neither considers the development of policies and standards for combatting counterfeit trading, nor does it handle the implementation of such polices in punishing offenders. Thus, it is not concerned with copyright infringement, digital piracy, and fraudulent documents; rather it examines the prevention of counterfeiting on a technological basis. Also, there are various categories of counterfeit goods in different domains: a precise taxonomy for each domain is out of the scope of this work. Furthermore, this work does not include the development of blockchain explorer as well as automated mechanism for enrolling products on the network.

CHAPTER TWO

LITERATURE REVIEW

2.1 Review of Key Concepts

In this section, we have thoroughly reviewed some literatures to enhance our understanding of the key concepts in the development of a blockchain-based anti-counterfeiting system. The review covers concepts of blockchain, product inherent features and QR code technologies.

2.1.1 Blockchain Technology

A blockchain is a software that allows a network of computers to connect directly to one another without a trusted central authority as a transaction intermediary (Ma et al, 2020). It establishes a distributed or decentralized network of computers through which values can be exchanged instantly, stored securely, and at a lower cost. The data is copied to multiple nodes, and each of these nodes runs the copy of the blockchain.

Blockchain architecture is divided into three layers which are Applications, Decentralized Ledger and Peer-to-Peer Network (Tushar et al, 2020). Application layer provides a human readable interface where users can keep track of their transactions. Decentralized Ledger is the middle layer in a blockchain architecture that confirms a consistent and temper-proof global ledger. In this layer, transactions can be grouped into blocks which are cryptographically linked to one another.

The bottom layer is the Peer-to-Peer (P2P) Network where node types play different roles and various messages are exchanged to the main Decentralized Ledger. A P2P network consists of a group of devices that collectively store and share files. Each participant (node) acts as an individual peer. Typically, all nodes have equal power and perform the same tasks (Binance Academy, 2019).

In essence, a P2P system is maintained by a distributed network of users. Usually, they have no central administrator or server because each node holds a copy of the files -

acting both as a client and as a server to other nodes. Thus, each node can download files from other nodes or upload files to them. On P2P networks, the connected devices share files that are stored on their hard drives. Using software applications designed to mediate the sharing of data, users can query other devices on the network to find and download files. Once a user has downloaded a given file, they can then act as a source of that file (Tushar et al., 2020).

Since every node stores, transmits and receives files, P2P networks tend to be faster and more efficient as their user base grows larger. Also, their distributed architecture makes P2P systems very resistant to cyberattacks. Unlike traditional models, P2P networks do not have a single point of failure (Tushar et al., 2020).

Blockchain system is further based on distributed consensus mechanisms, smart contracts, cryptographic mechanisms and mining process. (Sayyad et al, 2021). A brief description of the aforementioned mechanisms as well as the discussion of blockchain properties will further boost our understanding of blockchain concept.

(A) Distributed Consensus Algorithms

A consensus algorithm is a mechanism that allows users or nodes to coordinate in a distributed setting. It needs to ensure that all nodes in the system can agree on a single source of truth, even if some agents fail (Binance Academy, 2018). Generally, blockchain consensus algorithms are responsible for keeping the network secure and for verifying and validating transactions. The consensus mechanism determines who is able to add new blocks of transactions, and one of its primary aims is to ensure that the chain is not re-written. Common examples of a consensus mechanisms are proof-of-work (PoW), Proof of Stake (PoS), Delegated Proof-of-Stake (DPoS), Proof of Burn (PoB) and Proof of Authority (PoA).

(i) Proof of Work (PoW): This algorithm is based on the idea of solving a complex mathematical puzzle to generate a solution block. It requires a lot of computational

power, and the miner who solves the puzzle to mine a block gets rewarded (Hu et al., 2021).

(ii) Proof of Stake (PoS): This algorithm awards the highest chance of mining a block to the person with the highest number of coins in the validators pool. In the subsequent rounds of validations, the chances for the previously selected validator keep reducing until other validators also get the chance to verify a block (Saleh, 2020).

(iii) Delegated Proof-of-Stake (DPoS): In Delegated Proof-of-Stake, stakeholders select delegates and outsource validation of the block to them. Stakeholders will cast a vote to select delegates (Hu, Yan, Han, & Yu, 2021).

(iv) Proof of Burn (PoB): Essentially, PoB looks like a Proof of Work algorithm but with reduced rates of energy consumption. The block validation process of PoB-based networks does not require the use of powerful computational resources and does not depend on powerful mining hardware. Instead, cryptocurrencies are intentionally burned as a way to invest resources in the blockchain, so the candidate miners are not required to invest physical resources. In other words, by performing coin burns, users are able to demonstrate their commitment to the network, gaining the right to mine and validate transactions.

(v) Proof of Authority (PoA): It is a reputation-based consensus algorithm that introduces a practical and efficient solution for blockchain networks (especially the private ones). The PoA consensus algorithm leverages the value of identities, which means that block validators are not staking coins but their own reputation instead. Therefore, PoA blockchains are secured by the validating nodes that are arbitrarily selected as trustworthy entities. The Proof of Authority model relies on a limited number of block validators and this is what makes it a highly scalable system. Blocks and transactions are verified by pre-approved participants, who act as moderators of the system (Shamili & Muruganantham, 2022).

(B) Smart Contracts

Smart contracts are simply programs stored on a blockchain that are automatically executed when predetermined terms and conditions are met. They are typically used to automate the execution of an agreement so that all participants can be immediately certain of the outcome, without any intermediary's involvement or time loss. Technically, a smart contract is a collection of code (its functions) and data (its state) that resides at a specific address on the blockchain (Kashyap, 2021).

Smart contracts work by following simple "if/when…then…" statements that are written into code on a blockchain. A network of computers executes the actions when predetermined conditions have been met and verified. These actions could include releasing funds to the appropriate parties, registering a product, or issuing a ticket. The blockchain is updated when the transaction is completed (IBM, 2022).

In blockchain, smart contracts play very essential roles, they help to make the transactions taking place more safe and secure, and function in an organized manner. They also help other components like applications running on these platforms to be more accessible. In other words, using smart contracts make transactions traceable, transparent, and irreversible (Arora, 2022).

When a smart contract is deployed, it creates an instance (contract account) on the blockchain network. Deployment of a smart contract is done by sending a transaction to the network with bytecode. An emulator can be used to deploy a smart contract on a local network e.g., Ganache-cli (Wadhwa, 2021).

According to Arora (2022) developing and deploying smart contracts require the following steps:

Step 1: Business teams collaborate with developers to define their criteria for the smart contract's desired behavior in response to certain events or circumstances.

Step 2: The developers then use a smart contract writing platform to create and test the logic. After the application is written, it is sent to a separate team for testing.

Step 3: The contract is then deployed on an existing blockchain or other distributed ledger infrastructure once it has been authorized.

Step 4: The smart contract is configured to listen for event updates from an "oracle," which is a cryptographically secure streaming data source, once it has been deployed.

Step 5: Once it obtains the necessary combination of events from one or more oracles, the smart contract executes.

(C) Cryptographic Mechanism

Blockchain technology makes use of cryptography in multiple different ways – for wallets, transactions, security, and privacy-preserving protocols. To accomplish these, it employs different cryptographic techniques including public-key cryptography, hashing and Merkle trees (Lai, 2018).

(i) Public-key cryptography (also called asymmetric cryptography) uses public and private key encryption and decryption computer algorithms to secure user data. The public key may be widely distributed, but the private key is meant to be known only by its owner. Public-key cryptography is often used for encrypting messages between two people or two computers in a secure way. Anyone can use someone's public key to encrypt a message, but the only way to decrypt the message is by using the corresponding private key (Lai, 2018).

Encryption allows raw data being transmitted on blockchain network to be converted to an unreadable format that makes no sense to third party readers. When one user sends message to another user, the sender will use the public key of the receiver to encrypt the data, and then the receiver will use his private key to decrypt and read the message.

Another application of asymmetric cryptography algorithms is that of authenticating data through the use of digital signatures. Basically, a digital signature is a hash created using the information in a message. When that message is received, the signature can be checked by the recipient using the sender's public key. This way, they can authenticate the source of the message and ensure that it

has not been tampered with. In some cases, digital signatures and encryption are applied together, meaning the hash itself may be encrypted as part of the message (Lai, 2018).

(ii) Hashing is a process whereby an algorithm (hash function) receives an input of data of any size and returns an output (hash) that contains a predictable and fixed size or length. Regardless of the input size, the output will always be the same length. But if the input changes, the output will be completely different. Conversely, if the input does not change, the resulting hash will always be the same, no matter how many times you run the hash function (Binance Academy, 2020).

Within blockchains, these output values, known as hashes, are used as unique identifiers for data blocks. The hash of each block is generated based on both the data contained within that block and the hash of the previous block, and that is what creates a chain of linked blocks. The block hash is dependent on the data contained within that block, meaning that any change made to the data would require a change to the block hash. A diagram of chain of blocks using hashing architecture is shown in Plate 2.1.

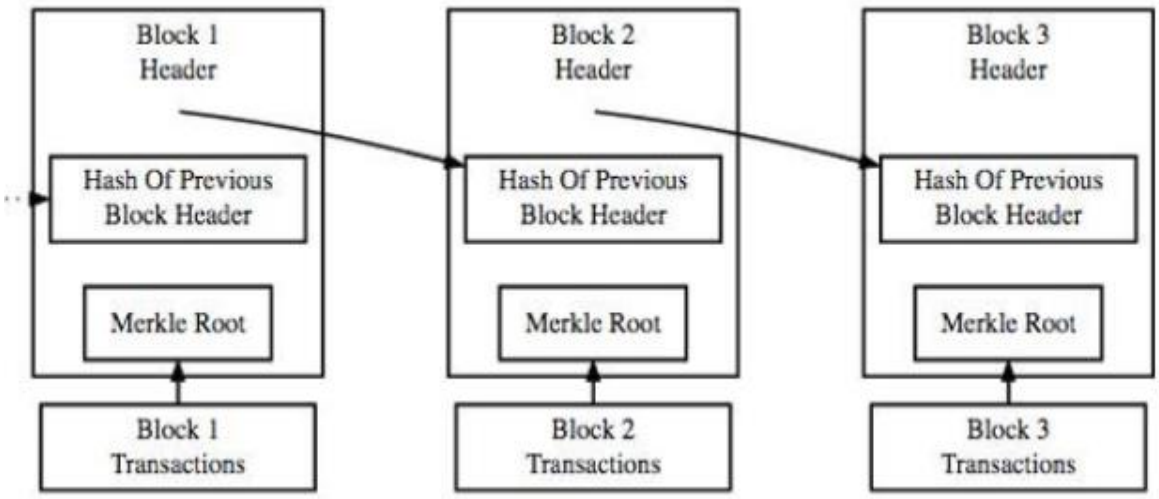

Plate 2.1: Using hashing to link or chain blocks (Lai, 2018).

Hashing is also leveraged in the consensus algorithms used to validate transactions. On the Bitcoin blockchain, for example, the Proof of Work (PoW) algorithm utilizes a hash function called SHA-256. As the name implies, SHA-256 takes data input and returns a hash that is 256 bits or 64 characters long. (Binance Academy, 2020).

(iii) Merkle tree is a fundamental part of blockchain technology. It is a mathematical data structure composed of hashes of different blocks of data, and which serves as a summary of all the transactions in a block. It is a tree in which every "leaf" (node) is labelled with the cryptographic hash of a data block, and every node that is not a leaf (called a branch, inner node, or inode) is labelled with the cryptographic hash of the labels of its child nodes. Put simply, each leaf node consists of a cryptographic hash of its original data, and every parent node is a hash of the combination of its child node hashes. A diagram of a Merkle tree with 4 leaf nodes is shown Plate 2.2.

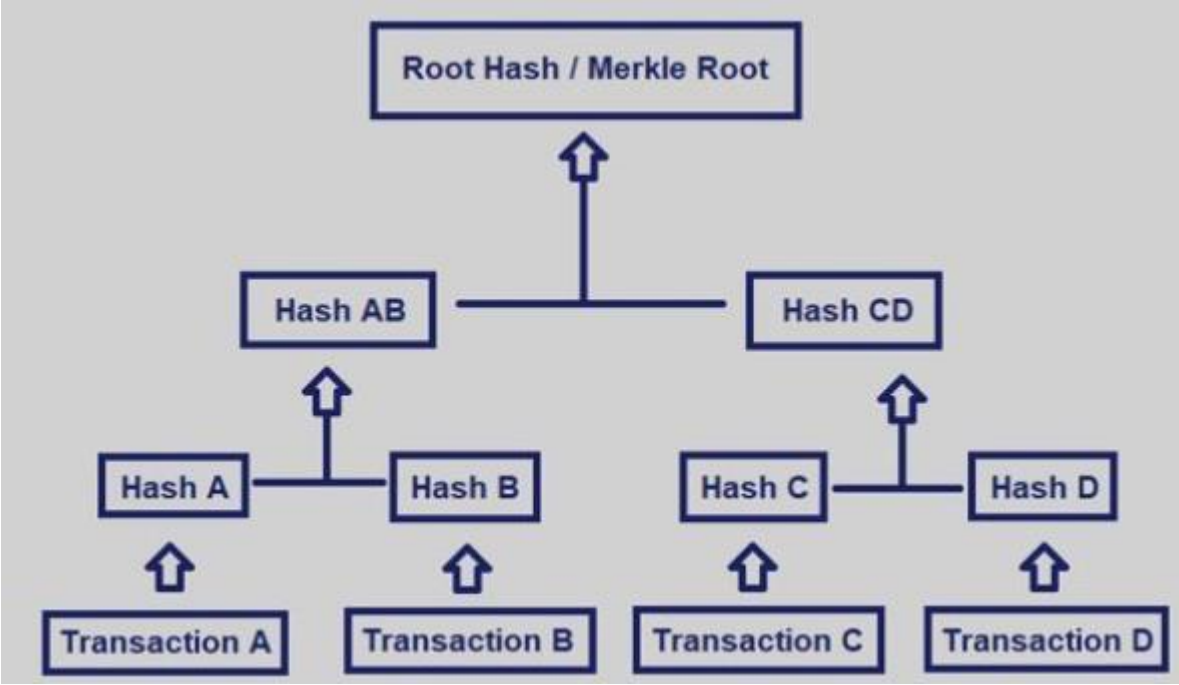

Plate 2.2: A diagram of a Merkle tree with 4 leaf nodes (Lai, 2018).

Each leaf node in Plate 2.2 represents a hash of the data for transactions A, B, C, and D. Then hash A and hash B are combined and hashed to produce hash AB, and hash CD is produced in the same way. Finally, hash AB and hash CD are combined and hashed to form the Merkle root of the tree.

A Merkle tree allows efficient and secure verification of the contents of a large data structure. For instance, using the Merkle root and applying the properties of cryptographic hash functions, one can quickly tell if transactions in a given block have been tampered with and the specific transaction that is being tampered. If a single transaction in a confirmed block is altered, the Merkle root would end up being completely different from the "correct" Merkle root and the tampering would be obvious (Lai, 2018).

Merkle trees also allow users to verify that their transaction has been included in a block without downloading the entire blockchain. Processes such as Simplified Payment Verification are able to traverse branches in the Merkle tree and check if a certain transaction has been hashed into that tree.

(D) Mining Process

In Blockchain technology, the process of adding transactional details to the present digital/public ledger is called 'mining.' Though the term is associated with Bitcoin, it is used to refer to other Blockchain technologies as well. Mining involves generating the hash of a block transaction, which is tough to forge, thereby ensuring the safety of the entire Blockchain without needing a central system (Rawat, 2021).

Any new block in the chain is formed by hashing the transactions sent in the network by participants. Take for instance, when they are requesting to send cryptocurrencies or to save files. The block should have a block number (its count in the chain), data field, cryptographic hash associated with it, and a Nonce (Haber & Stornetta, 2022).

The Nonce (Number Used Once) is used to generate a cryptographic hash that meets a certain criterion to be valid. For instance, let us say a requirement that for the hash output

to be valid it has to have four zeros in the lead (like the case in this output: 00001acbm010gfh1010xxx). Otherwise, it will be invalid. It is made valid by using the nonce. A nonce is a random number that has to be changed manually and many times by the way of guesswork such that when it is fed into the algorithm or hash function together with the rest of the block data. It must give a valid block that obeys the rule or target, for instance starting with the four zeros (Haber & Stornetta, 2022).

This is actually what miners do in Proof of Work algorithms, the mining software keeps on guessing the number starting with one, incrementally. It keeps feeding the guesswork until it produces a hash output that meets the specified criterion or target. The expiry duration it takes to make a correct guess for a particular block data set varies from blockchain to blockchain, with Bitcoin being 10, Ethereum 3 seconds, etc. Numerous nodes in the blockchain network try to figure out the right hash value to meet a pre-determined condition using computational algorithms. The node or miner that does the correct guesswork is rewarded with cryptocurrency in the case of Proof of Work. To put it more plainly, Blockchain miners attempt to solve a mathematical puzzle, which is referred to as a proof of work problem. Whoever solves it first gets a reward. Once the block is mined, it is added to the previous chain, making it immutable or unchangeable but also publicly available via the blockchain explorers (Haber & Stornetta, 2022).

(E) Properties of Blockchain

The properties or features of blockchain are depicted in Plate 2.3.

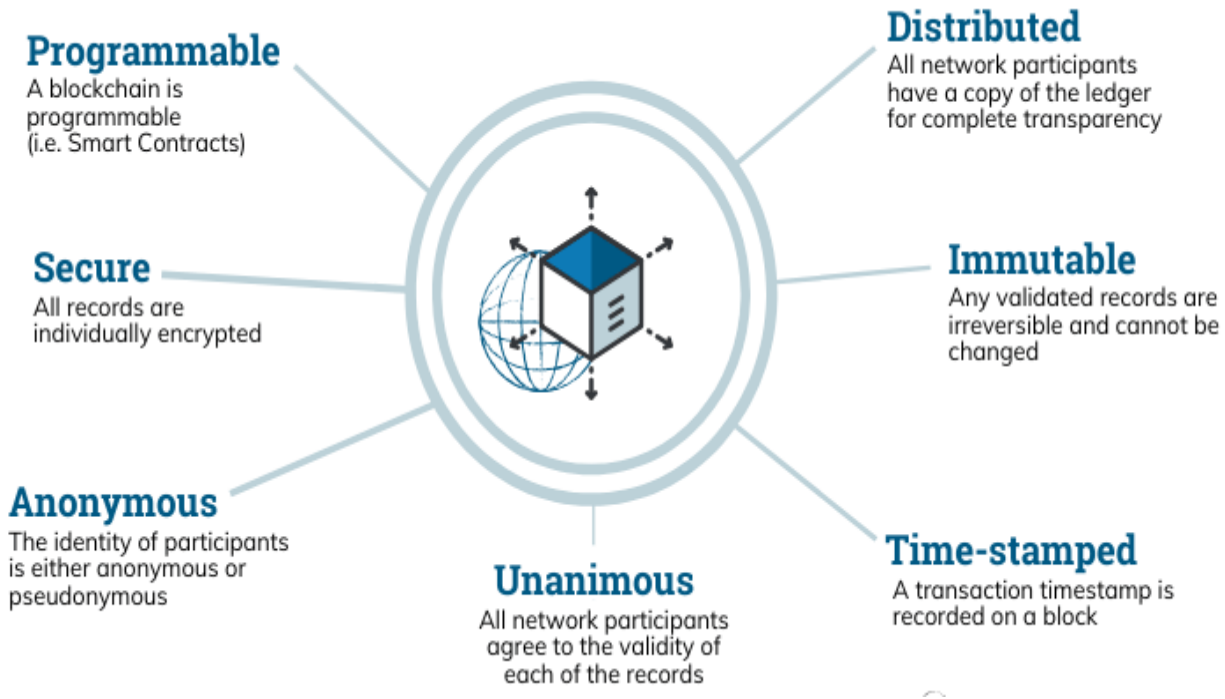

Plate 2.3: Features of blockchain (Euromoney Learning, 2020).

Other properties are flexibility, transparency and decentralization. Following the descriptions of Ma (2020), the characteristics of Blockchain technology are briefly discussed below.

(i) Decentralization: Through decentralized operations and storage, each node of the blockchain implements the verification, delivery, and management of information at the local side. Blockchain technology does not rely on an additional third-party control; it has no centralized control.

(ii) Immutability: After a block has been approved and added to the blockchain, it cannot be tampered with. Accordingly, once a block in the blockchain is altered, it will be detected immediately and rejected by other nodes.

(iii) Transparency: The data in blockchain is completely public and anyone can access it. Within the information flow, one can clearly see who is passing data to whom as blockchain maintains a continuous transaction log file.

(iv) Flexibility: The technology of blockchain is open source and anyone can modify it into his own version. There are already numerous flexible blockchain

platforms available, and users can also redevelop a new blockchain platform if they desire so. Blockchain is an unlimited technology meaning that users can create multiple applications based on blockchain.

(v) Anonymous/Privacy: In blockchain, each user is anonymous, and each user can have multiple addresses. When the system is operating, only one address is used as the identity, and the anonymous address can hardly be mapped to the real person, thereby protecting the user's privacy.

(vi) Security: Public key encryption is used in blockchain to protect data security. Users can generate their own key pairs, including a private key and a public key. Private key is used to sign data, and public key is used to verify the authenticity of the signed data. As long as the user prevents the private key from leaking, the data will remain secured.

(vii) Unanimous/Consensus: Consensus refers to the ability of the nodes within a distributed blockchain network to agree on the true state of the network and on the validity of transactions. Typically, the process of achieving consensus is dependent on the so-called consensus algorithms (Binance Academy, 2020).

(viii) Distributed Technology: Blockchain networks utilize distributed technology that collects a digital record of any event and store it in a distributed database that is shared among all the users connected with it.

(ix) Timestamped: Blockchain timestamp is a small data stored in each block as a unique serial and whose main function is to determine the exact moment in which the block has been mined and validated by the blockchain network. Implementing a timestamp makes it impossible to repeat a block in the future, since in addition to the time, the date of creation of the block is also stored, therefore, there is no possibility that it will be repeated.

(x) Programmable: Blockchain is programmable, meaning that a programmer can actually write a program in a high-level language that expresses the terms of a smart contract that is being carried out on the blockchain. Programmability is what

makes blockchain truly transformational. Programming is a necessary component to realizing the full potential of blockchain.

2.1.2 Product Inherent Features

Product inherent features are characteristics or properties that are indissolubly linked with the product, and induced by the production process itself. The specific conditions of production, manufacturing technologies and materials generate specific features (such as plasticity, elasticity, thermal conductivity, shape, color, surface texture, transparency), which identify the product uniquely. Since a counterfeiter gains margin by the use of inferior production processes and materials, the differences between genuine product and counterfeit can be captured from these inherent features in an automated fashion. On the other hand, high-quality branded products, as the target of counterfeiting, have usually, due to the production processes and materials used, and in view of its processing machinery and equipment, a grade of high quality. The specific conditions of production, manufacturing technologies and materials generate specific features, which serve as proof of the product identity (Horn, 2012).

In general, only the person familiar with the manufacture of the product can combine these inherent characteristics in their entirety so that it can differentiate the genuine product from a clear counterfeit. The technological approach is the detection of these features in an automated fashion through the combination of digital sensing and machine learning.

The product inherent feature approach is based on the stationary and mobile capture of key product features indissolubly linked with the product which enable its production process to be traced. The inherent characteristics that the high-quality production process impregnate in the genuine product are combined with one another to serve as proof of product identity. They form the basis on which electronic certificates of authenticity can be issued without the need for complicated explicit security markings. The identity characteristics captured by this range of sensors serve both for the product identification

and product authentication. This gives enhanced protection against counterfeiting as the inherent characteristics can neither be copied nor removed from the product. (Blankenburg, 2015).

As discussed in (Horn, 2012), the detection mechanisms of common product inherent features such as shape, surface texture and odour are briefly described below.

(A) Shape: One distinguishable feature of brand products is the shape itself. To capture the shape of a real-world object in three dimensions a 3D scanner, or range camera, is be used. Three-dimensional (3D) data capture of shape is necessary for counterfeit detection, because it provides important additional information. Most 3D shape matching approaches or applications use Iterative Closest Point Algorithm or its optimized variants to match objects.

(B) Surface Texture: Image texture is defined as a function of the spatial variation of pixel intensities. The ability to characterize visual textures and extract the features inherent to them is considered to be a powerful tool for counterfeit detection. A textural signature capable of capturing these features, and in particular capable of coping with various changes in the environment, would be highly suited to describing and recognizing image textures. A description method based on fractal geometry known as the multifractal spectrum is a useful tool in characterizing image texture.

(C) Odour: Much effort has been spent on how odour could be measured. The European Standard EN-13725 defined a method for the objective determination of the odour concentration of a gaseous sample using so called dynamic olfactometry. It is currently the only standardized method for the evaluation of odour impressions.

2.1.3 Copy Sensitive QR Codes

A Quick Response (QR) code is an enhancement on traditional unidimensional barcodes that stores vast information as a series of pixels in a square-shaped grid, and can be read

easily by a digital device such as a smartphone. They are decoded by specialized software that is able to extract data from the patterns that are present in the matrix (Hayes, 2021). QR codes are frequently used to track information about products in a supply chain, and often used in cryptocurrency wallet addresses, marketing and advertising campaigns. QR codes are often used to contain web address information and links, but they can also be used to direct Smartphone users to a multitude of other media too (e.g., videos, images etc.). Every QR code consists of a number of black squares and dots which represent certain pieces of information. When a Smartphone scans this code, it translates that information into something that can be easily understand by humans. There is plethora of online QR code generators that make the entire QR code generation process extremely straightforward. A typical QR code is shown in Plate 2. 4.

Plate 2.4: A typical Quick Response (QR) code (Hayes, 2021)

In order to enhance the security of QR code, a Copy Sensitive Graphical Codes (CSGC) could be inserted into it. The detection of counterfeits using a CSGC relies on an "information loss principle", which states that every time a digital image is printed or scanned, some information is lost about the original digital image. A CSGC is a maximum entropy image that attempts to take advantage of this information loss. Since producing a counterfeit CSGC requires an additional scanning and printing processes, it

will have less information than an original CSGC. By measuring the information in the scanned CSGC, the detector can determine whether the CSGC is an original print or a copy (Zhang, & Nenghai, 2019). Additionally, each printer or scanner has its own signature. Using these two properties, the authentication test can distinguish the original CSGC (printed once) from copied/counterfeited (printed several times).

2.2 Review of Related Works

In this section, we have carried out an exhaustive review a good number of related works in order to further enhance our knowledge in the development of a blockchain-based anti-counterfeiting system that uses an enhanced algorithm, and employs product inherent features as well as location coordinates copy-sensitive Quick Response (QR) codes. The review covers apposite areas such as Ethereum-based blockchain networks, blockchain consensus algorithms, smart contracts, decentralized blockchain applications (dApp), blockchain explorer, modelling blockchain-based systems, techniques for product inherent features, copy-sensitive QR codes, cryptography, and blockchain-enabled supply chain anti-counterfeiting and traceability.

2.2.1 Modelling Blockchain-based Systems

The approach for risk modelling of blockchain ecosystem was described in (Kabashkin, 2017). Practical realization of simulation on the base of Petri Net model was proposed. The process of E-net model design for analysis of blockchain platform and critical aspects of a blockchain solution included the next main components: formal construction of the individual blocks process formation in the blockchain; formal construction for transformation probability of various possible risks in the blockchain ecosystem into the generator of risk events; formal metamodeling of blockchain operation with critical aspects of a blockchain solution; base set of modelling elements for above-mentioned transformation procedure; and software tools for simulation experiment.

Jurgelaitis, Drungilas, Ceponiene, Butkiene, and Vaičiukynas (2019) proposed a method based on Model Driven Architecture, which could be used for defining and specifying

blockchain structure and behavior. Such approach could be used as one of the ways for describing blockchain-based systems in a more general language in order to facilitate blockchain development process.

Memon, Li, Ahmed, Khan, Nazir and Mangrio (2018) implemented the simulation of mining process in Blockchain based systems using queuing theory. They took the parameters of one of the mature Cryptocurrency, Bitcoin's real data and simulated using M/M/n/L queuing system in JSIMgraph. They achieved realistic results; and expect that it will open up new research direction in theoretical research of Blockchain based systems.

In (Ling, Le, Wang, Ding & Gao, 2021), an analytical framework was developed to model blockchain radio access network (B-RAN) and to provide some basic fundamental analysis. Starting from block generation, a queuing model based on a time-homogeneous Markov chain was established. From the queuing model, the performance of B-RAN was evaluated with respect to latency and security considerations. Finally, to validate the proposed model, experimental results were presented via an innovative prototype.

The authors in (Memon, Li & Ahmed, 2019) proposed a queuing theory-based model for understanding the working and theoretical aspects of the blockchain. They validated the model using the actual statistics of two popular cryptocurrencies, Bitcoin and Ethereum, by running simulations for two months of transactions. The obtained performance measures parameters such as the Number of Transactions per block, Mining Time of Each Block, System Throughput, Memorypool count, Waiting Time in Memorypool, Number of Unconfirmed Transactions in the Whole System, Total Number of Transactions, and Number of Generated Blocks; these values were compared with actual statistics. It was found that the results gained from the proposed model were in good agreement with actual statistics.

Three complementary modeling approaches based on well-known software engineering models were presented and applied to a blockchain-oriented software (BOS) system by

Rocha and Ducasse (2018). The goal of the proposed model to start discussions on specialized blockchain modeling notations was well achieved.

Seebacher and Maleshkova (2018) proposed a model-driven approach, combining an ontology and a layer model, that is capable of capturing the properties of existing blockchain-driven business networks. The layers were used to facilitate the comprehensive description of such networks. They also introduced the Blockchain Business Network Ontology, formalizing the concepts and properties for describing the integral parts of a blockchain network. Finally, they showed the practical applicability of their work by evaluating and applying it to an available blockchain use case.

The authors (Vladyko, Spirkina & Elagin, 2021) proposed approaches to modeling blockchain-based systems and, as an example, proposed to use a simulation system to assess the performance of a blockchain network and its elements. An overview of analytical and simulation modeling solutions focusing on mass service systems was conducted and presented. Simulation comparison results are presented.

2.2.2 Blockchain Consensus Algorithms

Lashkari and Musilek (2021) carried out a comparative analysis of 130 consensus algorithms using a novel architectural classification. The distribution of the reviewed algorithms was analyzed in terms of the proposed classification and different application domains, along with the applicability of each class among the top 10 platforms in the most prominent blockchain application domains. The analysis concluded that envisaging future prospects for consensus algorithms is an important part of distributed ledger technology.

The authors in (Shahaab, Lidgey, Hewage & Khan, 2019) reviewed 66 known consensus protocols, and classified them into philosophical and architectural categories, also providing a visual representation. As a case study, they focused on the public sector and highlighted potential protocols. They also listed these protocols against basic features and

sector preference in a tabular format to facilitate selection. Finally, they argued that no protocol is a silver bullet, therefore consensus algorithms should be selected carefully, considering the sector requirements and environment.

Yang, Zhou, Wu, Long, Xiong and Zhou (2019) introduced the computing power competition of Proof of Work (PoW) into Delegated Proof of Stake (DPoS) in order to design an improved consensus algorithm named Delegated Proof of Stake with Downgrade (DDPoS). Through further modification, the impact of both computing resources and stakes on generating blocks was reduced to achieve higher efficiency, fairness, and decentralization in consensus process. Then a downgrade mechanism was proposed to quickly replace the malicious nodes to improve the security. The simulation experiments in blockchain system show that the proposed consensus algorithm is significantly more efficient than PoW and PoS. Moreover, through the downgrade mechanism, the proposed consensus algorithm can detect and downgrade malicious nodes timely to ensure the security and good operation of system.

Santiago, Ren, Lee and Ryu (2021) designed a Byzantine fault-tolerant consensus protocol for sharded blockchain networks that does not rely on expensive leader-driven communication. The proposed protocol selects a single block proposer at a time and uses threshold signatures as a voting mechanism to confirm the validity of the proposed block. By using a gossip-like communication scheme, each node can collect and recover the group signature. With only one block proposer per consensus round, there is no possibility of conflicting blocks and resultant forks. Performance analysis shows that the proposed protocol enabled hundreds of nodes to participate in the agreement process, and finalized large blocks in approximately 10 seconds.

An improved Proof-of-Trust (PoT) consensus scheme with the underlying technology of blockchain, which is appropriate for crowdsourcing service scenarios, was proposed in (Zhu, Li, Fang & Chen, 2020). Firstly, the PoT consensus selects nodes with high credibility using subjective logic reputation algorithm. Only selected nodes have the

chance to generate blocks, participate in verification, and perform crowdsourcing tasks. Secondly, the choice scheme of generate-block nodes was further optimized through the unpredictability of timestamp and digital signature. Moreover, an incentive mechanism based on game theory was designed in this consensus. The analysis and simulation results demonstrated the effectiveness, feasibility and scalability of the proposed approach. The miner selection process of improved Proof-of-Trust (PoT) consensus scheme is shown in Plate 2.5.

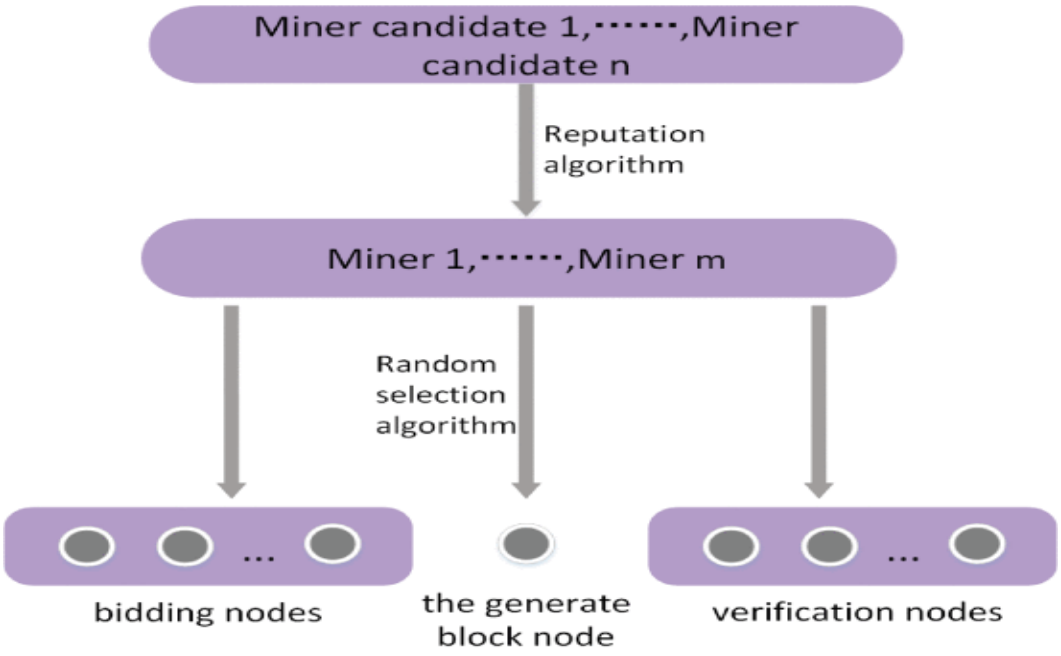

Plate 2.5: Miner selection process of improved Proof-of-Trust (PoT) consensus scheme (Zhu et al., 2020).

Du, Chen & Ma (2020) proposed a new type of consensus algorithm: mixed Byzantine fault tolerance (MBFT), that uses sharding and layered technology. MBFT functionally partitions the nodes that participate in the consensus process and improves the scalability and efficiency without sacrificing security. MBFT also introduces a random node selection mechanism and a credit mechanism to improve security and fault tolerance. The

result of a performance evaluation proved that MBFT has good security and scalability as well as high throughput.

In (Mihaljevic, 2020), the author proposed a technique that provides a flexibility for selection of the energy and space resources which should be employed by a node participating in the consensus procedure. The technique originated from the cryptographic time-memory-data trade-off approaches for cryptanalysis. The proposed consensus technique is based on a puzzle where the problem of inverting one-way function is solved by employing a dedicated Time-Memory-Data Trade-Off paradigm. The technique was implemented in Go language, and included as an alternative consensus option in Ethereum platform.

2.2.3 Blockchain Cryptographic Algorithms

Liang, Zhang, Lei, Tang, Li and Zomaya (2021) proposed a homomorphic encryption-based Blockchain for circuit copyright protection that effectively addresses the issues in the protection of circuit copyright transactions, such as low security of private data, low efficiency in transaction data storage, cooperation and supervision. They established a homomorphic encryption-based mathematical model by utilizing Blockchain and intelligent contract, and next, the algorithms that include Blockchain generation, homomorphic chain encryption/decryption, and intelligent contract are designed. The experimental results show that the proposed algorithm reduced the transmission cost and improved the efficiency of data storage and supervision. In addition, it was resilient to several common attacks (e.g., double-spending attacks), yet it incurred low cost/overhead and has a higher level of security when compared to three other competing algorithms.

A blockchain privacy protection scheme based on ring signature was constructed by Li et al. (2020). The solution built a privacy data storage protocol based on the ring signature on the elliptic curve, and used the complete anonymity of the ring signature to ensure the security of data and user identity privacy in blockchain applications. The correctness and safety proof analysis of the proposed scheme were also carried out.

Yaji, Bangera and Neelima (2018) considered Goldwasser-Micali and Paillier encryption schemes for the comparative evaluation study with a focus on data privacy techniques. It was proved that the above two encryption schemes have less processing time and provide more strength to the possible attacks. Finally, the pros and cons of the Goldwasser-Micali, Paillier and non-homomorphic encryption schemes that are expected to add value to blockchain technology were extensively discussed.

A blockchain-based trusted data management scheme (BlockTDM) in edge computing was proposed by Zhaofeng, Xiao-chang, Jain, Khan, Hongmin and Zhen (2020). The proposed scheme includes mutual authentication protocol, flexible consensus, smart contract, block and transaction data management, blockchain nodes management, and deployment. The BlockTDM scheme can support matrix-based multichannel data segment and isolation for sensitive or privacy data protection, and moreover, the authors designed user-defined sensitive data encryption before the transaction payload stores in blockchain system, and implemented conditional access and decryption query of the protected blockchain data and transactions through smart contract. Analysis and evaluations manifest that the proposed BlockTDM scheme provides a general, flexible, and configurable blockchain-based paradigm for trusted data management with tamper-resistance, which is suitable for edge computing with high-level security and creditability.

The research carried out in (Durga, Poovammal, Ramana, Jhanveri, Singh & Yoon, 2022) proposed a novel chaotic encryption-based blockchain-IoT architecture to clinch the security and privacy of data. Since smart sensors and image sensors are used widely in an IoT environment, the proposed scheme was tested with different image sets to evaluate performance metrics such as Number of Pixel Change Rate (NPCR), Unified Averaged Changed Intensity (UACI), Correlation Coefficients, and entropy under different attack scenarios. The result obtained indicates an NPCR of 99.65%, a UACI of 34%, and an entropy value close to 8. These values revealed that the novel chaotic encryption-based blockchain-IoT architecture will be safe from IoT attacks.

Chen, Kuo and Wu (2021) examined the possibility (or cost) of embedding the ideal lattice-based FHE into the Ethereum Blockchain for building up a new trustworthy framework with security and privacy protection capability. Due to the limitations of current Blockchain, the execution of FHE is conducted off-chain; at the same time, on-chain members can call FHE-based functions to directly compute the ciphertext domain operations after their Smart Contracts have been deployed to the Blockchain. To illustrate and benchmark the examining results of the framework, an FHE and Blockchain-based Vickrey auction system was also developed, in which the online bidding prices were kept secret. Simultaneously, the determination of the winner and the transferring of payments were conducted autonomously by smart contracts.

Zhang et al. (2019) presented a practical scheme by adding Identity-Based encryption system, which effectively improves the data privacy for non-transaction applications. Analyses show that the proposed scheme has a high security level which can prevent both disguise and passive attacks, and is functional, effective and practical in many applications for non-transactional scenarios.

A blockchain architecture based on the proxy re-encryption scheme was proposed by Zonda and Meddeb (2020). The scheme was integrated within smart contracts to provide a very efficient, fast, and secure platform. The proposed architecture was implemented in an Hyperledger Blockchain and tested in a real transport and mobility use case.

Lai, Hsueh and Wu (2019) proposed a protocol and a reliable encryption scheme to make time-sensitive message be opened on time at a fully decentralized environment, which is then integrated with the blockchain to adapt to different computing power situations. The method also provided the capability of incorporating with appropriate incentives for encouraging participants to contribute their computing resources, which makes the system more suitable for real world applications.

A proxy re-encryption approach to secure data sharing in cloud environments was proposed by Agyekum, Xia, Sifah, Cobblah, Xia and Gao (2021). Data owners can

outsource their encrypted data to the cloud using identity-based encryption, while proxy re-encryption construction will grant legitimate users access to the data. With the Internet of Things devices being resource-constrained, an edge device acts as a proxy server to handle intensive computations. Also, the authors made use of the features of information-centric networking to deliver cached content in the proxy effectively, thus improving the quality of service and making good use of the network bandwidth. Further, the proposed system model was based on blockchain. The security analysis and evaluation of the scheme showed that the approach can ensure data confidentiality, integrity, and security.

