

Blockchain-based product authenticity verification

By

Yeu Qi Rui

A REPORT

SUBMITTED TO

Universiti Tunku Abdul Rahman

in partial fulfillment of the requirements

for the degree of

BACHELOR OF INFORMATION TECHNOLOGY (HONOURS)

COMMUNICATIONS AND NETWORKING

Faculty of Information and Communication Technology
(Kampar Campus)

June 2025

COPYRIGHT STATEMENT

© 2025 Yeu Qi Rui. All rights reserved.

This Final Year Project report is submitted in partial fulfillment of the requirements for the degree of **Bachelor of Information Technology (Honours) Communications and Networking** at Universiti Tunku Abdul Rahman (UTAR). This Final Year Project proposal represents the work of the author, except where due acknowledgement has been made in the text. No part of this Final Year Project proposal may be reproduced, stored, or transmitted in any form or by any means, whether electronic, mechanical, photocopying, recording, or otherwise, without the prior written permission of the author or UTAR, in accordance with UTAR's Intellectual Property Policy.

ACKNOWLEDGEMENTS

I would like to express my sincere thanks and appreciation to my supervisor, Ts Dr Gan Ming Lee, who has given me this bright opportunity to engage in IT blockchain project. It is my first step to establish a career in IT blockchain field. A million thanks to you.

To a very special person in my life, Ng Hai Rou, for her patience, unconditional support, and love, and for standing by my side during hard times. Finally, I must say thanks to my parents and my family for their love, support, and continuous encouragement throughout the course.

ABSTRACT

This project proposes a blockchain-based product authentication system designed to enhance security, traceability, and transparency in supply chains, addressing the growing problem of counterfeit products in sectors such as pharmaceuticals, food, and electronics. Traditional centralised systems are prone to manipulation and lack transparency, resulting in counterfeit goods entering the market, which poses significant risks to consumers and businesses alike. The proposed system integrates blockchain technology with smart contracts to provide a decentralised, tamper-proof method for verifying product authenticity. Blockchain ensures that product data, including production and distribution history, is recorded immutably, while ECC offers efficient encryption for securing sensitive data. Additionally, the use of Ganache is utilised for decentralised data storage, ensuring data redundancy and security. The system incorporates blockchain details to enable secure consumer verification of product authenticity in real time. The methodology involves system architecture design, blockchain node setup, smart contract deployment, and local website development for verification purposes. The expected outcome of this project is an innovative, robust system that improves supply chain integrity, prevents the circulation of counterfeit products, and enhances consumer trust in product authenticity.

Area of Study (Maximum 2): **Blockchain technology**

Keywords (Maximum 10): **Smart contracts, Anti-Counterfeiting, Traceability, Decentralised Ledger, Friendly UI**

TABLE OF CONTENTS

| | |
|--------------------------------------|-------------|
| TITLE PAGE | i |
| COPYRIGHT STATEMENT | ii |
| ACKNOWLEDGEMENTS | iii |
| ABSTRACT | iv |
| TABLE OF CONTENTS | v |
| LIST OF FIGURES | vx |
| LIST OF TABLES | xii |
| LIST OF ABBREVIATIONS | xiii |
| | |
| CHAPTER 1 INTRODUCTION | 1 |
| 1.1 Problem Statement and Motivation | 2 |
| 1.2 Problem Objective | 5 |
| 1.3 Project Scope and Direction | 6 |
| 1.4 Contributions | 7 |
| 1.5 Report Organization | 8 |

| | |
|--|---------------|
| CHAPTER 2 LITERATURE REVIEW | 9 |
| 2.1 Previous Works on Blockchain in Verifying Authenticity of Product | 9 |
| 2.1.1 Product authenticity verification using blockchain and OTP | 9 |
| 2.1.2 Blockchain-based traceability system for agricultural products | 11 |
| 2.1.3 Blockchain-based solution for medication on anti-counterfeiting and traceability | 14 |
| 2.1.4 Product authentication technology integrating blockchain and traceability structure | 17 |
| 2.1.5 Blockchain-based platforms for diamond authentication and certification in luxury supply chains | 21 |
| 2.2 Limitation of Previous Studies | 23 |
| 2.2.1 Limitations of product authenticity verification using blockchain and OTP | 23 |
| 2.2.2 Limitations of blockchain-based traceability system for agricultural products | 24 |
| 2.2.3 Limitations of blockchain-based solution for medication on anti-counterfeiting and traceability | 24 |
| 2.2.4 Limitations of product authentication technology integrating blockchain and traceability structure | 25 |
| 2.2.5 Limitations of blockchain-based platforms for diamond authentication and certification in luxury supply chains | 26 |
| 2.2.6 Summary | 26 |
| 2.3 Proposed Solutions | 28 |
| CHAPTER 3 SYSTEM METHODOLOGY/APPROACH | 31 |
| 3.1 System Requirements | 31 |
| 3.1.1 Hardware | 31 |