2.2.4 Creating Smart Contracts

The authors, Pinna, Ibba, Baralla, Tonelli and Marchesi (2019) performed a comprehensive empirical study of smart contracts deployed on the Ethereum blockchain. The objective of the analysis was to provide empirical results on smart contracts features, smart contract transactions within the blockchain, the role of the development community, and the source code characteristics. They collected a set of more than 10000 smart contracts source codes and a dataset of meta-data regarding their interaction with the blockchain from etherscan.io. Thereafter, they examined the collected data computing different statistics on naming policies, smart contract ether balance, number of smart contract transactions, functions, and other quantities characterizing the use and purpose of smart contracts. The results obtained show that blockchain software is rapidly changing and evolving and it is no longer devoted only to crypto-values applications but to general purpose computation.

Khan et al. (2021) presented a comprehensive survey of blockchain-enabled smart contracts from both technical and usage points of view. To do so, they presented a taxonomy of existing blockchain-enabled smart contract solutions, categorized the included research papers, and discussed the existing smart contract-based studies. Based on the findings from the survey, they identified a set of challenges and open issues that need to be addressed in future studies.

The authors, Varela-Vaca and Quintero (2021), identified and categorized the state-of-the-art related to smart contract languages, in terms of the existing languages and their main features. The review was conducted as a multivocal mapping study that followed the guidelines for conducting multivocal literature reviews as well as the established policies for conducting mapping studies. As a result of the implementation of the review protocol, 4,119 papers were gathered, and 109 of them were selected for extraction. Consequently, 101 different smart contract languages were identified and classified according to a variety of criteria. Furthermore, a discussion on the findings and their implications for future research was outlined.

The study (Dwivedi, Norta, Wulf, Leiding, Saxena & Udokwu, 2021) reviewed existing Smart Contract Languages (SCL) and identified properties that are critical to any future SCL for drafting legally binding contracts. This was achieved by conducting a Systematic Literature Review (SLR) of white- and grey literature published between 2015 and 2019. Using the SLR methodology, 45 Selected and 28 Supporting Studies detailing 45 state-of-the-art SCLs were selected. Finally, 10 SCL properties that enable legally compliant decentralized autonomous organizations (DAOs) were discovered, and specifications for developing SCLs were explored.

Harz and Knottenbelt (2018) made significant efforts to improve the security of smart contracts by introducing new programming languages and advance verification methods. First, they introduced several smart contract languages focusing on security features. To that end, they presented an overview concerning paradigm, type, instruction set, semantics, and metering. Second, they examined verification tools and methods for smart contract and distributed ledgers. Accordingly, they introduced their verification approach, level of automation, coverage, and supported languages. Finally, they presented future research directions including formal semantics, verified compilers, and automated verification.

The research carried out by Kannengiesser, Lins, Sander, Winter, Frey and Sunyaev (2021) identified 29 challenges (e.g., code visibility, code updateability, and encapsulation) associated with smart contracts, and proffered 60 solutions (e.g., gas limit specification, off-ledger computations, and shadowing). Moreover, it developed 20 software design patterns (SDPs) in collaboration with smart contract developers. The SDPs help developers adjust their programming habits and thus support them in their daily development practices. The results obtained provide actionable knowledge for smart contract developers to overcome the identified challenges and offer support for comparing smart contract integration concepts across three fundamentally different Distributed Ledger Technologies protocols (i.e., Ethereum, EOSIO, and Hyperledger Fabric).

The work in (Mao et al., 2019) designed a visual and user-defined smart contract designing systems. It makes the development of domain-specific smart contracts simpler and visualization for contract users. The system implemented the domain-specific features extraction about the crawled data sets of smart contract programs by TF-IDF and K-means++ clustering algorithm. Then, it achieved the automatic generation of unified basic function codes by Char-RNN based on the domain-specific features. The system adopted Google Blockly, and linked the generated codes with UI controls. Finally, it provided a set of specialized templates of basic functions for users to design smart contracts with friendly interface.

Cai, Qu, Liu and Yu (2019) proposed a smart contract based on light-weighted quantum blind signature. Firstly, a smart contract architecture was built, and information processing and information transmitting for quantum blind signature were analysed. Then, life cycle and signature rules of quantum blind signature as well as quantum blind signature protocol for single signer in light-weighted smart contract were designed. Finally, based on the former single-person signature algorithm, a more complex multi-person quantum blind signature algorithm was proposed, and its security was analysed.

The proposed quantum signature schemes based on quantum entanglement features, will improve the security of blockchain smart contracts against quantum attacks.

The authors (Dwivedi, Pattanaik, Deval, Dixit, Norta & Draheim, 2021) specified a smart-legal-contract markup language (SLCML) for legal and business constructs to draft a legally-binding decentralized autonomous organizations (DAO). They first presented a formal smart-contract language (SCL) ontology to describe the legal and business semantics of a DAO. Secondly, they translated the SCL ontology into SLCML. They demonstrated and evaluated SLCML language through the specification of a real life-inspired Sale-of-Goods contract. Finally, the SLCML use-case code was translated into Solidity to demonstrate its feasibility for blockchain platform implementations.

Kim and Lee (2020) proposed an automated method to assess contract analyzers of smart contracts by diversifying test cases. They also presented a systematic method to diversify test cases by combining smart-contract-specific bugs and static analysis barriers. In the experimental results, they identified nine erroneous alarms in the state-of-the-art contract analyzers with automatically generated test cases on five vulnerabilities.

J. Liu and Z. Liu (2019) proposed a taxonomy toward the topic of security verification of blockchain smart contracts and discussed the pros and cons of each category of related studies. Through in-depth analysis of these studies, they realized that the verification of smart contracts based on the formal method is the more significant and effective method to validate whether a smart contract is credible and accurate. They further presented representative studies of formal verification of smart contracts in detail to demonstrate that using a formal method to validate blockchain smart contracts must have a promising and meritorious future.

An overview on the state-of-the-art of smart contracts was presented in (Zheng et al., 2020). They first provided a brief review on smart contract and blockchain technologies. They also pointed out the challenges in smart contracts in different aspects of creation,

deployment, execution, and completion of smart contracts. Finally, they discussed recent advances in solving these challenges.

Kemmoe, Stone, Kim & Son (2020) presented the state-of-the-art technologies of the smart contract protocol. Specifically, they surveyed the most recent advances, then classified them into four categories based on their goals and purposes to effectively deliver knowledge of the current status of smart contracts in domains such as cryptography, access management, and social applications, as well as the structure of smart contracts. In each of these categories, they highlighted the shortcomings of some recent works. Lastly, they proposed interesting future research directions that may help in the advancement and proliferation of smart contracts.

Jiao, Kan, Lin, Sanán, Liu & Sun (2018) introduced an executable operational semantics of Solidity formalized in the K-framework. They presented the semantics of the core features of Solidity, namely memory operations, new contract instance creations and function calls, with an emphasis on the semantics of the exception handling features. The semantics was designed from a general point of view to adapt to the language evolution of smart contracts. Experiment results show that the proposed Solidity semantics has already completely covered the supported high-level core language features specified by the official Solidity documentation, and the covered semantics is consistent with the official Solidity compiler Remix. Furthermore, the proposed semantics ensures correct and secure high-level execution behaviors of smart contracts to reason about compiler bugs and assist developers in writing secure smart contracts.

Sayeed, Marco-Gisbert and Caira (2020) classified blockchain exploitation techniques into four categories based on the attack rationale; attacking consensus protocols, bugs in the smart contract, malware running in the operating system, and fraudulent users. They then focused on smart contract vulnerabilities, analyzing the seven most important attack techniques to determine the real impact on smart contract technology. Finally, they

revealed that even adopting the ten most widely used tools to detect smart contract vulnerabilities does not guarantee high security level.

Tian et al. (2020) proposed a smart contract classification approach based on Bi-LSTM model and Gaussian LDA, which can use a variety of information as inputs of the model, including source code, comments, tags, account and other content information. Bi-LSTM was utilized to capture grammar rules and context information in source code, while Gaussian LDA model was employed to generate comments feature where the semantics of the comments are enriched by embeddings. They also used attention mechanism to focus on the more relevant features in smart contracts for tags, and fuse account information to provide additional information for classification. The experimental results show that the classification performance of the proposed model is superior to other baseline models.

The application of Oracle to fetch off-chain data results in rising deployment costs and requires Ethereum smart contract developers to follow format in programing contract, this constraint decreases the readability of smart contract. Thus, Liu et al (2018) proposed an off-chain data fetching architecture which is cost-effective and highly elastic for smart contract. It also compatible with exited contract, which makes Ethereum smart contract owner able to automate the reply process.

Molina-Jiménez, Sfyrakis, Solaiman, Wong, Chun and Crowcroft (2018) discussed the implementation of smart contracts on hybrid architectures. They showed how a smart contract can be split and executed partially on an off–blockchain contract compliance checker and partially on the rinkeby ethereum network. To test the solution, they exposed it to sequences of contractual operations generated mechanically by a contract validator tool.

2.2.5 Ethereum-based Blockchain Networks

Wood (2018) discussed Ethereum blockchain platform with respect to its design, implementation issues, the opportunities it provides and its future. He described its technological capacity to provide to the end-developer a tightly integrated end-to-end system for building software on a hitherto unexplored compute paradigm in the mainstream.

The authors in (Khan et al., 2020) proposed a content protection and transaction method using Blockchain Ethereum Technology. An encryption algorithm was incorporated in the proposed system to make transactions transparent, and it was also implemented on the content itself to prevent from smart forgery and hacking. The experimental results signify that the proposed method has strong potential to enhance transactions transparency by minimizing the security threats in digital content transactions.

In (Wang et al., 2019), a cloud user identity management protocol based on Ethereum blockchain was proposed, followed by an establishment of a simple credit management system framework. The new protocol is an improved version of Consolidated Identity Management (CIDM) referred to as Ethereum-based Identity Management (EIDM) protocol. In the improved protocol, JSON Web Token (JWT) in OAuth 2.0 was used to introduce smart contracts into EIDM protocol, and the credit management system was added to the system so that it can provide a credible identity authentication protocol for cloud users and service providers. In the end, an analysis on the security of the new protocol showed that the EIDM protocol presents more diversified security guarantees relative to the CIDM protocol.

Gimenez-Aguilar, Fuentes, González-Manzano and Camara (2021) used Ethereum to establish a covert channel considering all transaction fields and smart contracts. Zephyrus, an information hiding mechanism based on steganography, was developed. Its capacity, cost and stealthy nature were assessed empirically through a prototype implementation that is publicly available. Disregarding the time taken to send the

transaction to the blockchain, experimental results show that 40 Kbits can be embedded in 0.57 seconds and retrieved in 2.8 seconds.

Gutierrez-Aguero et al. (2021) designed a method and a tool for dealing with digital identities within Blockchain environments that are compliant with regulations. The new method uses the benefits of key derivation systems to ensure a non-binding interaction between users and the information model associated with their identity. The proposed method was demonstrated in the Ethereum context and illustrated in a burnable pseudo-identity which is a Web3js based implementation for representing digital identities. The creation stage involves an initial seed, from a secure entropy source, and a user secret. The secret is used during the identity storage to symmetrically encrypt it. The creation stage is shown in Plate 2.6.

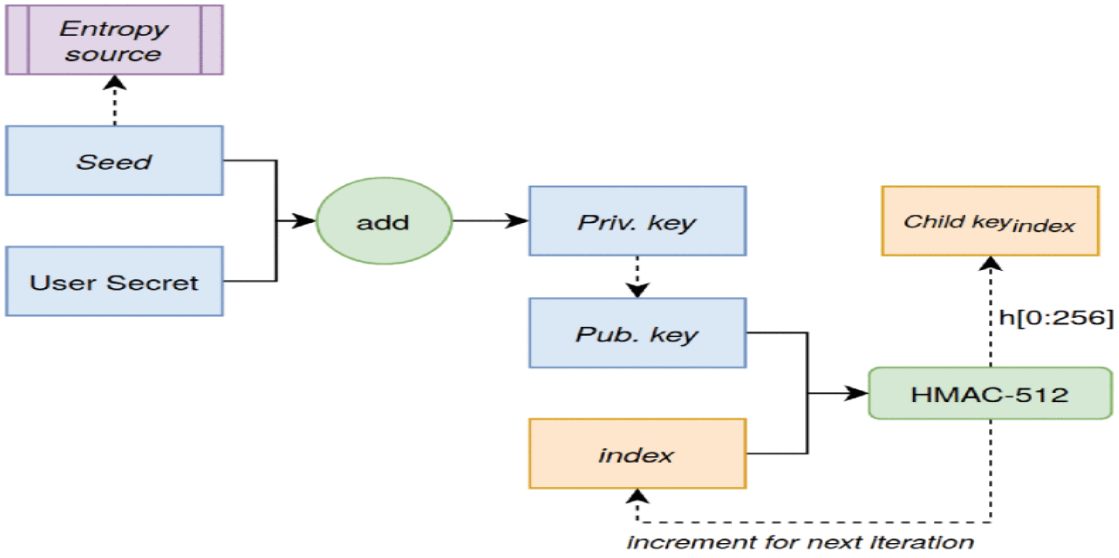

Plate 2.6: Burnable pseudo-identity creation stage (Gutierrez-Aguero et al., 2021)

The authors in (Abichandani, Lobo, Kabrawala & McIntyre, 2021) carried out an experimental validation of an Ethereum blockchain-based software and hardware architecture that enables secure communication for multiple small unmanned aerial vehicles (sUAVs). The experiments involved three DJI M100 quadrotors that shared

images captured during flight based on smart contracts created using Ethereum's Turing complete programming language. The smart contract was designed so that only the intended recipient sUAV could access a specific image. The effect of image size, difficulty level, and consensus algorithms on image transfer times during flight were noted and point to the feasibility of the developed system in practical missions.

Ashraf, Ma, Jiang & Chan (2020) presented GasFuzzer, a work on exploring the effects of gas allowance manipulation to expose gas-oriented exception security vulnerabilities. GasFuzzer consists of two phases. The first phase introduces a gas-greedy strategy to favor transactions having higher gas consumption for mutation to obtain test transactions with different gas consumptions. The second phase introduces a novel notion of fractional gas consumption coverage and a novel gas-leveling strategy. It applies them to mutate the gas allowances of some of the transactions resulting in the highest gas consumptions produced in the first phase followed by applying the allowance-mutated transactions together with those which remained non-mutated to fuzz test the smart contract. The authors reported an evaluation of GasFuzzer via an experiment on 3170 real-world smart contracts deployed on the public Ethereum Blockchain. The findings show that GasFuzzer with gas-greedy strategy can detect 7 more Exceptions Disorder security vulnerabilities than the state-of-the-art black-box fuzzer, and GasFuzzer with gas-leveling strategy and gas coverage criterion can detect 6 additional cases of Exceptions Disorder security vulnerabilities.

Kim, Jang, Park and Lee (2021) designed, implemented, simulated, and validated error-correction code proof-of-work (ECCPoW) algorithm for Ethereum, called ETH-ECC. In the implementation, they explained how ECCPoW algorithm has been integrated into Ethereum 1.0 as a new consensus algorithm. Furthermore, they devised and implemented a new method for controlling the difficulty level in ETH-ECC. In the simulation, they tested the performance of ETH-ECC using a large number of node tests and demonstrated

that the ECCPoW Ethereum works well with automatic difficulty-level change capability in real-world experimental settings.

Jang et al. (2020) conducted a study to systematically determine the fundamental causes of problems that users encounter when they interact with blockchain technology, one of the promising emerging technologies today, and to suggest relevant design strategies. To this end, usability evaluation was conducted for a decentralized exchange application. To ensure the effective identification of the significant usability problems, heuristic evaluation with four experts and usability testing with 23 experimental participants were carried out. The results obtained show that more user-centered design is necessary to enable the widespread use of decentralized applications. Thus, actionable design strategies that facilitate the effective utilization of emerging technologies were suggested.

Attaran & Gunasekaran (2019) explored the changing dimensions of blockchain, highlighted the importance of this technology, reviewed its timeline from inception to maturity, identified determinants of implementation success, and covered some of the potential benefits of this technology. Finally, the study highlighted the successful implementation of blockchain solutions in the manufacturing and service industries.

Seidel (2018) introduced the basic concept of distributed trust, described some early instances, and highlighted how organizational theories need to be updated to no longer rely upon fundamental assumptions about trust which are becoming outdated. Also, it was revealed that distributed trust is a very fertile ground for organizations research as the core tenet of the field – what roles and functions should group together within an organization - is being called into question at the most fundamental level.

The work in (Shermin, 2017) emphasized blockchain potentials for innovation in all industries, particularly the financial industry, and in a more general way, for organizations. Blockchain system was modelled using the game theory. It was also

revealed that blockchain technology challenges existing organizations, and through its characteristics has an impact on different costs as well as on the design of contracts.

2.2.6 Copy-Sensitive QR Codes

The authors (Tkachenko, Kucharczakr, Destruel, Strauss & Puech 2018) proposed copy-sensitive graphical codes made of particular patterns. The two Level QR (2LQR) code uses specific textured patterns in order to ensure the sensitivity to duplication process. The authentication test of this code is based on comparing the correlation values between the original and the printed-and-scanned codes with a pre-determined threshold. The authors propose to reduce the number of false-negative results of the authentication test by using a very competitive Super-Resolution (SR) technique. The experimental results show the significant improvement of correlation values when using the images printed-and-scanned once, without increasing the correlation values of duplicated codes. Therefore, the 2LQR code copy sensitivity is not affected by the suggested quality improvement process.

A similar work was carried out by Tkachenko and Destruel (2018) in which they studied the security of two level QR code which is constructed using specific textured patterns that are sensitive to print-and-scan impact. Such code is a good candidate because it generalizes several concepts from several codes. The authors took a falsifier point of view that aims to reconstruct the two level QR code and to fool the authentication system detector. As the two level QR code contains sets of the same textured patterns, the opponent has access to different printed-and-scanned versions of these textured patterns. These sets of patterns were used for structure estimation. The results show that the increasing number of printed-and-scanned patterns cannot improve the estimation results.

Yadav, Tkachenko, Trémeau and Fournel (2019) focused on estimation of copy sensitive graphical codes by using neural networks, which is an in-trend approach. They tested a state-of-the-art architecture efficiency in the binarization of handwritten characters. They used some random binary codes for their experiments. In order to estimate the original

code, they made use of an existing auto-encoder based architecture called Selectional Auto-Encoder (SAE). SAE are fully convolutional network without any fully connected layers. It consists of 5 layers of encoder and 5 layers of decoder. The hierarchy of layers in the encoder part of SAE consists of series of convolution and down-sampling operations till the hidden layer h, whereas in the decoder part it consists of series of convolution and up-sampling operations till reconstruction of the image to its original size. The results show that a small number of samples can be efficiently exploited in order to estimate the original structure of the digital CSGC.

Johar (2020) developed an application of QR code scanner that can scan the QR code whether it is original or copy. In addition, the research also developed a system of QR code generator, and the system generates the code to become Dynamic QR Code.

The authors (Huang, Zhou, Chang & Xie, 2018) proposed an optimized cyclic weight (OCW) decoding algorithm based on novelty ideas. The OCW algorithm utilizes the properties of quadratic residue (QR) codes and the advantages of the syndrome-weight algorithm to facilitate fast decoding of the binary systematic (47, 24, 11) QR code. The memory size of the OCW algorithm was reduced to only 0.45% of that in the original cyclic weight algorithm. In addition, it was carried out in simple and repeatable steps, which is of great importance for hardware implementation. The synthesized results in ALTERA QUARTUS V17.0 show that the proposed architecture operates at high throughput and low latency.

Iliyasu, (2019) proposed delineating an encryptable area (EA) that excludes key areas needed to retain physical appearance and properties of an innocuous QR Code. Further, they adduce a zoning structure that demarcates the EA into seven zones. Their analysis show that careful adulteration of contents of at least two zones are enough to produce encrypted versions of the QR codes. Second, each zone is partitioned into m tiles, each a 3×3 sub-block and then local interactions emanating from the occupancy (or strength) of the tiles are used to determine the composition of first-and second-tier rulesets. Third, to

steer the evolution of the QR Codes, they proposed the use of zone and cell-wise dextral boundary conditions (DBC) that combine a troika of cells permeating contents of a tile at state t to determine the left-most cell entry at state t+1 of its evolution. Further, they impose a pixel-wise constraint that ensures that each encrypted tile has a discordance that is no less than the in-built error correction tolerance of the code. This property guarantees adequate scrambling of the QR code to mitigate unauthorized access to it and the information it conceals. Meanwhile, considering the properties (balanced, linear and reversible) and nature of their rulesets, the proposed protocol recovers QR codes that are seamlessly scannable as conduits leading authorized users to confidential information. They validated the protocol by implementing both the encryption and recovery procedures on different versions of QR Codes.

A novel data hiding method for covert communication via the QR code image based on adjustments of the shapes of the QR code modules using image processing techniques was proposed by Wu D. C. and Wu, Y. M. (2020). A module block consisting of two module pairs was taken as the unit for message-bit embedding by adjusting the vertical and horizontal internal boundaries in the module pairs. Eliminations of the resulting boundary line zigzaggedness and irregular overlapping and holing phenomena in the module block were also carried out. The resulting stego-image with secret bits embedded can be scanned by a barcode reader to obtain the facial data recorded in the QR code while a program implementing the message extraction process of the proposed method can process the stego-image to extract the hidden secret message. Good experimental results, tests of stego-image readability by QR code scanners and resistance to noise attacks, and a comparison with other methods from the viewpoint of data embedding rate show the feasibility and superiority of the proposed method for real covert communication applications.

The authors in (Huang, Chang, Li & Liu, 2020) proposed a new secret message embedding scheme to embed sensitive message in the public message of a meaningful

cover QR code. The new embedding mechanism designs a data bit flipping rule to hide a hexadecimal secret digit into data codeword based on (8, 4) Hamming code. The fault tolerance of cover QR code would correct the errors caused by the embedding procedure. The valid and meaningful marked QR code would divert people's attention. Experimental result shows that, the proposed secret embedding scheme achieves a high secret payload, and its embedding efficiency is close to 2.9, which is much better than the state-of-art works.

The work (Ali & Farhan, 2020) provided a novel method to improve the data storage of a quick response code (QR code) by applying encrypted lossless compression technology. A key aspect of the work was to propose a new methodology to overcome the weaknesses of the limited size of the traditional QR code, which has long been an important issue in a wide range of areas. The new algorithm incorporated a simple technique for overcoming this difficulty by inserting confidential information into a QR code message. The QR code was updated through the addition of levels that help to share secure messages of various sizes and to authenticate documents for verification and validation. In the work, the newly proposed QR code does not reconstruct the configuration or structure of the QR code. Rather, it provided better security because it relied on the features of the Huffman compression algorithm to reduce the size of the input data and the principles of encryption through the XOR function, which is done through a variable encryption key. The experimental results show the superiority of the new method over previous methods.

In order to solve the security problem of QR code payment, Zhou, Hu, Zhang and Cai (2021) designed dynamic QR code payment system that supports SM2, SM3, and SM4 cryptographic algorithms, and generates dynamic QR code information in real time during the transaction process. Through dynamic algorithm distribution, the randomness and uniqueness of QR code generation are guaranteed, and it is suitable for multi-scene application transactions. The algorithm correctness test result shows that the system has

achieved the expected effect. The performance test results show that the hardware of the security module implements the algorithm flow and improves the payment performance. Compared with some other algorithms, the processing time is shorter, the running speed is faster, and the system is more secured.

Liu et al. (2019) introduced a novel QR code with three-layer information that utilizes the characteristics of the Hamming code and the error correction mechanism of the QR code to protect the secret information. The first layer information, that is, the public information, is decoded by a standard QR reader. The XOR operation is performed on all shares to obtain the second layer information. Finally, the pixels of the same position in all shares are taken as a set of Hamming codes, respectively, and the third layer information was extracted by a matrix multiplication operation. Compared to other related schemes, the proposed scheme has the advantage of high information embedding efficiency and strong robustness against common image post-processing attacks.

In this work, (Alajmi, 2020), presented a steganographic system that uses the container not only to hide the payload, but also to give misleading information to the adversary. To achieve this goal, he used quick response (QR) code as a container. QR codes generated by the proposed system can carry its ordinary message in addition to the payload. Anyone can read the message, but the payload can only be obtained using a secret key. The message and the payload are unrelated; i.e., any message can be generated regardless of the payload and vice versa. The authors can take advantage of that by generating a message that gives misleading information to the adversary. The proposed system was tested and the result shows that the generated QR code is valid i.e., indistinguishable from an ordinary QR code.

Fu, Cheng and Yu (2018) proposed a (k, n)-VCS combining with QR codes. To enlarge the allowable maximum size of secret image, a probabilistic sharing model is utilized. Based on it, a secret sharing method is presented with high relative difference. Furthermore, they embedded the initial shares into cover QR codes by using encoding

redundancy. After that, each share is meaningful and can be read by any standard QR code reader. Different from previous work, error correction capacities of the covers are perfectly preserved. Finally, experimental results and comparisons are provided to show the feasibility and advantages of the proposed scheme.

2.2.7 Product Inherent Features

In Wei, Lin and Liao (2019), person re-identification (ReID) was addressed by learning an inherent feature representation (inherent code) that is unique to each individual. They proposed new learning objectives to learn the inherent code for each person based on deep learning. Specifically, the proposed deep-net model was trained by jointly optimizing the multiple objectives that pulls the instances of the same person closer while pushing the instances belonging to different persons far from each other. Owing to such complementary designs, the deep-net model yielded a robust code for each individual and hence better solved person ReID.

The explicit roles of robot skin in satisfying the escalating demands of features on inherent safety, sensory feedback, natural interaction, and energy autonomy were analyzed in (Pang, Yang & Pang, 2021). Furthermore, a comprehensive review of the recent progress in functionalized robot skin in components level, including proximity, pressure, temperature, sensory feedback, and stiffness tuning, was presented. Results show that the co-design of these sensing and actuation functionalities will enable robot skin to provide improved safety, intuitive feedback, and natural interfaces.

Pinto, Cardoso and Lourenço (2018) conducted a deep review and discussion of 93 state-of-the-art publications on their proposed methods, signal datasets, and publicly available electrocardiogram (ECG) collections. The extracted knowledge was used to present the fundamentals and the evolution of ECG biometrics, describe the current state of the art, and draw conclusions on prior art approaches and current challenges. This would inspire and guide future research in ECG biometrics.

Yang et al. (2020) propose a method for wafer defect detection. The novel method includes two phases, namely wafer segmentation and defect detection. In wafer segmentation phase, the target wafer image was segmented based on the affine iterative closest algorithm with spatial feature points guided (AICP-FP). In wafer defect detection phase, with the inherent characteristics of wafers, a simple and effective algorithm based on machine vision was proposed. The simulations demonstrate that, with these two phases, the higher accuracy and higher speed of wafer defect detection can be achieved at the same time. For real industrial system, this novel method can satisfy the real-time detection requirements of automatic production line.

Lin et al. (2019) presented a new method of cross fusing feature, named multi-semantic pyramids (MSP), for detecting different-scale objects. Various scale objects were predicted, respectively, by corresponding semantic pyramids (SP), each SP can produce rich semantic features for predicting via reusing inherent multi-scale feature maps from the network backbone. Through promoting the reuse of inherent feature layers, MSP improved detection performance with marginal extra cost. In addition, since the reuse connection of the MSP facilitates the conduction of the gradient, the convergence of the network was greatly improved.

Sreeshma, Manu and GopaKumar (2018) identified an optimal sequence structure combination for computational analysis of IncRNAs. They also proposed a novel secondary structure quantization which consider the existence of various structure elements. The feature combinations were used as input to classification of IncRNAs from coding RNAs, and significant improvement in the results were obtained.

Wang, et al. (2019) proposed a novel feature extraction method known as geometric algebra based oriented fast and rotated brief (GA-ORB), for multispectral images based on the theory GA. First, the scale information in both spectral and spatial spaces of multispectral images was obtained in GA, where the inherent spectral structures were retained successfully. Then, referring to the ORB, the images were computed in different

scales and the interest points were detected and described in the GA space. The experimental results show that the GA-ORB outperformed some previous algorithms with respect to distinctiveness and robustness in extracting and matching interest points, and it could be computed much faster.

Jakubović and Velagic (2018) considered a problem of feature matching and object detection in two images using brute-force matchers. The proposed framework exploited several concurrent algorithms for feature detection and descriptor extraction, such as ORB (Oriented FAST and Rotated BRIEF), BRISK (Binary Robust Invariant Scalable Keypoints), SIFT (Scale Invariant Feature Transform) and SURF (Speeded-Up Robust Features). The feature matching was accomplished by the Brute-Force approach combined with the k-Nearest Neighbors algorithm. The obtained matches were utilized by the robust RANSAC (Random Sample Consensus) method for estimating the transformation between two consecutive images. Therefore, the RANSAC method is employed to improve the outliers removal. The proposed algorithm was designed and implemented using OpenCV library. Its effectiveness and quality were verified through analyses of its execution speed and accuracy of the feature matching.

Karami, Prasad and Shehata (2017) compared the performance of three different image matching techniques, i.e., SIFT, SURF, and ORB, against different kinds of transformations and deformations such as scaling, rotation, noise, fish eye distortion, and shearing. For this purpose, they manually applied different types of transformations on original images and computed the matching evaluation parameters such as the number of key points in images, the matching rate, and the execution time required for each algorithm.

A novel fast image stitching algorithm based on Oriented FAST and Rotated BRIEF (ORB) features was proposed by Wang et al. (2017). The algorithm firstly selects ORB algorithm which adds the direction information to the FAST detector for image feature extracting and matching. Secondly, Random Sample Consensus (RASANC) algorithm

was used to eliminate false matching points. Finally, weighted average method was used to speed up the image fusion.

Based on Speeded-Up Robust Features (SURF) SURF and the theory of Geometric Algebra (GA), a novel feature extraction algorithm named GA-SURF was proposed for multispectral images by Wang et al. (2019). First, a Hessian matrix based on GA was calculated for locating interest points in spatial and spectral space. The box filters based on GA were used to simplify the calculation of Hessian matrix and generate image pyramids. Then, following the procedures of SURF, interest points were located by the image pyramids and described in GA space. Experimental results demonstrate that, compared with other feature extraction algorithms for multispectral images, GA-SURF can be computed much faster and are more robust and distinctive.

2.2.8 Decentralized Blockchain Applications (dApps)

The authors in (Cai et al., 2018) traced the development of blockchain systems to reveal the importance of decentralized applications (dApps) and the future value of blockchain. They also surveyed the state-of-the-art dApps and discuss the direction of blockchain development to fulfill the desirable characteristics of dApps.

Yiu et al. (2021) carried out a compact and concise survey on the state-of-the-art research of decentralizing applications with blockchain in the 5G and beyond perspective. They provided four burning 5G and beyond challenges and discuss five aspects of motivation for decentralizing applications with blockchain. Particularly, they presented the capabilities of blockchain for decentralizing applications through reviewing dApps for 5G and beyond. Finally, they clearly distinguished three blockchain paradigms and discuss how developers can make right choices for 5G and beyond.

The work in (Trojanowska, Kedziora, Hanif & Song, 2020) considered Ethereum decentralized design process issues, starting from the methodology used, going through the description of requirements and specification, ending up with the implementation. It compared guidelines from Ethereum Smart Contract Best Practices by ConsenSys, Smart

Contract Security Verification Standard created by SecuRing, Security Considerations from Solidity documentation, Ethereum Smart Contracts Security Recommendations from Guylando Knowledge Lists, and Smart Contract Weakness Classification and Test Cases. Finally, an overview of the issues associated with the security of Ethereum decentralized applications as well as how they can be detected during security verification phase, and countermeasures to address them were presented.

The research (Zichichi, Ferretti & D'Angelo 2021) developed MOVO, a decentralized application (dApp) for smart mobility. The dApp consists of an Android application intended for use inside a vehicle, which helps the user/driver collect contextually generated data (e.g., a driver's stress level, an electric vehicle's battery level), which can then be shared through the use of Distributed Ledger Technologies (DLT). MOVO dApp includes the following three modules: a module for collecting data from vehicles and smartphones sensors; a component for interacting with DLT, and Decentralized File Storages (DFS) for storing and validating sensor data; and a module for "offline" interaction between devices. Finally, the research provided an experimental evaluation that confirms the viability of the MOVO dApp in real mobility scenarios.

Gao (2019) presented a guided and automated framework called Sungari to test dApps. The insights behind Sungari are two-fold. First, it employes random events to infer an abstract relation between frontend and blockchain. Second, it uses the relation to generate event sequences in a guided manner that can cover blockchain smart contracts as quick as possible. In a real-world dApp case study, Sungari outperformed random approach by covering 33% more application use cases.

2.2.9 Blockchain Explorer

Kuzuno and Karam (2017) proposed a practical bitcoin analytical process and an analyzing system that stands alone and manages all data on the blockchain in real-time with tracing and visualizing techniques rendering transactions decipherable and useful for law enforcement investigation and training. The system adopts combination of analyzing

methods that provides statistics of address, graphical transaction relation, discovery of paths and clustering of already known addresses. The system was evaluated in the three criminal cases which includes marketplace, ransomware and DDoS extortion. Then, it was determined that the proposed system could help investigation process and training.

The authors in (Lee et al, 2019) proposed a monitoring system and an explorer to show the results collected from the system. They considered that though blockchain technology provides transparency because all participants have distributed ledgers with the same data, the anonymity of blockchain can be exploited for illegal trades in the Darknet market. Therefore, a monitoring system for transactions occurring on the blockchain was needed.

Zhong, Wei, Xu, Zhao, Zhou, Luo and Shi (2020) introduced a new online cryptocurrency transaction data viewing tool called SilkViser. Guided by detailed scenario and requirement analyses, they created series of appreciating visualization designs, such as paper ledger-inspired block and blockchain visualizations and ancient copper coin-inspired transaction visualizations, to help users understand cryptocurrency transaction mechanisms and recognize advanced transaction information. They also provided a set of lightweight interactions to facilitate easy and free data exploration. Moreover, a controlled user study was conducted to quantitatively evaluate the usability and effectiveness of SilkViser. Results indicate that SilkViser can satisfy the requirements of NUsers and EUsers.

Heo and Shin (2021) argued that the main chain block data, usually available from block explorer services, does not serve as a sufficient source of transaction and block dynamics that are only visible from a large-scale event measurement. Thus, they measured the transaction and block arrival events of the two popular public blockchains, i.e., Bitcoin and Ethereum, to investigate the hidden dynamics of blockchain networks. They discovered an invalid transaction propagation problem that can be exploited to launch a Denial-of-Service attack on a network.

2.2.10 Testing Blockchain-Enabled Supply Chain

Wang et al. (2019) proposed a product traceability system based on blockchain technology, in which all product transferring histories are perpetually recorded in a distributed ledger by using smart contracts and a chain is formed that can trace back to the source of the products. In particular, they designed an event response mechanism to verify the identities of both parties of the transaction, so that the validity of the transaction can be guaranteed. And all events are permanently stored in the form of logs as a basis for handling disputes and tracking responsible entities. Furthermore, a system prototype was constructed based on the testing framework of Truffle. The contract code was deployed on a test network, TestRpc (that runs in local memory) and a decentralized web page interface was implemented based on the prototype. Finally, the system security analysis and experimental results show that the solution is feasible.

The research (Sayyad et al., 2021) developed the prototype for identification of counterfeit products using blockchain technology. The system prototype is a distributed application (DApp) with a supporting blockchain network. The network was developed on hyper-ledger fabric and uses DPoS/PBFT consensus algorithm by default. Finally, it presented the software implementation process in which the product code is scanned using the developed application and then verify if the given product is counterfeit or not. The architecture of the developed system is shown in Plate 2.7.

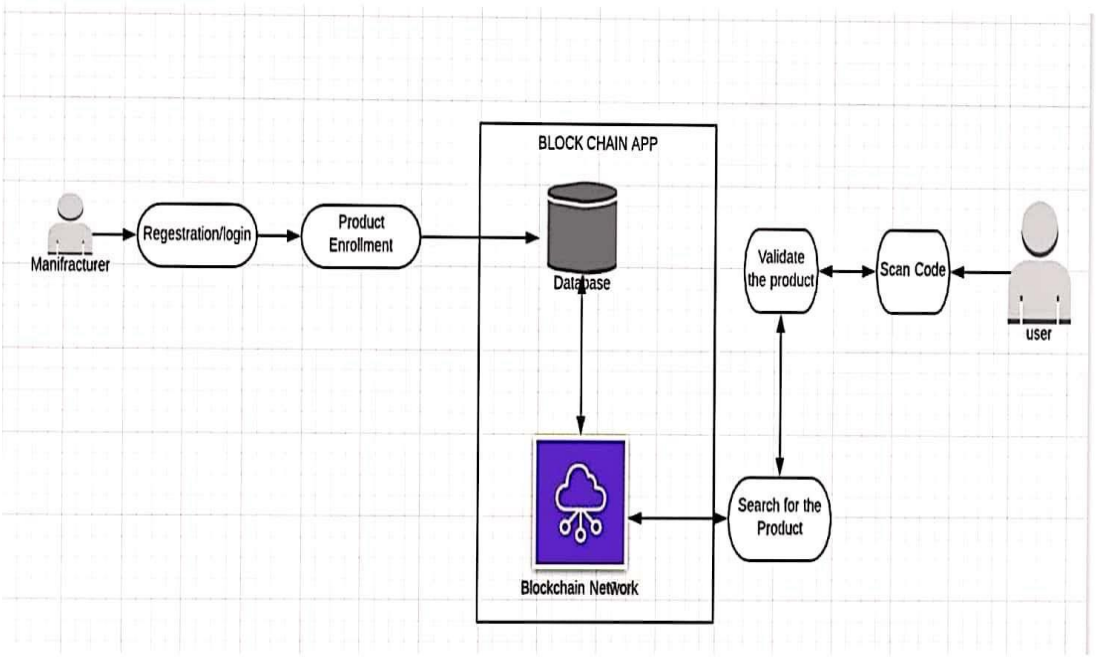

Plate 2.7: Architecture of Sayyad system (Sayyad et al., 2021).

The author (Daoud, 2019) described a decentralized product certificate information that ensures consumer inform themselves about product specification especially the validity information of the certificate. He implemented a protocol that turns a Blockchain into an automated access-control. Transactions in the system are used to carry data, such as storing, querying and sharing product certificate data. Finally, he discussed possible future extensions to Blockchains that could harness the novel technology into a well-rounded solution for trusted computing problems in society.

The research (Ma et al., 2021) adopted the decentralized Blockchain technology approach to ensure that consumers do not fully rely on the merchants to determine if products are genuine. In that way, manufacturers can use this system to provide genuine products without having to manage direct-operated stores, which can significantly reduce the cost of product quality assurance. The proposed system utilized Ethereum as the back end Blockchain operating system, and used Ethereum's proprietary programming language

Solidity as the high-level programming language for writing smart contracts. The basic model of the system is shown in Plate 2.8.

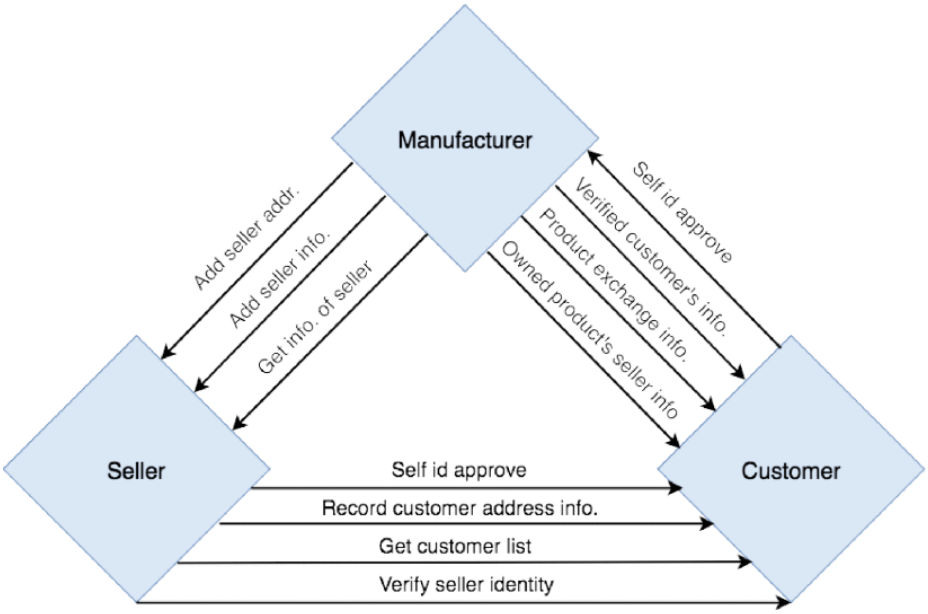

Plate 2.8: Basic model of Ma system (Ma et al., 2021).