| | |
|---|---------------|
| 3.1.2 Software | 32 |
| 3.2 Project Costing | 33 |
| 3.2.1 Hardware Costs | 33 |
| 3.2.2 Software Costs | 34 |
| 3.2.3 Cloud Hosting Costs | 34 |
| 3.2.4 Other Costs | 35 |
| 3.2.5 Summary of Cost Estimation | 35 |
| 3.3 System Architecture Diagram and Explanation | 36 |
| 3.4 Use Case Diagram and Explanation | 38 |
| 3.5 Project Timeline | 40 |
| CHAPTER 4 SYSTEM DESIGN | 42 |
| 4.1 Setting Up | 42 |
| 4.2 System Design | 42 |
| 4.2.1 Truffle | 42 |
| 4.2.2 Solidity | 44 |
| 4.2.3 Ganache | 45 |
| 4.2.4 Microsoft Visual Studio Code (React) | 48 |

| | |
|---|---------------|
| CHAPTER 5 SYSTEM IMPLEMENTATION | 50 |
| 5.1 Settings and Configuration | 50 |
| 5.2 System Operation | 55 |
| 5.3 Implementation Issues and Challenges | 73 |
| 5.4 Concluding Remarks | 74 |
| CHAPTER 6 SYSTEM EVALUATION AND DISCUSSION | 75 |
| 6.1 System Testing and Performance Metrics | 75 |
| 6.2 Objectives Evaluation | 82 |
| 6.3 Concluding Remark | 84 |
| CHAPTER 7 CONCLUSION AND RECOMMENDATION | 85 |
| 7.1 Conclusion | 85 |
| 7.2 Recommendations | 86 |
| REFERENCES | 89 |
| APPENDIX | |
| A.1 Poster | A-1 |

LIST OF FIGURES

| Figure Number | Title | Page |
|----------------------|--|-------------|
| Figure 2.1 | Proposed system model for blockchain | 6 |
| Figure 2.2 | The flow diagram of the proposed supply chain | 7 |
| Figure 2.3 | Blockchain based traceability system architecture | 8 |
| Figure 2.4 | Traceability information privacy protection data flow diagram | 9 |
| Figure 2.5 | User interface and product label | 10 |
| Figure 2.6 | Blockchain-based medication anti-counterfeiting and traceability model | 11 |
| Figure 2.7 | The improved PBFT consensus mechanism flowchart | 12 |
| Figure 2.8 | Sequence diagram of participating entities interacting with smart contract | 13 |
| Figure 2.9 | Product certification technology architecture | 14 |
| Figure 2.10 | Product traceability certification architecture diagram | 15 |
| Figure 2.11 | Authentication system process | 16 |
| Figure 2.12 | The Hyperledger Sawtooth to track the product lifecycle and authenticity | 21 |
| Figure 2.13 | The diamond-lapse program by BTS platform Everledger | 22 |
| Figure 2.14 | The sample report of the origin and people involved in diamond retailing | 23 |
| Figure 3.1 | The system architecture diagram of the project | 37 |
| Figure 3.2 | User Case Diagram | 38 |
| Figure 4.1 | The screenshot of the truffle installation | 42 |
| Figure 4.2 | The screenshot of the compilation of contracts | 43 |
| Figure 4.3 | The screenshot of the written contract using Solidity | 44 |
| Figure 4.4 | The solidity code to record the product and save it into the blockchain | 44 |
| Figure 4.5 | The ganache workspace | 46 |

| | | |
|-------------|---|----|
| Figure 4.6 | The accounts that existed in the local ganache | 46 |
| Figure 4.7 | The transaction details after the smart contracts are deployed | 47 |
| Figure 4.8 | Installing React using the command prompt | 48 |
| Figure 4.9 | The Visual Studio Code used to debug and write code | 48 |
| Figure 4.10 | The user details to verify and view the previous history | 49 |
| Figure 5.1 | Ganache local workspace | 51 |
| Figure 5.2 | The ganache server details | 51 |
| Figure 5.3 | The accounts on the local Ganache application | 52 |
| Figure 5.4 | The output at cmd after migrations of contracts | 53 |
| Figure 5.5 | The output at cmd after launching React | 54 |
| Figure 5.6 | The truffle-config settings in the Visual Studio Code | 55 |
| Figure 5.7 | The manufacturer portal frontend using React | 56 |
| Figure 5.8 | The Register New Product function | 56 |
| Figure 5.9 | The Register New Product function after completion | 57 |
| Figure 5.10 | The Register New Product function after completion (local ganache) | 57 |
| Figure 5.11 | The Register New Product when duplicate registration occurs | 58 |
| Figure 5.12 | The Register New Product when the manufacturer does not have access | 58 |
| Figure 5.13 | The Update Product Details | 59 |
| Figure 5.14 | The Update Product Details when completed | 59 |
| Figure 5.15 | The Update Product Details when the user is not actual manufacturer | 59 |
| Figure 5.16 | The Product Verification function | 60 |
| Figure 5.17 | The Verify Product function before verification | 60 |
| Figure 5.18 | The Product Verification function after verification | 61 |
| Figure 5.19 | The Verify Product function after verification | 61 |