Srivatsa, Aakash and Sahisnu (2021) proposed and implemented a system for product authentication using blockchain technology based on Ethereum platform by making use of smart contracts. The actual implementation of the smart contracts and all the core logic was done using Solidity. Writing transactions to the Ethereum blockchain requires a transaction fee called gas. To test the working of the application, ganache and truffle were used. Interaction with smart contracts was done using truffle console(command), which provides a platform for calling functions and verifying the state inside the smart contracts.

Yiu (2021a) performed a thorough security analyses on an existing product anti-counterfeiting and traceability system—NFC-Enabled Anti-Counterfeiting System (NAS). Tag-cloning, tag-disabling, tag's data modification, man-in-the-middle relay attack, spoofing attack on product data, and denial-of-service were identified as being among the riskiest threats to NAS as well as other existing anti-counterfeiting and

traceability systems built on a centralized architecture. Based on the findings and opportunities identified from the analyses, key areas of decentralization, fundamental system requirements, and feasible mechanisms for developing a decentralized product anti-counterfeiting and traceability system were defined. It was also established that with the use of peer-to-peer blockchain networks and distributed storage technologies, it will be possible to eliminate an absurd amount of cost for on-chain storage and provide a much higher level of privacy, reliability, and quality of service compared with existing anti-counterfeiting and traceability solutions with centralized architecture.

In (Yiu, 2021b) proof-by-demonstration research method was adopted to develop Decentralized NFC-Enabled Anti-Counterfeiting System (dNAS). The decentralized system components of dNAS actually consist of a blockchain network, smart contracts, an InterPlanetary File System (IPFS) network and a blockchain service. The dNAS blockchain network was developed based on the Go-Ethereum PoA implementation. The network implementation started by creating Ethereum accounts for each of the blockchain nodes with a key pair, and its key-store file was generated during the process. The smart contract source code, deployed to the blockchain network, was implemented in Solidity, with Truffle framework as its development environment and automated testing library. Performance analysis prove that the developed dNAS is a secure and immutable scientific-data provenance tracking and management platform on which provenance records, providing compelling properties of the data integrity of luxurious goods, are recorded, verified and validated automatically.

The work (Yiu, 2021c) carried out performance evaluation on Decentralized NFC-Enabled Anti-Counterfeiting System (dNAS) in areas such as the decentralized record management, data and process integrity, system security, as well as availability and integration. Discussion of results was also covered to provide qualitative and quantitative system analysis, benchmarking different existing enterprise blockchain protocols to the novel dNAS developed, with risks and limitations identified for creating potential

opportunities on future development and advancement of dNAS. Based on the system evaluation, it was determined that decentralized solutions, such as dNAS, applied in supply chain anticounterfeiting and traceability are more effective and useful as expected.

Zhu et al. (2020) designed an access control policy model based on smart contract to prevent medication information from being altered or disclosed at nodes of a blockchain. The system architecture includes three parts: blockchain network environment, a smart contract, and a web client. A point-accumulation upgrade/downgrade mechanism was introduced to improve the consensus mechanism. Python 3.7 in the Anaconda Spyder environment was used to code the algorithm to simulate the medication information storage and traceability process along the medication supply chain. Two kinds of smart contract vulnerability security checking tools – "SmartCheck" and "Oyente" – were utilized to scan possible vulnerabilities and errors in the contract code, consequently eliminating the weaknesses in the code. Simulation results show that efficiency and security were enhanced by the improved consensus algorithm and access control mechanism. Consequently, the developed system could render high level of security and privacy protection which is critical to the integrity of a medication information management system.

Ehioghae, Idowu and Ebiesuwa (2021) developed an enhanced drug anti-counterfeiting and verification system, using blockchain. Two germane smart contracts – shipDrug and receiveDrug – were defined to serve as a means of safely moving drugs from one supply chain actor to the other, to secure the drug supply chain from the infiltration of counterfeit drugs. Hyperledger Fabric was used for the implementation of the proposed system. The backend Application Programming Interface (API), which interacts directly with the smart contracts on the blockchain network on behalf of the user, was also written in NodeJS. The user interface was written with HTML, CSS, and JavaScript. Hyperledger Caliper, a blockchain performance benchmark framework, was used to evaluate the system based on key metrics such as transaction latency, transaction throughput, and

resource utilization (CPU and Memory). The result revealed that an optimal throughput of 46 drug verifications per second was obtained, with less utilization of CPU and memory resources.

2.3 Identified Research Gaps

Based on the literatures reviewed so far, the following research gaps, associated with state-of-the-art anti-counterfeiting solutions, were identified.

i. None of the blockchain-based product anti-counterfeiting systems has employed product inherent features.
ii. The works reviewd did not utilize copy sensitive Quick Response (QR) code.
iii. Existing blockchain-based systems do not make use of efficient consensus algorithm.
iv. None of the existing systems utilized location information i.e., GPS coordinates in combination with blockchain technology.

This work seeks to address these identified research gaps by pursuing enhanced security, decentralization, immutability, efficiency and optimal quality of service as well as low-cost while achieving the specified objectives of this research.

CHAPTER THREE

MATERIALS AND METHOD

3.1 Materials

Materials utilized in this work are categorized into two: software materials and hardware materials.

3.1.1 Software Materials

The software materials used in this work are:

(i) Ethereum Virtual Machine

(ii) Solidity compiler

(iii) Modelio

(iv) Truffle suite

(v) Hypertext Markup Language 5 (HTML5)

(vi) Cascading Style Sheets (CSS)

(vii) Visual Studio Code (VS Code)

(viii) MetaMask extension

(ix) Solidity Extension

(x) Node.js

(xi) React

(xii) Kotlin

3.1.2 Hardware Materials

The hardware materials used in this work are:

(i) Three computer systems

(ii) Two Android phones

(iii) Three different products

(iv) Printer

3.2 Method

In this work, a composite methodology has been adopted. It involves:

i. Object-Oriented System analysis and design utilizing UML patterns.

ii. Blockchain development techniques adopting a combination of 'Throwaway Prototype' and 'Rapid Unified Process (RUP).'

3.2.1 Modeling of the Blockchain-Based Anti-Counterfeiting System (BBACS)

Having considered the nature of this research and the advantageous features of object-oriented programming such as data abstraction, encapsulation, scalability, code reusability, polymorphism, etc., the developed system is modelled using appropriate object-oriented analysis and design principles and patterns.

3.2.1.1 Object-Oriented Analysis of BBACS

This includes system description and functional requirements of the system.

(A) Description of BBACS

The system combines product inherent features, copy-sensitive QR Code, location information and Track and Trace technologies. It utilizes a method for detecting counterfeit by capture of inherent features indissolubly linked with the product induced by the production process itself. The specific conditions of production, manufacturing technologies and materials generate specific features which identify the product uniquely. Since a counterfeiter gains margin by the use of inferior production processes and materials, the differences between genuine product and counterfeit can be captured from these inherent features in an automated fashion.

Copy Sensitive QR code is employed by the system to detect counterfeit products. In order to enhance the security of QR code, an inherent feature of the product and a Copy Sensitive Graphical Codes (CSGC) are inserted into it. The detection of counterfeits using a CSGC relies on an "information loss principle", which states that every time a digital image is printed or scanned, some information is lost about the original digital image. A CSGC is a maximum entropy image that attempts to take advantage of this

information loss. Since producing a counterfeit CSGC requires an additional scanning and printing processes, it will have less information than an original CSGC. By measuring the information in the scanned CSGC, the detector can determine whether the CSGC is an original print or a copy. Additionally, each printer or scanner has its own signature. Using these two properties, the authentication test can distinguish the original CSGC (printed once) from copied/counterfeited (printed several times).

The system utilizes a new enhanced consensus algorithm that is, unlike existing protocols, fully decentralized and balances between efficiency and security. The system prototype is a distributed application (DApp) with a supporting blockchain network. We chose Ethereum as our backend platform due to the fact that our system would require multi-state and flexibility in block storage to record data as well as short block time.

The BBACS execution is in the following four stages.

(i) Enrolment of manufacturers and distributors on the network: The first stage in the system execution is to bring all manufacturers and distributors into the blockchain network. Manufacturers and distributors' authentication is done via registration. In so doing, each manufacturer and distributor will furnish the system with his business name, contact (e.g. phone number(s), email address and location address), location information i.e., Global Positioning System (GPS) coordinates, etc. In the end, proper identification (ID) and password is issued to each manufacturer and distributor.

(ii) Product enrolment on the network: A manufacturer will make a request to add a product on the network. The product will be enrolled on the network if the requester is a genuine manufacturer. After the product is registered on the network, it will create a smart contract and a unique code of the product in which the details of the inherent features and other parameters embedded in the QR Code for the product will be maintained in an encrypted form.

(iii) Shipping of Product: In this third stage, the manufacturer will ship the product to a distributor and the product status is set as shipped; it will not change the

ownership of the product until a request from both parties is approved to buy and sell the product. As soon as both parties approve mutually, its ownership in the blockchain network will be transferred in the form of smart contract automatically after the payment is successful.

(iv) End user gets details about product: In this final stage, end users will scan the unique QR Code embedded on the product using an App installed in an android device. The App automatically transmits information retrieved from the QR code as well as the transaction location information (i.e., the GPS coordinates) to the blockchain platform. The blockchain network compares the information received from the App with the details stored during the enrollments of the product, the distributor and the manufacturer. Based on the outcome of the comparison, the blockchain network forwards a message via the App to the end user stating whether the product is genuine or fake.

The architecture of BBACS is presented in Figure 3.1.

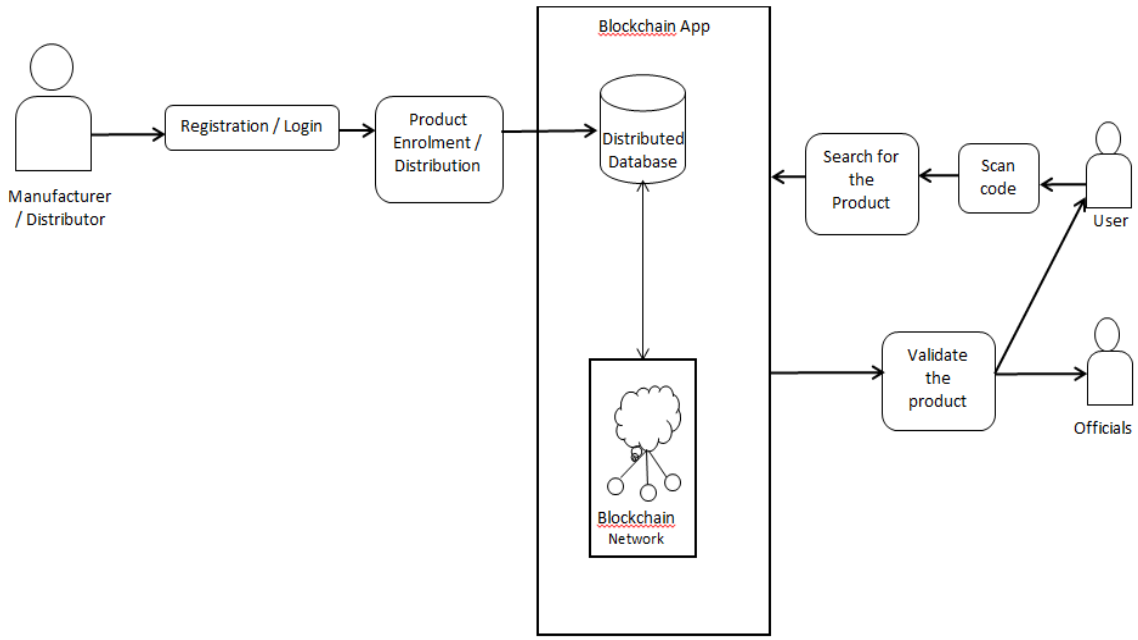

Figure 3.1: Proposed System Architecture

(B) Functional Requirements of BBACS

i. The system should allow manufacturers to register to the blockchain network.

ii. The system should allow registered manufacturers to be approved after ascertaining their authenticity.

iii. The blockchain should issue unique ID to manufacturers (distributors alike) which will enable them to log onto the network.

iv. The blockchain will allow approved manufacturers to register their products on the network.

v. The blockchain will allow distributors to register to a specific manufacturer on the network.

vi. The blockchain will allow manufacturers authenticate and approve distributors on the network.

vii. The system will enable approved distributors to record transactions (purchase of genuine products) on the blockchain network.

viii. The system will enable end users to register and access the blockchain network using the developed application.

ix. The system will enable manufacturers, distributors (and the general public) to validate the authenticity of a product using the blockchain network.

x. The blockchain will generate a copy-sensitive QR code for every product that is registered on the network.

xi. The system will enable mutual agreement between manufacturers and distributors before ownership of a product (or products) can be transferred from the manufacturer to the distributor.

xii. The system will keep track of every product that is registered on the network by monitoring the changes in their states e.g., registered, shipped, purchased.

xiii. The system will allow consumers scan the QR code of a product in order to ascertain its details and history in the blockchain network. This will proof whether or not the product is authentic.

xiv. The system will alert admins, manufacturers, distributors, end users and officials if a counterfeit product is detected.

3.2.1.2 Unified Modeling Language (UML) Representation of BBACS

The BBACS is modeled using Modelio. Modelio is an open-source Universal Modeling Language (UML) tool developed by Modeliosoft, It supports the UML2 standards. Modelio is a modeling environment, supporting a wide range of models and diagrams and providing many services facilitating the modeling of system architectures. Use Case diagrams, Activity diagrams, Conceptual Class diagrams as well as System Entities, Data, and Operations of the proposed system are designed using Modelio.

The UML representation of BBACS includes use cases, tabular presentation of major entities, data, and operations as well as conceptual class diagram, use case diagram and activity diagram.

(A) Use Cases

Use Case descriptions present the logical flow of interaction between the blockchain-based anti-counterfeiting system and its users for complex activities described under the subheading “Description of BBACS” in section 3.2.1.1. These Use Cases which interpret the proposed system in clear and concise terms are presented in Tables 3.1 to 3.6.

Table 3.1: Use Case for Registering Manufacturer

Use Case Number: 1	**Use Case Name**: Register Manufacturer
Precondition: Manufacturer approaches the system to be registered.	**Trigger:** Manufacturer attempts to enroll in BBACS network
Description: This use case details the steps involved in registering a manufacturer.	**Primary Actor**: Manufacturer/Admin.
Main Flow: A. System prompts manufacturer to input basic details	

B. Manufacturer inputs basic details C. System checks basic details are correct **a.** If details are not correct **i.** Stop registration process. **ii.** Notify the manufacturer. **iii.** Go back to step A. **b.** Else if details are correct. **i.** Proceed to step D. D. System prompts Admin to authenticate the manufacturer E. System checks if authentication is successful a. If authentication is not successful i. Stop registration process ii. Notify the manufacturer iii. Go back to step A b. Else if authentication is successful i. Proceed to step F. F. The system completes the registration process by performing the following operations: i. System issues ID to the manufacturer ii. System stores details of the registration
Special Requirements: Company registration certificate; confirmation of office address and location information (i.e., longitude and latitude)
Post Condition: Manufacturer has successfully registered and details stored in blockchain.

Table 3.2: Use Case for Registering Distributor

Use Case Number: 2	**Use Case Name**: Register Distributor
Precondition: Distributor approaches the	**Trigger:** Distributor attempts to enroll in

system to be registered.	BBACS network
Description: This use case details the steps involved in registering a distributor.	**Primary Actor**: Distributor/Manufacturer

Main Flow:

A. System prompts distributor to input basic details

B. Distributor inputs basic details

C. System checks basic details are correct

   a. If details are not correct

      i. Stop registration process.

      ii. Notify the distributor.

      iii. Go back to step A.

      Else if details are correct.

      i. Proceed to step D.

D. System prompts manufacturer to authenticate the distributor

E. System checks if authentication is successful

      If authentication is not successful

      i. Stop registration process

      ii. Notify the distributor

      iii. Go back to step A

   Else if authentication is successful

      i. Proceed to step F.

F. The system completes the registration process by performing the following operations:

      i. System issues ID to the distributor

      ii. System stores details of the registration

Special Requirements: Business registration certificate; confirmation of business address and location information (i.e., longitude and latitude)

Post Condition: Distributor has successfully registered and details stored in blockchain.

Table 3.3: Use Case for Registering Product

Use Case Number: 3	**Use Case Name**: Register Product
Precondition: Manufacturer approaches the system to register a product.	**Trigger:** Manufacturer attempts to enroll a product on BBACS network
Description: This use case details the steps involved in registering a product.	**Primary Actor**: Manufacturer/Product.

Main Flow:

A. System prompts manufacturer to input basic product details as well as scan product texture

B. Manufacturer inputs basic details

C. System checks basic details are correct

   i. If details are not correct

   ii. Stop registration process.

   iii. Notify the manufacturer.

   iv. Go back to step A.

   Else if details are correct.

   i. Proceed to step D.

D. The system completes the registration process by performing the following operations:

   i. System issues ID to the product

   ii. System generates QR code for product

   iii. System stores details of the registration

Special Requirements: Product texture

Post Condition: Manufacturer has successfully registered a product and the details stored in blockchain.

Table 3.4: Use Case for Confirming Product Genuineness

Use Case Number: 4	**Use Case Name**: Confirm Product Genuineness
Precondition: End-user approaches a distributor to buy a product	**Trigger:** End-user (customer) attempts to confirm the genuineness of a product using BBACS network
Description: It details the steps involved in confirming the genuineness of a product.	**Primary Actor**: End-user/Product.

Main Flow:

A. End User scans the QR code embedded on the product

B. System automatically uploads product details generated from the scan to the network

C. System compares product details generated from the scan with that stored in the blockchain during product registration.

i. If details are not the same

ii. Display "Product not found on network; product is likely counterfeit"

iii. Notify the enforcement officials.

iv. Store details of the unsuccessful confirmation process

v. Go back to step A.

Else if details are the same

i. Display "Product is genuine"

ii. System stores details of the product confirmation process

Special Requirements: Scan QR code embedded on the product

Post Condition: System has successfully confirmed the genuineness of a product

Table 3.5: Use Case for Transferring Product Ownership

Use Case Number: 5	**Use Case Name**: Transfer Ownership
Precondition: Product is successfully sold out	**Trigger:** Manufacturer or distributor requests to change product ownership
Description: It details the steps involved in changing the ownership of a product	**Primary Actor**: Manufacturer/Distributor/End-User
Main Flow: A. System receives transfer of ownership request from purchaser B. System gets confirmation from vendor a. If confirmation is not received i. System aborts transfers ownership process b. Else if confirmation is received i. Proceed to step C. C. System transfers ownership to purchaser D. System stores details of transfers of ownership	
Special Requirements: Payment for product	
Post Condition: System has successfully transferred ownership to purchaser	

Table 3.6: Use Case for Creating New Block

Use Case Number: 6	**Use Case Name**: Create New Block
Precondition: Data to form a new block is available	**Trigger:** Nodes attempts to create a new block
Description: This use case details the steps involved in creating a new block	**Primary Actor**: Network nodes e.g. admin, manufacturers, distributors, etc.

Main Flow:

A. System prompts nodes to elect a signer to create a new block

B. Each node checks for node X with the highest number of products in the transactions that make up new the block

C. Nodes vote node X to be a signer

D. System checks to confirm that not less than 51% nodes voted node X

   a. If less than 51% nodes voted node X

      i. Wait for more nodes to vote

   b. Else if not less than 51% nodes voted node X

      i. Proceed to step E.

E. System assigns node X the role to create a new block

F. Node X creates a new block

G. Other nodes validates the new block

H. System stores new block in chain

Special Requirements: Signer must have the highest number of products in the transactions that make up the new block, not less than 51% vote must be given to signer

Post Condition: Signer has successfully created a new block

(B) **BBACS Entities, Data, and Operations:** Major entities, data, and operations that make up each use case are captured. These entities, data, and operations are later represented as classes (objects) which constitute the entire system and are shown in Table 3.7.

Table 3.7: System Entities, Data, and Operations

ADMIN - Admin Username - Admin Password	**MANUFACTURER** - Manufacturer Username - Manufacturer Password - Manufacturer Address - Manufacturer Phone no - Manufacturer GPS - Manufacturer email	**DISTRIBUTOR** - Distributor Username - Distributor Password - Distributor Address - Distributor Phone no - Distributor GPS - Distributor email
+ Approve manufacturer + Approve distributor + Approve product	+ Register with Admin + Approve Distributor + Register Product + Record transactions	+ Register with Manufacturer + Register with Admin + Record transactions
PRODUCT - Product ID - Inherent features - QR code - Production date - Expiry date	**END-USER** - End-User Username - End-User Password - End-User Address - End-User Phone no - End-User email	**OFFICER** - Officer Username - Officer Password - Officer Organization - Officer Address - Officer Phone no - Officer email
	+ Register via App + Record transactions	+ Investigate reports + Prosecute offenders + Informs the public
TRANSACTION - Date - Time - Location		

(C) **Conceptual Class and Use Case Diagrams of BBACS:** The conceptual class and use case diagrams of the system are shown in Figures 3.2 and 3.3 respectively.

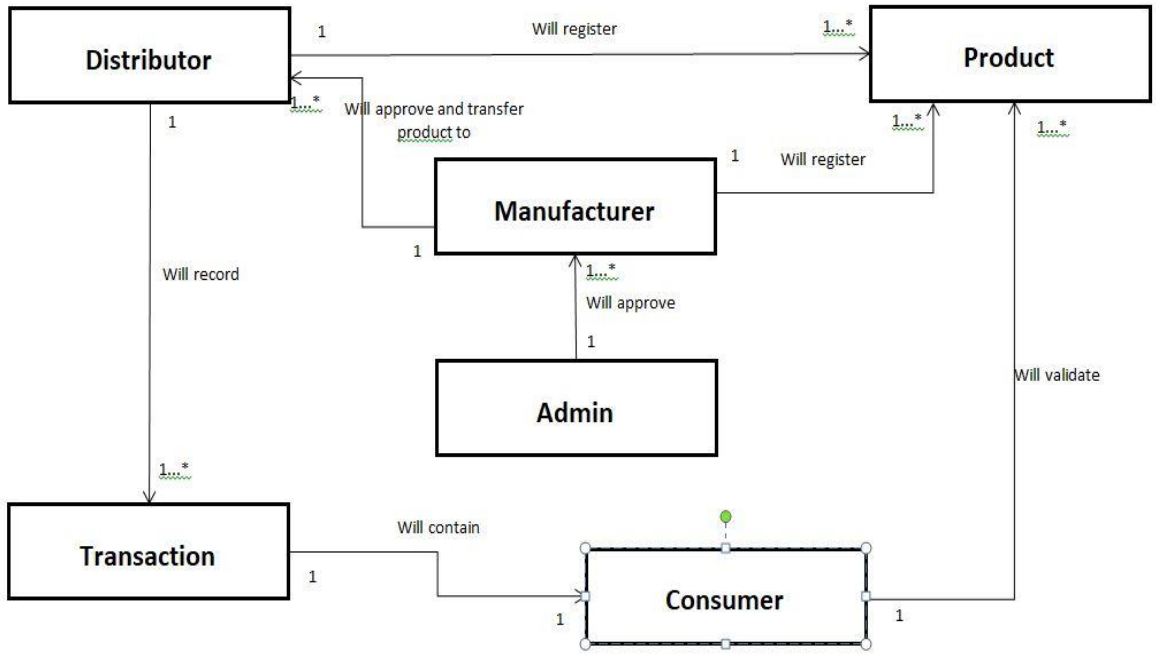

Figure 3.2: Conceptual Class Diagram of the System

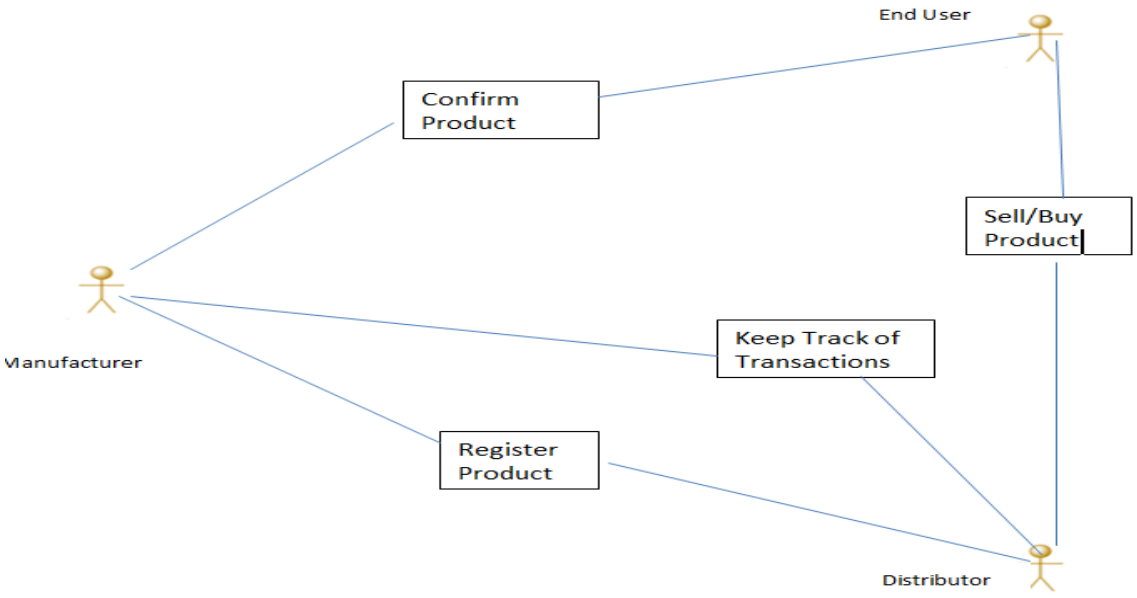

Figure 3.3: Use Case Diagram of the System

(D) **Activity Diagram of BBACS:** The activity diagram of BBACS is shown in Figure 3.4.

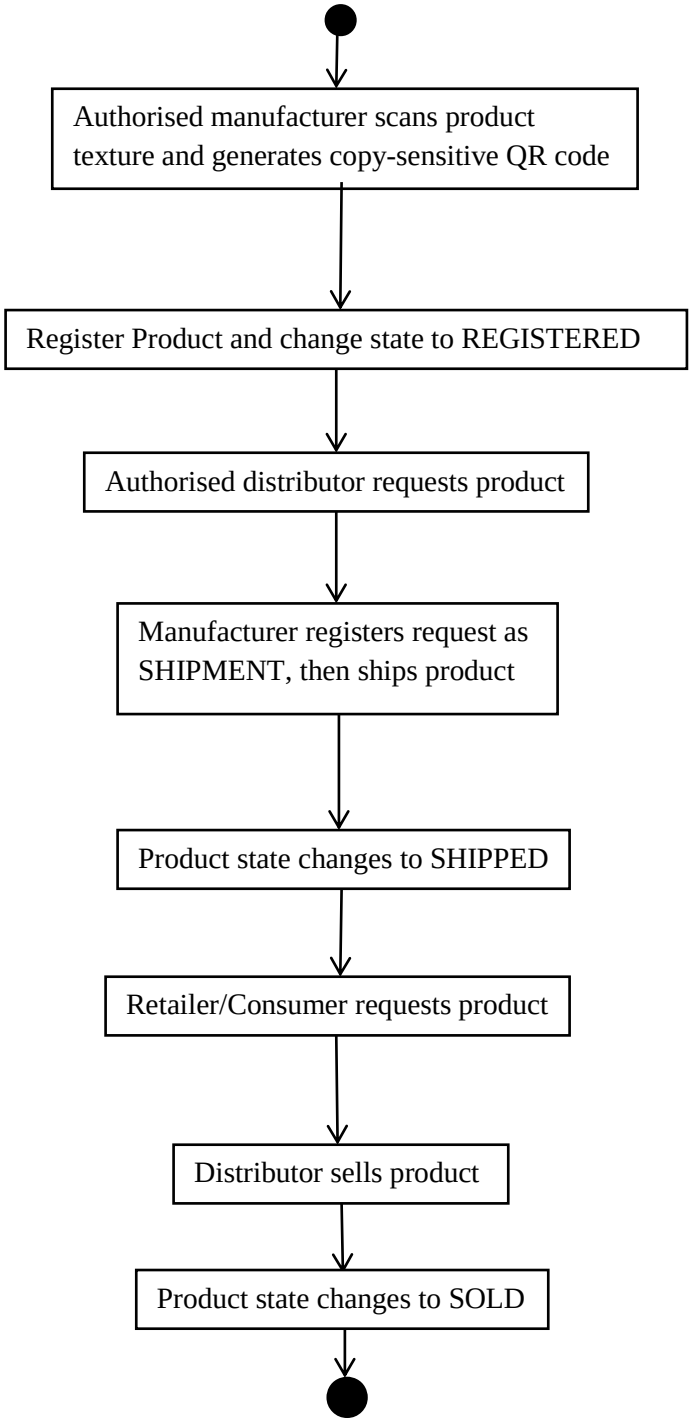

Figure 3.4: Activity Diagram of BBACS

3.2.2 Setting up a Private Ethereum Blockchain Network

In this work, Truffle suite will serve as the development environment as well as the testing framework for developing our private Ethereum blockchain network and supporting applications. One of the primary goals of Truffle Suite focuses on ensuring better accessibility in private blockchain development process. The three components of Truffle Suite namely Ganache, Truffle and Drizzle are utilized in the development of BBACS. Each component serves a distinct functionality, which empowers the overall ecosystem for developing blockchain applications.

In developing BBACS, Ethereum Virtual Machine (EVM) is deployed as the computation engine for Ethereum to manage the state of the blockchain and enable smart contract functionality. The EVM is contained within the client software (e.g., Genache) that is needed in order to run a node on Ethereum. It acts as the virtual machine which is the bedrock of Ethereum's entire operating structure. In other words it is the part of the Ethereum that runs execution and smart contract deployment. The role of the EVM is to deploy a number of extra functionalities to the Blockchain to ensure users face limited issues on the distributed ledger. Every node runs on the EVM to maintain consensus across the blockchain. The EVM is completely isolated meaning the code inside the EVM has no access to network, file system or other processes.

In setting up a private Ethereum Blockchain network for the product anti-counterfeiting system, the following steps are taken:

Step 1: Download and install Ganache

Ganache is the principal component of Truffle suite that will be utilized in setting up our personal Ethereum blockchain network.

To download and install Ganache:

i. Go to https://trufflesuite.com

ii. Click on Get started with Ganache

iii. Click on Download (Windows)

iv. Save the installation file

v. Double click on the installation file to install it.

A screenshot of successful installation is displayed in Plate 3.1.

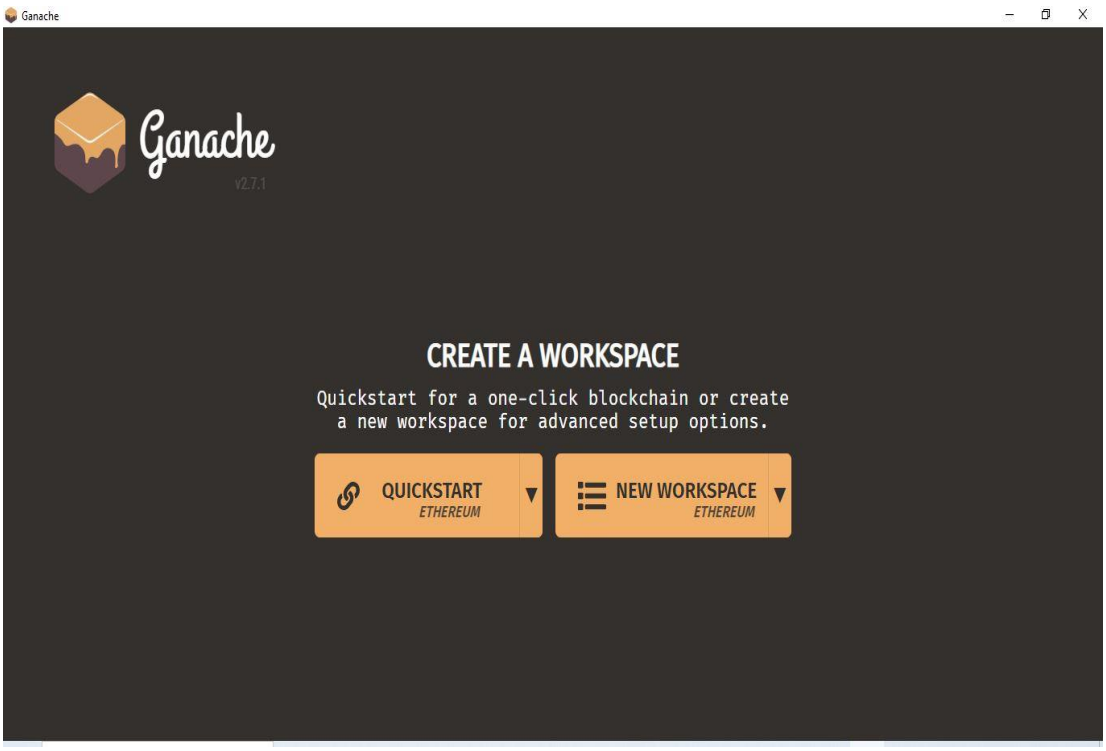

Plate 3.1: A screenshot showing successful installation of Ganache

Step 2: Download and install Visual Studio Code (VS Code)

In this work, VS code is the main programming environment where we will write, edit and debug our codes. VS Code is a lightweight integrated development environment (IDE) or source-code editor optimized for building and debugging modern web and cloud applications. It is used to code in any programming language, without switching editors. VS Code has support for many languages, including Python, Java, C++, JavaScript, and more. Its features include support for debugging, syntax highlighting, intelligent code completion, snippets, code refactoring, and embedded Git. We downloaded VS code from

http://code.visualstudio.com. Then, the installer was run. A screenshot of successful installation is displayed in Plate 3.2.

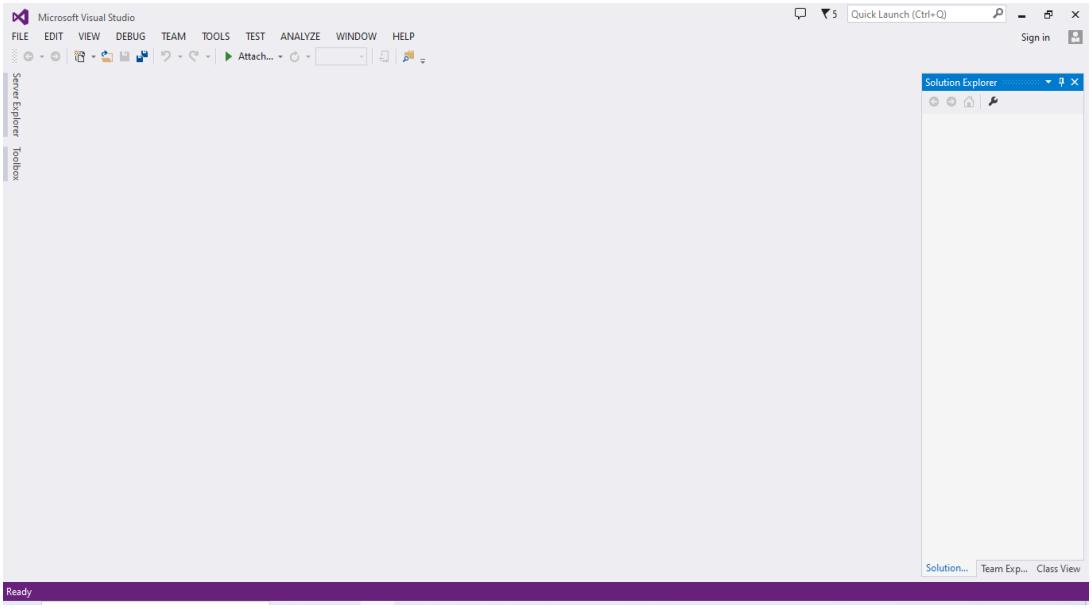

Plate 3.2: A screenshot showing successful installation of VS code

Step 3: Develop the configuration file for the Ganache network.

This is accomplished by writing a program file named “anticounterfeiting” in VS code environment. The entire source code of the configuration file is found in Appendix 1.

Step 4: Run the Ganache

This was achieved by selecting Ethereum and clicking on New Workspace. Thereafter IP addresses were configured for connecting network nodes. A screenshot of successful running of Ganache is displayed in Plate 3.3.

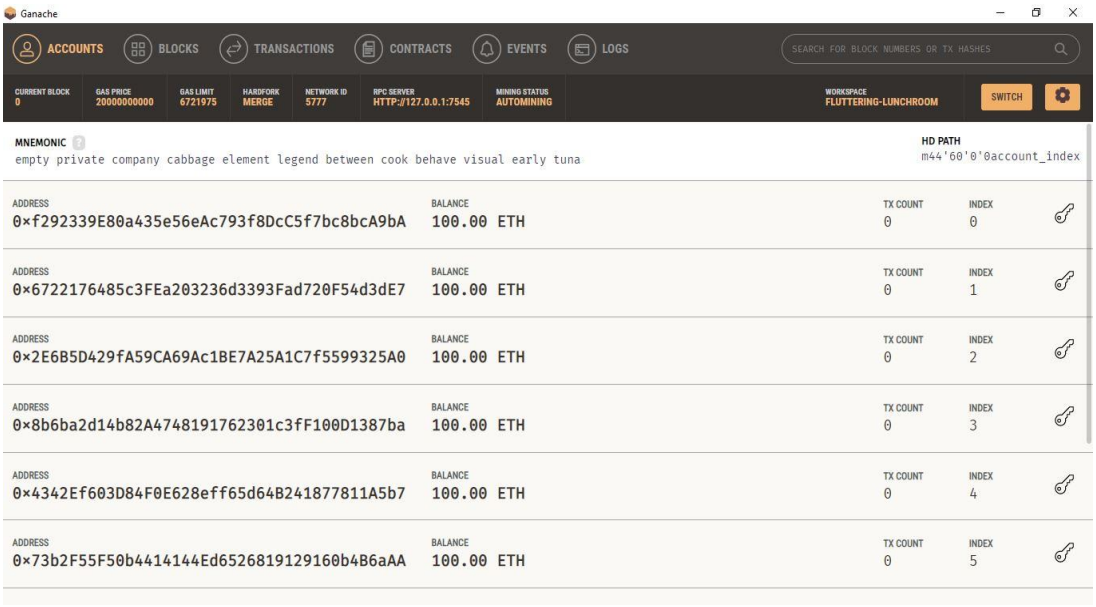

Plate 3.3: A screenshot showing successful running of Ganache

Step 5: Create a workbench on Genache

We named the workspace “anticounterfeiting,” and added the configuration file that was created earlier. A screenshot of successful creation of workbench on Ganache is displayed in Plate 3.4.

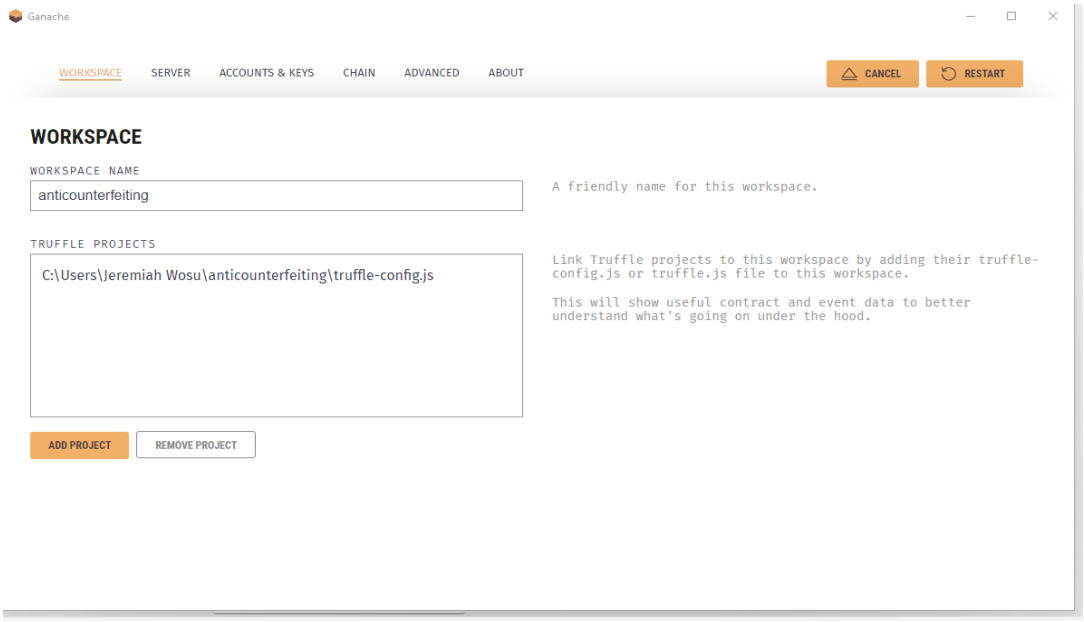

Plate 3.4: A screenshot showing successful creation of workbench on Ganache

Step 6: Install Truffle and Node.js on the Configuration File

In this work, Truffle will be employed to deploy smart contracts as well as to provide a gateway which enables smart contracts to communicate with the Genache private Ethereum network that will be setup. Truffle is the smart contract development framework. It has in-built capabilities for compilation, integration, and deployment of smart contracts along with binary management features. The network management features in the Truffle development framework helps developer to deploy applications to any number of private and public networks. Another prominent feature associated with the Truffle development framework is the configurable build pipeline which offers support for tight integration.