| | | |
|-------------|--|----|
| Figure 5.20 | The Transfer Product Ownership function (can transfer) | 62 |
| Figure 5.21 | The Transfer Product Ownership function (can't transfer) | 62 |
| Figure 5.22 | The Transfer Product Ownership function after completion | 63 |
| Figure 5.23 | The details of the transfer show on the product history | 63 |
| Figure 5.24 | The Role Management function | 64 |
| Figure 5.25 | The Role Management function after completion | 64 |
| Figure 5.26 | The User Dashboard overview | 65 |
| Figure 5.27 | The Verify Product function overview | 66 |
| Figure 5.28 | The Physical Verification | 67 |
| Figure 5.29 | The Physical Verification when user is not the product owner | 67 |
| Figure 5.30 | The Physical Verification after completion | 68 |
| Figure 5.31 | The Verify Product after verification | 68 |
| Figure 5.32 | The Transfer Ownership function (product owner) | 69 |
| Figure 5.33 | The Transfer Ownership function (not product owner) | 70 |
| Figure 5.34 | The Transfer Ownership function after completion | 70 |
| Figure 5.35 | The Verify Product after Transfer Ownership function | 71 |
| Figure 5.36 | The transaction blocks on the local ganache after registering product | 72 |
| Figure 5.37 | The transaction blocks on the local ganache after transferring product | 72 |

LIST OF TABLES

| Table Number | Title | Page |
|---------------------|---|-------------|
| Table 2.1 | The summary of limitations in each study | 28 |
| Table 3.1 | Specifications of Desktop | 32 |
| Table 3.2 | Specifications of Smartphone | 33 |
| Table 3.3 | Software Components and Corresponding Purpose | 33 |
| Table 3.4 | Hardware Costs for the Project | 34 |
| Table 3.5 | Software Costs for the Project | 35 |
| Table 3.6 | Cloud Hosting Costs for the Project | 35 |
| Table 3.7 | Other Costs for the Project | 36 |
| Table 3.8 | Summary Costs for the Project | 36 |
| Table 3.9 | Timeline for FYP 1 | 40 |
| Table 3.10 | Timeline for FYP 2 | 41 |
| Table 5.1 | The system testing and results | 76 |

LIST OF ABBREVIATIONS

| | |
|--------------|---|
| <i>QR</i> | Quick Response |
| <i>NFC</i> | Near Field Communication |
| <i>AI</i> | Artificial Intelligence |
| <i>IoT</i> | Internet of Things |
| <i>OTP</i> | One Time Password |
| <i>RFID</i> | Radio Field Identification |
| <i>QCO</i> | Quality Control Officer |
| <i>CBC</i> | Cipher Block Chaining |
| <i>PBFT</i> | Practical Byzantine Fault Tolerance |
| <i>RBAC</i> | Role-Based Access Control |
| <i>CC</i> | Certification Center |
| <i>IPFS</i> | InterPlanetary File System |
| <i>PCR</i> | Product Certification Result |
| <i>PoW</i> | Proof of Work |
| <i>HACCP</i> | Hazard Analysis and Critical Control Points |
| <i>ECC</i> | Elliptic Curve Cryptography |

CHAPTER 1 INTRODUCTION

Background

The rapid emergence of counterfeit products across various industries and countries, ranging from pharmaceuticals and electronics to luxury goods, has become a significant challenge for consumers, who often lack the necessary tools and knowledge to validate the authenticity of products. Traditional centralised systems for product authentication and supply chain management are increasingly susceptible to manipulation, data breaches, and inefficiencies, which can result in the circulation of fake products. These counterfeit goods not only cause substantial financial losses for businesses but also pose serious risks to consumer health and safety. As such, developing effective strategies to combat counterfeit products is imperative, and blockchain technology offers a promising solution.

According to [1], Blockchain is a decentralised, distributed, and public digital ledger technology that maintains a continuously growing list of ordered records, called blocks, which are linked using cryptography. Each block inside the system contains a cryptographic hash of the previous block, a timestamp, and transaction data, making the blockchain incredibly secure and tamper-proof. This system was initially introduced by the pseudonymous Satoshi Nakamoto in 2008 [2], as the underlying technology for Bitcoin, Blockchain solves the double-spending problem without the need for a trusted central authority. While it started with cryptocurrency, people are starting to know and explore the blockchain's potential applications extend far beyond imagination, including payment processing, supply chain monitoring, digital identity management, secure data sharing, copyright and royalty protection, and many more. Its key benefits include significant time and cost savings, enhanced security against fraud and cybercrime, and the elimination of duplicated efforts through shared ledgers. Blockchain employs smart contracts for self-executing agreements, permissions to secure transactions, and various consensus mechanisms to validate network activity. However, while often considered secure, blockchain is not immune to vulnerabilities, such as 51% of attacks that can compromise its integrity. The technology continues to evolve, with projects like Hyperledger driving cross-industry adoption, emphasising the importance of transparency, performance, and security in blockchain applications.

The Integration of blockchain Into”prod’ct authentication processes offers several key advantages. Firstly, it decentralises the verification process, eliminating the need for a central authority and thereby reducing the risk of data manipulation and fraud. Secondly, blockchain’s immutable nature ensures that once data is recorded, it cannot be altered or deleted, providing a permanent and trustworthy record of a product’s journey through the supply chain. Thirdly, the transparency provided by blockchain allows all stakeholders—including manufacturers, distributors, retailers, and consumers—to access real-time information about the provenance and authenticity of a product.