Node.js is a cross-platform JavaScript runtime environment that allows developers to run JavaScript code outside of a web browser. Nodejs comes with a lot of modules and mostly used in web development. V8 engine is at the heart of Node.js. The Node.js runtime environment includes several Node APIs to power the Node.js environment in addition to the V8 engine.

The installation of Truffle and Node.js on the Configuration File was achieved using the command line interface as shown in Plate 3.5.

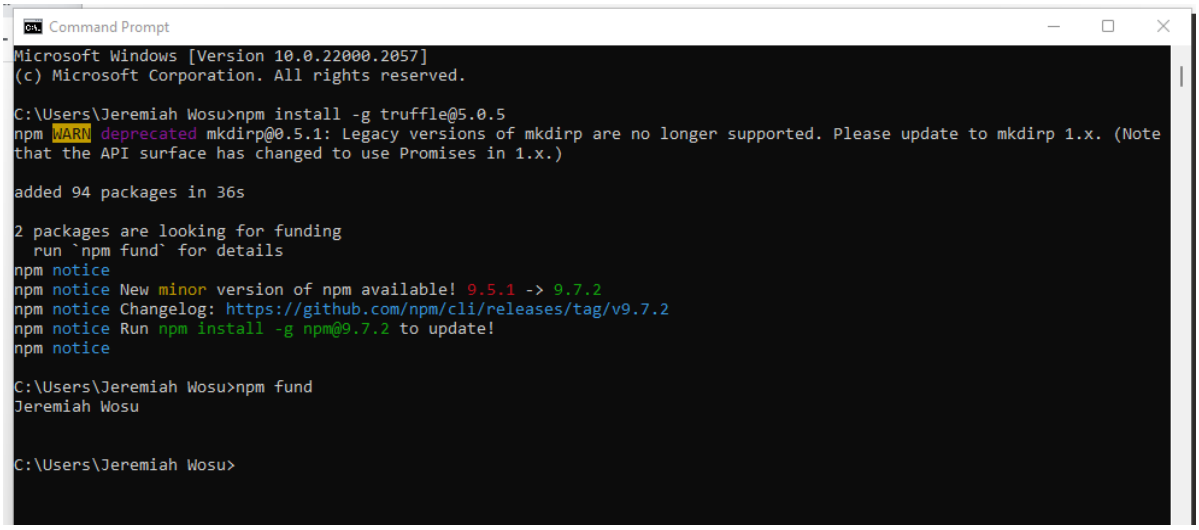

Plate 3.5: A screenshot showing successful installation of Truffle and Node.js

Step 7: Download and install MetaMask

MetaMask is an extension for Chrome which enables interaction between Ethereum dApp, blockchain network and user interface. It is an extension for accessing Ethereum enabled distributed applications, or "Dapps" in a web browser. The extension injects the Ethereum web3 API into every website's javascript context, so that dapps can read from the blockchain. MetaMask also lets the user create and manage their own identities (via private keys, local client wallet and hardware wallets), so when a Dapp wants to perform a transaction and write to the blockchain, the user gets a secure interface to review the transaction, before approving or rejecting it. It warns users when navigating sites that are suspected to be involved in phishing activities or that have a name matching the popular phishing targets. A screenshot of successful installation of MetaMask extension is displayed in Plate 3.6.

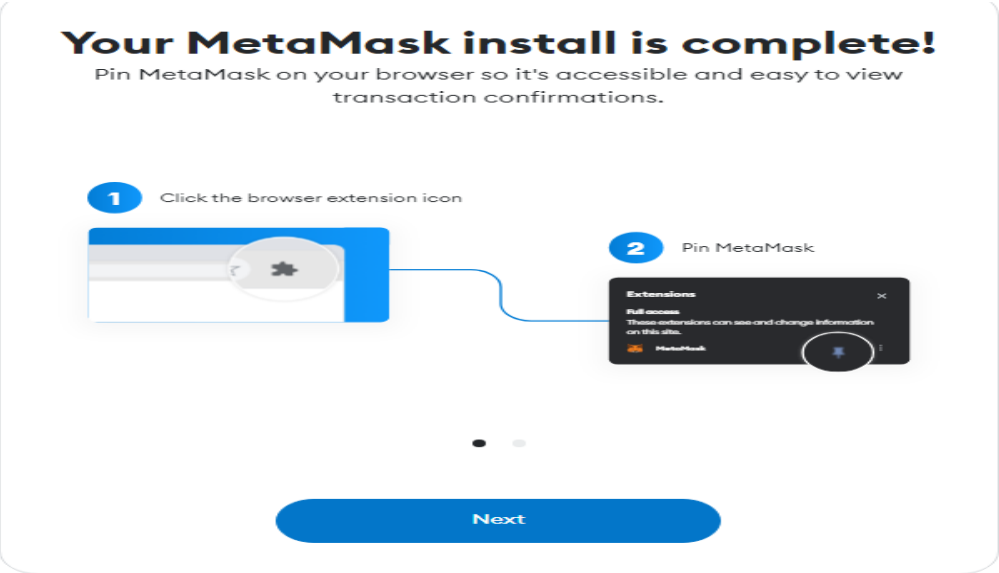

Plate 3.6: A screenshot showing successful installation of MetaMask extension

Step 8: Connect MetaMask to the Ganache blockchain network and create wallet

MetaMask was connected to the Ganache blockchain network and a secure wallet was created as shown in Plate 3.7.

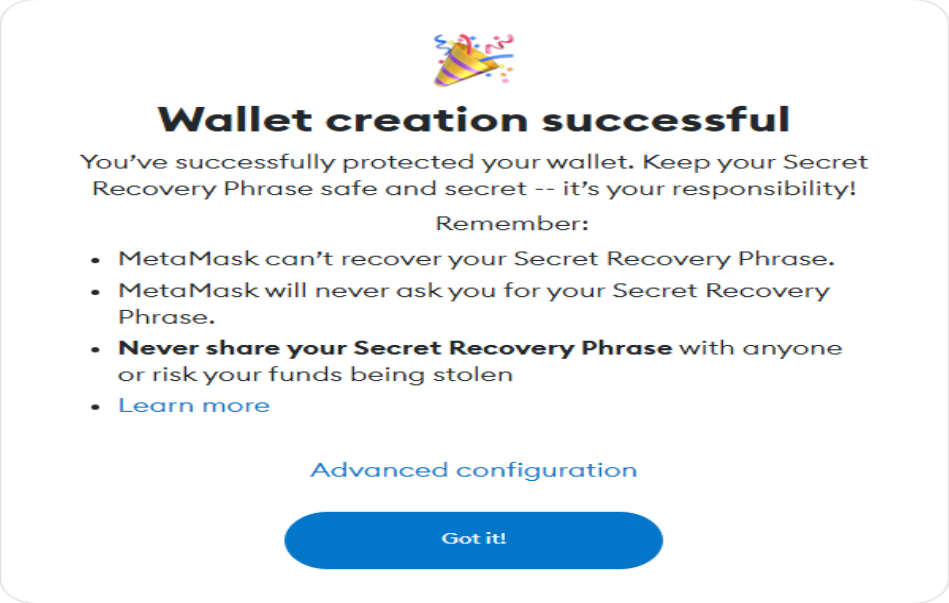

Plate 3.7: A screenshot showing successful creation of wallet.

3.2.3 Designing an Efficient Blockchain Consensus Algorithm

A custom consensus algorithm named Proof of Product Contribution (PoPC) is designed. It is fully decentralized and balances between efficiency and security. It does not verify the block signer election through the computation of complex mathematical problems. Instead, verification is done by considering the amount of product contribution and with some random criteria. This behavior will allow the whole blockchain to run without the use of many quantities of energy, but achieving a random consensus through all the network nodes. In PoPC, the node (i.e., manufacturer) with the highest number of products involved in different transactions that makeup a block is granted the privilege to create a new block. If there is a tie between two or more nodes, the node that its product was first involved in a transaction will be chosen. Other nodes will confirm the genuineness of the transactions contained in the new block before validating it.

PoPC main focus is to find a way for a randomized node election which incurs in a complete consensus through the network. To achieve this, the algorithm uses time, slots and synchrony as main features. Time is divided in slots, and for each slot it is supposed to be a minimum of one block. At all time, nodes or participants are synchronized among them.

PoPC is based on following a number of steps in order to decide the block signer. Each step can be understood as an elimination round. Each round starts when finishes the previous one, i.e. when we have found the new block signer. Once this signer broadcasts the new block, a new round starts.

At the beginning of each step, a hash (sig(r,s,q(r-1))) is calculated and converted into a decimal between 0 and 1, i.e. "p". Where "r" is the number of the block, "s" is the current step (at the beginning is 1) and q(r-1) is a random value taken from the previous block. If the value extracted from the hash, and converted into a decimal, is lower than "p", this node should participate in the step "s". Each participant has a master key which purpose is to generate new temporal keys. Every new generated private key is destroyed after signing a message with it.

Step 1, Block signer election: The node should prepare a candidate block, sign it and then send it to the network with sig(r,1,q(r-1)).

Step 2, Graded Consensus step 1: As long as we receive sig(r,1,q(r-1)) messages, the node must calculate the hash of this messages and then the candidate block will be the one with the minimum hash calculated. Then the new candidate block is signed with a temporal private key and sent to the network with sig(r,2,q(r-1)). Finally, the temporal private key is destroyed.

Step 3, Graded Consensus step 2: As long as we receive sig(r,2,q(r-1)) messages, the node should verify the following: If more than 1/2 of the arrived signatures belongs to the same bloc, the node signs this bloc and proceeds by sending the block signed and sig(r,2,q(r-1)). Otherwise an empty string is signed and send it to the network with sig(r,2,q(r-1)). Finally the temporal private key is destroyed.

Step 4, Graded Consensus output: If more than 1/2 of the new received messages belong to the same block, this is the candidate and a new variable "g" with value 2 is set. If more than 1/2of the new receiver messages belong to the same block, this is the candidate and a new variable "g" with value 1 is set. Otherwise there is no consensus so candidate block is an empty block and new variable "g" value is 0. Then, if "g" = 2, "b" = 0, otherwise "b" = 1. Finally the node signs "b" and sends it with sig(r,4,q(r-1)).

Step 5, Graded Consensus with output 0: If more than 1/2 of all received "b" messages are equal to zero, the node stops the search and sets the definitive block the one received. If more than 1/2 of the received "b" messages are equal to 1, the node stops the search and sets a void block because of the no consensus situation. Otherwise "b" is equal to zero and "b" is sent with sig(r,5,q(r-1)).

Step 6, Graded Consensus with output 1: Same as Step 5 but if there is no consensus, "b" equal to 1 is sent with sig(r,6,q(r-1)).

Step 7, Graded Consensus with output different than 0 or 1: Same as Step 5 but if there is no consensus, "b" is equal to the lower hash of the sig(r,6,q(r-1)) messages received. Then "b" is sent with sig(r,7,q(r-1)).

Step 8, Last step (low probability): Execute again BBA and Step 5 but if there is no consensus, the node stops. No consensus found so the block remains void, "b" equals to 1 and sig(r,8,q(r-1)) is sent.

The algorithm for implementing PoPC is represented in Figure 3.5.

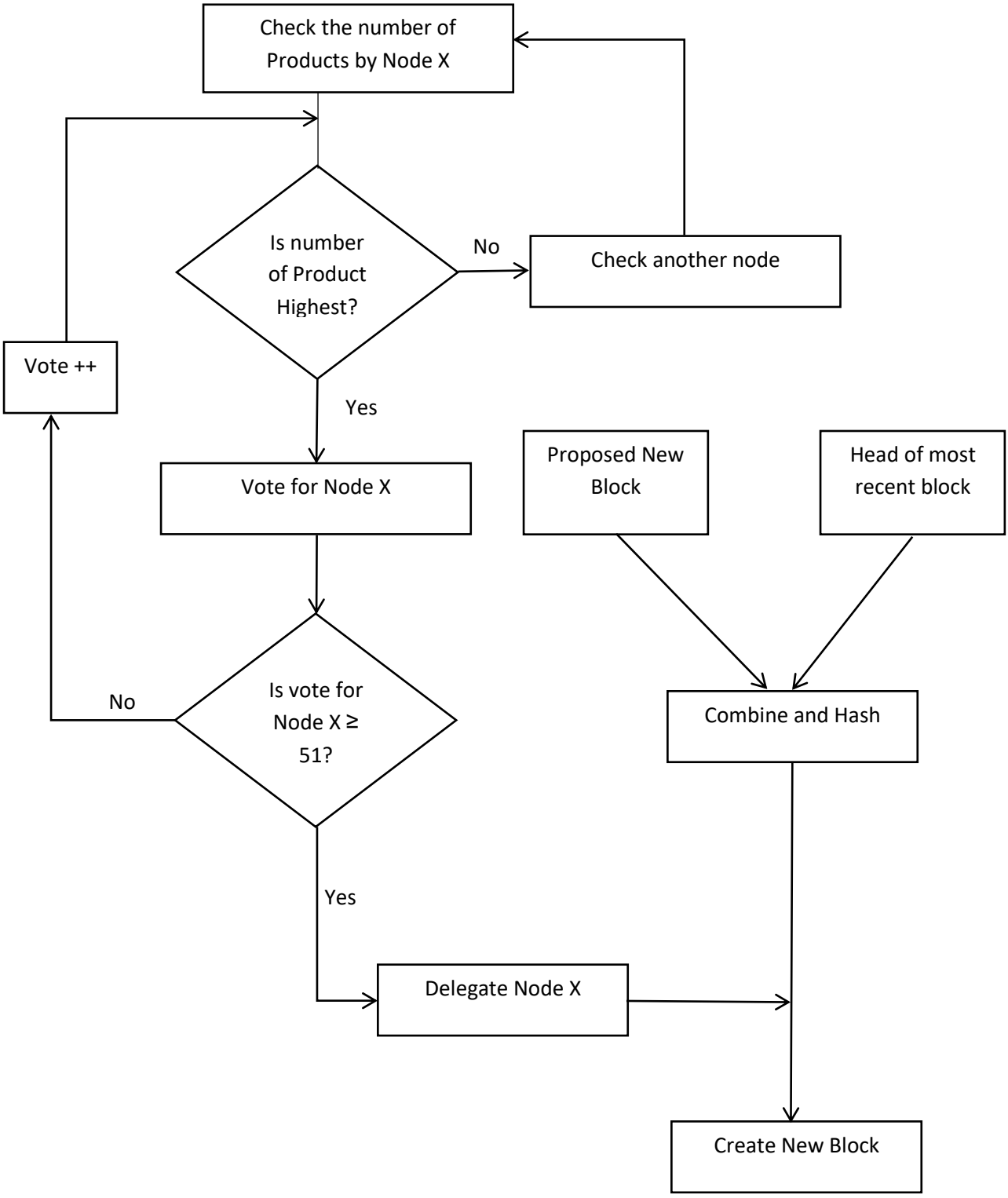

Figure 3.5: Algorithm for Implementing PoPC

PoPC is coded in Java programming language. The entire source code of PoPC is found in Appendix 2.

3.2.4 Writing Smart Contracts

A smart contract in the sense of Solidity is a collection of code (its functions) and data (its state) that resides at a specific address on the Ethereum blockchain. In this work smart contracts are coded using Solidity programming language.

Solidity is an object-oriented programming language created specifically by the Ethereum Network team for constructing and designing smart contracts on Blockchain platforms. It is used to create smart contracts that implement business logic and generate a chain of transaction records in the blockchain system. It has a lot of similarities with C and C++ and is pretty simple to learn and understand. Like other programming languages, Solidity programming also has variables, functions, classes, arithmetic operations, string manipulation, and many other concepts.

In view of its simplicity and Turing completeness as well as its ability to allow complex data types and member variables, Solidity programming language will be employed in writing smart contracts in this work.

Smart contracts written in Solidity must run through a compiler so that the Ethereum Virtual Machine (EVM) can understand and execute appropriate commands. In this work, codes written using Solidity are compiled using a Solidity compiler, which outputs byte code and other artifacts needed for deployment of smart contracts.

To operate a Solidity smart contract in Offline mode, the following software were downloaded and installed:

1. Truffle globally: This was installed earlier. (see step 6 under sub-section 3.2.2) Truffle is the smart contract development framework. It has in-built capabilities for compilation, integration, and deployment of smart contracts along with binary management features.

2. Solcjs: solcjs is a Solidity compiler. It was installed using npm command. In this work, codes written using Solidity are compiled using a Solidity compiler, which outputs byte code and other artifacts needed for deployment of smart contracts.

3. Solidity extension: This was installed on VS code by selecting it from the list of extensions, and clicking install. This extension provides IntelliSense, autocompletion, and templates for DApps, and works within VS IDE.

The source code for all the smart contracts created in this work is found in Appendix 3. The algorithms and flowcharts of key BBACS smart contracts are presented below.

(A) Smart Contract for Registering Manufacturers

Algorithm

A. System prompts manufacturer to input basic details
B. Manufacturer inputs basic details
C. System checks basic details are correct
    a. If details are not correct
        i. Stop registration process.
        ii. Notify the manufacturer.
        iii. Go back to step A.
    b. Else if details are correct.
        i. Proceed to step D.
D. System prompts Admin to authenticate the manufacturer
E. System checks if authentication is successful
    a. If authentication is not successful
        i. Stop registration process
        ii. Notify the manufacturer
        iii. Go back to step A
    b. Else if authentication is successful
        i. Proceed to step F.

F. The system completes the registration process by performing the following operations:

i. System issues ID to the manufacturer

ii. System stores details of the registration

The flowchart for this smart contract is presented in Figure 3.6.

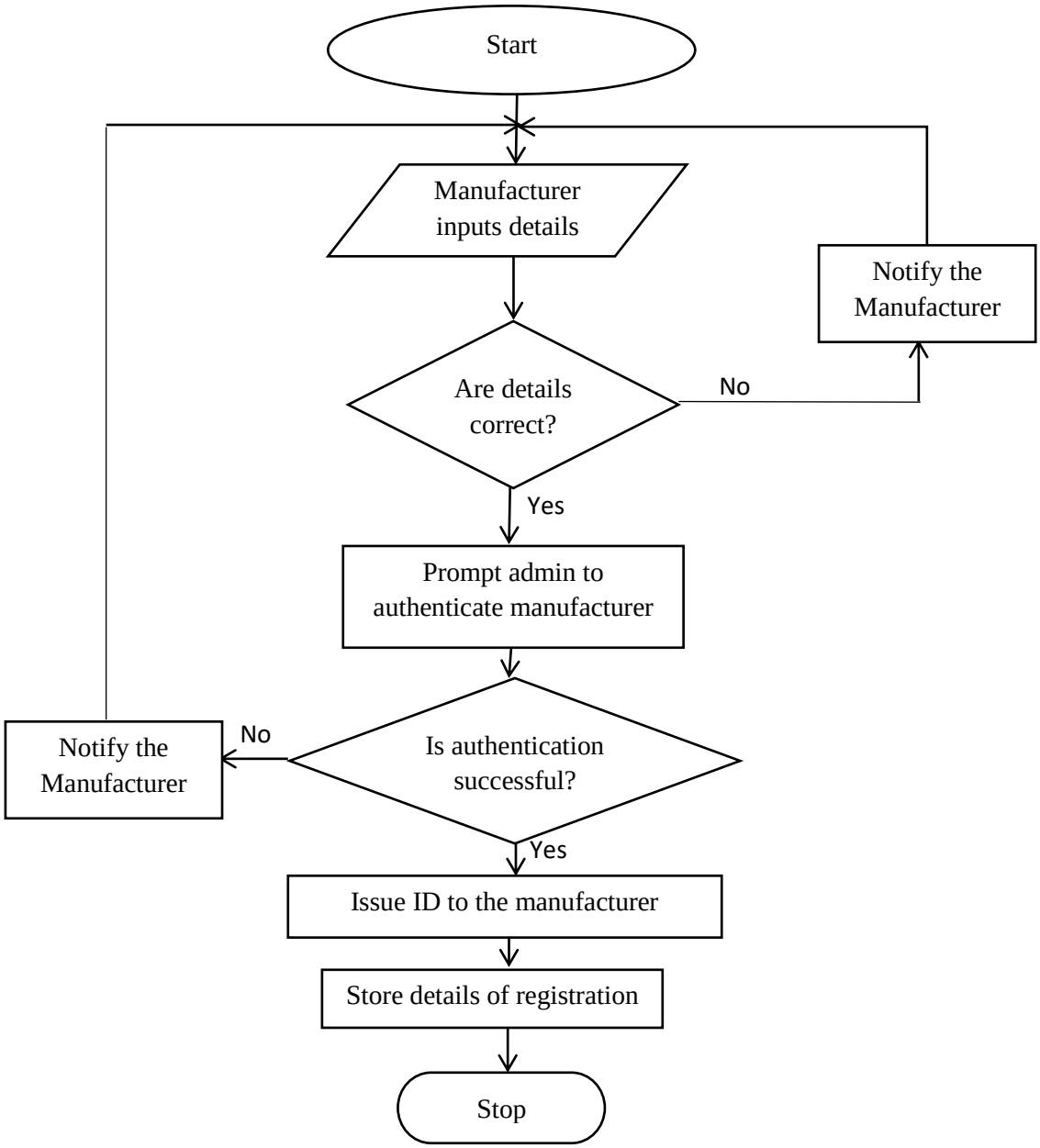

Figure 3.6: Flowchart for Registering Manufacturers

A screenshot of the smart contract code for registering Manufacturers is displayed in Plate 3.8.

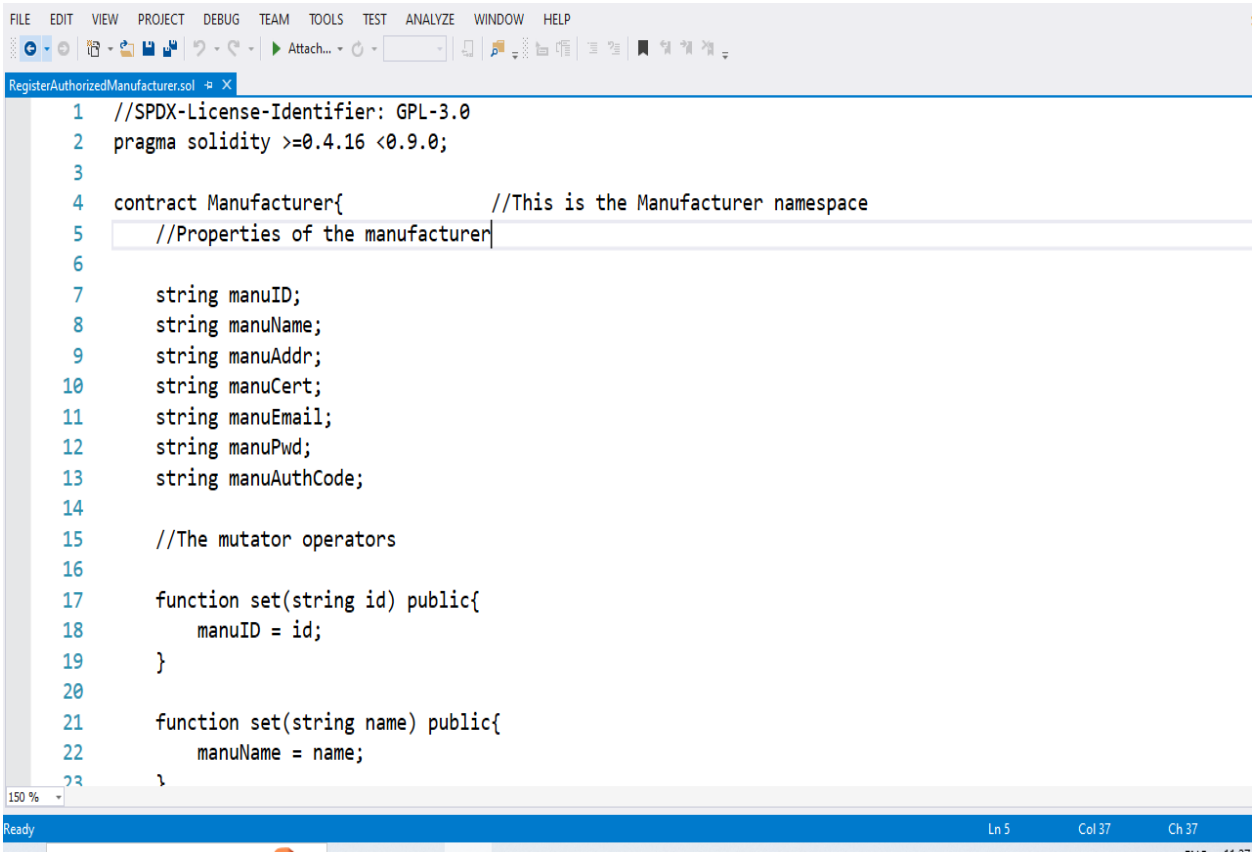

Plate 3.8: A Screenshot of Smart Contract Code for Registering Manufacturers

(B) Smart Contract for Registering Distributor

Algorithm

A. System prompts distributor to input basic details
B. Distributor inputs basic details
C. System checks basic details are correct
    a. If details are not correct
        i. Stop registration process.
        ii. Notify the distributor.

iii. Go back to step A.

b. Else if details are correct.

i. Proceed to step D.

D. System prompts manufacturer to authenticate the distributor

E. System checks if authentication is successful

a. If authentication is not successful

i. Stop registration process

ii. Notify the distributor

iii. Go back to step A

b. Else if authentication is successful

i. Proceed to step F.

F. The system completes the registration process by performing the following operations:

i. System issues ID to the distributor

ii. System stores details of the registration

The flowchart for this smart contract is presented in Figure 3.7

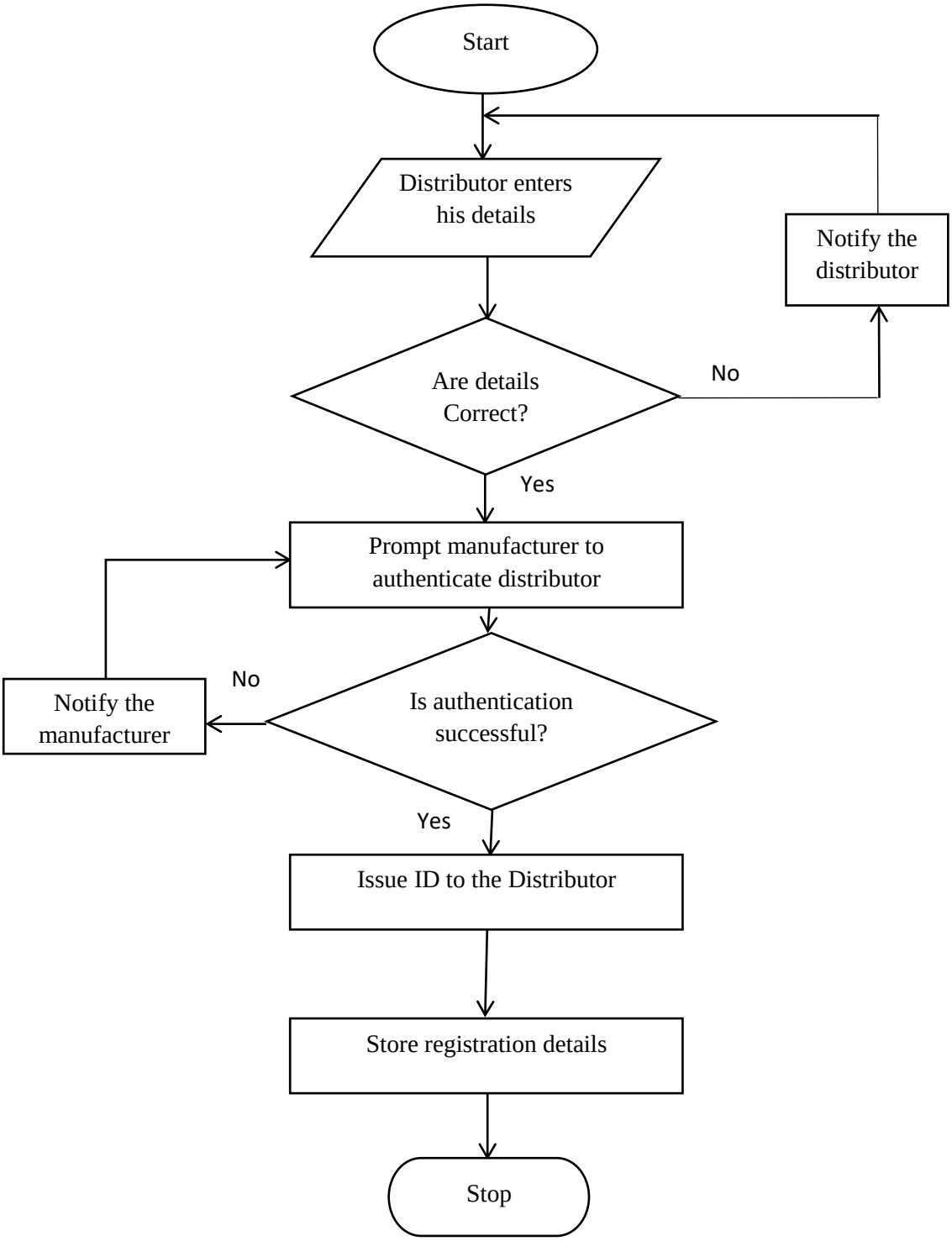

Figure 3.7: Flowchart for Registering Distributorsrs

A screenshot of the smart contract code for registering distributors is displayed in Plate 3.9.

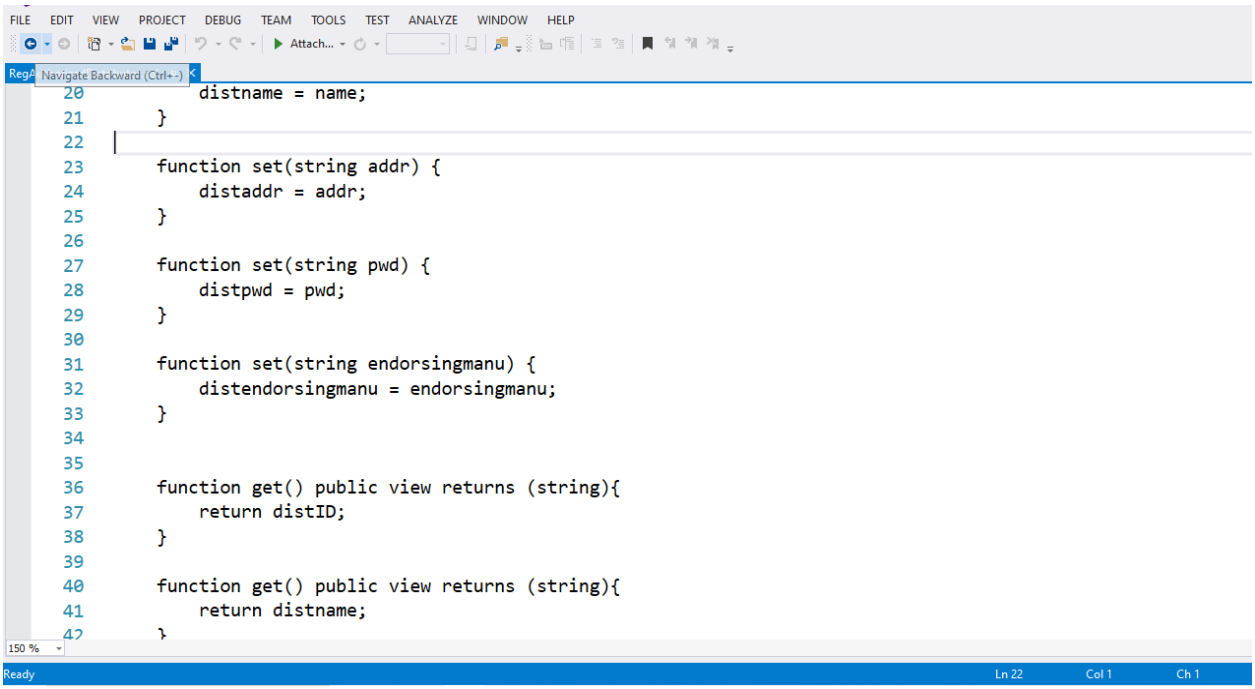

Plate 3.9: A Screenshot of Smart Contract Code for Registering Distributors

(C) Smart Contract for Registering Product

Algorithm

A. Manufacturer to input basic product details as well as scan product texture
B. Manufacturer inputs basic details
C. System checks basic details are correct
    a. If details are not correct
        i. Stop registration process.
        ii. Notify the manufacturer.
        iii. Go back to step A.
    b. Else if details are correct.
        ii. Proceed to step D.
D. The system completes registration process by performing these operations:

i. System issues ID and generates QR code for product

ii. System stores details of the registration

The flowchart for this smart contract is presented in Figure 3.8

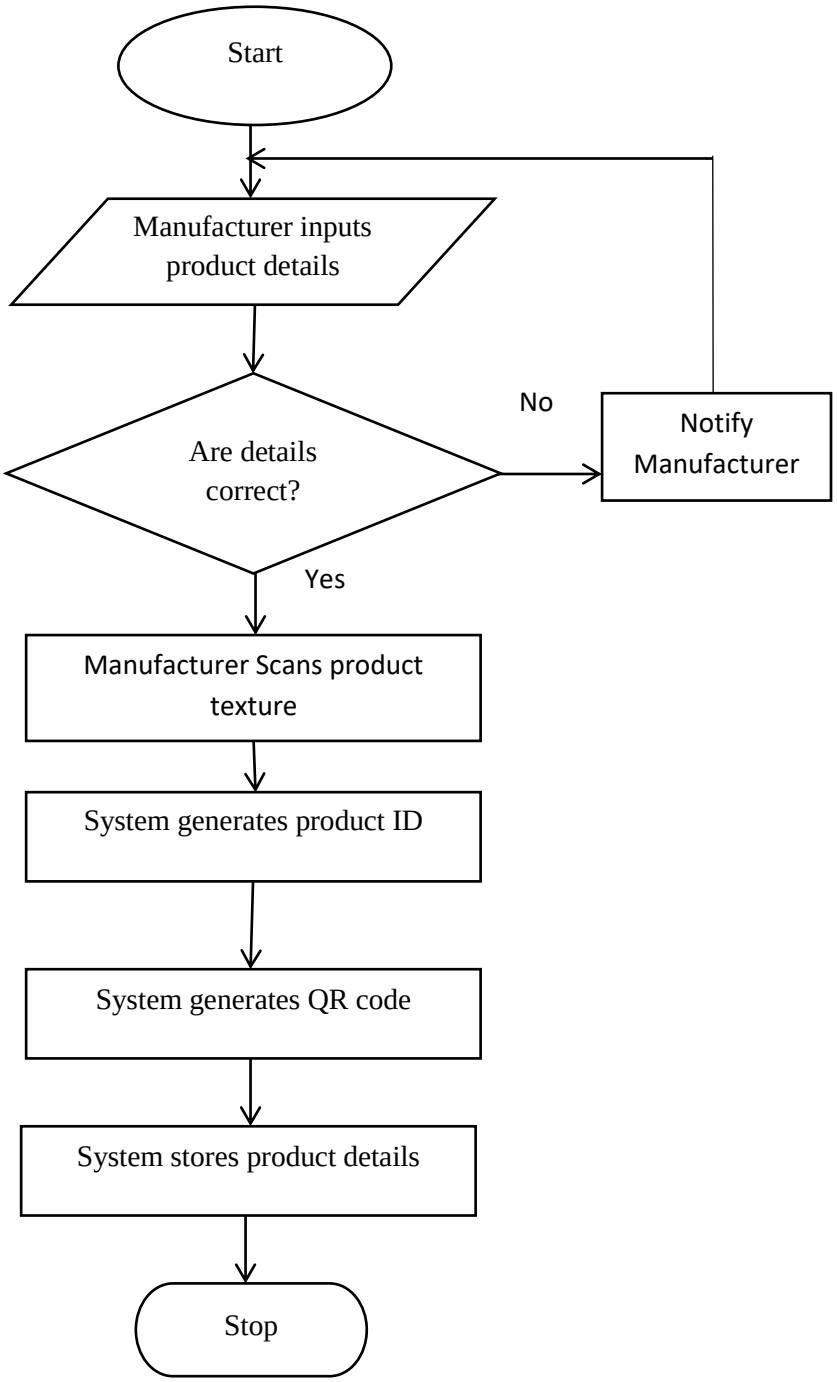

Figure 3.8: Flowchart for Registering Products

(D) Smart Contract for Product Confirmation Algorithm

A. End User scans the QR code embedded on the product

B. System automatically uploads product details generated from the scan to the network

C. System compares product details generated from the scan with that stored in the blockchain during product registration.

   a. If details are not the same

      i. Display “Product not found on BCACS network; product may likely be counterfeit”

      ii. Notify the enforcement officials.

      iii. Store details of the unsuccessful confirmation process

      iv. Go back to step A.

   b. Else if details are the same

      i. Display “Product is genuine”

      ii. System stores details of the product confirmation process

The flowchart for this smart contract is presented in Figure 3.9

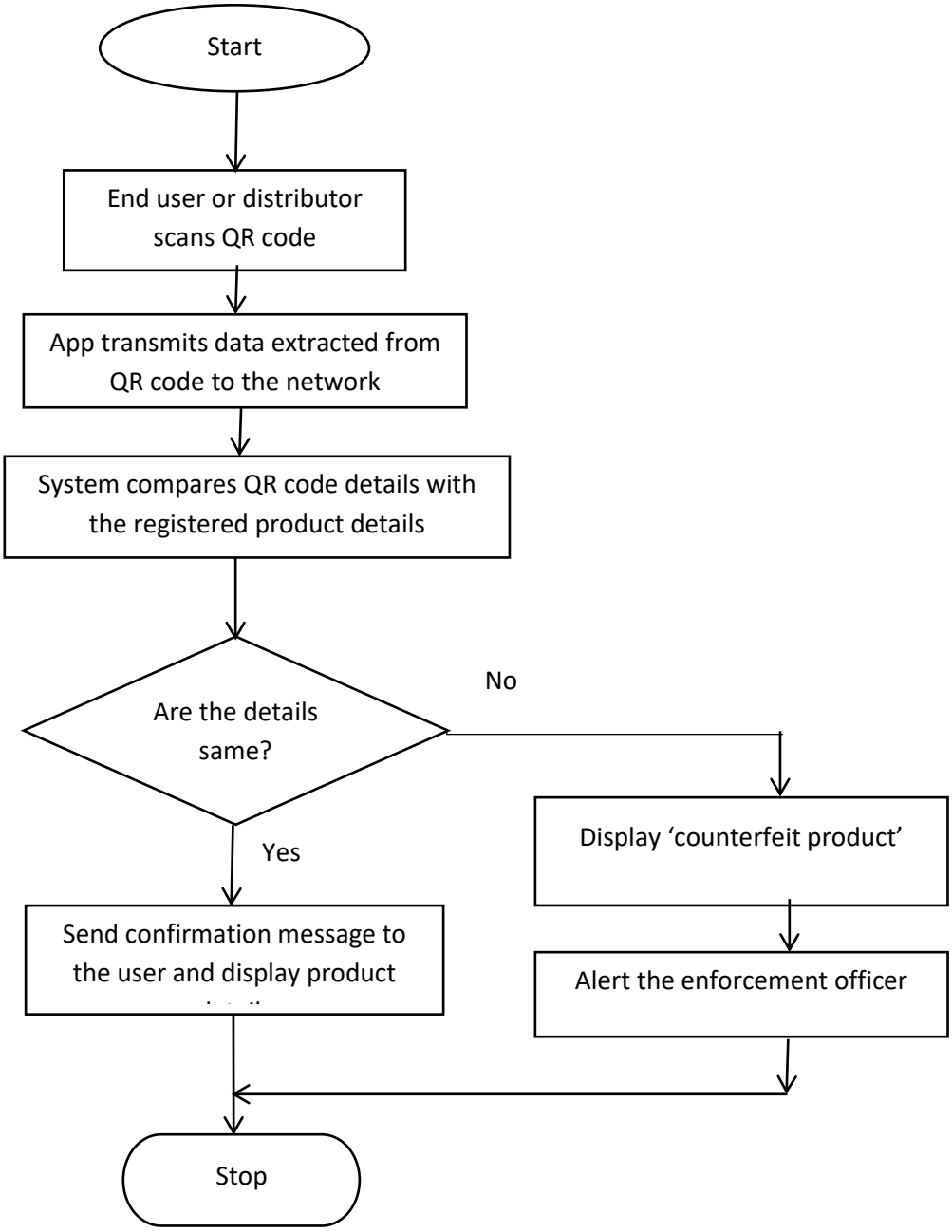

Figure 3.9: Flowchart of Smart Contract for Product Confirmation

(E) Smart Contract for Transfer of Product Ownership

Algorithm

A. System receives transfer of ownership request from purchaser

B. System gets confirmation from vendor

a. If confirmation is not received

i. System aborts transfers ownership process

b. Else if confirmation is received

i. Proceed to step C.

C. System transfers ownership to purchaser

D. System stores details of transfers of ownership

The flowchart for this smart contract is presented in Figure 3.10

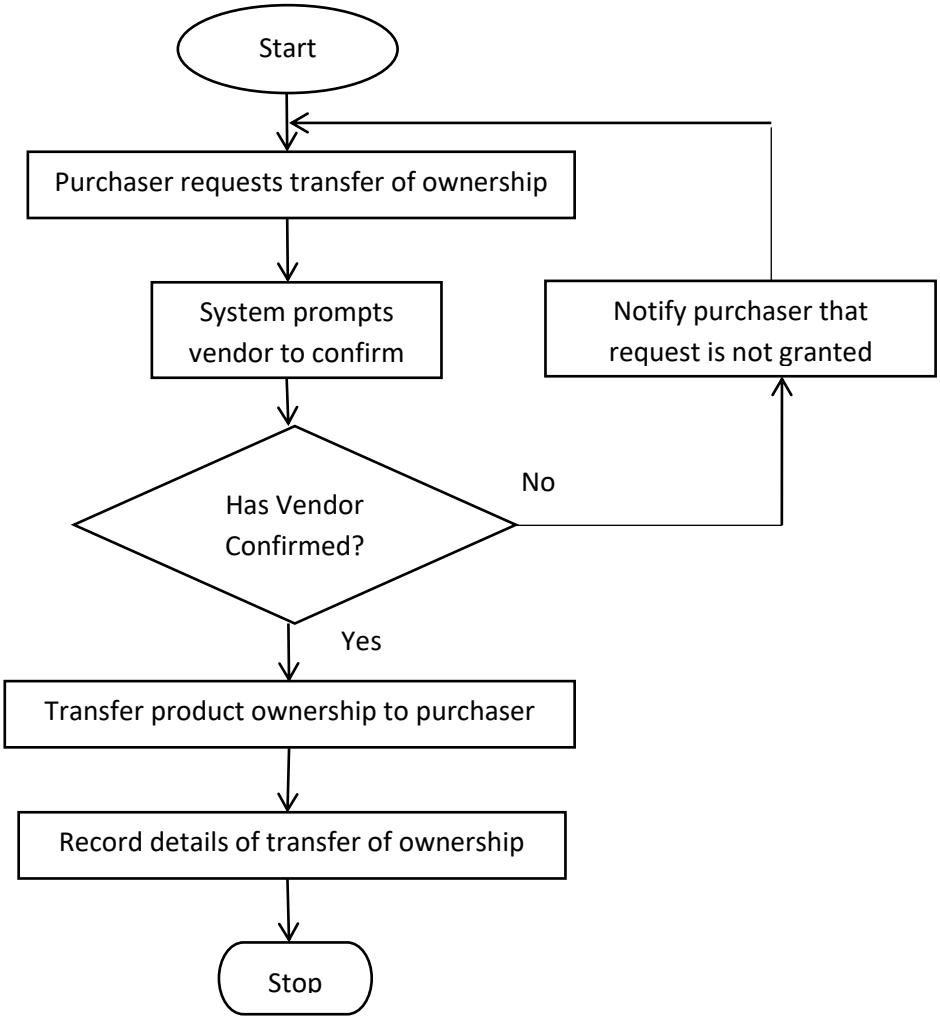

Figure 3.10: Flowchart for transfering Product Ownership

(F) Smart Contract for Creating New Block

Algorithm

A. System prompts nodes to elect a signer to create a new block

B. Each node checks for node X with the highest number of products in the transactions that make up new the block

C. Nodes vote node X to be a signer

D. System checks to confirm that not less than 51% nodes voted node X

   a. If less than 51% nodes voted node X

      i. Wait for more nodes to vote

   b. Else if not less than 51% nodes voted node X

      i. Proceed to step E.

E. System assigns node X the role to create a new block

F. Node X creates a new block

G. Other nodes validates the new block

H. System stores new block in chain

The flowchart for this smart contract is presented in Figure 3.11

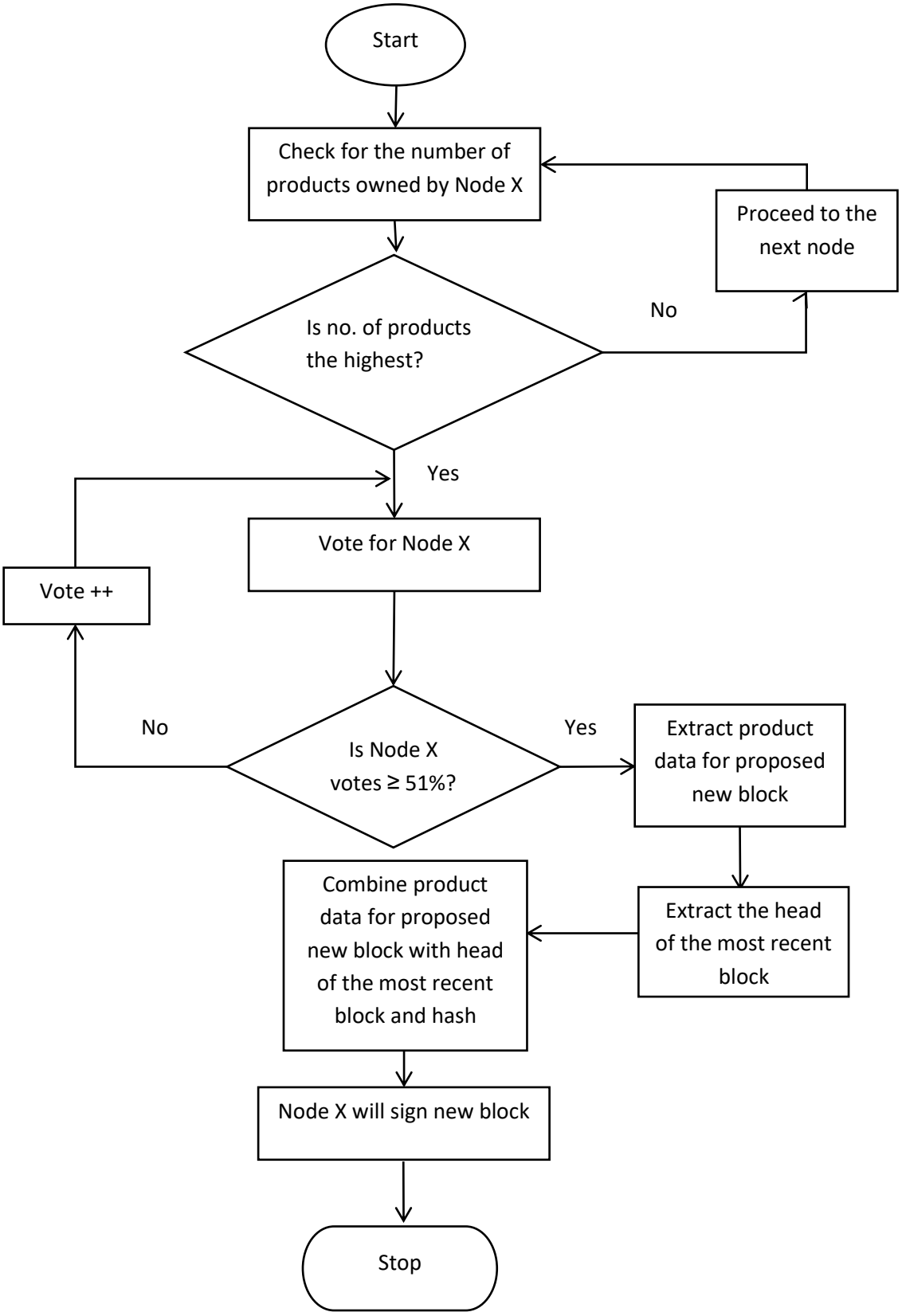

Figure 3.11: Flowchart for Creating New Blocks

3.2.5 Developing a Distributed Application

A distributed application constitutes of smart contracts and user interfaces (UI). The method for creating smart contracts have been presented in the preceding sub-section. Thus, this section will fucus only on the development of UI. This work utilized HTML5, CSS, Nodejs and React in creating UI.

Hypertext Markup Language 5 (HTML5) is the fifth version, or release, of the Hypertext Markup Language (HTML). It allows the modification of the appearance of web pages, as well as making adjustments to their appearance. It also used to structure and present content for the web.

Cascading Style Sheets (CSS) is the code that styles web content. CSS is a style sheet language. This work utilized CSS to selectively style HTML elements.

Node.js: This was installed earlier. (See step 6 under sub-section 3.2.2) It is a cross-platform JavaScript runtime environment that allows developers to run JavaScript code outside of a web browser.

React is one of the most flexible front-end frameworks used to build blockchain apps. React is often used with other libraries such as Redux for building single-page applications. The React code is modular and easy to maintain, which saves development time and business costs.

The source codes for developing the dApp UI is found in Appendix 4. A sample code is displayed in Plate 3.10

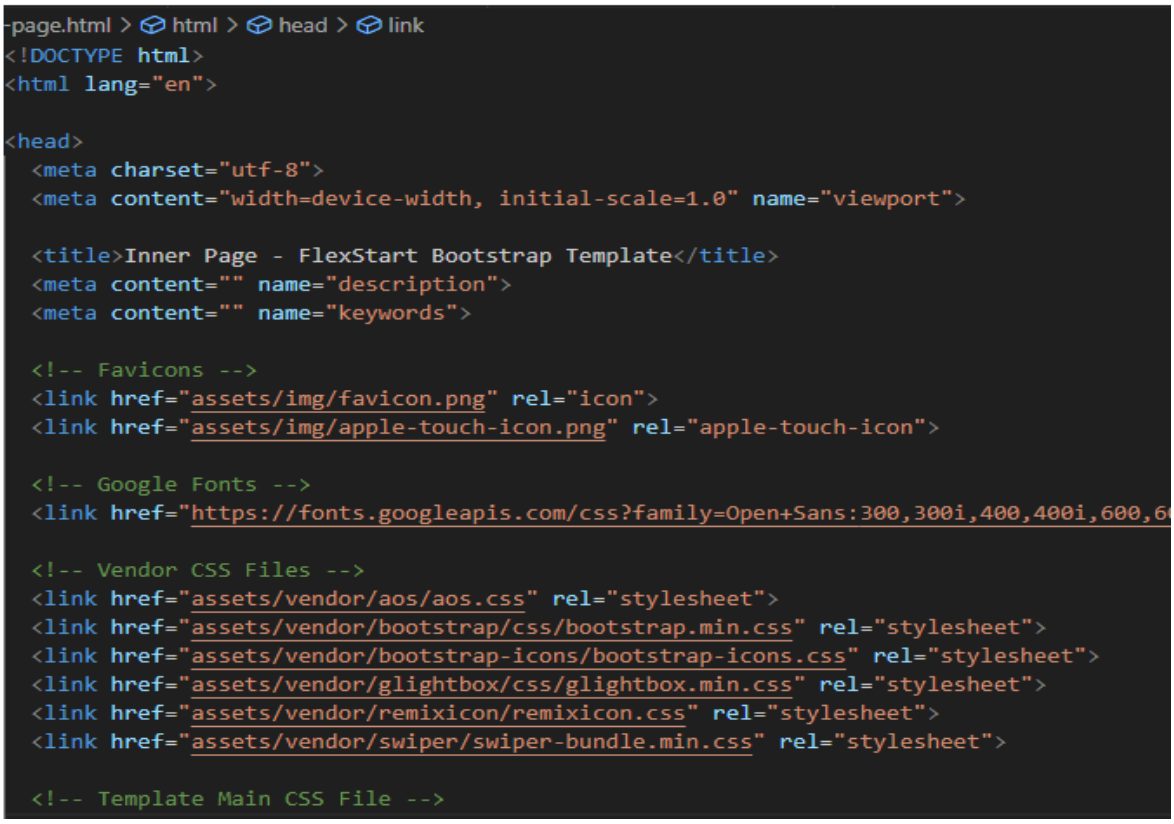

Plate 3.10: A sample program code developing user interface

3.2.6 Developing Application for Generating and Scanning Special QR Code

This work will employ Kotlin to develop the applications (Apps) for generating and scanning QR codes as well as the associated web services. Kotlin is a cross-platform, statically typed, general-purpose high-level programming language with type inference. It is designed to interoperate fully with Java. Kotlin mainly targets the Java virtual machine (JVM), but also compiles to JavaScript (e.g., for frontend web applications using React) or native code (e.g., for native iOS apps sharing business logic with Android apps).

Kotlin simplifies the development of cross-platform projects. It reduces time spent writing and maintaining the same code for different platforms while retaining the

flexibility and benefits of native programming. Kotlin applications will work on different operating systems, such as iOS, Android, Windows, Linux, etc.

The application is developed in two parts, namely:

(A) **Application for Generating Special QR Code**

The special QR code is generated by utilizing a combination of product texture image and the production details. To generate this special QR code, the following steps are taken:

(i) Capture product texture as an image
(ii) Downsize the image
(iii) Convert image to Gray scale
(iv) Apply matrix filters
(v) Do max-polling
(vi) Flatten matrix result
(vii) Extract product details
(viii) Combine data from (vi) with data from (vii) using Regex json
(ix) Feed data obtained from (viii) into a QR generator

These steps are explained below.

1. Product Details and Image Capture: The application starts by prompting the user to provide product details. These details could include information such as product name, manufacturing date, expiry date, batch number, price, and any other relevant data. Once the user submits these details, the application activates the device's camera to capture an image of the product.
2. Image Processing: After capturing the image, the application performs image processing operations, similar to those used in convolutional neural networks (CNNs). These operations help enhance the image quality and extract relevant features. The specific techniques employed include resizing the image to a standardized size and applying convolutional filters to emphasize important patterns. Image processing

plays a crucial role in enhancing the captured product image and extracting relevant features. The techniques employed in this step are inspired by convolutional neural networks (CNNs), which are commonly used for image analysis and recognition tasks. Although a full-fledged CNN is not implemented in this process, some of the fundamental concepts are applied.

3. Resizing: Resizing the image is an essential preprocessing step to ensure consistency in the subsequent operations. By resizing the image to a standardized size, such as 24x24 pixels, we simplify the subsequent calculations while maintaining the vital visual characteristics of the product.

4. Convolutional Filters: Convolutional filters, also known as kernels, are applied to the image to extract specific features. These filters act as sliding windows that traverse the image and perform mathematical operations on small patches of pixels. The filters emphasize patterns, edges, textures, or other relevant features within the image. By applying convolutional filters, the processed image highlights important visual cues that aid in subsequent analysis.

5. Max pooling: Max pooling is a down-sampling technique used in CNNs to reduce the spatial dimensions of feature maps obtained from convolutional operations. It helps to extract the most prominent features while reducing the computational complexity. Max pooling divides the feature map into non-overlapping rectangular regions (often referred to as pooling windows or pooling kernels) and retains only the maximum value within each region. This down-samples the feature map by a factor determined by the size and stride of the pooling windows. Max pooling serves two primary purposes in image processing:

(i) Translation Invariance: By selecting the maximum value within each pooling window, max pooling helps to capture the most significant features while being invariant to small translations or shifts in the input image. This property allows the CNN to recognize features irrespective of their precise location within the image.

(ii) Dimension Reduction: The downsampling effect of max pooling reduces the spatial dimensions of the feature map, reducing the number of parameters and computations required in subsequent layers. This dimension reduction aids in extracting more abstract and higher-level features from the image.

In the context of our QR scanner application, max pooling is applied after the convolutional filters. The feature map obtained from the convolutional operations is divided into pooling windows, and the maximum value within each window is retained, forming a downsampled feature map.

6. Matrix Representation: The resulting downsampled feature map is converted into a matrix representation with reduced dimensions, similar to the matrix conversion step discussed earlier. This matrix captures the essential features extracted through convolution and max pooling, further aiding in subsequent processing steps.

By incorporating max pooling into the image processing step, our QR scanner application is effectively downsample and capture the most salient features of the product image. This downsampling reduces computational complexity and improves the efficiency of subsequent operations, ultimately contributing to the generation of accurate and informative QR codes

7. Matrix Conversion: Following the image processing step, the image is converted into a matrix representation. This matrix represents the image with dimensions typically reduced to 24x24 pixels. The downsizing helps to reduce the complexity of subsequent operations while retaining the essential information needed for QR code generation.

8. Product Information Encryption: Simultaneously, the extracted product details are converted into a JSON string, creating a structured representation of the information. This JSON string is combined with the image matrix as input for an encryption algorithm. The encryption process protects the data and ensures its confidentiality during transmission and storage.

9. QR Code Generation: The encrypted output from the encryption algorithm is passed to a QR code generator. The QR code generator takes the encrypted data as input and generates a QR code image that encodes the information. When scanned, the QR Result is the encrypted information which is used to uniquely identify the product.

10. Saving the QR Code: Once the QR code is generated, the application saves it for future use or sharing. The saved QR code can be displayed on the screen, printed, or transmitted to be stored in the private blockchain network.

This is represented with the Activity Diagram shown in Figure 3.12

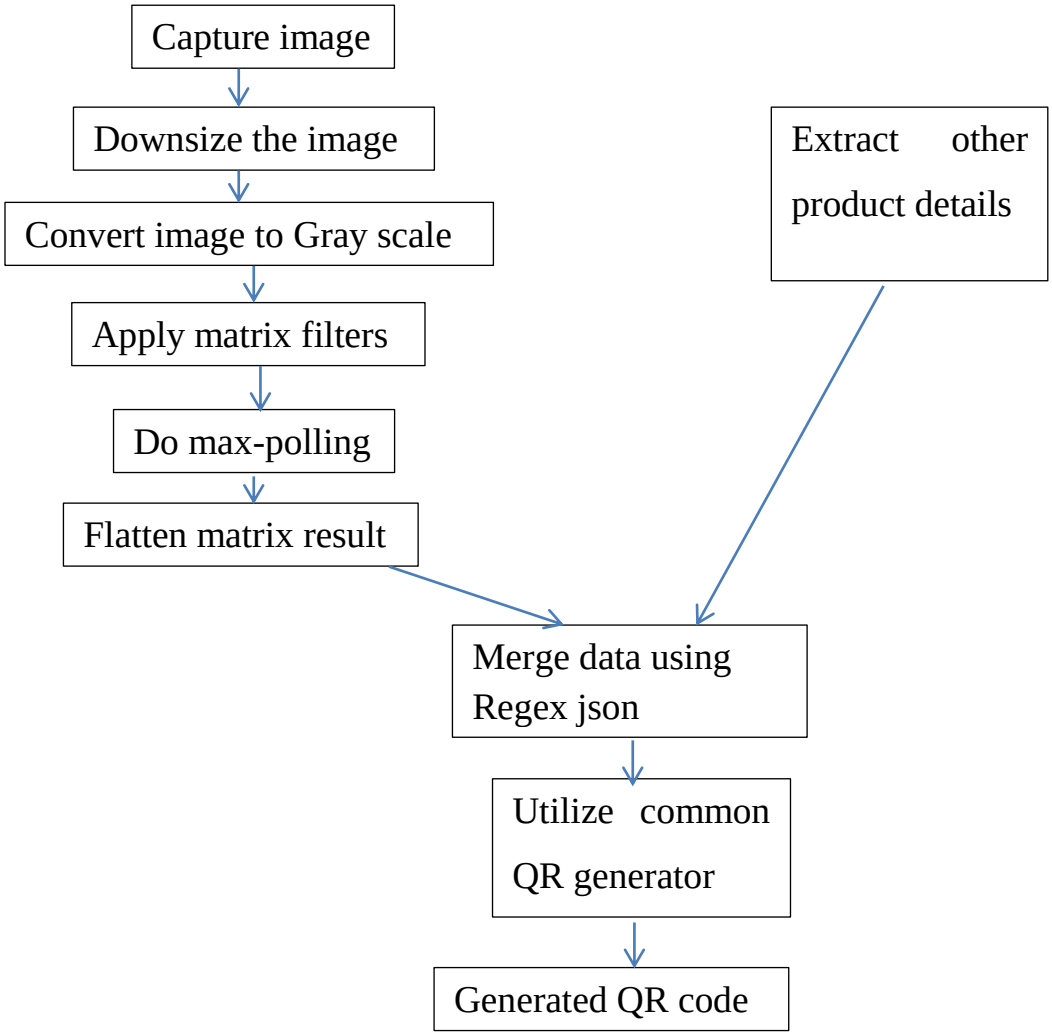

Figure 3.12: Activity Diagram for Generating Special QR Code

(B) Application for Scanning Special QR Code

An application for scanning special QR code was developed by reversing the process applied in developing application for generating special QR code outlined above. The source codes for developing these applications and the associated web services are found in Appendix 5.

3.2.7 BBACS Integration and Testing

(A) **System Integration**

Truffle suite environment is utilized for integrating the developed applications into the blockchain network. In order to integrate developed software files and applications into the Ganache private blockchain network, the following things were done:

1. The developed configuration file named anticounterfeiting was added to the workbench created on our Ganache blockchain network.
2. Truffle framework was installed to enable our smart contracts compile and be deployed to our Ganache blockchain network.
3. MetaMask was downloaded and installed to enable dApp to interact with our Ganache blockchain network.
4. We developed two web services (APIs) to enable the applications for QR code generation and scanning to communicate with our Ganache blockchain network.

(B) **Testing the Developed System**

Testing blockchain applications is the systematic evaluation of the blockchain's various functional components such as smart contracts, consensus algorithm, etc. Blockchain testing aids in the development of various quality stages, ranging from system performance to the security of the blockchain application. The following tests were carried out on the developed system:

1. Test to verify that the UML design pattern that was used in modelling the system would yield a functional system. This was proved by systematically following the design in the development of our system.
2. Test to confirm that Ganache is functioning properly. This was achieved by establishing that Truffle is communicating with our Ganache network. In order to carry out this test, we changed the directory in the CLI to our program directory i.e., anticounterfeiting, using the keyword Truffle migrate, and compiling the code with the keyword Truffle compile.
3. Test to ascertain the speed and security of the developed consensus mechanism. The time for the creation of new blocks in the network were monitored using Truffle. Also, Sybil Attack was lunched on the network to determine its security strength.
4. Test to ensure that smart contracts are compiled and deployed to our Ganache network. Smart contracts were compiled and deployed to the network using solidity compiler and Truffle respectively. The smart contracts were tested as to whether they will execute automatically.
5. Test to establish that our dApp is interacting with our Ganache network. This was carried out by connecting MetaMask to our Ganache network, and letting users (e.g., manufacturer and distributor) to login and input data via our dApp.
6. Test to confirm that the applications for generating and scanning QR codes are functioning properly. This was achieved by using these applications to generate and scan QR codes.
7. Integrated test to confirm that the BBACS is functioning properly. This was achieved by registering manufacturers and distributors in the network. Then, manufacturers registered their products using the QR code generating application. Finally, end users used the QR code scanning application to scan QR code embedded on product and were alerted as to whether the product is genuine or not.

The results obtained from these tests are presented and analyzed in chapter four.

CHAPTER FOUR
RESULTS AND DISCUSSION

4.1 Results

The immutability of stored data (i.e., data integrity), security strength, speed of execution, as well as the uniqueness of QR codes embedded on products are considered the Key Parameter Indices (KPIs) for testing this system. These KPIs were tested for the developed system in sub-section 3.2.7. Results obtained are presented in this section, and discussed in subsequent section i.e., section 4.2.

4.1.1: UML Design Pattern

The result obtained by utilizing UML design pattern in modelling our blockchain-based system is presented in Figure 4.1.

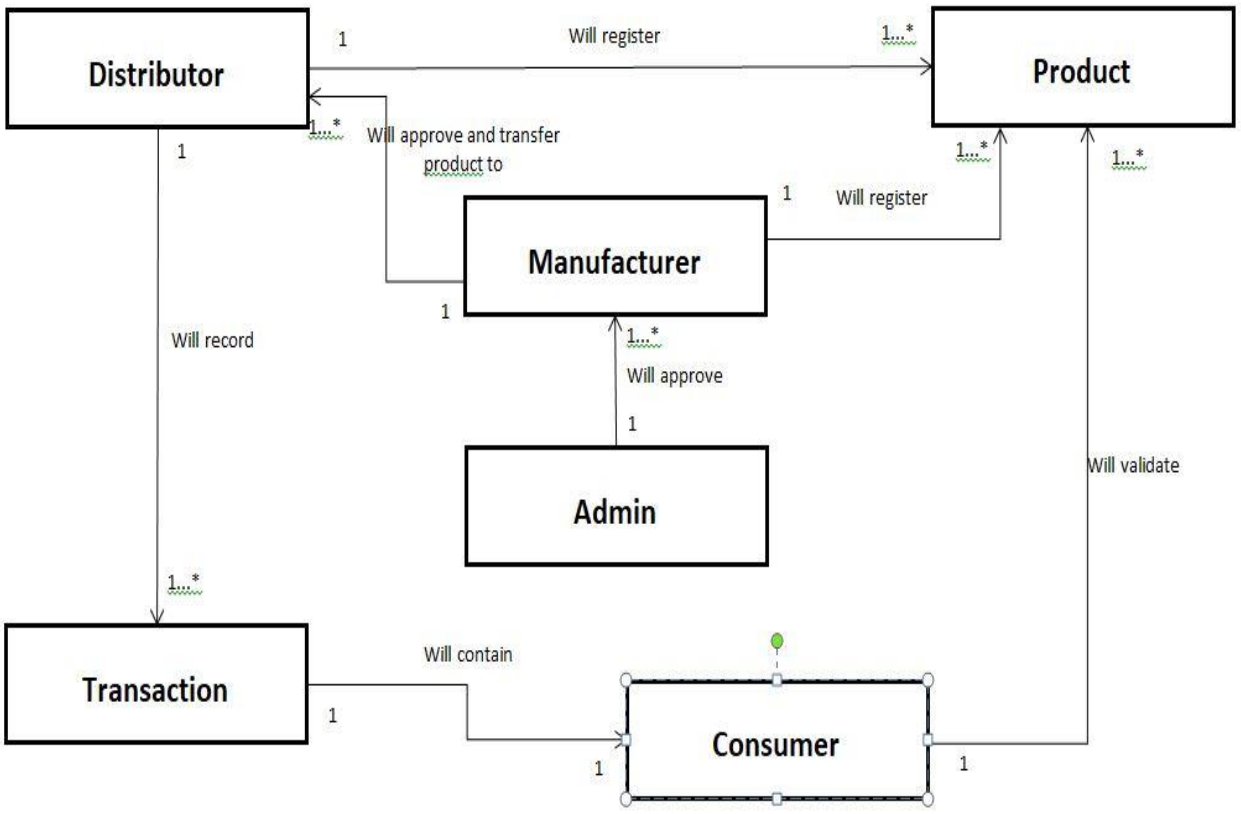

Figure 4.1: Model of the Developed System

4.1.2: Functioning of the Developed Private Ganache Blockchain Network

Snapshots confirming that the private Ganache blockchain network that was setup is functioning properly are presented in Plate 4.1 and Plate 4.2.

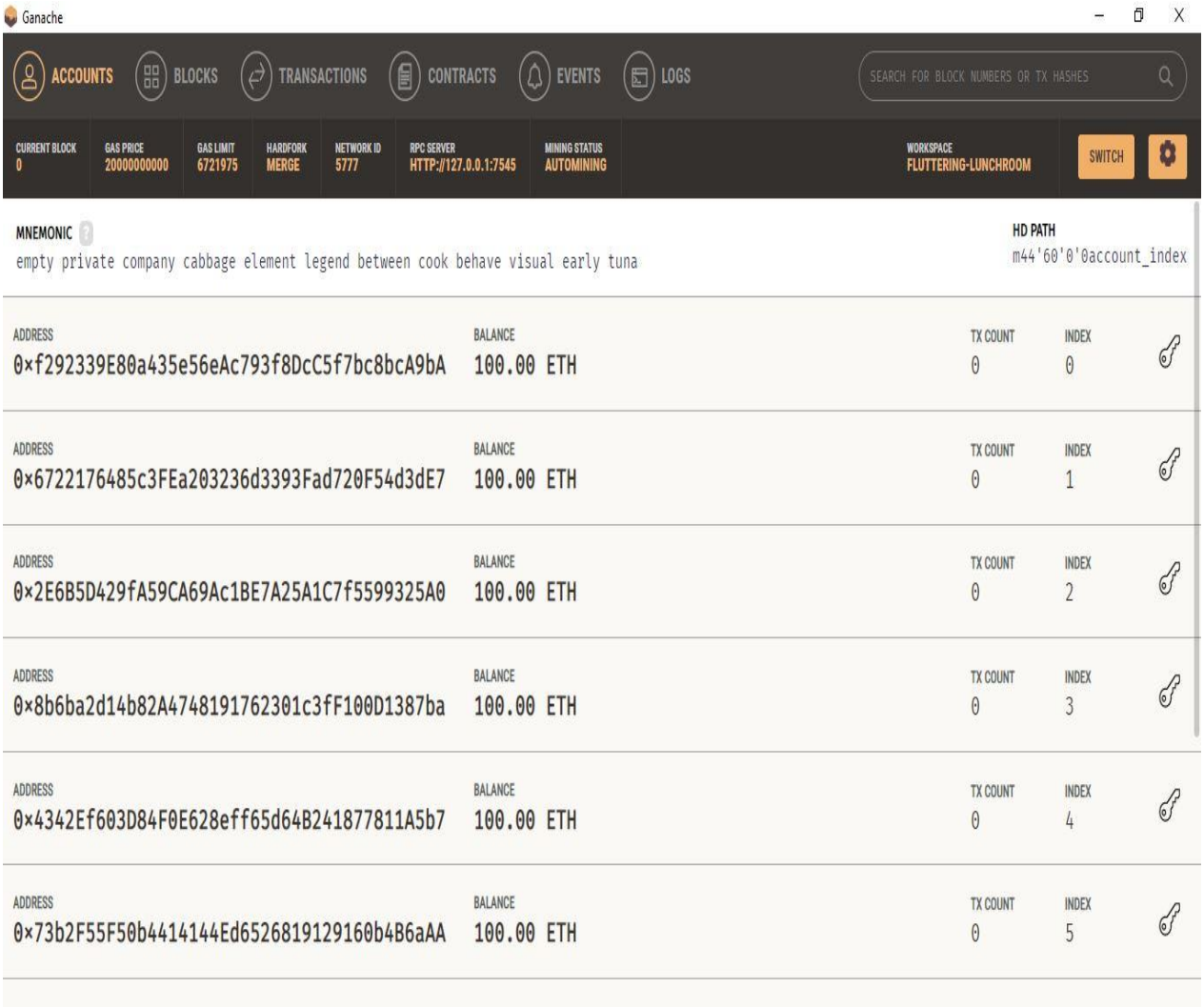

Plate 4.1: A screenshot showing successful running of Ganache

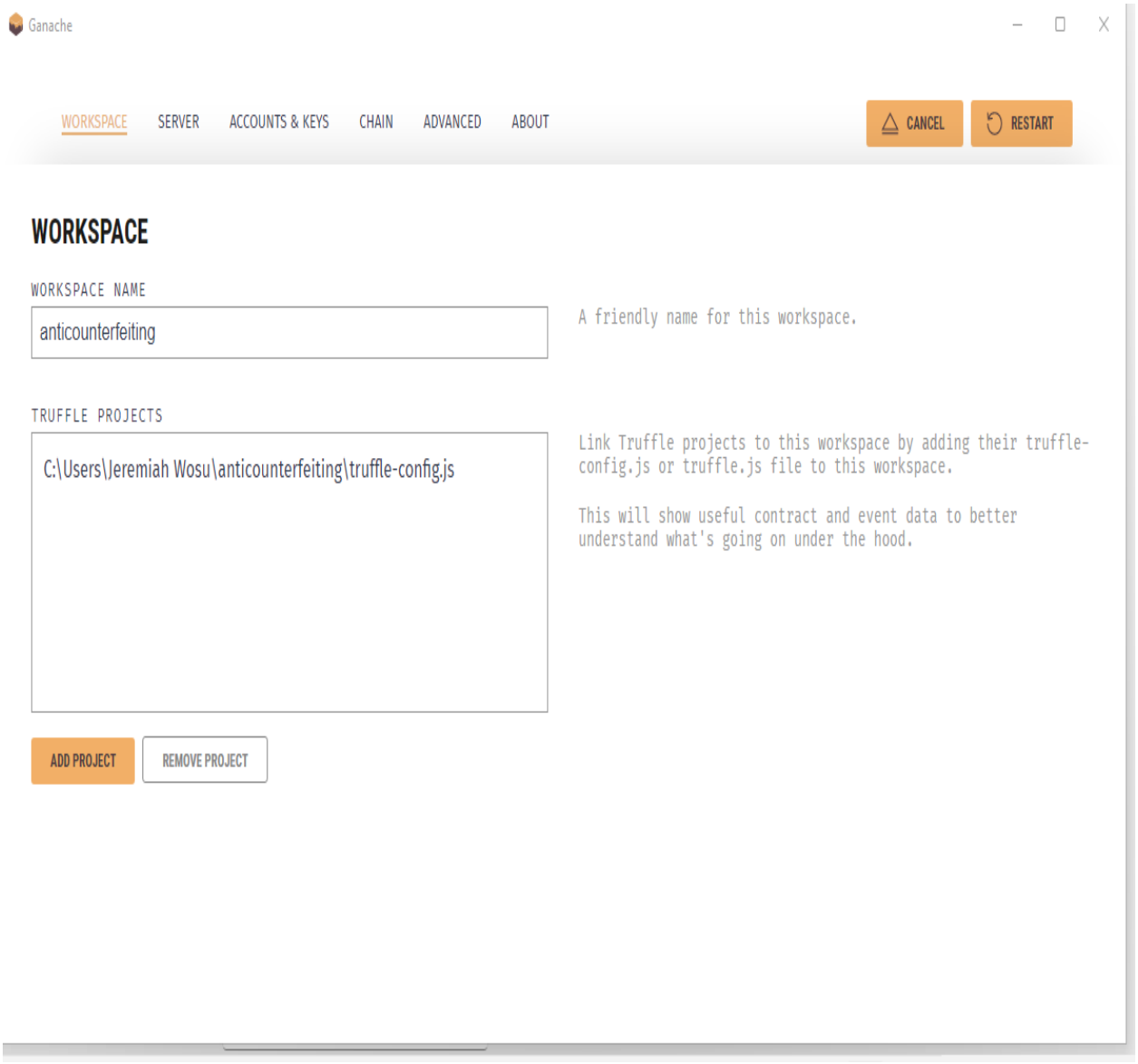

Plate 4.2: A screenshot showing successful creation of workbench on Ganache

4.1.3: Speed and Security Strength of The Developed Consensus Mechanism

Plate 4.3 is a snapshot showing the speed of execution as well as the resistance to attack of PoPC. Table 4.1 shows the result obtained by testing the developed PoPC consensus algorithm in terms of speed of execution, energy consumption and ability to resist attacks, while Table 4.2 shows the result obtained comparing PoPC with PoW, PoS, DPoS and PBFT in terms of speed of execution and ability to resist attacks.

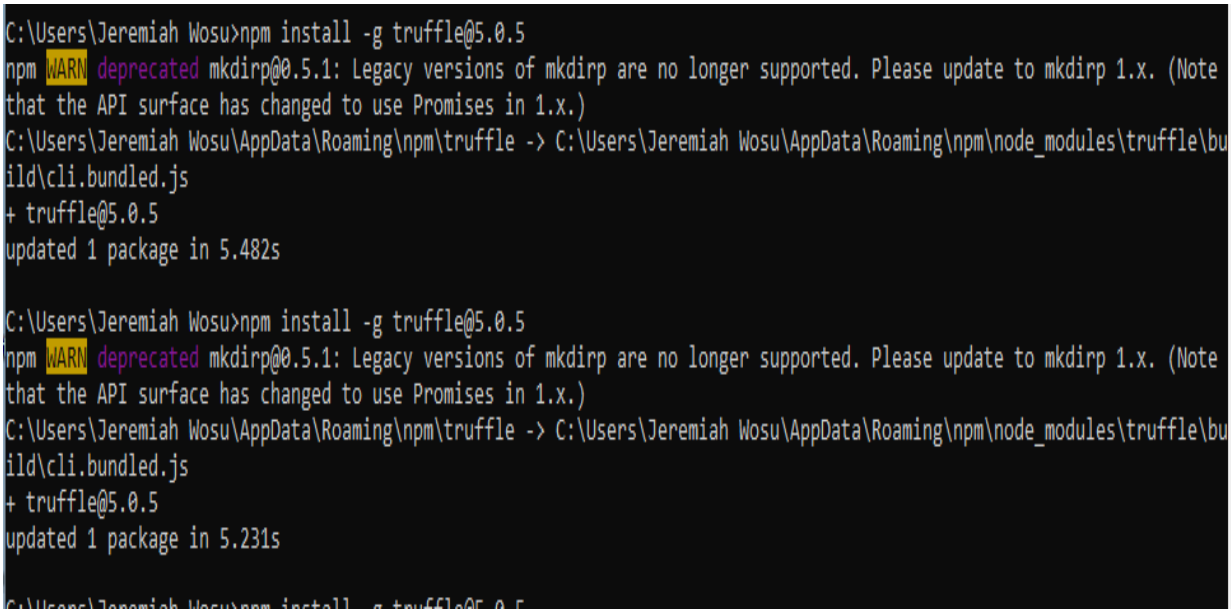

Plate 4.3: Snapshot Showing Speed of Execution and Resistance to Attack

Table 4.1 Speed of Execution, Data Integrity, Energy Consumption and Ability to Resist Attacks

Test No.	Speed of Execution (s)	Data Integrity (%)	Resist Attacks	Energy Consumption
1	5.12	98	High	Low
2	4.90	97	High	Low
3	5.21	97	High	Low
4	5.03	99	High	Low
5	5.12	98	High	Low
6	5.11	99	High	Low
7	4.84	98	High	Low
8	5.17	97	High	Low
9	4.86	98	High	Low
10	5.14	99	High	Low
Average	5.05	98	High	Low

4.1.4: Smart Contracts Compilation and Deployment to Ganache Network

Smart contracts were written in Solidity and ran through Solidity compiler so that the Ethereum Virtual Machine (EVM) can understand and execute appropriate commands. Snapshot showing that smart contracts were compiled and deployed is presented in plate 4.4

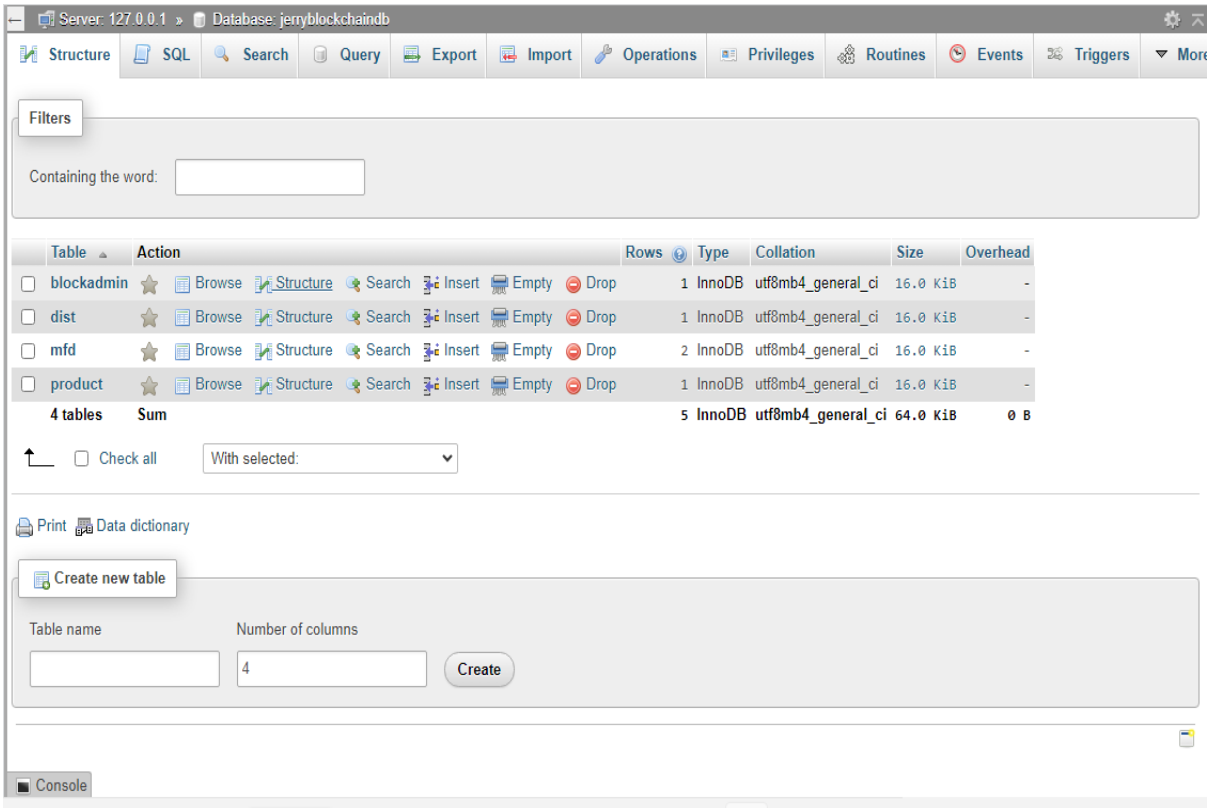

Plate 4.4: Snapshot showing that smart contracts were compiled and deployed

4.1.5: DApp Communicating With The Ganache Network

A snapshot indicating that dApp is communicating with our Ganache network is presented in Plate 4.5.