Beyond its inherent security features, blockchain can be integrated with other emerging technologies such as the Internet of Things (IoT), Near-Field Communication (NFC), and Artificial Intelligence (AI) to further enhance the authentication process. For example, IoT devices can monitor and record environmental conditions throughout the supply chain, while NFC tags can provide consumers with instant access to authentication information via their smartphones.

This study aims to explore the application of blockchain technology in product authentication within supply chains to verify the authenticity of the product. By developing a blockchain-based framework, the study seeks to address the limitations of current centralised systems and improve the overall trustworthiness and efficiency of supply chains. The proposed system will not only enhance security but also provide a more transparent and reliable method for verifying product authenticity, ultimately protecting both businesses and consumers from the growing threat of counterfeit goods.

1.1 Problem Statement and Motivation

Problem Statement

The first problem that exists is that traditional product authentication methods, which rely on centralised systems, are exposed easily to manipulation and data breaches, and this kind of system lacks transparency. Counterfeit products present serious risks, including financial losses, health hazards, and damage to brand reputation.

Therefore, the need for a decentralised, tamper-proof system is critical to ensure reliable, verifiable, and transparent product authentication throughout the entire supply chain—from manufacturing to delivery. For instance, in the case of edible products, the addition or omission of chemicals can have severe consequences for consumers who are unaware of how the product has been processed. So, an authentication system would allow manufacturers to quickly identify and recall problematic batches, while consumers could verify the safety and integrity of the products they purchase. Consumers can use technology such as mobile app applications, NFCs, SMS, or even unique code generation to secure authenticity. This level of transparency would make it much harder for counterfeit or defective products to enter the market, protecting both consumers and businesses alike.

The next problem is the lack of traceability of the product worldwide. The article in [3] mentioned that the Haagen-Dazs vanilla ice cream that had been pulled from supermarket shelves had higher carcinogenic element Ethylene Oxide (ETO) levels than normal. This would possibly threaten the consumer's life, and it should be taken seriously. The Health Ministry of Malaysia has made statements that the products will not be marketed in the country. If there is a blockchain system that can help monitor and detect such action, the government can take quicker action before the scenario escalates.

The lack of traceability of products makes the distribution of counterfeit products, particularly in critical sectors like healthcare, pose a severe threat to public safety and trust. A notable example is the widespread circulation of fake COVID-19 vaccines in India, as discussed by Choudhary et al [4]. This issue not only endangers public health but also exacerbates vaccine hesitancy, undermining efforts to control the pandemic. Thousands of individuals in regions like West Bengal and Maharashtra were administered counterfeit vaccines, leading to a significant loss of trust in the vaccination process and worsening the spread of COVID-19 over time.

Motivation

The growing prevalence of counterfeit products and the challenges in verifying the authenticity of goods on the market require a better solution to address this increase in fake products being sold on the market. Current centralised systems fail to provide the necessary transparency and security to prevent the spread of fake products. For instance, consumers often struggle to determine whether the items they purchase are genuine, especially when counterfeit goods closely mimic the quality of the originals. One example is what was mentioned in the article [5] where in China, Putian, there existed a business mainly to produce fake shoes and sell them to customers worldwide. This is a very serious issue since buyers like us might not be able to tell the difference between authentic products and high-grade fake items produced there. This study aims to address these issues by leveraging blockchain technology to establish a decentralised, secure, and transparent system for tracking products from their origin to the end consumer, ensuring authenticity at every stage.

To address the problem of counterfeit products, particularly in critical sectors like healthcare, food manufacturing or even electronics, an authentication system that uses blockchain technology will be implemented. This system will verify the authenticity of products at every stage of the supply chain, ensuring that only genuine products are exported to the market. By using a decentralised and tamper-proof blockchain system, this solution will prevent counterfeit or fake products from entering the market, thereby protecting public health, restoring trust, and enhancing overall safety.

Driven by the need to enhance the security and reliability of supply chains, this research seeks to reduce the incidence of counterfeit products, safeguard consumer rights, and improve supply chain efficiency. Additionally, the project will explore the integration of dynamic authentication methods, such as QR code scanning and application-based verification, to strengthen the authentication process throughout the product's journey from production to delivery.

1.2 Project Objective

1. Enhanced Product Authentication

Implement a decentralised blockchain-based system to verify and authenticate products through unique serial number registration and tamper-proof on-chain records, ensuring that consumers can securely confirm product authenticity.

2. Improved Traceability

Utilise blockchain technology to provide complete tracking of the product lifecycle, including registration, updates, and ownership transfers, with immutable transaction records that ensure transparency and accountability.

3. Prevention of Counterfeiting

Leverage smart contracts to prevent duplicate registrations and enforce registrar-based permissions, ensuring that only authorised entities can register products and thereby minimising the risk of counterfeit entries.

4. Data Security and Access Control

Ensure data integrity and security by leveraging blockchain immutability and role-based access control, restricting operations such as registration, updating, and verification to authorised users only.

5. User-Friendly Verification

Develop a React-based web application integrated with MetaMask that allows users to easily search for products using serial numbers, verify authenticity, and view product details through an intuitive interface.

6. Reliable Distributed Storage

Use blockchain technology as a secure and immutable ledger to store product data, ensuring availability, tamper resistance, and transparency of all recorded product transactions.

Project Scope and Direction

The project scope in this paper can be divided into four main sections, which are blockchain ledger for product traceability, smart contracts for automation and compliance, blockchain account details as parameters for verification, and data encryption for security.