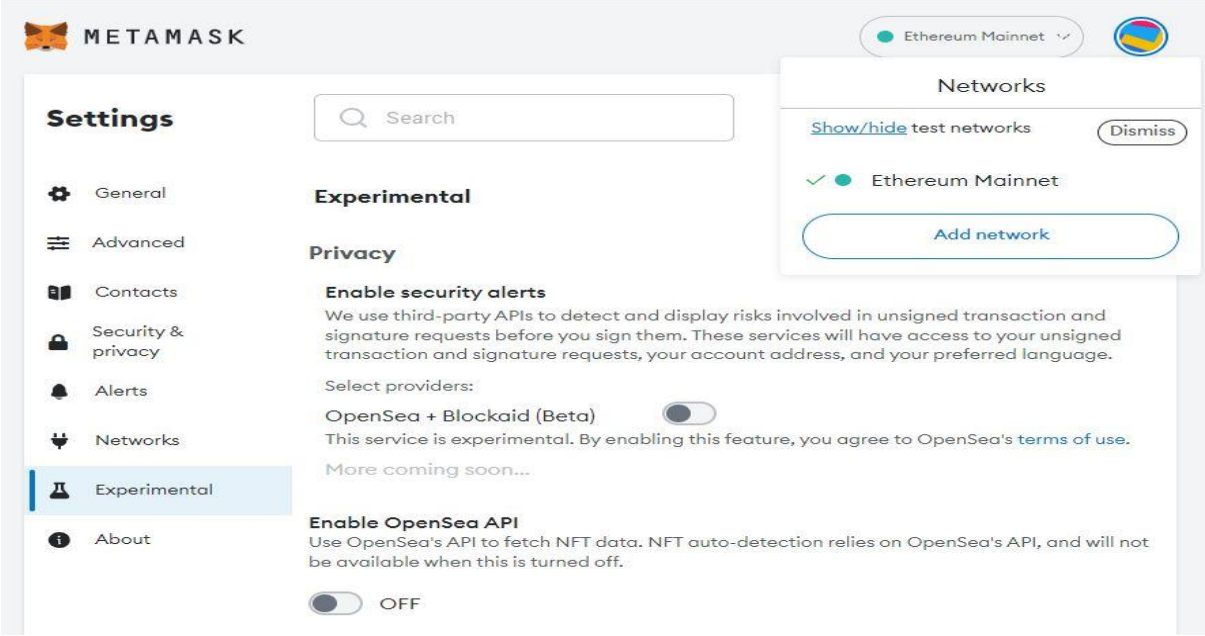

Plate 4.5: A snapshot indicating that dApp is communicating with our Ganache network.

4.1.6: **Applications for Generating And Scanning QR Codes**

Snapshots that show that the applications for generating and scanning QR codes are functioning properly is shown in Plate 4.6 and Plate 4.7.

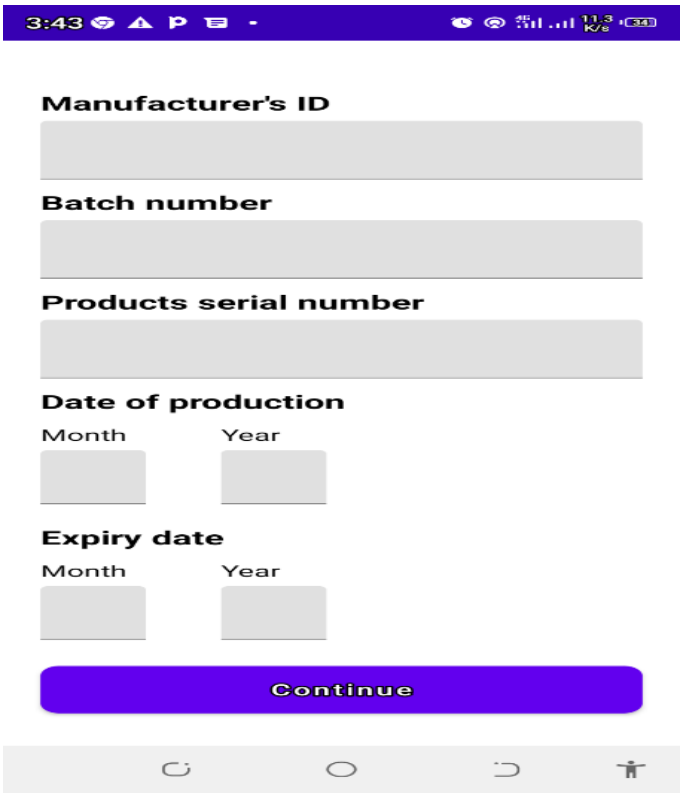

Plate 4.6: App is requesting manufacturer to input product details for generating QR code

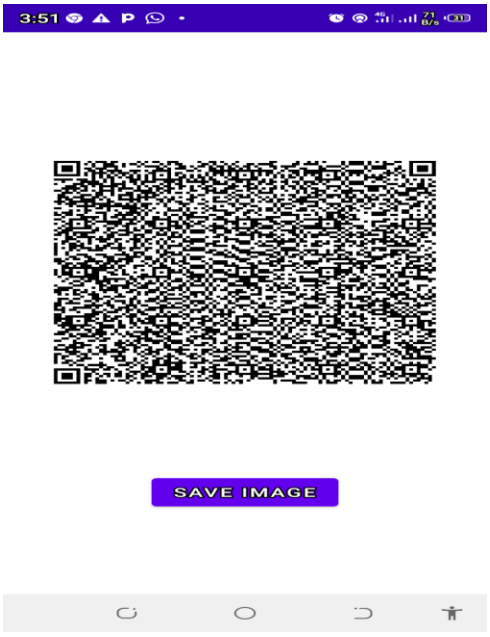

Plate 4.7: QR code generated by the appplication

4.1.7: BBACS Integration Testing

Snapshots that show that BBACS is functioning properly is shown in Plate 4.8, Plate 4.9 and Plate 4.10.

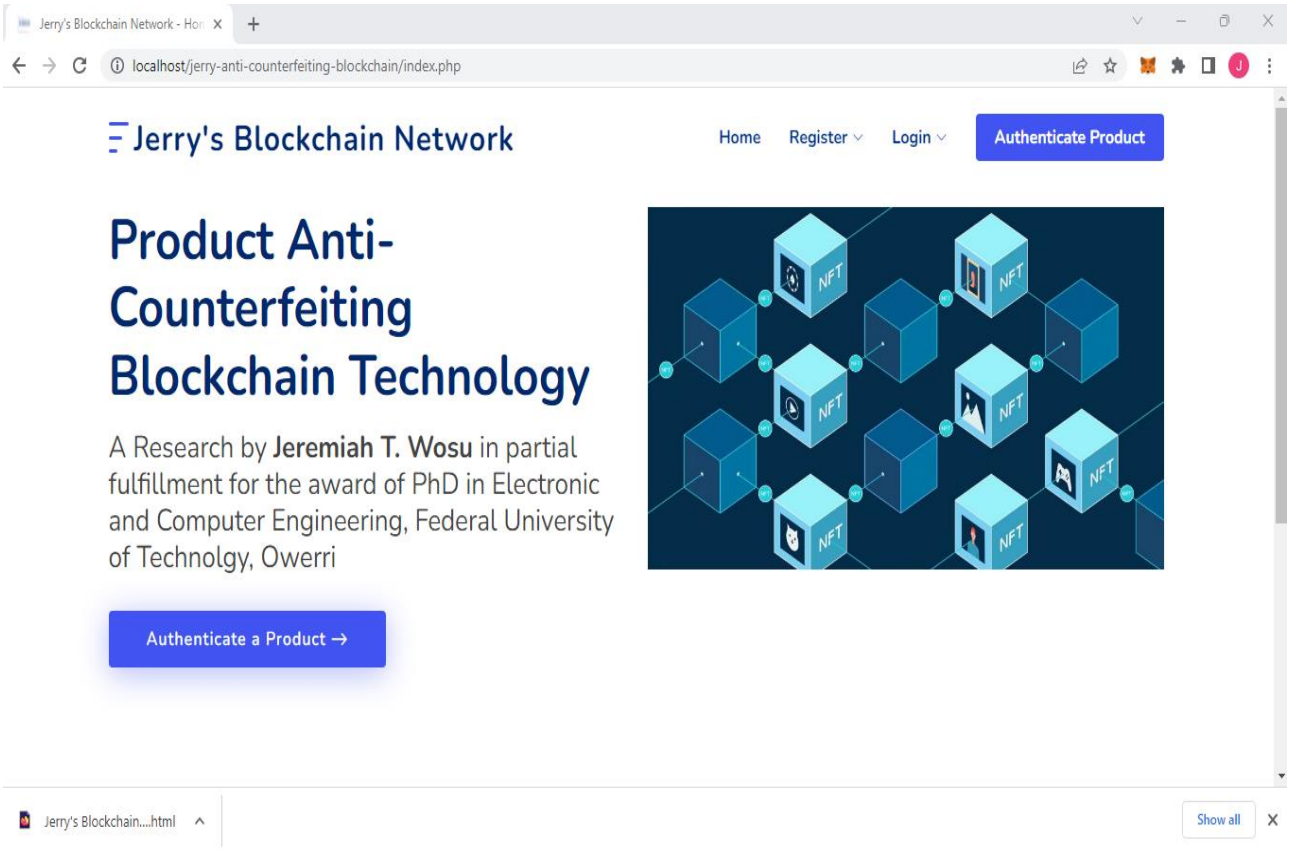

Plate 4.8: Snapshot showing BBACS Home page

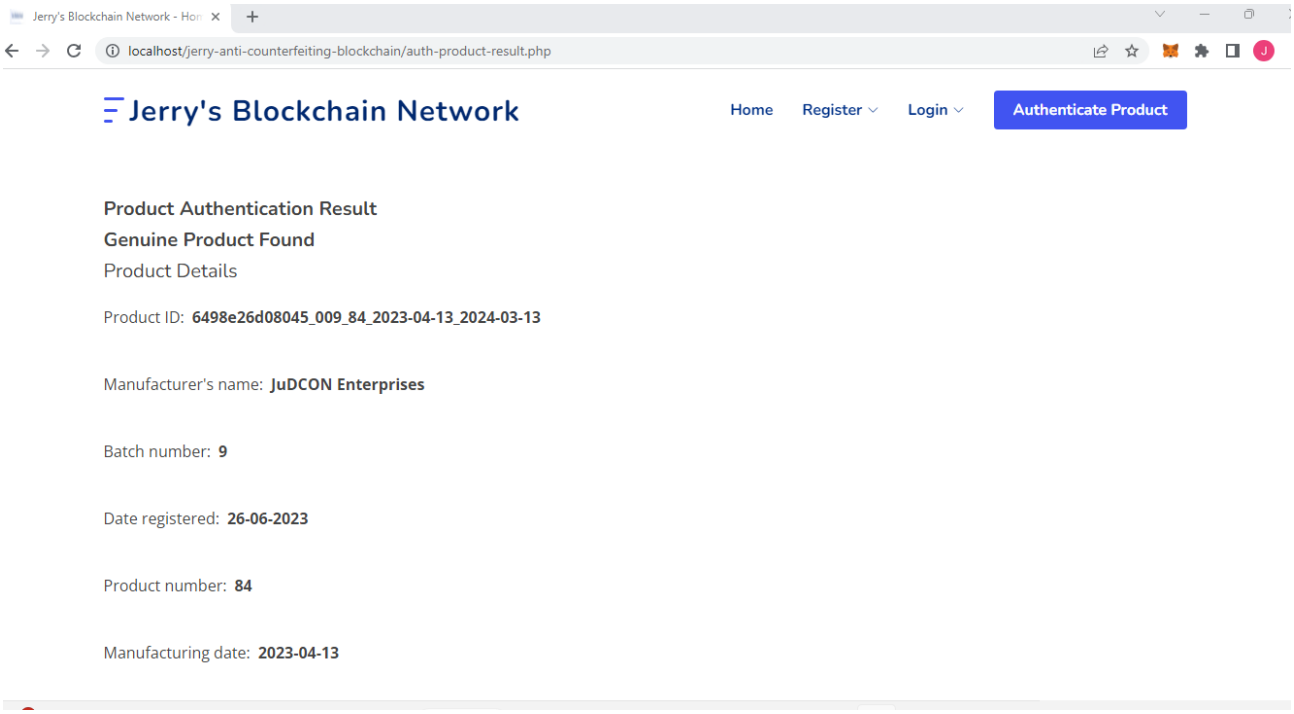

Plate 4.9: Snapshot showing successful product authentication i.e., genuine product

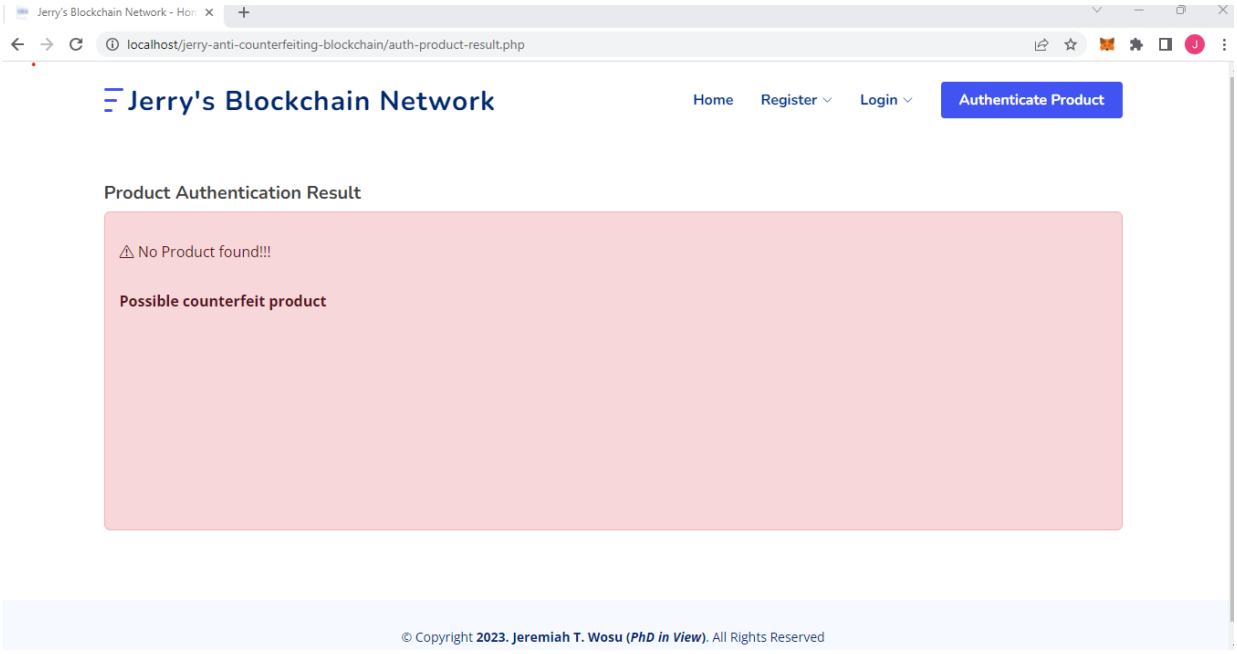

Plate 4.10: Snapshot showing failed product authentication i.e., counterfeit product

4.2 Discussions

The result presented using Figure 4.1 verifies that the UML design pattern adopted in modelling the system yielded a functional system (Cai et al, 2018; Jurgelaitis et al, 2019). Systematically following the design led to a successful development of BBACS.

This result presented in section 4.1.2 using Plates 4.1 and Plate 4.2 reveals that Truffle is communicating with our Ganache network. Thus, an indication that that Ganache is functioning properly (Attaran & Gunasekaran, 2019).

From the Table 4.1, it is obvious that PoPC consensus mechanism balances between efficiency and security. It has the following advantages:

(i) Energy is not wasted as it is in proof-of-work and many other consensus mechanisms. Typically, the election of a new signer is made by the use of random criteria and weighted depending on the product contribution of a node. So, making more attempts per second do not increase the possibility of being selected as the new signer. Hence, a reduction in time and energy can be noticed (Gao, 2019).

(ii) Faster chains are possible by using PoPC algorithm. Without the need of computing a lot of hashes to find the correct one, is possible to create a chain with a higher rate of block generation per 5 seconds (Hu et al, 2021).

(iii) More decentralization. PoPC algorithm allows for more decentralized blockchain networks which in turn enhances the security and flexibility of the system (Jang et al, 2020).

Furthermore, a comparison of PoPC with PoW, PoS, DPoS and PBFT as presented in Tables 4.2 to 4.6 reveals that PoPC is very fast, consumes less energy, yet highly secured (Lashkari & Musilek, 2021; Santiago, et al, 2021; Saleh, 2020; Seebacher & Maleshkova, 2018; Shamili & Muruganantham, 2022).

Table 4.2 PoW Speed of Execution, Energy Consumption and Ability to Resist Attacks

Test No.	Speed of Execution (s)	Resist Attacks	Energy Consumption
1	603	Very High	High
2	606	Very High	High
3	592	Very High	High
4	598	Very High	High
5	601	Very High	High
6	602	Very High	High
7	596	Very High	High
8	607	Very High	High
Average	600.5	Very High	High

Table 4.3 PoS Speed of Execution, Energy Consumption and Ability to Resist Attacks

Test No.	Speed of Execution (s)	Resist Attacks	Energy Consumption
1	56	High	Low
2	57	High	Low
3	55	High	Low
4	55	High	Low
5	56	High	Low
6	57	High	Low
7	55	High	Low
8	57	High	Low
Average	56	High	Low

Table 4.4 DPoS Speed of Execution, Energy Consumption and Ability to Resist Attacks

Test No.	Speed of Execution (s)	Resist Attacks	Energy Consumption
1	77	High	Low
2	76	High	Low
3	74	High	Low
4	75	High	Low
5	76	High	Low
6	73	High	Low
7	75	High	Low
8	74	High	Low
Average	75	High	Low

Table 4.5 PBFT Speed of Execution, Energy Consumption and Ability to Resist Attacks

Test No.	Speed of Execution (s)	Resist Attacks	Energy Consumption
1	62	High	Low
2	62	High	Low
3	62	High	Low
4	63	High	Low
5	60	High	Low
6	63	High	Low
7	63	High	Low
8	62	High	Low
Average	62.1	High	Low

Table 4.6: Comparing PoPC with PoW, PoS, DPoS and PBFT in terms of speed of execution and ability to resist attacks.

Algorithm	Speed Of Execution (s)	Energy Consumption	Resist Attacks
PoPC	5	Low	High
PoW	600	High	Very High
PoS	56	Low	High
DPoS	75	Low	High
PBFT	62	Low	High

From Tables 4.6 it is obvious that PoPC is extremely faster than other blockchain consensus protocols, yet it is highly secured.

However, PoPC has one major drawback: The main drawback of PoPC algorithms is that a potential monopoly from the major product contributors i.e., major manufacturers of the network could be possible.

Results presented in section 4.4 using Plate 4.4 clearly indicates that smart contracts were compiled and successfully deployed (Ashraf et al, 2020; Kim & Lee, 2020). Also, Plate 4.5 established that our dApp is interacting with our Ganache network as MetaMask was able connect to our Ganache network, and let users (e.g., manufacturer and distributor) login and input data via our dApp. Furthermore, Plate 4.6 and Plate 4.7 confirmed that the applications for generating and scanning QR codes for authenticating products enrolled in the developed blockchain system are functioning properly (Agyekum et al, 2021).

Results presented using Plate 4.8, Plate 4.9 and Plate 4.10 confirmed that the BBACS is functioning properly. This was achieved by registering manufacturers and distributors in the network. Then, manufacturers registered their products using the QR code generating

application. Finally, end users used the QR code scanning application to scan QR code embedded on product and were alerted as to whether the product is genuine or counterfeit.

CHAPTER FIVE

CONCLUSION AND RECOMMENDATIONS

5.1 Conclusion

This work has been able to achieve its primary objective of developing a blockchain-based anti-counterfeiting system with enhanced consensus algorithm leveraging product inherent features, copy-sensitive QR codes and location information. First, it rummaged that blockchain technology has the potential to redefine the battle against counterfeiting as it is chronologically updated and cryptographically sealed, thereby guaranteeing immutable transactions, which are publicly available and distributed globally. Thus, the work proposed a user-friendly blockchain-based solution for detecting and reporting counterfeit products.

The developed system is unique as it combines product texture (which is an inherent feature), copy-sensitive QR Code, location information (i.e., longitude and latitude coordinates) as well as Track and Trace technologies. It utilizes a method for detecting counterfeit by capture of texture which is indissolubly linked with the product.

This work designed and applied a new enhanced consensus algorithm named proof of product contribution (PoPC) that is fully decentralized and balances between efficiency and security. The system prototype is a distributed application (dApp) with a supporting blockchain network. Ganache, a private Ethereum blockchain network was setup to serve as the backend platform due to the fact that our system requires multi-state and flexibility in block storage to record data as well as short block time. Furthermore, smart contracts were writing in Solidity programming language, compiled and deployed to the developed anti-counterfeiting blockchain network via Truffle.

The developed distributed application allows manufacturers and distributors to enroll on the private Ganache blockchain-based anti-counterfeiting network. Registered

manufacturers are also granted the privilege to register their products utilizing the developed QR code generator. Finally, end-users apply the developed QR code scanner to scan QR codes embedded on products, and verify from the network whether the product is genuine or counterfeit.

5.2 Recommendations for Further Work

Recommendation is made for the adoption and rapid deployment of the blockchain-based solution developed in this work for fighting counterfeiting in Nigeria. Thus, the blockchain network and the associated APIs should be hosted on the internet. This will help in curbing the unprecedented level of counterfeiting as well as the associated health and economic implications in this country.

Although this work has made remarkable contributions towards the applicability of blockchain technology in identifying fake products, further works with objectives of improving what has been presented in this thesis are encouraged. Hence, the following recommendations are made:

i. Enrollment of products on the network: This work developed a method for registering products on the network, whereby manufacturers manually enroll products by entering product details and scanning product texture to generate QR code. Future work should endeavor to develop automated means of enrolling products to enable manufacturers register large number of their products within the shortest time possible.
ii. PoPC mechanism should be designed to utilize random numbers in selecting new block signers to avoid monopoly in the system
iii. A blockchain explorer should be developed to publish confirmed cases of counterfeiting,

5.3 Contribution to Knowledge

This work has greatly contributed to the knowledge by developing a decentralized blockchain-based anti-counterfeiting system with enhanced consensus algorithm. The custom consensus algorithm, PoPC, is the first of its kind. It is completely decentralized and balances between efficiency and security.

This work is also unique in that it is the first and only one at present to leverage product inherent features, copy-sensitive QR codes and location information in combination with blockchain technology in the development of an anti-counterfeiting system.

References

Abichandani, P., Lobo., D., Kabrawala, S., & McIntyre, W. (2021). Secure Communication for Multiquadrotor Networks Using Ethereum Blockchain. *IEEE Internet of Things Journal*, 8(3), 1783-1796. https://doi.org/10.1109/JIOT.2020.3015716

Agyekum, K. O-B. O., Xia, Q., Sifah, E. B., Cobblah, C. N. A., Xia H., & Gao, J. (2021). A Proxy Re-Encryption Approach to Secure Data Sharing in the Internet of Things Based on Blockchain. IEEE Systems Journal, https://doi.org/10.1109/JSYST.2021.3076759

Alajmi, M., Elashry, I., El-Sayed, H. S., & Farag-Allah, O. S. (2020). Steganography of Encrypted Messages Inside Valid QR Codes. *IEEE Access*, 8, 27861-27873. https://doi.org/10.1109/ACCESS.2020.2971984

Ali, A. M. & Farhan, A. K. (2020). Enhancement of QR Code Capacity by Encrypted Lossless Compression Technology for Verification of Secure E-Document. *IEEE Access*, 8, 27448-27458. https://doi.org/10.1109/ACCESS.2020.2971779

Arora, S. (2022). What is Blockchain Technology and How Does It Work? https://www.simplilearn.com/tutorials/blockchain-tutorial/blockchain-technology

Ashraf, I., Ma, X., Jiang, B., & Chan, W. K. (2020). GasFuzzer: Fuzzing Ethereum Smart Contract Binaries to Expose Gas-Oriented Exception Security Vulnerabilities". *IEEE Access*, 8, 99552-99564. https://doi.org/10.1109/ACCESS.2020.2995183

Attaran, M., & Gunasekaran, A. (2019). Blockchain-enabled technology: The emerging technology set to reshape and decentralize many industries. International Journal of Applied Decision Sciences, 12(4), 424-444. https://doi.org/10.1504/IJADS.2019.102642

Binance Academy. (2018). What Is a Blockchain Consensus Algorithm? Binance Academy. https://academy.binance.com/en/articles/what-is-a-blockchain-consensus-algorithm

Binance Academy. (2019). Peer-to-Peer Networks Explained. Binance Academy. https://academy.binance.com/en/articles/peer-to-peer-networks-explained

Binance Academy. (2020). What Makes a Blockchain Secure? https://academy.binance.com/en/articles/what-makes-a-blockchain-secure

Cai, W., Wang, Z., Ernst, J. B., Hong, Z., Feng, C., & Leung, V. C. M. (2018). Decentralized Applications: The Blockchain-Empowered Software System. *IEEE Access*, 6, 53019-53033. https://doi.org/10.1109/ACCESS.2018.2870644

Cai, Z., Qu, J., Liu P., & Yu, J. (2019). A Blockchain Smart Contract Based on Light-Weighted Quantum Blind Signature. *IEEE Access, vol. 7, pp. 138657-138668,* https://doi.org/10.1109/ACCESS.2019.2941153

Chen, P., Kuo, T., & Wu, J. (2021). A Study of the Applicability of Ideal Lattice-Based Fully Homomorphic Encryption Scheme to Ethereum Blockchain. *IEEE Systems Journal, 15, 1528-1539.* https://doi.org/10.1109/JSYST.2021.3064053

Daoud, E. (2019). Decentralizing of Transparency: Using Blockchain to Reduce Counterfeiting. https://doi.org/10.33965/es2019_201904L011

Du, M., Chen, Q., & Ma, X. (2020). MBFT: A New Consensus Algorithm for Consortium Blockchain. *IEEE Access*, 8, 87665-87675. https://doi.org/10.1109/ACCESS.2020.2993759

Durga, R., Poovammal, E., Ramana, K., Jhanveri, R. H., Singh, S., & Yoon, B. (2022). CES Blocks -A Novel Chaotic Encryption Schemes Based Blockchain System for an IoT Environment. IEEE Access, https://doi.org/10.1109/ACCESS.2022.3144681

Dwivedi, V., Norta, A., Wulf, A., Leiding, B., Saxena S., & Udokwu, C. (2021). A Formal Specification Smart-Contract Language for Legally Binding Decentralized Autonomous Organizations. *IEEE Access, vol. 9, pp. 76069-76082,* https://doi.org/10.1109/ACCESS.2021.3081926

Dwivedi, V.K., Pattanaik, V., Deval, V., Dixit, A., Norta, A., & Draheim, D. (2021). Legally Enforceable Smart-Contract Languages. *ACM Computing Surveys* (CSUR), 54, 1 - 34.

Ehioghae, E,. Idowu, S., & Ebiesuwa, O. (2021). Enhanced Drug Anti-Counterfeiting and Verification System for the Pharmaceutical Drug Supply Chain using Blockchain. *International Journal of Computer Applications.* 174 975-8887. https://doi.org/10.5120/ijca2021921105

Euromoney Learning. (2020). Blockchain Explained: What is Blockchain? https://www.euromoney.com/learning/blockchain-explained/what-is-blockchain

Fu, Z., Cheng, Y., & Yu, B. (2018). Visual Cryptography Scheme with Meaningful Shares Based on QR Codes. *IEEE Access*, 6, 59567-59574. https://doi.org/10.1109/ACCESS.2018.2874527

Gao, J. (2019). Guided, Automated Testing of Blockchain-Based Decentralized Applications". *IEEE/ACM 41st International Conference on Software Engineering: Companion Proceedings (ICSE-Companion)*, (138-140). https://doi.org/10.1109/ICSE-Companion.2019.00059

Gimenez-Aguilar, M., De Fuentes, J. M., González-Manzano, L. & Camara, C. (2021). Zephyrus: An Information Hiding Mechanism Leveraging Ethereum Data Fields. *IEEE Access*, 9, 118553-118570. https://doi.org/10.1109/ACCESS.2021.3106713

Gutiérrez-Agüero, I., Anguita, S., Larrucea, X., Gomez-Goiri, A., & Urquizu, B. (2021). Burnable Pseudo-Identity: A Non-Binding Anonymous Identity Method for

Ethereum. *IEEE Access*, 9, 108912-108923. https://doi.org/10.1109/ACCESS.2021.3101302

Haber, S., & Stornetta, W. S. (2022). Blockchain Explained: What Is Blockchain & How Does It Work? https://www.softwaretestinghelp.com/blockchain-tutorial/

Harz, D., & Knottenbelt, W.J. (2018). Towards Safer Smart Contracts: A Survey of Languages and Verification Methods. ArXiv, abs/1809.09805.

Hayes A. (2021). Quick Response (QR) Code. *Financial Technology & Automated Investing*. https://www.investopedia.com/terms/q/quick-response-qr-code.asp.

Heo, H., & Shin, S. (2021). Behind Block Explorers: Public Blockchain Measurement and Security Implication. 2021 IEEE 41st International Conference on Distributed Computing Systems (ICDCS), 216-226. https://doi.org/10.1109/ICDCS51616.2021.00029

Horn, C., Blankenburg, M., & Kruger, J. (2012). Automated Detection of Counterfeit Consumer Goods using Product Inherent Features. www.semanticscholar.org/paper/Automated-Detection-of-Counterfeit-Consumer-Goods-Horn-Blankenburg/9e91b27a5ce87913668beaa2102cf840be18caa1

Hu, Q., Yan, B., Han, Y., & Yu, J. (2021). An Improved Delegated Proof of Stake Consensus Algorithm. *Procedia Computer Science*. https://doi.org/10.1016/J.PROCS.2021.04.109

Huang, J., Zhou, T., Chang, H., & Xie, D. (2018). An Optimized Cyclic Weight Algorithm of (47, 24, 11) QR Code and Hardware Implementation. *IEEE Access*, 6, 36995-37002.https://doi.org/10.1109/ACCESS.2018.2849700.

Huang, P., Chang, C., Li, Y., & Liu, Y. (2020). Efficient QR Code Secret Embedding Mechanism Based on Hamming Code. *IEEE Access*, 8, 86706-86714. https://doi.org/10.1109/ACCESS.2020.2992694

IBM. (2022). What are smart contracts on blockchain? https://www.ibm.com/topics/smart-contracts

Iliyasu, A. M. (2019). Cellular-Automated Protocol to Safeguard Confidentiality of QR Codes. *IEEE Access*, 7, 144451-144471. https://doi.org/10.1109/ACCESS.2019.2943754

Jakubović, A., & Velagic, J. (2018). Image Feature Matching and Object Detection Using Brute-Force Matchers. *2018 International Symposium ELMAR, 83-86.*

Jang, H., Han, S., & Kim, J.H. (2020). User Perspectives on Blockchain Technology: User-Centered Evaluation and Design Strategies for DApps. *IEEE Access, 8, 226213-226223.*

JetBrains, (2020). *"Kotlin for cross-platform mobile development". JetBrains: Developer Tools for Professionals and Teams. Retrieved 20 August 2020.*

Jiao, J., Kan, S., Lin, S., Sanán, D., Liu, Y.P., & Sun, J. (2018). Semantic Understanding of Smart Contracts: Executable Operational Semantics of Solidity. ArXiv, abs/1804.01295. https://doi.org10.1109/SP40000.2020.00066

Johar, M. D. (2020). Secure Quick Response Based on Dynamic Quick Response Code. *Journal of Computational and Theoretical Nanoscience*, 17, 1090-1098.

Jurgelaitis, M., Drungilas, V., Ceponiene, L., Butkiene, R., & Vaičiukynas, E. (2019). Modelling principles for blockchain-based implementation of business or scientific processes. IVUS.

Kabashkin I. (2017) Risk Modelling of Blockchain Ecosystem. NSS 2017. Lecture Notes in Computer Science, vol 10394. Springer, Cham. https://doi.org/10.1007/978-3-319-64701-2_5

Kaifeng, Y., Zhang, Y., Chen, Y., Li, Y., Zhao, L., Rong, C., & Chen, L. (2021). A Survey of Decentralizing Applications via Blockchain: The 5G and Beyond

Perspective. *IEEE Communications Surveys & Tutorials*, 23(4), 2191-2217, Fourthquarter 2021. https://doi.org/10.1109/COMST.2021.3115797.

Kannengiesser, N., Lins, S., Sander, C., Winter, K., Frey, H., & Sunyaev, A. (2021). Challenges and Common Solutions in Smart Contract Development. *IEEE Transactions on Software Engineering*. https://doi.org/10.1109/TSE.2021.3116808

Karami, E., Prasad, S., & Shehata, M.S. (2017). Image Matching Using SIFT, SURF, BRIEF and ORB: Performance Comparison for Distorted Images. ArXiv, abs/1710.02726.

Kashyap, B. (2021). Introduction to smart contracts: What is a smart contract? https://ethereum.org/en/developers/docs/smart-contracts/

Keith, W., & Jith, L. Z. (2020). The Evolution of Counterfeit Luxury Consumption. *Research Handbook on Luxury Branding (*pp. 265 – 281).

Kemmoe, V.Y., Stone, W., Kim, J., Kim, D., & Son, J. (2020). Recent Advances in Smart Contracts: A Technical Overview and State of the Art. *IEEE Access, 8, 117782-117801.*

Khan, S.N., Loukil, F., Ghedira, C., Benkhelifa, E., & Bani-Hani, A.I. (2021). Blockchain smart contracts: Applications, challenges, and future trends. *Peer-to-Peer Networking and Applications*, (1 – 25).

Khan, U., An, Z. Y., & Imran, A. (2020). A Blockchain Ethereum Technology-Enabled Digital Content: Development of Trading and Sharing Economy Data. *IEEE Access*, 8, 217045-217056. https://doi.org/10.1109/ACCESS.2020.3041317

Kim K. B., & Lee, J. (2020). Automated Generation of Test Cases for Smart Contract Security Analyzers. *IEEE Access, vol. 8, pp. 209377-209392* https://doi.org10.1109/ACCESS.2020.3039990

Kim, H., Jang, J., Park, S., & Lee, H. N. (2021). Error-Correction Code Proof-of-Work on Ethereum". *IEEE Access*, 9, 35942-135952. https://doi.org/10.1109/ACCESS.2021.3113522

Kotlin, (2020). *"Kotlin for JavaScript - Kotlin Programming Language". Retrieved 20 August 2020.*

Kuzuno, H., & Karam, C. (2017). Blockchain explorer: An analytical process and investigation environment for bitcoin. 2017 APWG Symposium on Electronic Crime Research (eCrime), 9-16. https://doi.org/10.1109/ECRIME.2017.7945049

Lai, V. (2018). Introduction to Cryptography in Blockchain Technology. https://crushcrypto.com/cryptography-in-blockchain/

Lai, W., Hsueh, C., & Wu, J. (2019). A Fully Decentralized Time-Lock Encryption System on Blockchain. 2019 IEEE International Conference on Blockchain (Blockchain), 302-307. https://doi.org/10.1109/Blockchain.2019.00047

Lashkari, B., & Musilek, P. (2021). A Comprehensive Review of Blockchain Consensus Mechanisms. *IEEE Access*, 9, 43620-43652. https://doi.org/10.1109/ACCESS.2021.3065880

Lee, C., Kim, H., Maharjan, S., Ko, K., & Hong, J.W. (2019). Blockchain Explorer based on RPC-based Monitoring System. 2019 IEEE International Conference on Blockchain and Cryptocurrency (ICBC), 117-119. https://doi.org/10.1109/BLOC.2019.8751468

Li, X., Mei, Y., Gong, J., Xiang, F., & Sun, Z. (2020). A Blockchain Privacy Protection Scheme Based on Ring Signature. IEEE Access, 8, 76765-76772. https://doi.org/10.1109/ACCESS.2020.2987831

Liang, W., Zhang, D., Lei, X., Tang, M., Li, K., & Zomaya, A. (2021). Circuit Copyright Blockchain: Blockchain-Based Homomorphic Encryption for IP Circuit

Protection. IEEE Transactions on Emerging Topics in Computing, 9, 1410-1420. https://doi.org/10.1109/tetc.2020.2993032

Lin, Q., Zhao, J., Fu, G., & Yuan, Z. (2019). Fast Multi Semantic Pyramids via Cross Fusing Inherent Features for Different-Scale Detection. *IEEE Access, 7, 98374-98386*. https://doi.org10.1109/ACCESS.2019.2930083

Ling, X., Le, Y., Wang, J., Ding, Z., & Gao, X. (2021). Practical Modeling and Analysis of Blockchain Radio Access Network. *IEEE Transactions on Communications, 69, 1021-1037*

Liu J., & Liu, Z. (2019). A Survey on Security Verification of Blockchain Smart Contracts. *IEEE Access, vol. 7, pp. 77894-77904.* https://doi.org10.1109/ACCESS.2019.2921624

Liu, S., Fu, Z. & Yu, B. (2019). Rich QR Codes with Three-Layer Information Using Hamming Code. *IEEE Access*, 7, 78640-78651. https://doi.org/10.1109/ACCESS.2019.2922259

Liu, X., Chen, R., Chen, Y., & Yuan, S. (2018). Off-chain Data Fetching Architecture for Ethereum Smart Contract. 2018 International Conference on Cloud Computing, Big Data and Blockchain (ICCBB), 1-4.

Ma, J., Lin, S., Chen, X., Sun, H., Chen, Y., & Wang, H. (2020). A Blockchain-Based Application System for Product Anti-Counterfeiting. *IEEE Access*, 8, 77642-77652. https://doi.org/10.1109/ACCESS.2020.2972026

Mao, D., Wang, F., Wang, Y., & Hao, Z. (2019). Visual and User-Defined Smart Contract Designing System Based on Automatic Coding. *IEEE Access*, 7, 73131-73143. https://doi.org/10.1109/ACCESS.2019.2920776

Memon, R.A., Li, J., & Ahmed, J. (2019). Simulation Model for Blockchain Systems Using Queuing Theory. Electronics. https://doi.org/10.3390/ELECTRONICS8020234

Memon, R.A., Li, J., Ahmed, J., Khan, A., Nazir, M.I., & Mangrio, M.I. (2018). Modeling of Blockchain Based Systems Using Queuing Theory Simulation. 2018 15th International Computer Conference on Wavelet Active Media Technology and Information Processing (ICCWAMTIP), 107-111.

Mihaljevic, M. J. (2020). A Blockchain Consensus Protocol Based on Dedicated Time-Memory-Data Trade-Off. *IEEE Access*, 8, 141258-141268. https://doi.org/10.1109/ACCESS.2020.3013199

Molina-Jiménez, C., Sfyrakis, I., Solaiman, E., Ng, I., Wong, M.W., Chun, A., & Crowcroft, J. (2018). Implementation of Smart Contracts Using Hybrid Architectures with On and Off–Blockchain Components. 2018 IEEE 8th International Symposium on Cloud and Service Computing (SC2), 83-90.

Odumade, D. (2020). 10 Most Commonly Counterfeited Products in Nigeria. https://chekkitapp.com/blog/10-most-commonly-counterfeited-products-in-nigeria/

Pacific Research Institute. (2020). Counterfeit Drugs Harm Patients, Economy, Innovation. Center for Medical Economics and Innovation at Pacific Research Institute. https://medecon.org/new-brief-counterfeit-drugs-harm-patients-economy-innovation/

Pang, G., Yang, G., & Pang, Z. (2021). Review of Robot Skin: A Potential Enabler for Safe Collaboration, Immersive Teleoperation, and Affective Interaction of Future Collaborative Robots. IEEE Transactions on Medical Robotics and Bionics, vol. 3, no. 3, pp. 681-700. https://doi.org10.1109/TMRB.2021.3097252

Pinna, A., Ibba, S., Baralla, G., Tonelli, R., & Marchesi, M. (2019). A Massive Analysis of Ethereum Smart Contracts Empirical Study and Code Metrics. *IEEE Access*, 7, 78194-78213. https://doi.org/10.1109/ACCESS.2019.2921936

Pinto, J. R., Cardoso J. S., & Lourenço, A. (2018). Evolution, Current Challenges, and Future Possibilities in ECG Biometrics. IEEE Access, vol. 6, pp. 34746-34776. https://doi.org10.1109/ACCESS.2018.2849870

Rawat, A. S. (2021). Blockchain Mining: Types and Uses. Analyticssteps. https://www.analyticssteps.com/blogs/blockchain-mining-types-and-uses

Rocha, H., & Ducasse, S. (2018). Preliminary Steps Towards Modeling Blockchain Oriented Software. 2018 IEEE/ACM 1st International Workshop on Emerging Trends in Software Engineering for Blockchain (WETSEB), 52-57. https://doi.org/10.1145/3194113.3194123

Saleh, F. (2020). Blockchain Without Waste: Proof-of-Stake. *Information Systems & Economics eJournal*. https://doi.org/10.2139/ssrn.3183935

Santiago, C., Ren, S., Lee, C., & Ryu, M. (2021). Concordia: A Streamlined Consensus Protocol for Blockchain Networks. *IEEE Access*, 9, 13173-13185. https://doi.org/10.1109/ACCESS.2021.3051796

Sayeed, D., Marco-Gisbert, H., & Caira, T. (2020). Smart Contract: Attacks and Protections. IEEE Access, 8, 24416-24427. https://doi.org10.1109/ACCESS.2020.2970495

Sayyad, T J., Adil, T. M., Vaishnavi, S. R., Mukesh, J. P., & Devisha, B. P. (2021). Fake Product Identification Using Blockchain Technology. *International Journal of Future Generation Communication and Networking*, 14(1).

Seebacher, S., & Maleshkova, M. (2018). A Model-driven Approach for the Description of Blockchain Business Networks. HICSS. https://doi.org/10.24251/HICSS.2018.442

Seidel, M.L. (2018). Questioning centralized organizations in a time of distributed trust. *Journal of Management Inquiry, Vol. 27, No. 1, pp.40–44.*

Shahaab, A., Lidgey, B., Hewage, C., & Khan, I. (2019). Applicability and Appropriateness of Distributed Ledgers Consensus Protocols in Public and Private Sectors: A Systematic Review. *IEEE Access*, 7, 43622-43636. https://doi.org/10.1109/ACCESS.2019.2904181

Shamili, P., & Muruganantham, B. (2022). Design and Implementation Considerations of POA Network Architecture in the Ethereum Blockchain. *Webology*. https://doi.org/10.14704/web/v19i1/web19358

Shermin, V. (2017). Disrupting governance with blockchains and smart contracts. *Strategic Change, Vol. 26, No. 5, pp.499–509.*

Sreeshma, C. M., Manu, M., & GopaKumar, G. (2018). Identification of Long Non-coding RNA from inherent features using Machine Learning Techniques. *2018 International Conference on Bioinformatics and Systems Biology (BSB), pp. 97-102*. https://doi.org10.1109/BSB.2018.8770699

Srivatsa, D., & Aakash, N., & Sahisnu, S. (2021). A Product Authentication Scheme for Supply Chain system via Smart Contracts using Blockchain Technology and Facial Recognition. *Journal of Physics*: Conference Series. 1767. 012057. 10.1088/1742-6596/1767/1/012057.

Tavares, I. D. (2020). Counterfeiting Of Fake Drugs in Africa: Current Situation, Causes and Countermeasures. Mondaq. https://www.mondaq.com/nigeria/trademark/988968/counterfeiting-of-fake-drugs-in-africa-current-situation-causes-and-countermeasures

Tian, G., Wang, Q., Zhao, Y., Guo, L., Sun, Z., & Lv, L. (2020). Smart Contract Classification with a Bi-LSTM Based Approach. IEEE Access, 8, 43806-43816. https://doi.org10.1109/ACCESS.2020.2977362

Tkachenko, I., & Destruel, C. (2018). Exploitation of redundancy for pattern estimation of copy-sensitive two level QR code. *IEEE International Workshop on Information Forensics and Security (WIFS)*, (1-6). https://doi.org/10.1109/WIFS.2018.8630792.

Tkachenko, I., Kucharczakr, F., Destruel, C., Strauss, O., & Puech, W. (2018). Copy Sensitive Graphical Code Quality Improvement Using a Super-Resolution Technique. *25th IEEE International Conference on Image Processing (ICIP)*, (3808-3812). https://doi.org/10.1109/ICIP.2018.8451839

Trojanowska, N., Kedziora, M., Hanif, M., & Song, H. (2020). Secure Decentralized Application Development of Blockchain-based Games. *IEEE 39th International Performance Computing and Communications Conference (IPCCC)*, (1-8). https://doi.org/10.1109/IPCCC50635.2020.9391556.

Tushar, W., Saha, T.K., Yuen, C., Smith, D.B., & Poor, H.V. (2020). Peer-to-Peer Trading in Electricity Networks: An Overview. *IEEE Transactions on Smart Grid*, 11, 3185-3200. https://doi.org/10.1109/TSG.2020.2969657

Varela-Vaca, Á.J., & Quintero, A.M. (2021). Smart Contract Languages. *ACM Computing Surveys* (CSUR), 54, 1 - 38.

Vladyko, A., Spirkina, A., & Elagin, V. (2021). Towards Practical Applications in Modeling Blockchain System. Future Internet, 13(5), 125. MDPI AG. Retrieved from https://doi.org/10.3390/fi13050125

Wadhwa, G. (2021). How to Simply Deploy a Smart Contract on Ethereum? https://www.geeksforgeeks.org/how-to-simply-deploy-a-smart-contract-on-ethereum/?ref=leftbar-rightbar

Wang, M., Niu, S., & Yang, X.S. (2017). A novel panoramic image stitching algorithm based on ORB. 2017 International Conference on Applied System Innovation (ICASI), 818-821.

Wang, R., Shi, Y., & Cao, W. (2019). GA-SURF: A new Speeded-Up robust feature extraction algorithm for multispectral images based on geometric algebra. Pattern Recognit. Lett., 127, 11-17.

Wang, R., Zhang, W., Shi, Y., Wang, X., & Cao, W. (2019). GA-ORB: A New Efficient Feature Extraction Algorithm for Multispectral Images Based on Geometric Algebra. *IEEE Access, 7, 71235-71244.* https://doi.org10.1109/ACCESS.2019.2918813

Wang, S., Li, D., Zhang, Y., & Chen, J. (2019). Smart Contract-Based Product Traceability System in the Supply Chain Scenario. *IEEE Access*, 7, 115122-115133. https://doi.org/10.1109/ACCESS.2019.2935873.

Wang, S., Pei, R., & Zhang, Y. (2019). EIDM: A Ethereum-Based Cloud User Identity Management Protocol. *IEEE Access*, 7, 115281-115291. https://doi.org/10.1109/ACCESS.2019.2933989

Wei, W., Lin, J., Lin, Y., & Liao, H.M. (2019). What Makes You Look Like You: Learning an Inherent Feature Representation for Person Re-Identification. *16th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS), 1-6.*

Wood, G. (2018). Ethereum: A Secure Decentralized Generalized Transaction Ledger (EIP-150). https://www.yellowpaper.io.

Wu, D. C.& Wu, Y. M. (2020). Covert Communication via the QR Code Image by a Data Hiding Technique Based on Module Shape Adjustments. *IEEE Open Journal of the Computer Society*, 1, 12-34. https://doi.org/10.1109/OJCS.2020.2984473.

Yadav, R., Tkachenko, I., Trémeau, A., & Fournel, T. (2019). *Estimation of Copy-sensitive Codes Using a Neural Approach*. 7th ACM Workshop on information Hiding and Multimedia Security, Paris, France.

Yaji, S., Bangera, K., & Neelima, B. (2018). Privacy Preserving in Blockchain Based on Partial Homomorphic Encryption System for AI Applications. 2018 IEEE 25th International Conference on High Performance Computing Workshops (HiPCW), 81-85. https://doi.org/10.1109/HIPCW.2018.8634280

Yang, F., Zhou, W., Wu, Q., Long, R., Xiong, N. N., & Zhou, M. (2019). Delegated Proof of Stake with Downgrade: A Secure and Efficient Blockchain Consensus Algorithm with Downgrade Mechanism. *IEEE Access*, 7, 118541-118555. https://doi.org/10.1109/ACCESS.2019.2935149.

Yang, J., Xu, Y., Rong, H., Du, S., & Zhang, H. (2020). A Method for Wafer Defect Detection Using Spatial Feature Points Guided Affine Iterative Closest Point Algorithm. IEEE Access, 8, 79056-79068.

Yiu, N. C. K. (2021a). Toward Blockchain-Enabled Supply Chain Anti-Counterfeiting and Traceability. *Future Internet*, 13(4), 86. https://doi.org/10.3390/fi13040086

Yiu, N. C. K. (2021b). Decentralizing Supply Chain Anti-Counterfeiting and Traceability Systems Using Blockchain Technology. *Future Internet*, 13(4), 84. https://doi.org/10.3390/fi13040084

Yiu, N. C. K. (2021c). An Empirical Analysis of Implementing Enterprise Blockchain Protocols in Supply Chain Anti-Counterfeiting and Traceability, *Computer Science ArXiv*. https://doi.org/10.13140/RG.2.2.20322.04805

Zhang, M., Wang, S., Zhang, P., He, L., Li, X., & Zhou, S. (2019). Protecting Data Privacy for Permissioned Blockchains using Identity-Based Encryption. 2019 IEEE 3rd Information Technology, Networking, Electronic and Automation Control Conference (ITNEC), 602-605. https://doi.org/10.1109/ITNEC.2019.8729244

Zhang, P., Zhang, W., & Nenghai, Y. (2019). Copy Detection Pattern-Based Authentication for Printed Documents with Multi-Dimensional Features. *7th International Conference on Information, Communication and Networks (ICICN)*, (150–157). https://doi.org/10.1109/icicn.2019.8834939

Zhaofeng, M., Xiao-chang, W., Jain, D.K., Khan, H., Hongmin, G., & Zhen, W. (2020). A Blockchain-Based Trusted Data Management Scheme in Edge Computing. IEEE Transactions on Industrial Informatics, 16, 2013-2021.

Zheng, Z., Xie, S., Dai, H., Chen, W., Chen, X., Weng, J., & Imran, M.A. (2020). An Overview on Smart Contracts: Challenges, Advances and Platforms. *Future Gener. Comput. Syst., 105, 475-491*.

Zhong, Z., Wei, S., Xu, Y., Zhao, Y., Zhou, F., Luo, F., & Shi, R. (2020). SilkViser: A Visual Explorer of Blockchain-based Cryptocurrency Transaction Data. ArXiv, abs/2009.02651.

Zhou, Y., Hu, B., Zhang, Y. & Cai, W. (2021). Implementation of Cryptographic Algorithm in Dynamic QR Code Payment System and Its Performance. *IEEE Access*, 9, 122362-122372. https://doi.org/10.1109/ACCESS.2021.3108189.

Zhu, P., Hu, J., Zhang, Y., & Li, X. (2020). A Blockchain Based Solution for Medication Anti-Counterfeiting and Traceability. *IEEE Access*, 8, 184256-184272.

Zhu, X., Li, Y., Fang, L., & Chen, P. (2020). An Improved Proof-of-Trust Consensus Algorithm for Credible Crowdsourcing Blockchain Services. *IEEE Access*, 8, 102177-102187. https://doi.org/10.1109/ACCESS.2020.2998803.

Zichichi, M., Ferretti, S., & D'Angelo, G. (2021). MOVO: a dApp for DLT-based Smart Mobility. *International Conference on Computer Communications and Networks (ICCCN)*, (1-6). https://doi.org/10.1109/ICCCN52240.2021.9522257

Zonda, D., & Meddeb, M. (2020). Proxy re-encryption for privacy enhancement in Blockchain: Carpooling use case. 2020 IEEE International Conference on

Blockchain (Blockchain), 482-489. https://doi.org/10.1109/Blockchain50366.2020.00070

Appendix 1: Configuration Source Code for the Blockchain Network

```
require('babel-register');
require('babel-polyfill');
module.exports = {
  networks: {
    development: {
      host: "127.0.0.1",          //The IP Address
      port: 7545,                 //The localhost port
      network_id: "Ganache" // the name of my network on the blockchain
    },
  },
  contracts_directory: './src/contracts/', //
  contracts_build_directory: './src/abis/',
  compilers: {
    solc: {
      optimizer: {
        enabled: true,
        runs: 200
      }
    }
  }
}
```

```
//JavaScript Background Service Worker
const isLocalhost = Boolean(
  window.location.hostname === 'localhost' ||
    // [::1] is the IPv6 localhost address.
    window.location.hostname === '[::1]' ||
    // 127.0.0.1/8 is considered localhost for IPv4.
    window.location.hostname.match(
      /^127(?:\.(?:25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?)){3}$/
    )
);

export function register(config) {
  if (process.env.NODE_ENV === 'production' && 'serviceWorker' in navigator) {
    // The URL constructor is available in all browsers that support SW.
    const publicUrl = new URL(process.env.PUBLIC_URL, window.location.href);
    if (publicUrl.origin !== window.location.origin) {
      // Our service worker won't work if PUBLIC_URL is on a different origin
      // from what our page is served on. This might happen if a CDN is used to
      // serve assets; see https://github.com/facebook/create-react-app/issues/2374
      return;
    }
    window.addEventListener('load', () => {
      const swUrl = `${process.env.PUBLIC_URL}/service-worker.js`;
```

```
    if (isLocalhost) {
      // This is running on localhost. Let's check if a service worker still exists or not.
      checkValidServiceWorker (swUrl, config);
      // Add some additional logging to localhost, pointing developers to the
      // service worker/PWA documentation.
      navigator.serviceWorker.ready.then(() => {
        console.log(
          'This web app is being served cache-first by a service ' +
            'worker. To learn more, visit https://bit.ly/CRA-PWA'
        );
      });
    } else {
      // Is not localhost. Just register service worker
      registerValidSW(swUrl, config);
    }
  });
 }
}
function registerValidSW(swUrl, config) {
 navigator.serviceWorker
  .register(swUrl)
  .then(registration => {
    registration.onupdatefound = () => {
```

```
const installingWorker = registration.installing;
if (installingWorker == null) {
  return;
}
installingWorker.onstatechange = () => {
  if (installingWorker.state === 'installed') {
    if (navigator.serviceWorker.controller) {
      // At this point, the updated precached content has been fetched,
      // but the previous service worker will still serve the older
      // content until all client tabs are closed.
      console.log(
        'New content is available and will be used when all ' +
          'tabs for this page are closed. See https://bit.ly/CRA-PWA.'
      );

      // Execute callback
      if (config && config.onUpdate) {
        config.onUpdate(registration);
      }
    } else {
      // At this point, everything has been precached.
      // It's the perfect time to display a
      // "Content is cached for offline use." message.
```

```
            console.log('Content is cached for offline use.');
            // Execute callback
            if (config && config.onSuccess) {
              config.onSuccess(registration);
            }
          }
        }
      };
    };
  })
  .catch(error => {
    console.error('Error during service worker registration:', error);
  });
}
function checkValidServiceWorker(swUrl, config) {
 // Check if the service worker can be found. If it can't reload the page.
 fetch(swUrl)
  .then(response => {
    // Ensure service worker exists, and that we really are getting a JS file.
    const contentType = response.headers.get('content-type');
    if (
      response.status === 404 ||
      (contentType != null && contentType.indexOf('javascript') === -1)
```

```
      )
{
        // No service worker found. Probably a different app. Reload the page.
        navigator.serviceWorker.ready.then(registration => {
          registration.unregister().then(() => {
            window.location.reload();
          });
        });
      } else {
        // Service worker found. Proceed as normal.
        registerValidSW(swUrl, config);
      }
    })
    .catch(() => {
      console.log(
        'No internet connection found. App is running in offline mode.'
      );
    });
}

export function unregister() {
  if ('serviceWorker' in navigator) {
    navigator.serviceWorker.ready.then(registration => {
```

```
      registration.unregister();
    });
  }
}
{
  "name": "eth-anticounterfeiting",
  "version": "0.1.0",
  "description": "An Ethereum Product Anticounterfeiting system",
  "author": "jerrywosu@futo.edu.ng",       //Can be configured to any specification
  "dependencies": {
    "babel-polyfill": "6.26.0",
    "babel-preset-env": "1.7.0",
    "babel-preset-es2015": "6.24.1",
    "babel-preset-stage-2": "6.24.1",
    "babel-preset-stage-3": "6.24.1",
    "babel-register": "6.26.0",
    "bootstrap": "4.3.1",
    "chai": "4.2.0",
    "chai-as-promised": "7.1.1",
    "chai-bignumber": "3.0.0",
    "react": "16.8.4",
    "react-bootstrap": "1.0.0-beta.5",
    "react-dom": "16.8.4",
```

```
    "react-scripts": "2.1.3",
    "truffle": "5.0.5",
    "web3": "1.0.0-beta.55"
  },
  "scripts": {
    "start": "react-scripts start",
    "build": "react-scripts build",
    "test": "react-scripts test",
    "eject": "react-scripts eject"
  },
  "eslintConfig": {
    "extends": "react-app"
  },
  "browserslist": [
    ">0.2%",
    "not dead",
    "not ie <= 11",
    "not op_mini all"
  ]
}
{
  "contractName": "ProductAnticounterfeiting",
  "abi": [
```

```
    {
      "constant": true,
      "inputs": [],
      "name": "name",
      "outputs": [
        {
          "name": "",
          "type": "string"
        }
      ],
      "payable": false,
      "stateMutability": "view",
      "type": "function"
    },
    {
      "inputs": [],
      "payable": false,
      "stateMutability": "nonpayable",
      "type": "constructor"
    }
  "sourcePath": "C:/Users/Jeremiah
Wosu/anticounterfeiting/src/contracts/Smartcontracts.sol",
  "ast": {
```

```
  "absolutePath": "/C/Users/Jeremiah Wosu/anticounterfeiting/src/contracts/Smartcontracts.sol",
  "exportedSymbols": {
    "Marketplace": [
      12
    ]
  },
  "id": 13,
  "nodeType": "SourceUnit",
  "nodes": [
    {
      "id": 1,
      "literals": [
        "solidity",
        "^",
        "0.5",
        ".0"
      ],
      "nodeType": "PragmaDirective",
      "src": "0:23:0"
    },
    {
      "baseContracts": [],
      "contractDependencies": [],
```

```
"contractKind": "contract",
"documentation": null,
"fullyImplemented": true,
"id": 12,
"linearizedBaseContracts": [
 12
],
"name": "Marketplace",
"nodeType": "ContractDefinition",
"nodes": [
 {
  "constant": false,
  "id": 3,
  "name": "name",
  "nodeType": "VariableDeclaration",
  "scope": 12,
  "src": "54:18:0",
  "stateVariable": true,
  "storageLocation": "default",
  "typeDescriptions": {
   "typeIdentifier": "t_string_storage",
   "typeString": "string"
  },
```

```
  "typeName": {
    "id": 2,
    "name": "string",
    "nodeType": "ElementaryTypeName",
    "src": "54:6:0",
    "typeDescriptions": {
      "typeIdentifier": "t_string_storage_ptr",
      "typeString": "string"
    }
  },
  "value": null,
  "visibility": "public"
},
{
  "body": {
    "id": 10,
    "nodeType": "Block",
    "src": "102:55:0",
    "statements": [
      {
        "expression": {
          "argumentTypes": null,
          "id": 8,
```

```
"isConstant": false,
"isLValue": false,
"isPure": false,
"lValueRequested": false,
"leftHandSide": {
  "argumentTypes": null,
  "id": 6,
  "name": "name",
  "nodeType": "Identifier",
  "overloadedDeclarations": [],
  "referencedDeclaration": 3,
  "src": "113:4:0",
  "typeDescriptions": {
    "typeIdentifier": "t_string_storage",
    "typeString": "string storage ref"
  }
},
"nodeType": "Assignment",
"operator": "=",
"rightHandSide": {
  "argumentTypes": null,
  "hexValue":
"4461707020556e69766572736974794204d61726b6574706c616365",
  "id": 7,
```

```
        "isConstant": false,
        "isLValue": false,
        "isPure": true,
        "kind": "string",
        "lValueRequested": false,
        "nodeType": "Literal",
        "src": "120:29:0",
        "subdenomination": null,
        "typeDescriptions": {
          "typeIdentifier":
"t_stringliteral_da4b6052d4e2657e92b9e8ac7ae34efe0016cdd765c6837ed00d7314b2dd8
f36",
          "typeString": "literal_string \"Product Anticounterfeiting\""
        },
        "value": "Product Anticounterfeiting"
      },
      "src": "113:36:0",
      "typeDescriptions": {
        "typeIdentifier": "t_string_storage",
        "typeString": "string storage ref"
      }
    },
    "id": 9,
    "nodeType": "ExpressionStatement",
```

```
      "src": "113:36:0"
    }
  ]
},
"documentation": null,
"id": 11,
"implemented": true,
"kind": "constructor",
"modifiers": [],
"name": "",
"nodeType": "FunctionDefinition",
"parameters": {
  "id": 4,
  "nodeType": "ParameterList",
  "parameters": [],
  "src": "93:2:0"
},
"returnParameters": {
  "id": 5,
  "nodeType": "ParameterList",
  "parameters": [],
  "src": "102:0:0"
},
```

```
          "scope": 12,
          "src": "81:76:0",
          "stateMutability": "nonpayable",
          "superFunction": null,
          "visibility": "public"
        }
      ],
      "scope": 13,
      "src": "27:133:0"
    }
  ],
  "src": "0:160:0"
},
"legacyAST": {
  "absolutePath": "/C/Users/Jeremiah
Wosu/anticounterfeiting/src/contracts/Smartcontracts.sol",
  "exportedSymbols": {
    "Marketplace": [
      12
    ]
  },
  "id": 13,
  "nodeType": "SourceUnit",
  "nodes": [
```

```
{
  "id": 1,
  "literals": [
    "solidity",
    "^",
    "0.5",
    ".0"
  ],
  "nodeType": "PragmaDirective",
  "src": "0:23:0"
},
{
  "baseContracts": [],
  "contractDependencies": [],
  "contractKind": "contract",
  "documentation": null,
  "fullyImplemented": true,
  "id": 12,
  "linearizedBaseContracts": [
    12
  ],
  "name": "Marketplace",
  "nodeType": "ContractDefinition",
```

```
"nodes": [
  {
    "constant": false,
    "id": 3,
    "name": "name",
    "nodeType": "VariableDeclaration",
    "scope": 12,
    "src": "54:18:0",
    "stateVariable": true,
    "storageLocation": "default",
    "typeDescriptions": {
      "typeIdentifier": "t_string_storage",
      "typeString": "string"
    },
    "typeName": {
      "id": 2,
      "name": "string",
      "nodeType": "ElementaryTypeName",
      "src": "54:6:0",
      "typeDescriptions": {
        "typeIdentifier": "t_string_storage_ptr",
        "typeString": "string"
      }
```

```
  },
  "value": null,
  "visibility": "public"
},
{
  "body": {
    "id": 10,
    "nodeType": "Block",
    "src": "102:55:0",
    "statements": [
      {
        "expression": {
          "argumentTypes": null,
          "id": 8,
          "isConstant": false,
          "isLValue": false,
          "isPure": false,
          "lValueRequested": false,
          "leftHandSide": {
            "argumentTypes": null,
            "id": 6,
            "name": "name",
            "nodeType": "Identifier",
```

```
      "overloadedDeclarations": [],
      "referencedDeclaration": 3,
      "src": "113:4:0",
      "typeDescriptions": {
        "typeIdentifier": "t_string_storage",
        "typeString": "string storage ref"
      }
    },
    "nodeType": "Assignment",
    "operator": "=",
    "rightHandSide": {
      "argumentTypes": null,
      "hexValue":
"4461707020556e6976657273697479204d61726b6574706c616365",
      "id": 7,
      "isConstant": false,
      "isLValue": false,
      "isPure": true,
      "kind": "string",
      "lValueRequested": false,
      "nodeType": "Literal",
      "src": "120:29:0",
      "subdenomination": null,
      "typeDescriptions": {
```

```
                "typeIdentifier":
"t_stringliteral_da4b6052d4e2657e92b9e8ac7ae34efe0016cdd765c6837ed00d7314b2dd8
f36",
                "typeString": "literal_string \"Product Anticounterfeiting\""
              },
              "value": "Product Anticounterfeiting"
            },
            "src": "113:36:0",
            "typeDescriptions": {
              "typeIdentifier": "t_string_storage",
              "typeString": "string storage ref"
            }
          },
          "id": 9,
          "nodeType": "ExpressionStatement",
          "src": "113:36:0"
        }
      ]
    },
    "documentation": null,
    "id": 11,
    "implemented": true,
    "kind": "constructor",
    "nodeType": "FunctionDefinition",
```

```
      "parameters": {
        "id": 4,
        "nodeType": "ParameterList",
        "parameters": [],
        "src": "93:2:0"
      },
      "returnParameters": {
        "id": 5,
        "nodeType": "ParameterList",
        "parameters": [],
        "src": "102:0:0"
      },
      "scope": 12,
      "src": "81:76:0",
      "stateMutability": "nonpayable",
      "superFunction": null,
      "visibility": "public"
    }
  ],
  "scope": 13,
  "src": "27:133:0"
 }
],
```

```
    "src": "0:160:0"
  },
  "compiler": {
    "name": "solc",
    "version": "0.5.0+commit.1d4f565a.Emscripten.clang"
  },
  "networks": {},
  "schemaVersion": "3.0.2",
  "updatedAt": "2023-05-20T22:25:20.484Z",
  "devdoc": {
    "methods": {}
  },
  "userdoc": {
    "methods": {}
  }
}
```


Appendix 2: Source Code for the Consensus Algorithm

```
package au.id.tindall.distalg.raft.newblockcreator;
import org.junit.jupiter.api.AfterEach;
import org.junit.jupiter.api.BeforeEach;
import org.junit.jupiter.api.Nested;
import org.junit.jupiter.api.Test;
import org.junit.jupiter.api.extension.ExtendWith;
import org.mockito.Mock;
import org.mockito.junit.jupiter.MockitoExtension;
```

```
@ExtendWith(MockitoExtension.class)
class NewBlockCreator {

   private static final long TIMEOUT_MILLIS = 100L;
   private static final Instant CURRENT_TIME = Instant.now();
   @Mock
   private NewBlockCreator;

   private final AtomicReference<Instant> currentTime = new
AtomicReference<>(CURRENT_TIME);
   private NewBlockCreator;

   @BeforeEach
   void setUp() {
      NewBlockCreator = new NewBlockCreator(NewBlockCreator, currentTime::get);
      lenient().when(NewBlockCreator.next()).thenReturn(TIMEOUT_MILLIS);
   }

   @Nested
   class StartElectionTimeouts {

      @BeforeEach
      void setUp() {
         NewBlockCreator.startTimeouts();
      }

      @AfterEach
      void tearDown() {
         NewBlockCreator.stopTimeouts();
      }

      @Test
      void NewBlockCreator () {
         currentTime.set(currentTime.get().plusMillis(TIMEOUT_MILLIS - 1));
         assertThat(NewBlockCreator.shouldTimeout()).isFalse();
         currentTime.set(currentTime.get().plusMillis(2));
         assertThat(NewBlockCreator.shouldTimeout()).isTrue();
      }

      @Test
      void willcreateifvotes>50%() {
```

```
            assertThatThrownBy(() ->
NewBlockCreator.startTimeouts()).isInstanceOf(IllegalStateException.class);

        }
    }
}
  <title>Jerry's Blockchain Network - Home Page</title>

  <meta content="" name="description">

  <link
href="https://fonts.googleapis.com/css?family=Open+Sans:300,300i,400,400i,600,600i,7
00,700i|Nunito:300,300i,400,400i,600,600i,700,700i|Poppins:300,300i,400,400i,500,500i
,600,600i,700,700i" rel="stylesheet">

  <!-- Vendor CSS Files -->

  <link href="assets/vendor/aos/aos.css" rel="stylesheet">

  <link href="assets/vendor/bootstrap/css/bootstrap.min.css" rel="stylesheet">

  <link href="assets/vendor/bootstrap-icons/bootstrap-icons.css" rel="stylesheet">

  <link href="assets/vendor/glightbox/css/glightbox.min.css" rel="stylesheet">

  <link href="assets/vendor/remixicon/remixicon.css" rel="stylesheet">

  <link href="assets/vendor/swiper/swiper-bundle.min.css" rel="stylesheet">


  <!-- Template Main CSS File -->

  <link href="assets/css/style.css" rel="stylesheet">


</head>


<body>
```

```
<!-- ======= Header ======= -->

<header id="header" class="header fixed-top">

  <div class="container-fluid container-xl d-flex align-items-center justify-content-between">


    <a href="index.php" class="logo d-flex align-items-center">

      <img src="assets/img/logo.png" alt="">

      <span>Jerry's Blockchain Network</span>

    </a>


    <nav id="navbar" class="navbar">

      <ul>

        <li><a class="nav-link scrollto" href="index.php">Home</a></li>

        <li class="dropdown"><a href="#"><span>Register</span> <i class="bi bi-chevron-down"></i></a>

          <ul>

            <li><a href="mfd-reg.php">Register as Manufacturer</a></li>

            <li><a href="dist-reg.php">Register as Distributor</a></li>

          </ul>

        </li>

        <li class="dropdown"><a href="#"><span>Login</span> <i class="bi bi-chevron-down"></i></a>

          <ul>

            <li><a href="mfd-login.php">Login as Manufacturer</a></li>
```

```
            <li><a href="dist-login.php">Login as Distributor</a></li>
          </ul>
        </li>
        <li><a class="getstarted scrollto" href="auth-product.php">Authenticate Product</a></li>
      </ul>
      <i class="bi bi-list mobile-nav-toggle"></i>
    </nav><!-- .navbar -->

  </div>
</header><!-- End Header -->

<!-- ======= Hero Section ======= -->
<section id="hero" class="hero d-flex align-items-center">

  <div class="container">
    <div class="row">
      <div class="col-lg-6 d-flex flex-column justify-content-center">
        <h1 data-aos="fade-up">Product Anti-Counterfeiting Blockchain Technology</h1>
        <h2 data-aos="fade-up" data-aos-delay="400">
          A Research by <b>Jeremiah T. Wosu</b> in partial fulfillment for the award of
          PhD in Electronic and Computer Engineering, Federal University of Technolgy, Owerri
```

```
        </h2>
        <div data-aos="fade-up" data-aos-delay="600">
          <div class="text-center text-lg-start">
            <a href="auth-product.php" class="btn-get-started scrollto d-inline-flex align-items-center justify-content-center align-self-center">
              <span>Authenticate a Product</span>
              <i class="bi bi-arrow-right"></i>
            </a>
          </div>
        </div>
      </div>
      <div class="col-lg-6 hero-img" data-aos="zoom-out" data-aos-delay="200">
        <img src="img/blockchain-img.jpg" class="img-fluid" alt="">
      </div>
    </div>
  </div>

</section><!-- End Hero -->

<!-- ======= Counts Section ======= -->
<section id="counts" class="counts">
    <div class="container" data-aos="fade-up">

      <div class="row gy-4">
```

```
<div class="col-lg-3 col-md-6">
  <div class="count-box">
    <i class="bi bi-emoji-smile"></i>
    <div>
      <span data-purecounter-start="0" data-purecounter-end="232" data-purecounter-duration="1" class="purecounter"></span>
      <p>Manufacturers</p>
    </div>
  </div>
</div>
<div class="col-lg-3 col-md-6">
  <div class="count-box">
    <i class="bi bi-people" style="color: #bb0852;"></i>
    <div>
      <span data-purecounter-start="0" data-purecounter-end="15" data-purecounter-duration="1" class="purecounter"></span>
      <p>Distributors</p>
    </div>
  </div>
</div>

<div class="col-lg-3 col-md-6">
```

```
          <div class="count-box">
            <i class="bi bi-journal-richtext" style="color: #ee6c20;"></i>
            <div>
              <span data-purecounter-start="0" data-purecounter-end="521" data-purecounter-duration="1" class="purecounter"></span>
              <p>Products</p>
            </div>
          </div>
        </div>
      </div>

    </div>
  </section><!-- End Counts Section -->

  <!-- ======= Footer ======= -->
  <footer id="footer" class="footer">
    <div class="container">
      <div class="copyright">
        &copy; Copyright <strong><span>2023. Jeremiah T. Wosu (<i>PhD in View</i>)</span></strong>. All Rights Reserved
      </div>
    </div>
  </footer><!-- End Footer -->
```

```
  <a href="#" class="back-to-top d-flex align-items-center justify-content-center"><i class="bi bi-arrow-up-short"></i></a>

  <!-- Vendor JS Files -->
  <script src="assets/vendor/purecounter/purecounter_vanilla.js"></script>
  <script src="assets/vendor/aos/aos.js"></script>
  <script src="assets/vendor/bootstrap/js/bootstrap.bundle.min.js"></script>
  <script src="assets/vendor/glightbox/js/glightbox.min.js"></script>
  <script src="assets/vendor/isotope-layout/isotope.pkgd.min.js"></script>
  <script src="assets/vendor/swiper/swiper-bundle.min.js"></script>
  <script src="assets/vendor/php-email-form/validate.js"></script>

  <!-- Template Main JS File -->
  <script src="assets/js/main.js"></script>

</body>

</html>
```

Appendix 3: Source Code for Smart Contracts

```
session_start();

include 'connection.php';

if($_SESSION['adminid'] === "" || $_SESSION['adminpwd'] === ""){

   session_destroy();

   header('Location: admin-login.php');

}

$adminid = $_SESSION['adminid'];

$fn = "";

$sql = "SELECT DISTINCT * FROM blockadmin WHERE ID = '$adminid'";

$result = $conn->query($sql);

if($result->num_rows > 0){

   while($row = $result->fetch_assoc()){

      $fn = $row['fn'];

   }

}else{

   session_destroy();

   header('Location: admin-login.php');

}



$distph = $_SESSION['admindistapproveph'];
```

```
?>

<!DOCTYPE html>

<html lang="en">

<head>
  <meta charset="utf-8">
  <meta content="width=device-width, initial-scale=1.0" name="viewport">

  <title>Jerry's Blockchain Network - Manufacturer's Dashboard</title>
  <meta content="" name="description">
  <meta content="" name="keywords">

  <!-- Favicons -->
  <link href="img/logo-icon.PNG" rel="icon">
  <link href="assets/img/apple-touch-icon.png" rel="apple-touch-icon">

  <!-- Google Fonts -->
  <link
href="https://fonts.googleapis.com/css?family=Open+Sans:300,300i,400,400i,600,600i,700,700i|Nunito
:300,300i,400,400i,600,600i,700,700i|Poppins:300,300i,400,400i,500,500i,600,600i,700,700i"
rel="stylesheet">
```

```
  <!-- Vendor CSS Files -->
  <link href="assets/vendor/aos/aos.css" rel="stylesheet">
  <link href="assets/vendor/bootstrap/css/bootstrap.min.css" rel="stylesheet">
  <link href="assets/vendor/bootstrap-icons/bootstrap-icons.css" rel="stylesheet">
  <link href="assets/vendor/glightbox/css/glightbox.min.css" rel="stylesheet">
  <link href="assets/vendor/remixicon/remixicon.css" rel="stylesheet">
  <link href="assets/vendor/swiper/swiper-bundle.min.css" rel="stylesheet">

  <!-- Template Main CSS File -->
  <link href="assets/css/style.css" rel="stylesheet">

</head>

<body>

  <!-- ======= Header ======= -->
  <header id="header" class="header fixed-top">
    <div class="container-fluid container-xl d-flex align-items-center justify-content-between">

      <a href="admin-dashboard.php" class="logo d-flex align-items-center">
```

```
      <img src="assets/img/logo.png" alt="">
      <span>Jerry's Blockchain Network</span>
    </a>

    <nav id="navbar" class="navbar">
      <ul>
        <li class="dropdown"><a href="#"><span>Manufacturers</span> <i class="bi bi-chevron-down"></i></a>
          <ul>
            <li><a href="admin-auth-manufacturers.php">Approved Manufacturers</a></li>
            <li><a href="admin-unauth-manufacturers.php">Not Approved Manufacturers</a></li>
            <li><a href="admin-unauth-manufacturers.php">Disapproved Manufacturers</a></li>
          </ul>
        </li>
        <li class="dropdown"><a href="#"><span>Distributors</span> <i class="bi bi-chevron-down"></i></a>
          <ul>
            <li><a href="admin-auth-distributors.php">Approved Distributors</a></li>
            <li><a href="admin-unauth-distributors.php">Not Approved Distributors</a></li>
            <li><a href="admin-unauth-distributors.php">Disapproved Distributors</a></li>
          </ul>
        </li>
```

```
      <li><a class="getstarted scrollto" href="dist-auth-product.php">Authenticate a Product</a></li>
      <li>
        <form action="logout-dist.php" method="post">
          <button class="logoutbtn">Logout</button>
        </form>
      </li>
    </ul>
    <i class="bi bi-list mobile-nav-toggle"></i>
  </nav><!-- .navbar -->

 </div>
</header><!-- End Header -->

<br><br><br>
<main id="main">
 <!-- ======= About Section ======= -->
 <section id="about" class="about">

  <div class="container" data-aos="fade-up" data-aos-delay="200">
   <div class="row gx-0">
     <h5>Welcome, <?php echo $fn ?> [Administrator]</h5>
```

```
<hr>

<?php

   $sql = "SELECT DISTINCT * FROM dist WHERE ph = '$distph' AND approv_status = 'APPROVED'";

   $result = $conn->query($sql);


   if($result->num_rows > 0){

      while($row = $result->fetch_assoc()){

         ?>

         <p><b>Details of the Manufacturer</b></p>

         <p>Distributor's ID: <b><?php echo $row['distid'] ?></b></p>

         <p>Manufacturer's ID:<b><?php echo $row['mfdid'] ?></b></p>

         <p>Phone number: <b><?php echo $distph ?></b></p>

         <p>Name: <b><?php echo $row['name'] ?></b></p>

         <p>Email: <b><?php echo $row['em'] ?></b></p>

         <p>Address: <b><?php echo $row['addr'] ?></b></p>

         <p>CAC certificate number: <b><?php echo $row['cacnum'] ?></b></p>

         <p>Longitude: <b><?php echo $row['lng'] ?></b></p>

         <p>Latitude: <b><?php echo $row['lat'] ?></b></p>


         <?php
```

```
            }
          }else{
            ?>
            <div class="alert alert-danger">
               <p>No result found!</p>
            </div>
            <?php
          }
        ?>
    </div>
  </div>

 </section><!-- End About Section -->
</main><!-- End #main -->

<style>
 .btnapprove{
    background:green; color:white;
 }
 body{
  counter-reset: Serial;
```

```
}
.caccert{
   width:500px;
   height:500px; object-fit:contain;
}
.detbtn{
   background:teal; color:white; border:1px solid teal;
}
.table-div{
  overflow-x: auto;
  white-space: nowrap;
}
.logoutbtn{
   padding:0; border:1px solid red; background:red; color:white; margin:5px;
}
.input-ui{
   padding:5px; border:1px solid lightgray;
}
table {
     border-collapse: separate;
     font-family: arial, sans-serif;
```

```
  border-collapse: collapse;
  width: 100%;
 }
 tr td:first-child:before{
  counter-increment:Serial;
  content: counter(Serial);
 }

 td, th {
  border: 1px solid #dddddd;
  text-align: left;
  padding: 8px;
 }

 tr:nth-child(even) {
  background-color: #dddddd;
 }

</style>

<!-- ======= Footer ======= -->
```

```
<footer id="footer" class="footer">

  <div class="container">

    <div class="copyright">

      &copy; Copyright <strong><span>2023. Jeremiah T. Wosu (<i>PhD in View</i>)</span></strong>. All Rights Reserved

    </div>

  </div>

</footer><!-- End Footer -->


<a href="#" class="back-to-top d-flex align-items-center justify-content-center"><i class="bi bi-arrow-up-short"></i></a>


<!-- Vendor JS Files -->

<script src="assets/vendor/purecounter/purecounter_vanilla.js"></script>

<script src="assets/vendor/aos/aos.js"></script>

<script src="assets/vendor/bootstrap/js/bootstrap.bundle.min.js"></script>

<script src="assets/vendor/glightbox/js/glightbox.min.js"></script>

<script src="assets/vendor/isotope-layout/isotope.pkgd.min.js"></script>

<script src="assets/vendor/swiper/swiper-bundle.min.js"></script>

<script src="assets/vendor/php-email-form/validate.js"></script>

<!-- Template Main JS File -->

<script src="assets/js/main.js"></script>
```

```
</body>

</html>

<?php

   unset($_SESSION['updateerror']);

   unset($_SESSION['updatesuccess']);

?>



<?php

   session_start();

   include 'connection.php';



   if($_SESSION['adminid'] === "" || $_SESSION['adminpwd'] === ""){

      session_destroy();

      header('Location: admin-login.php');

   }



   $adminid = $_SESSION['adminid'];

   $fn = "";

   $sql = "SELECT DISTINCT * FROM blockadmin WHERE ID = '$adminid'";

   $result = $conn->query($sql);
```

```
    if($result->num_rows > 0){
        while($row = $result->fetch_assoc()){
            $fn = $row['fn'];
        }
    }else{
        session_destroy();
        header('Location: admin-login.php');
    }
    $mfdph = $_SESSION['approvedmfdph'];
?>
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="utf-8">
  <meta content="width=device-width, initial-scale=1.0" name="viewport">

  <title>Jerry's Blockchain Network - Manufacturer's Dashboard</title>
  <meta content="" name="description">
  <meta content="" name="keywords">

  <!-- Favicons -->
```

```
  <link href="img/logo-icon.PNG" rel="icon">
  <link href="assets/img/apple-touch-icon.png" rel="apple-touch-icon">

  <!-- Google Fonts -->
  <link
href="https://fonts.googleapis.com/css?family=Open+Sans:300,300i,400,400i,600,600i,700,700i|Nunito
:300,300i,400,400i,600,600i,700,700i|Poppins:300,300i,400,400i,500,500i,600,600i,700,700i"
rel="stylesheet">

  <!-- Vendor CSS Files -->
  <link href="assets/vendor/aos/aos.css" rel="stylesheet">
  <link href="assets/vendor/bootstrap/css/bootstrap.min.css" rel="stylesheet">
  <link href="assets/vendor/bootstrap-icons/bootstrap-icons.css" rel="stylesheet">
  <link href="assets/vendor/glightbox/css/glightbox.min.css" rel="stylesheet">
  <link href="assets/vendor/remixicon/remixicon.css" rel="stylesheet">
  <link href="assets/vendor/swiper/swiper-bundle.min.css" rel="stylesheet">

  <!-- Template Main CSS File -->
  <link href="assets/css/style.css" rel="stylesheet">

</head>
<body>
```

```
<!-- ======= Header ======= -->
<header id="header" class="header fixed-top">
  <div class="container-fluid container-xl d-flex align-items-center justify-content-between">
    <a href="admin-dashboard.php" class="logo d-flex align-items-center">
      <img src="assets/img/logo.png" alt="">
      <span>Jerry's Blockchain Network</span>
    </a>

    <nav id="navbar" class="navbar">
      <ul>
        <li class="dropdown"><a href="#"><span>Manufacturers</span> <i class="bi bi-chevron-down"></i></a>
          <ul>
            <li><a href="admin-auth-manufacturers.php">Approved Manufacturers</a></li>
            <li><a href="admin-unauth-manufacturers.php">Not Approved Manufacturers</a></li>
            <li><a href="admin-unauth-manufacturers.php">Disapproved Manufacturers</a></li>
          </ul>
        </li>
        <li class="dropdown"><a href="#"><span>Distributors</span> <i class="bi bi-chevron-down"></i></a>
          <ul>
            <li><a href="admin-auth-distributors.php">Approved Distributors</a></li>
```

```
          <li><a href="admin-unauth-distributors.php">Not Approved Distributors</a></li>
          <li><a href="admin-unauth-distributors.php">Disapproved Distributors</a></li>
        </ul>
      </li>
      <li><a class="getstarted scrollto" href="dist-auth-product.php">Authenticate a Product</a></li>
      <li>
        <form action="logout-dist.php" method="post">
          <button class="logoutbtn">Logout</button>
        </form>
      </li>
    </ul>
    <i class="bi bi-list mobile-nav-toggle"></i>
  </nav><!-- .navbar -->

 </div>
</header><!-- End Header -->

<br><br><br>
<main id="main">
 <!-- ======= About Section ======= -->
 <section id="about" class="about">
```

```
<div class="container" data-aos="fade-up" data-aos-delay="200">
 <div class="row gx-0">
    <?php
       if(isset($_SESSION['updateerror'])){
          ?>
          <div class="alert alert-danger">
             <p><?php echo $_SESSION['updateerror'] ?></p>
          </div>
          <?php
       }
       if(isset($_SESSION['updatesuccess'])){
          ?>
          <div class="alert alert-success">
             <p><?php echo $_SESSION['updatesuccess'] ?></p>
          </div>
          <?php
       }
    ?>
    <h5>Welcome, <?php echo $fn ?> [Administrator]</h5>
```

```
<hr>
<div class="col-sm-4">
  <form action="adminapprovemfd-approved.php" method="post">
    <input type="hidden" name="mfdph" id="" value="<?php echo $mfdph ?>">
    <button class="btn btn-danger">Disapprove Manufacturer</button>
  </form>
</div>

<?php
  $sql = "SELECT DISTINCT * FROM mfd WHERE ph = '$mfdph' AND approv_status = 'APPROVED'";
  $result = $conn->query($sql);

  if($result->num_rows > 0){
    while($row = $result->fetch_assoc()){
      ?>
      <p><b>Details of the Manufacturer</b></p>
      <p>Manufacturer's ID: <b><?php echo $row['mfdid'] ?></b></p>
      <p>Phone number: <b><?php echo $mfdph ?></b></p>
      <p>Name: <b><?php echo $row['mfdname'] ?></b></p>
      <p>Email: <b><?php echo $row['mfdem'] ?></b></p>
```

```
                <p>Address: <b><?php echo $row['mfdaddr'] ?></b></p>
                <p>CAC certificate number: <b><?php echo $row['cacnum'] ?></b></p>
                <img src="<?php echo $row['cac'] ?>" alt="CAC Certificate" class="caccert">
                <p>Longitude: <b><?php echo $row['lng'] ?></b></p>
                <p>Latitude: <b><?php echo $row['lat'] ?></b></p>

                <?php
              }
            }else{
              ?>
              <div class="alert alert-danger">
                <p>No result found!</p>
              </div>
              <?php
            }
          ?>
      </div>
    </div>

  </section><!-- End About Section -->
</main><!-- End #main -->
```

```
<style>
 .btnapprove{
    background:green; color:white;
 }
 body{
  counter-reset: Serial;
 }
 .caccert{
    width:500px;
    height:500px; object-fit:contain;
 }
 .detbtn{
    background:teal; color:white; border:1px solid teal;
 }
 .table-div{
  overflow-x: auto;
  white-space: nowrap;
 }
 .logoutbtn{
    padding:0; border:1px solid red; background:red; color:white; margin:5px;
```

```
}
.input-ui{
   padding:5px; border:1px solid lightgray;
}
table {
     border-collapse: separate;
     font-family: arial, sans-serif;
     border-collapse: collapse;
     width: 100%;
   }

   tr td:first-child:before{
     counter-increment:Serial;
     content: counter(Serial);
   }

   td, th {
     border: 1px solid #dddddd;
     text-align: left;
     padding: 8px;
   }
```

```
tr:nth-child(even) {
  background-color: #dddddd;
}
```