To ensure product traceability and transparency, a blockchain ledger is implemented to record the journey of each product from manufacturing to the end consumer. This ledger stores essential information such as product details, production dates, and ownership history, creating a complete and trustworthy record. All transactions — including product registration, updates, and ownership transfers — are immutably recorded, ensuring that every modification is transparent and verifiable.

In addition to traceability, smart contracts are utilised to automate supply chain operations and enforce compliance. These contracts define the rules for product registration, updates, and transfers, ensuring that only authorised accounts are able to perform such actions. This automation reduces the risk of errors and eliminates reliance on intermediaries.

To strengthen authenticity, blockchain account details are used as parameters for product verification. Each product is linked to a unique blockchain account, preventing duplicate registrations and enabling stakeholders to confirm ownership and authenticity directly on the blockchain.

Finally, data encryption methods are applied to secure sensitive product and user information. Encryption safeguards the integrity and confidentiality of all recorded and transmitted data, ensuring that only authorised parties can access protected information while maintaining trust in the system.

1.3 Contributions

The contributions that can be achieved in this project include solving the problem of the lack of security in the old method of centralised systems for authentication and providing more reliability and security to the entire process. By leveraging blockchain's decentralised ledger, the proposed system will eliminate the need for central authorities, thereby reducing the risk of data manipulation and improving trust among customers.

The second contribution is that it provides a dynamic authentication method. The system will incorporate dynamic authentication methods, such as blockchain account verification or even unique code generation, to verify the authenticity of products at each stage of the supply chain. This adds a layer of security, ensuring that only authorised parties can access and verify product information. This method is also easy to use and is user-friendly for all consumers, regardless of age. This makes the confirmation of counterfeit products clearer and more accessible. This can also effectively increase the confidence of consumers towards the product since the authenticity of the different products can be ensured.

The third major contribution is scalability and flexibility. The proposed framework will be designed to be scalable, accommodating various industries and product types. This makes it not only applicable to one but many different industries that include maybe sectors like healthcare, food supply, or even luxury brands. It will also be flexible enough to integrate with existing supply chain management systems, allowing businesses to adopt the new system without overhauling their current processes. This adaptability ensures that the system can evolve with changing industry needs and continue to provide robust security and authentication in the long term.

1.4 Report Organization

This report is organised into 7 chapters: Chapter 1 Introduction, Chapter 2 Literature Review, Chapter 3 System Methodology, Chapter 4 System Design, Chapter 5 System Implementation, Chapter 6 System Evaluation and Discussion and finally Chapter 7 Conclusion and Recommendations. The first chapter is the introduction of this project, which includes introduction, problem statement and motivation, project objectives, project scope and direction and project contribution, and report organisation. The second chapter is the literature review carried out on several existing blockchain-based voting systems to evaluate the strengths and weaknesses of each system. The third chapter discusses the overall system approach and methods for this project. The fourth chapter is regarding the system design that has been used to build the system, where the code will be explained and elaborated in each section of the system. The fifth chapter is the system implementation, where the project system is being demonstrated and shown. The sixth chapter is the evaluation of the whole system and the objectives as well. Lastly, chapter 7 summarises the work done and further improvements that could be made in the future

CHAPTER 2 LITERATURE REVIEW

2.1 Previous Works on Blockchain in Verifying the Authenticity of Product

2.1.1 Product authenticity verification using blockchain and OTP

In the research studies in [6], the author mentioned that the system integrates Blockchain technology with OTP verification to ensure the authenticity and integrity of products as they move through the supply chain, from the factory to the consumer. The key components of this system are the verification using OTP and the implementation of blockchain in the

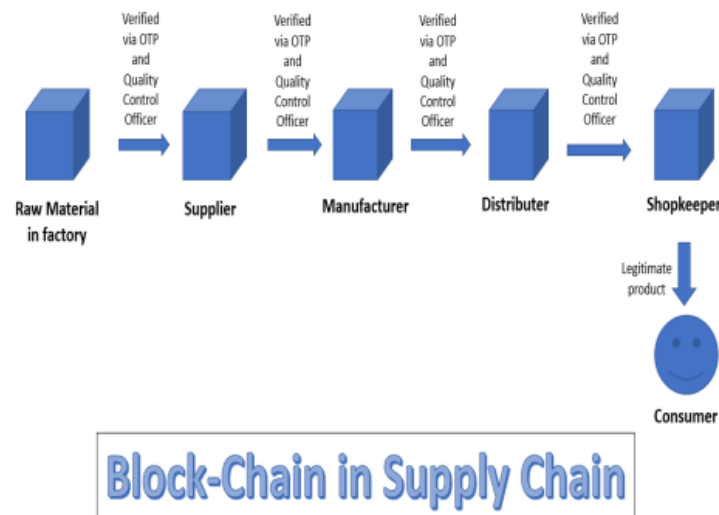


Figure 2.1: Proposed system model for blockchain

The picture above shows how the process of verification occurs during the flow of transporting products in a supply chain. There will be an OTP verification needed for every station of the supply chain and the user will be required to visit a website and log in to the account manually before being able to receive the OTP verification. The system will generate a code with a unique hash value to prevent others from using it. There is also the involvement of officers needed for spot-checking. The Quality Control Officer (QCO) is introduced by the authors in the research paper to randomly check the products that exist in the process to ensure that the consumers can receive the genuine and real products that they have ordered and bought.

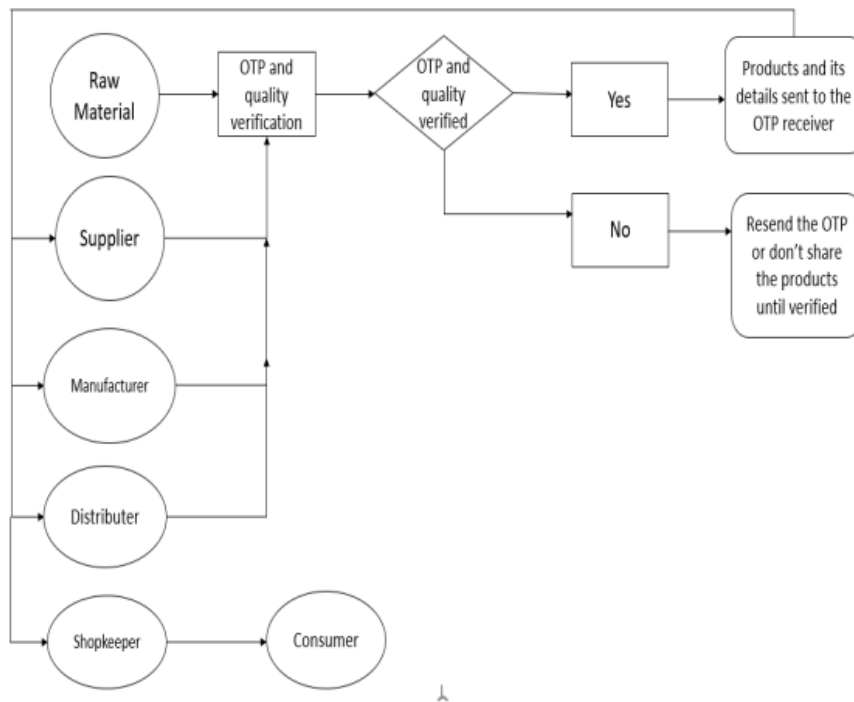


Figure 2.2: The flow diagram of the Proposed Supply Chain

The diagram above shows how the products will be transported and how the blockchain and OTP technology can be implemented within the supply chain to enhance the authenticity of the products. If the quality of the products does not meet the criteria, the OTP will not be sent, and the manufacturer has to dispose of the unqualified product so that the quality of the product is ensured.

2.1.2 Blockchain-based traceability system for agricultural products

The research studies in [7] have shown the traceability of agricultural products in the agricultural industry using blockchain. The method separates the supply chain into four parts: mainly production, processing, logistics, and sales. Each of these links will capture different essential data. For example, the production links will focus more on the details such as seedling information, environmental conditions, and more, while the processing links will focus more on the production details such as processing, packaging, and recording key information.