Appendix 4: Source Code for Developing dApp

```
<!-- Vendor CSS Files -->

  <link href="assets/vendor/aos/aos.css" rel="stylesheet">

  <link href="assets/vendor/bootstrap/css/bootstrap.min.css" rel="stylesheet">

  <link href="assets/vendor/bootstrap-icons/bootstrap-icons.css" rel="stylesheet">

  <link href="assets/vendor/glightbox/css/glightbox.min.css" rel="stylesheet">

  <link href="assets/vendor/remixicon/remixicon.css" rel="stylesheet">

  <link href="assets/vendor/swiper/swiper-bundle.min.css" rel="stylesheet">



  <!-- Template Main CSS File -->

  <link href="assets/css/style.css" rel="stylesheet">



</head>



<body>



  <!-- ======= Header ======= -->

  <header id="header" class="header fixed-top">

    <div class="container-fluid container-xl d-flex align-items-center justify-content-between">



      <a href="admin-dashboard.php" class="logo d-flex align-items-center">
```

```
      <img src="assets/img/logo.png" alt="">
      <span>Jerry's Blockchain Network</span>
    </a>

    <nav id="navbar" class="navbar">
      <ul>
        <li class="dropdown"><a href="#"><span>Manufacturers</span> <i class="bi bi-chevron-down"></i></a>
          <ul>
            <li><a href="admin-auth-manufacturers.php">Approved Manufacturers</a></li>
            <li><a href="admin-unauth-manufacturers.php">Not Approved Manufacturers</a></li>
            <li><a href="admin-disapproved-manufacturers.php">Disapproved Manufacturers</a></li>
          </ul>
        </li>
        <li class="dropdown"><a href="#"><span>Distributors</span> <i class="bi bi-chevron-down"></i></a>
          <ul>
            <li><a href="admin-auth-distributors.php">Approved Distributors</a></li>
            <li><a href="admin-unauth-distributors.php">Not Approved Distributors</a></li>
            <li><a href="admin-disapproved-distributors.php">Disapproved Distributors</a></li>
          </ul>
        </li>
```

```
      <li><a class="getstarted scrollto" href="dist-auth-product.php">Authenticate a Product</a></li>
      <li>
        <form action="logout-admin.php" method="post">
          <button class="logoutbtn">Logout</button>
        </form>
      </li>
    </ul>
    <i class="bi bi-list mobile-nav-toggle"></i>
  </nav><!-- .navbar -->

 </div>
</header><!-- End Header -->

<br><br><br>
<main id="main">
 <!-- ======= About Section ======= -->
 <section id="about" class="about">

  <div class="container" data-aos="fade-up" data-aos-delay="200">
   <div class="row gx-0">
     <h5>Welcome, <?php echo $fn ?> [Administrator]</h5>
```

```
<hr>

<p>List of all the approved distributors</p>

<form action="searchnotapprovedman.php" method="post">
   <input type="text" name="mfdname" id="" class="input-ui" required>
   <button class="btn btn-primary"><i class="bi bi-search"></i></button>
</form>
<br>
<div class="table-div">
   <?php
      $sql = "SELECT DISTINCT * FROM dist WHERE approv_status = 'APPROVED'";

      $query = mysqli_query($conn, $sql);
      $nums = mysqli_num_rows($query);
   ?>
   <p>Number of approved distributor:  <b><?php echo $nums; ?></b></p>
   <br>
   <table>
      <thead style="text-align:center;">
```

```
    <th>S/N</th>
    <th>Phone</th>
    <th>Name</th>
    <th>Email</th>
    <th>Address</th>
    <th>Approval Status</th>
    <th>Action</th>
</thead>
<tbody>
    <?php
        $sql = "SELECT DISTINCT * FROM dist WHERE approv_status = 'APPROVED'";
        $result = $conn->query($sql);

        if($result->num_rows > 0){
            while($row = $result->fetch_assoc()){
                ?>
                <tr>
                    <td><p></p></td>
                    <td><p><?php echo $row["ph"] ?></p></td>
                    <td><p><?php echo $row['name'] ?></p></td>
                    <td><p><?php echo $row["em"] ?></p></td>
```

```
                <td><p><?php echo $row["addr"] ?></p></td>
                <td><p><?php echo $row['approv_status'] ?></p></td>
                <td style="text-align:center">
                  <form action="admindistdetailapproved.php" method="post">
                    <input type="hidden" name="dph" id="" value="<?php echo $row["ph"] ?>">
                    <button class="detbtn">Details  <i class="bi bi-list"></i></button>
                  </form>
                </td>
              </tr>
            <?php
            }
          }else{
            ?>
            <div class="alert alert-danger">
              <p>No approved distributor found!</p>
            </div>
            <?php
          }
        ?>
    </tbody>
</table>
```

```
        </div>
      </div>
    </div>

  </section><!-- End About Section -->
</main><!-- End #main -->

<style>
  body{
    counter-reset: Serial;
  }
  .detbtn{
      background:teal; color:white; border:1px solid teal;
  }
  .table-div{
    overflow-x: auto;
    white-space: nowrap;
  }
  .logoutbtn{
      padding:0; border:1px solid red; background:red; color:white; margin:5px;
  }
```

```
.input-ui{
   padding:5px; border:1px solid lightgray;
}
table {
     border-collapse: separate;
     font-family: arial, sans-serif;
     border-collapse: collapse;
     width: 100%;
   }

   tr td:first-child:before{
     counter-increment:Serial;
     content: counter(Serial);
   }

   td, th {
     border: 1px solid #dddddd;
     text-align: left;
     padding: 8px;
   }
```

```
      tr:nth-child(even) {
        background-color: #dddddd;
      }

  </style>

  <!-- ======= Footer ======= -->
  <footer id="footer" class="footer">
    <div class="container">
      <div class="copyright">
        &copy; Copyright <strong><span>2023. Jeremiah T. Wosu (<i>PhD in View</i>)</span></strong>. All Rights Reserved
      </div>
    </div>
  </footer><!-- End Footer -->

  <a href="#" class="back-to-top d-flex align-items-center justify-content-center"><i class="bi bi-arrow-up-short"></i></a>

  <!-- Vendor JS Files -->
  <script src="assets/vendor/purecounter/purecounter_vanilla.js"></script>
  <script src="assets/vendor/aos/aos.js"></script>
```

```
  <script src="assets/vendor/bootstrap/js/bootstrap.bundle.min.js"></script>

  <script src="assets/vendor/glightbox/js/glightbox.min.js"></script>

  <script src="assets/vendor/isotope-layout/isotope.pkgd.min.js"></script>

  <script src="assets/vendor/swiper/swiper-bundle.min.js"></script>

  <script src="assets/vendor/php-email-form/validate.js"></script>


  <!-- Template Main JS File -->

  <script src="assets/js/main.js"></script>

</body>

</html>


<?php

   session_start();

   include 'connection.php';


   if($_SESSION['adminid'] === "" || $_SESSION['adminpwd'] === ""){

      session_destroy();

      header('Location: admin-login.php');

   }


   $adminid = $_SESSION['adminid'];
```

```
$fn = "";

$sql = "SELECT DISTINCT * FROM blockadmin WHERE ID = '$adminid'";

$result = $conn->query($sql);



if($result->num_rows > 0){

   while($row = $result->fetch_assoc()){

      $fn = $row['fn'];

   }

}else{

   session_destroy();

   header('Location: admin-login.php');

}
```

Appendix 5: Source Code for Generating and Scanning QR code

```
package com.thirdwinder.qrgenerator
import android.annotation.SuppressLint
import android.content.Intent
import android.content.pm.PackageManager
import android.location.Location
import android.location.LocationListener
import android.location.LocationManager
import android.location.LocationRequest
import android.media.Image
import android.os.Build
import androidx.appcompat.app.AppCompatActivity
import android.os.Bundle
import android.view.Surface
import android.widget.Toast
import androidx.core.app.ActivityCompat
import androidx.core.content.ContextCompat
import com.google.android.gms.location.FusedLocationProviderClient
import com.google.android.gms.location.LocationCallback
import com.google.android.gms.location.LocationServices
import com.google.mlkit.vision.barcode.BarcodeScannerOptions
import com.google.mlkit.vision.barcode.BarcodeScanning
import com.google.mlkit.vision.barcode.common.Barcode
import com.google.mlkit.vision.common.InputImage
import com.otaliastudios.cameraview.controls.WhiteBalance
import com.otaliastudios.cameraview.frame.Frame
import com.otaliastudios.cameraview.frame.FrameProcessor
import com.thirdwinder.qrgenerator.databinding.ActivityBarcodeScannerBinding

class BarcodeScanner : AppCompatActivity(), FrameProcessor {

    private val binding: ActivityBarcodeScannerBinding by lazy {
        ActivityBarcodeScannerBinding.inflate(layoutInflater)
    }

    private lateinit var fusedLocationClient: FusedLocationProviderClient
```

```
private lateinit var locationRequest: LocationRequest
private lateinit var locationCallback: LocationCallback
private lateinit var locationManager: LocationManager
private lateinit var locationListener: LocationListener
@SuppressLint("MissingPermission")
override fun onCreate(savedInstanceState: Bundle?) {
    super.onCreate(savedInstanceState)
    setContentView(binding.root)



    if (allPermissionsGranted()) {
        binding.cameraView.setLifecycleOwner(this)
        configuration()

        configureLocation()
    } else {
        ActivityCompat.requestPermissions(
            this,
            REQUIRED_PERMISSIONS,
            REQUEST_CODE_PERMISSIONS
        )
        binding.cameraView.setLifecycleOwner(this)
        configuration()
        configuration()
    }


}

@SuppressLint("MissingPermission")
private fun configureLocation() {
    locationManager = getSystemService(LOCATION_SERVICE) as LocationManager
    locationListener = object : LocationListener {
        override fun onLocationChanged(location: Location) {
            val latitude = location.latitude
```

```
            val longitude = location.longitude
            println("*****************")
            println("long: $longitude, lat: $latitude")
        }

        override fun onStatusChanged(provider: String, status: Int, extras: Bundle) {}
        override fun onProviderEnabled(provider: String) {}
        override fun onProviderDisabled(provider: String) {}
    }


    locationManager.requestLocationUpdates(
        LocationManager.GPS_PROVIDER,
        50L,
        100F,
        locationListener
    )
}


private val options = BarcodeScannerOptions.Builder()
    .setBarcodeFormats(
        Barcode.FORMAT_QR_CODE,
        Barcode.FORMAT_AZTEC
    )
    .build()

private val scanner = BarcodeScanning.getClient(options)


private fun degreesToFirebaseRotation(orientation: Int): Int {
    val rotation: Int = when (orientation) {
        in 45..134 -> {
            Surface.ROTATION_270
        }
        in 135..224 -> {
```

```
            Surface.ROTATION_180
        }
        in 225..314 -> {
            Surface.ROTATION_90
        }
        else -> {
            Surface.ROTATION_0
        }
    }
    return rotation
}


private fun handleImageAnalysis(image: InputImage) {

    scanner.process(image)
        .addOnSuccessListener { barcodes ->
            for (barcode in barcodes) {
                val rawValue = barcode.rawValue

            }
        }
        .addOnFailureListener {

        }


}

private fun frameProcessor(frame: Frame) {
    val size: com.otaliastudios.cameraview.size.Size = frame.size
    val userRotation: Int = frame.rotationToUser
    if (frame.dataClass === ByteArray::class.java) {
        val data: ByteArray = frame.getData()
        val inputImage = InputImage.fromByteArray(
            data, size.width, size.height,
```

```
            userRotation, InputImage.IMAGE_FORMAT_NV21
        )
        handleImageAnalysis(inputImage)
    } else if (frame.dataClass === Image::class.java) {
        val data: Image = frame.getData()
        val inputImage =
            InputImage.fromMediaImage(data, degreesToFirebaseRotation(userRotation))
        handleImageAnalysis(inputImage)
    }
    frame.release()
}

private fun configuration() {
    binding.cameraView.whiteBalance = WhiteBalance.AUTO
    binding.cameraView.addFrameProcessor(this)

}

private fun allPermissionsGranted() = REQUIRED_PERMISSIONS.all {
    ContextCompat.checkSelfPermission(
        this, it
    ) == PackageManager.PERMISSION_GRANTED
}

companion object {
    private const val REQUEST_CODE_PERMISSIONS = 10
    private val REQUIRED_PERMISSIONS =
        mutableListOf(
            android.Manifest.permission.CAMERA,
            android.Manifest.permission.ACCESS_FINE_LOCATION
        ).apply {
            if (Build.VERSION.SDK_INT <= Build.VERSION_CODES.P) {
                add(android.Manifest.permission.WRITE_EXTERNAL_STORAGE)
            }
        }.toTypedArray()
}
```

```
    override fun process(frame: Frame) {

        frameProcessor(frame)

    }

    private fun startLocationUpdates() {


    }

    private fun stopLocationUpdates() {
        fusedLocationClient.removeLocationUpdates(locationCallback)
    }


    override fun onPause() {
        super.onPause()
       locationManager.removeUpdates(locationListener)
    }
}
```

XML FILE

```
<?xml version="1.0" encoding="utf-8"?>
<FrameLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".BarcodeScanner">


    <com.otaliastudios.cameraview.CameraView
```

```
    android:id="@+id/cameraView"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:layout_gravity="center_vertical"
    app:cameraAudio="off"
    app:cameraFacing="back" />

<androidx.appcompat.widget.Toolbar
    android:id="@+id/toolbar"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"

    android:background="@android:color/transparent">

</androidx.appcompat.widget.Toolbar>

<FrameLayout
    android:layout_width="match_parent"
    android:layout_margin="20dp"
    android:layout_gravity="center"
    android:id="@+id/scannIndicator"
    android:layout_height="wrap_content">

    <com.thirdwinder.qrgenerator.ScanningIndicator
        android:layout_width="match_parent"
        android:layout_height="250dp" />
</FrameLayout>
<com.google.android.material.progressindicator.CircularProgressIndicator
    android:layout_width="wrap_content"
    android:indeterminate="true"
    android:id="@+id/progress"
    android:visibility="invisible"
    android:layout_height="wrap_content"
    android:layout_gravity="center"
    app:indicatorSize="30dp"
    />
```

```
</FrameLayout>
```

GRADLE

```
plugins {
    id 'com.android.application'
    id 'org.jetbrains.kotlin.android'
}

android {
    namespace 'com.thirdwinder.qrgenerator'
    compileSdk 33

    defaultConfig {
        applicationId "com.thirdwinder.qrgenerator"
        minSdk 21
        targetSdk 33
        versionCode 1
        versionName "1.0"

        testInstrumentationRunner "androidx.test.runner.AndroidJUnitRunner"
    }

    buildFeatures {
        viewBinding true
    }
    buildTypes {
        release {
            minifyEnabled false
            proguardFiles                getDefaultProguardFile('proguard-android-optimize.txt'),
'proguard-rules.pro'
        }
    }
    compileOptions {
```

```
        sourceCompatibility JavaVersion.VERSION_1_8
        targetCompatibility JavaVersion.VERSION_1_8
    }
    kotlinOptions {
        jvmTarget = '1.8'
    }
}

dependencies {
    implementation 'androidx.core:core-ktx:1.7.0'
    implementation 'androidx.appcompat:appcompat:1.6.1'
    implementation 'com.google.android.material:material:1.9.0'
    implementation 'androidx.constraintlayout:constraintlayout:2.1.4'
    implementation 'com.google.android.gms:play-services-location:21.0.1'
    testImplementation 'junit:junit:4.13.2'
    androidTestImplementation 'androidx.test.ext:junit:1.1.5'
    androidTestImplementation 'androidx.test.espresso:espresso-core:3.5.1'
    implementation 'com.google.zxing:core:3.4.1'
    implementation 'com.google.code.gson:gson:2.10'
    implementation 'com.google.mlkit:object-detection:17.0.0'
    implementation ("com.otaliastudios:cameraview:2.7.2")
    implementation 'com.google.mlkit:barcode-scanning:17.1.0'
}
```

MAIN ACTIVITY

```
package com.thirdwinder.qrgenerator

import android.content.Intent
import android.graphics.Bitmap
import androidx.appcompat.app.AppCompatActivity
import android.os.Bundle
import com.google.gson.Gson
import com.google.zxing.BarcodeFormat
import com.google.zxing.WriterException
```

```
import com.google.zxing.common.BitMatrix
import com.google.zxing.qrcode.QRCodeWriter
import com.thirdwinder.qrgenerator.algorithm.BasicData
import com.thirdwinder.qrgenerator.databinding.ActivityMainBinding
import com.thirdwinder.qrgenerator.ui.CameraActivity
import com.thirdwinder.qrgenerator.ui.GenerateCodePage

class MainActivity : AppCompatActivity() {

    private val binding: ActivityMainBinding by lazy {
        ActivityMainBinding.inflate(layoutInflater)
    }

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(binding.root)

        Intent(this, BarcodeScanner::class.java)
            .also {
                startActivity(it)
            }
        /*  binding.nextButton.setOnClickListener {
              val data = processData()
              Intent(this, GenerateCodePage::class.java)
                  .apply {
                      putExtra("data", data)
                  }.also {
                      startActivity(it)
                  }
          }*/
    }

    fun generateMatrix(rows: Int, cols: Int): List<List<Int>> {
        val matrix = mutableListOf<MutableList<Int>>()

        for (i in 0 until rows) {
```

```
            val row = mutableListOf<Int>()
            for (j in 0 until cols) {
                row.add((Math.random() * 100).toInt()) // Generate random values between 0
and 100
            }
            matrix.add(row)
        }

        return matrix
    }

    private fun processData(): String {
        val mid = binding.mid.text.toString()
        val batchNumber = binding.batchNumber.text.toString()
        val serielNumber = binding.serielNumber.text.toString()
        val data = BasicData(mid, batchNumber, serielNumber, "20/24", "10/23", listOf())
        val gson = Gson()
        return gson.toJson(data)
    }

}
```

QR UTIL

```
package com.thirdwinder.qrgenerator.util
import android.graphics.Bitmap
import android.graphics.Color
import android.graphics.Matrix
import android.util.Log
import com.google.zxing.BarcodeFormat
import com.google.zxing.EncodeHintType
import com.google.zxing.WriterException
import com.google.zxing.qrcode.QRCodeWriter
import java.util.*
object QRCodeUtils {
```

```
    private const val TAG = "QRCodeUtils"

    fun generateQRCode(content: String, size: Int): Bitmap? {
        try {
            val hints = EnumMap<EncodeHintType, Any>(EncodeHintType::class.java)
            hints[EncodeHintType.CHARACTER_SET] = "UTF-8"
            val qrCodeWriter = QRCodeWriter()
            val bitMatrix = qrCodeWriter.encode(content, BarcodeFormat.QR_CODE, size,
size, hints)
            val width = bitMatrix.width
            val height = bitMatrix.height
            val pixels = IntArray(width * height)
            for (y in 0 until height) {
                val offset = y * width
                for (x in 0 until width) {
                    pixels[offset + x] = if (bitMatrix[x, y]) Color.BLACK else Color.WHITE
                }
            }
            val bitmap = Bitmap.createBitmap(width, height, Bitmap.Config.ARGB_8888)
            bitmap.setPixels(pixels, 0, width, 0, 0, width, height)
            return bitmap
        } catch (e: WriterException) {
            Log.e(TAG, "Failed to generate QR code: ${e.message}")
        }
        return null
    }
}



package com.thirdwinder.qrgenerator.ui

import android.content.pm.PackageManager
import android.graphics.Bitmap
import android.graphics.BitmapFactory
```

```
import android.graphics.Canvas
import android.graphics.ColorMatrix
import android.graphics.ColorMatrixColorFilter
import android.graphics.Matrix
import android.graphics.Paint
import android.media.Image
import android.net.Uri
import android.os.Build
import androidx.appcompat.app.AppCompatActivity
import android.os.Bundle
import android.util.Log
import android.view.Surface
import android.view.View
import android.widget.Toast
import androidx.core.app.ActivityCompat
import androidx.core.content.ContextCompat
import com.google.mlkit.vision.common.InputImage
import com.google.mlkit.vision.objects.ObjectDetection
import com.google.mlkit.vision.objects.ObjectDetector
import com.google.mlkit.vision.objects.defaults.ObjectDetectorOptions
import com.otaliastudios.cameraview.CameraException
import com.otaliastudios.cameraview.CameraListener
import com.otaliastudios.cameraview.PictureResult
import com.otaliastudios.cameraview.controls.WhiteBalance
import com.otaliastudios.cameraview.frame.Frame
import com.otaliastudios.cameraview.frame.FrameProcessor
import com.thirdwinder.qrgenerator.R
import com.thirdwinder.qrgenerator.databinding.ActivityCameraBinding
import java.io.File
import java.io.IOException

class CameraActivity : AppCompatActivity(), FrameProcessor, View.OnClickListener {

    private val binding: ActivityCameraBinding by lazy {
        ActivityCameraBinding.inflate(layoutInflater)
    }
```

```
private lateinit var recognizer: ObjectDetector
override fun onCreate(savedInstanceState: Bundle?) {
    super.onCreate(savedInstanceState)
    setContentView(binding.root)

    binding.captureButton.setOnClickListener(this)
    binding.close.setOnClickListener(this)
    if (allPermissionsGranted()) {
        binding.camera.setLifecycleOwner(this)
        configuration()
    } else {
        ActivityCompat.requestPermissions(
            this,
            REQUIRED_PERMISSIONS,
            REQUEST_CODE_PERMISSIONS
        )
        binding.camera.setLifecycleOwner(this)
        configuration()
    }
}

private fun configuration() {
    binding.camera.whiteBalance = WhiteBalance.AUTO
   // binding.camera.addFrameProcessor(this)
}


private fun allPermissionsGranted() = REQUIRED_PERMISSIONS.all {
    ContextCompat.checkSelfPermission(
        this, it
    ) == PackageManager.PERMISSION_GRANTED
}

companion object {
    private const val REQUEST_CODE_PERMISSIONS = 10
```

```
    private val REQUIRED_PERMISSIONS =
        mutableListOf(
            android.Manifest.permission.CAMERA,
        ).apply {
            if (Build.VERSION.SDK_INT <= Build.VERSION_CODES.P) {
                add(android.Manifest.permission.WRITE_EXTERNAL_STORAGE)
            }
        }.toTypedArray()
}

override fun process(frame: Frame) {
    frameProcessor(frame)
}

override fun onStart() {
    super.onStart()
    val options = ObjectDetectorOptions.Builder()
        .setDetectorMode(ObjectDetectorOptions.SINGLE_IMAGE_MODE)
        .enableMultipleObjects()
        .enableClassification()
        .build()
    recognizer = ObjectDetection.getClient(options)

}

private fun degreesToFirebaseRotation(orientation: Int): Int {
    val rotation: Int = when (orientation) {
        in 45..134 -> {
            Surface.ROTATION_270
        }
        in 135..224 -> {
            Surface.ROTATION_180
        }
        in 225..314 -> {
            Surface.ROTATION_90
        }
```

```
        else -> {
            Surface.ROTATION_0
        }
    }
    return rotation
}


override fun onClick(p0: View?) {

    when (p0) {
        binding.close -> {
            onBackPressed()
        }
        binding.captureButton -> {
            println("********** button clicked")
            captureImage()
        }
    }
}

private fun handleImageAnalysis(image: InputImage) {
    /*  recognizer.process(image)
          .addOnSuccessListener {
              // Task completed successfully
              println("********************** object detected ***************")

          }
          .addOnFailureListener {
              // Task failed with an exception
              Log.e("MainActivity","")

          }*/

}
```

```
private fun frameProcessor(frame: Frame) {
    val size: com.otaliastudios.cameraview.size.Size = frame.size
    val userRotation: Int = frame.rotationToUser
    if (frame.dataClass === ByteArray::class.java) {
        val data: ByteArray = frame.getData()
        val inputImage = InputImage.fromByteArray(
            data, size.width, size.height,
            userRotation, InputImage.IMAGE_FORMAT_NV21
        )
        handleImageAnalysis(inputImage)
    } else if (frame.dataClass === Image::class.java) {
        val data: Image = frame.getData()
        val inputImage =
            InputImage.fromMediaImage(data, degreesToFirebaseRotation(userRotation))
        handleImageAnalysis(inputImage)
    }
    frame.release()
}

override fun onPause() {
    super.onPause()
    binding.camera.close()
}

private fun captureImage() {
    binding.camera.addCameraListener(object : CameraListener() {
        override fun onPictureTaken(result: PictureResult) {
            val file = File.createTempFile("${System.currentTimeMillis()}", ".jpg")
            println("********** picture taken")
            try {

                result.toFile(file) { _file ->
                    println("********** picture taken and saved")
                    val decoded = BitmapFactory.decodeFile(file.path)
                    val resized = resizeImage(decoded, 256, 256)
                    binding.previewImage.setImageBitmap(resized)
```

```
                println("image resized")
                if (_file != null) {

                }
            }

        } catch (e: IOException) {
            e.printStackTrace()
            Toast.makeText(this@CameraActivity, e.localizedMessage,
Toast.LENGTH_SHORT)
                .show()
        }
    }

    override fun onCameraError(exception: CameraException) {
        super.onCameraError(exception)
        exception.printStackTrace()
    }
  })
}

fun resizeImage(originalBitmap: Bitmap, newWidth: Int, newHeight: Int): Bitmap {
    val width = originalBitmap.width
    val height = originalBitmap.height
    val scaleWidth = newWidth.toFloat() / width
    val scaleHeight = newHeight.toFloat() / height
    val matrix = Matrix()
    matrix.postScale(scaleWidth, scaleHeight)
    val result = Bitmap.createBitmap(originalBitmap, 0, 0, width, height, matrix, true)
    val colorMatrix = ColorMatrix()
    colorMatrix.setSaturation(0f)
    val paint = Paint()
    val filter = ColorMatrixColorFilter(colorMatrix)
    paint.colorFilter = filter
    val newBitmap = Bitmap.createBitmap(result.width, result.height,
Bitmap.Config.ARGB_8888)
```

```
        val canvas = Canvas(newBitmap)
        canvas.drawBitmap(result, 0f, 0f, paint)
        return newBitmap
    }
}
```

```
package com.thirdwinder.qrgenerator.ui

import android.graphics.Bitmap
import androidx.appcompat.app.AppCompatActivity
import android.os.Bundle
import android.os.Environment
import com.google.gson.Gson
import com.google.zxing.BarcodeFormat
import com.google.zxing.WriterException
import com.google.zxing.common.BitMatrix
import com.google.zxing.qrcode.QRCodeWriter
import com.thirdwinder.qrgenerator.R
import com.thirdwinder.qrgenerator.algorithm.BasicData
import com.thirdwinder.qrgenerator.databinding.ActivityGenerateCodePageBinding
import java.io.File
import java.io.FileOutputStream
import java.io.OutputStream

class GenerateCodePage : AppCompatActivity() {

    private val binding: ActivityGenerateCodePageBinding by lazy {
        ActivityGenerateCodePageBinding.inflate(layoutInflater)
    }

    var resultBitmap: Bitmap? = null

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
```

```
        setContentView(binding.root)
        val data = intent.getStringExtra("data") ?: return
        try {

            val matrix = generateMatrix(16, 16).flatten()
            val actualData = Gson().fromJson(data, BasicData::class.java).copy(imgData = matrix)
            val bitMatrix = generateQRCode(Gson().toJson(actualData))
            val qrCodeBitmap = createBitmapFromBitMatrix(bitMatrix)
            resultBitmap = qrCodeBitmap
            binding.qrCodeImageView.setImageBitmap(qrCodeBitmap)
        } catch (e: WriterException) {
            e.printStackTrace()
        }

        binding.save.setOnClickListener {
            if (resultBitmap != null) {
                saveBitmapAsImage(resultBitmap!!, "qrcode")
            }
        }
    }


    private fun generateMatrix(rows: Int, cols: Int): List<List<Int>> {
        val matrix = mutableListOf<MutableList<Int>>()

        for (i in 0 until rows) {
            val row = mutableListOf<Int>()
            for (j in 0 until cols) {
                row.add((Math.random() * 100).toInt()) // Generate random values between 0 and 100
            }
            matrix.add(row)
        }

        return matrix
```

```
    }

    private fun generateQRCode(data: String): BitMatrix {
        val qrCodeWriter = QRCodeWriter()
        return qrCodeWriter.encode(data, BarcodeFormat.QR_CODE, 1024, 1024)
    }

    private fun createBitmapFromBitMatrix(matrix: BitMatrix): Bitmap {
        val width = matrix.width
        val height = matrix.height
        val bitmap = Bitmap.createBitmap(width, height, Bitmap.Config.RGB_565)

        for (x in 0 until width) {
            for (y in 0 until height) {
                bitmap.setPixel(x, y, if (matrix[x, y]) 0xFF000000.toInt() else
0xFFFFFFFF.toInt())
            }
        }

        return bitmap
    }


    private fun saveBitmapAsImage(bitmap: Bitmap, filename: String) {
        val file = File.createTempFile(filename,".png")
        println("*******${file.path}")
        val outputStream: OutputStream?

        try {
            outputStream = FileOutputStream(file)
            bitmap.compress(Bitmap.CompressFormat.PNG, 100, outputStream)
            outputStream.flush()
            outputStream.close()
            println("Image saved successfully.")
        } catch (e: Exception) {
            e.printStackTrace()
```

```
            println("Error saving image.")
        }
    }

}


package com.thirdwinder.qrgenerator.algorithm

data class BasicData(
    val manufacturerId: String,
    val batchNumber: String,
    val serialNumber: String,
    val productionDate: String,
    val expiryDate: String,
    val imgData:List<Int>
)


<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    tools:context=".MainActivity">


    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="20dp"
        android:layout_marginTop="50dp"
        android:text="Manufacturer's ID"
        android:textSize="17sp"
```

```
    android:textStyle="bold" />

<com.google.android.material.textfield.TextInputLayout
    android:id="@+id/midContainer"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_marginStart="20dp"
    android:layout_marginTop="5dp"
    android:layout_marginEnd="20dp">

    <com.google.android.material.textfield.TextInputEditText
        android:id="@+id/mid"
        android:layout_width="match_parent"
        android:layout_height="wrap_content" />
</com.google.android.material.textfield.TextInputLayout>


<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginStart="20dp"
    android:layout_marginTop="10dp"
    android:text="Batch number"
    android:textSize="17sp"
    android:textStyle="bold" />

<com.google.android.material.textfield.TextInputLayout
    android:id="@+id/batchNumberContainer"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_marginStart="20dp"
    android:layout_marginTop="5dp"
    android:layout_marginEnd="20dp">

    <com.google.android.material.textfield.TextInputEditText
        android:id="@+id/batchNumber"
```

```
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:inputType="number" />
</com.google.android.material.textfield.TextInputLayout>


<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginStart="20dp"
    android:layout_marginTop="10dp"
    android:text="Products serial number"
    android:textSize="17sp"
    android:textStyle="bold" />

<com.google.android.material.textfield.TextInputLayout
    android:id="@+id/serialNumberContainer"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_marginStart="20dp"
    android:layout_marginTop="5dp"
    android:layout_marginEnd="20dp">

    <com.google.android.material.textfield.TextInputEditText
        android:id="@+id/serielNumber"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:inputType="number" />

</com.google.android.material.textfield.TextInputLayout>

<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginStart="20dp"
    android:layout_marginTop="10dp"
```

```
    android:text="Date of production"
    android:textSize="17sp"
    android:textStyle="bold" />

<RelativeLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_marginTop="10dp">

    <TextView
        android:id="@+id/monthText"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignStart="@id/month"
        android:layout_alignParentTop="true"
        android:text="Month" />

    <com.google.android.material.textfield.TextInputLayout
        android:id="@+id/month"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@id/monthText"
        android:layout_marginStart="20dp"
        android:layout_marginTop="5dp"
        android:layout_marginEnd="20dp">

        <com.google.android.material.textfield.TextInputEditText
            android:layout_width="wrap_content"
            android:layout_height="55dp"
            android:maxLength="2"
            android:inputType="number"
            android:textAlignment="center"
            android:maxLines="1" />
    </com.google.android.material.textfield.TextInputLayout>

    <com.google.android.material.textfield.TextInputLayout
```

```
        android:id="@+id/year"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@id/yearText"
        android:layout_marginStart="20dp"
        android:layout_marginTop="5dp"
        android:layout_marginEnd="20dp"
        android:layout_toEndOf="@id/month">

        <com.google.android.material.textfield.TextInputEditText
            android:layout_width="wrap_content"
            android:layout_height="55dp"
            android:inputType="number"
            android:textAlignment="center"
            android:maxLength="4"
            android:maxLines="1" />
    </com.google.android.material.textfield.TextInputLayout>

    <TextView
        android:id="@+id/yearText"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignStart="@id/year"
        android:text="Year" />
</RelativeLayout>


<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginStart="20dp"
    android:layout_marginTop="20dp"
    android:text="Expiry date"
    android:textSize="17sp"
    android:textStyle="bold" />
```

```
<RelativeLayout
   android:layout_width="match_parent"
   android:layout_height="wrap_content"
   android:layout_marginTop="10dp">

   <TextView
      android:id="@+id/monthText1"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:layout_alignStart="@id/month1"
      android:layout_alignParentTop="true"
      android:text="Month" />

   <com.google.android.material.textfield.TextInputLayout
      android:id="@+id/month1"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:layout_below="@id/monthText1"
      android:layout_marginStart="20dp"
      android:layout_marginTop="5dp"
      android:layout_marginEnd="20dp">

      <com.google.android.material.textfield.TextInputEditText
         android:layout_width="wrap_content"
         android:layout_height="55dp"
         android:maxLength="2"
         android:inputType="number"
         android:textAlignment="center"
         android:maxLines="1" />
   </com.google.android.material.textfield.TextInputLayout>

   <com.google.android.material.textfield.TextInputLayout
      android:id="@+id/year1"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:layout_below="@id/yearText1"
```

```
        android:layout_marginStart="20dp"
        android:layout_marginTop="5dp"
        android:layout_marginEnd="20dp"
        android:layout_toEndOf="@id/month1">

        <com.google.android.material.textfield.TextInputEditText
            android:layout_width="wrap_content"
            android:layout_height="55dp"

            android:inputType="number"
            android:textAlignment="center"
            android:maxLength="4"
            android:maxLines="1" />
    </com.google.android.material.textfield.TextInputLayout>

    <TextView
        android:id="@+id/yearText1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignStart="@id/year1"
        android:text="Year" />
  </RelativeLayout>

  <com.google.android.material.button.MaterialButto
    android:layout_width="match_parent"
    android:layout_height="60dp"
    android:layout_margin="20dp"
    android:text="Continue"
    android:textAllCaps="false"
    app:cornerRadius="10dp" />

</LinearLayout
```