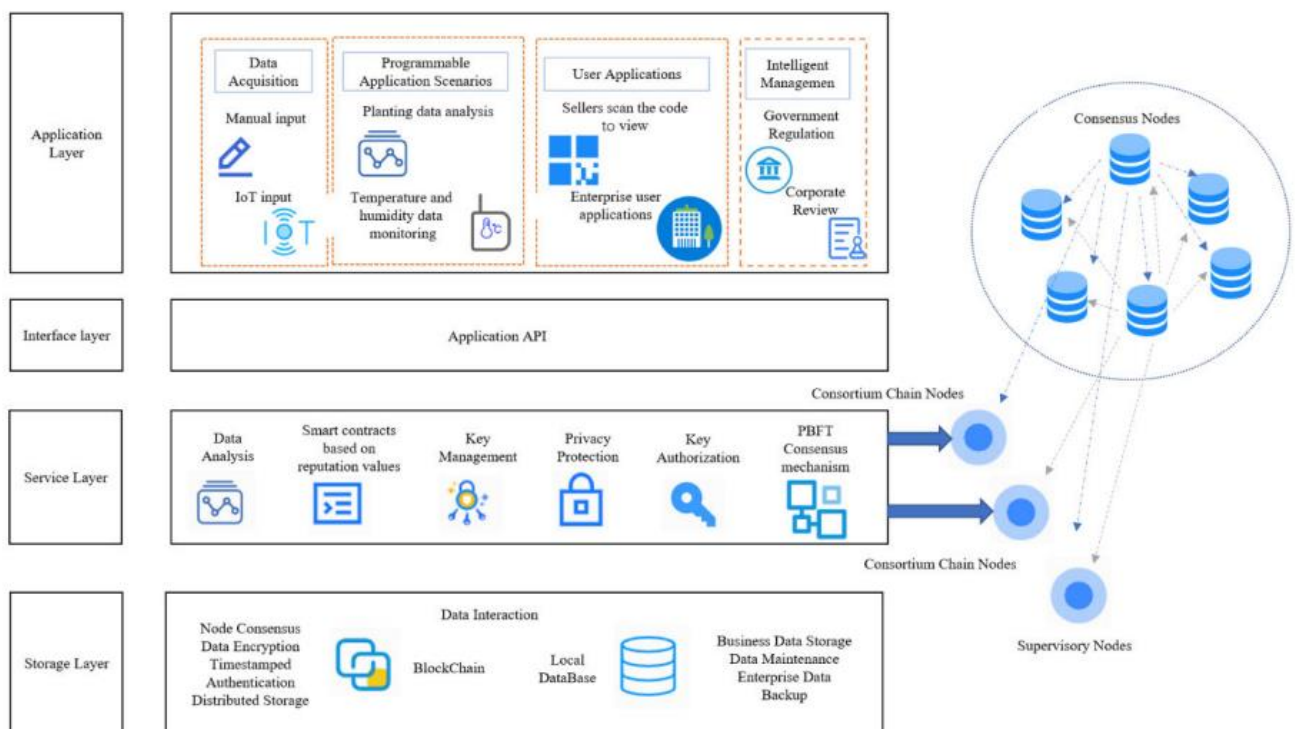


Figure 2.3: Blockchain-based traceability system architecture

To provide more clarity and transparency to the system, this 4-tier architecture is introduced whereby the layer starts from storage, service, interface, and application. The storage layer is mainly responsible for storing the information on the database locally while the encrypted private information is stored on the blockchain. The service layer handles data analysis and smart contracts needed for the transaction. The interface layer includes the implementation of the data upload into the blockchain system and data query functions. Lastly, the application layer provides functionalities for different users for different purposes, whether it is data upload, traceability queries, or even governmental supervision.

The system protects the data by different encryption such as Cipher Block Chaining (CBC) using AES. The data encrypted is more secure as a smart contract will be generated together with an encryption key known as Key 1, which is further encrypted using Elliptic Curve Cryptographic (ECC) before being uploaded to the blockchain to enhance the security. The data can only be accessed by authorised data, and no editing can be done to ensure the sensitive data remains secure throughout the whole process [8].

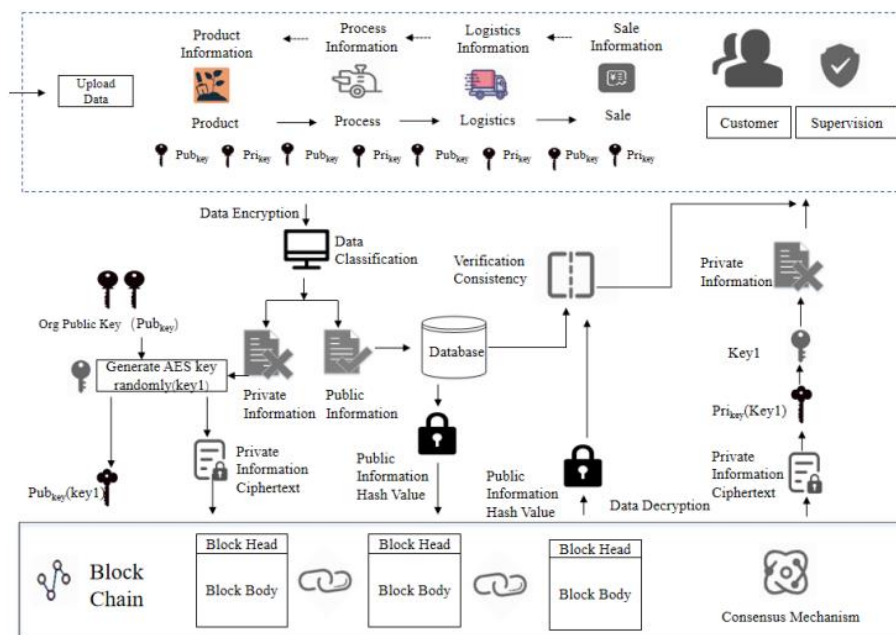


Figure 2.4: Traceability information privacy protection data flow diagram

The system combats counterfeiting by allowing consumers to verify the authenticity of product information. Public information is stored in a local database and hashed. The hash is then stored on the blockchain.

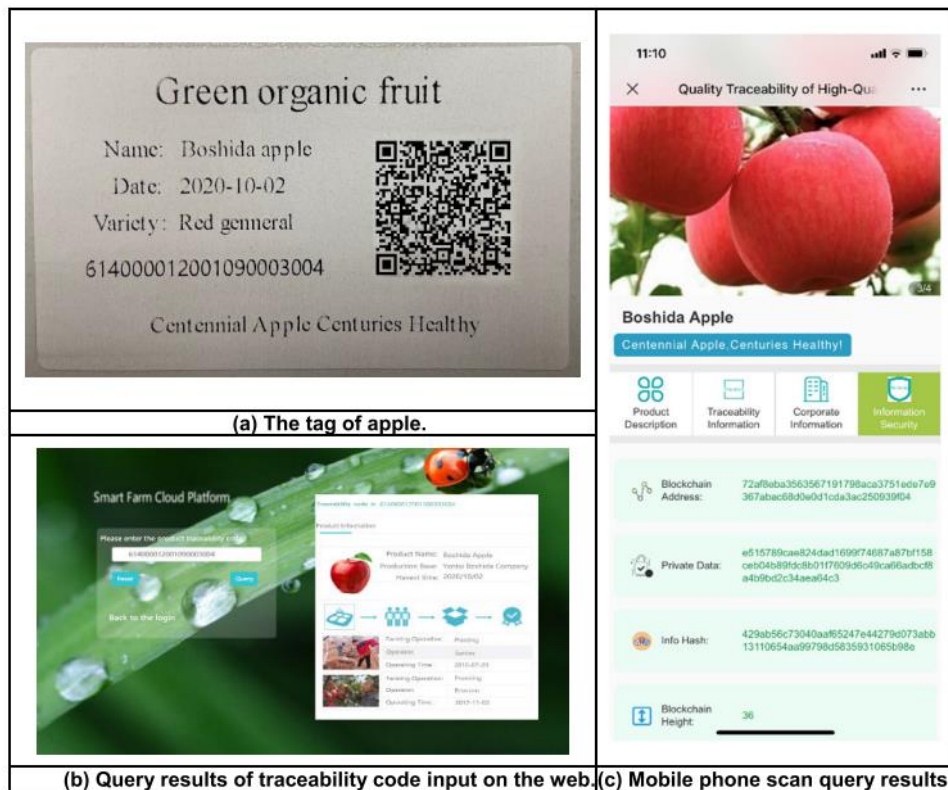


Figure 2.5: User interface and product label

Users can scan a QR code to access product information and verify its authenticity by comparing the hash value on the blockchain. The system employs the Practical Byzantine Fault Tolerance (PBFT) consensus mechanism, which is well-suited for consortium chains with fewer nodes, ensuring data consistency and security [9]. The system's user interface includes modules for managing farms, production operations, crop traceability, and blockchain management, among others. The user can create and manage traceability information and view blockchain status in real-time through the pages created.

2.1.3 Blockchain-based solution for medication on anti-counterfeiting and traceability

The study in [10] mentions the blockchain implementation on medication for anti-counterfeiting and traceability. The key point mentioned is first the blockchain technology utilised to record and store data across the entire medication supply chain, from manufacturer to consumer. Every node in the supply chain autonomously uploads data into the system, ensuring that each step is documented and traceable.

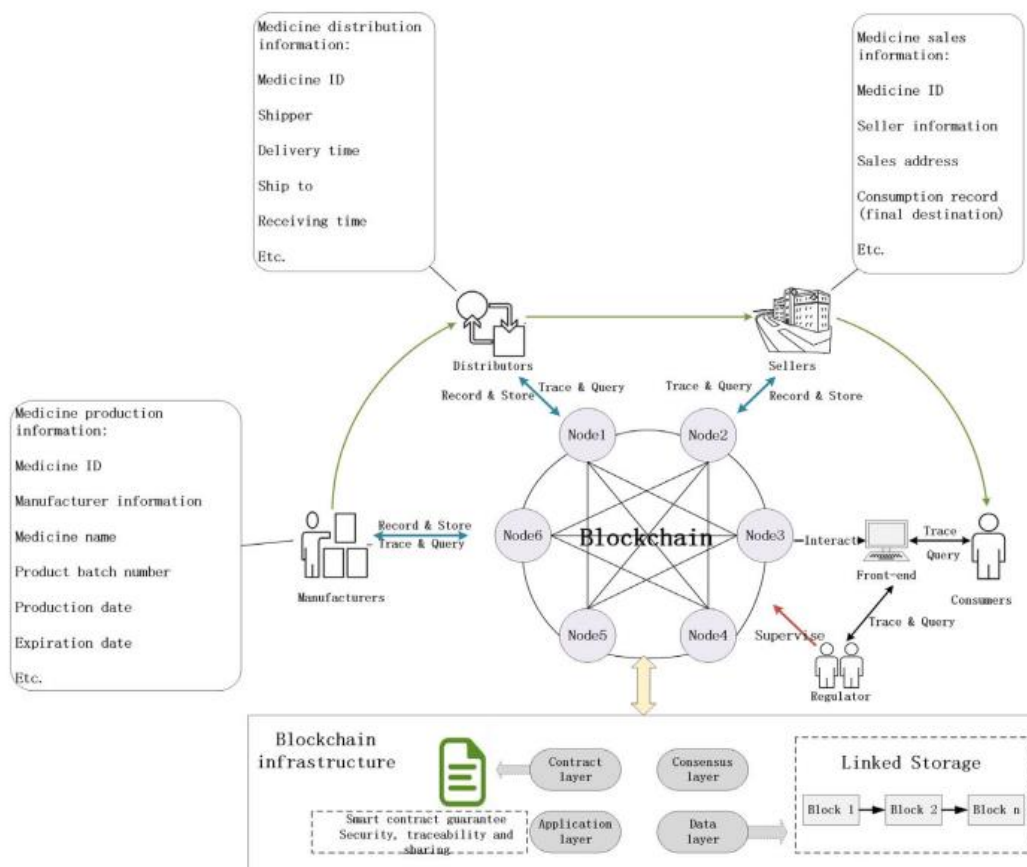


Figure 2.6: Blockchain-based medication anti-counterfeiting and traceability model.

The diagram above shows the flow diagram for the blockchain model proposed. The model uses hashing to ensure that the medical information is unique, fixed and is non-reversible or modifiable. The verification process is also easy as the users can straight away compare the hash value with a newly calculated one, so any alteration in the data can be detected.

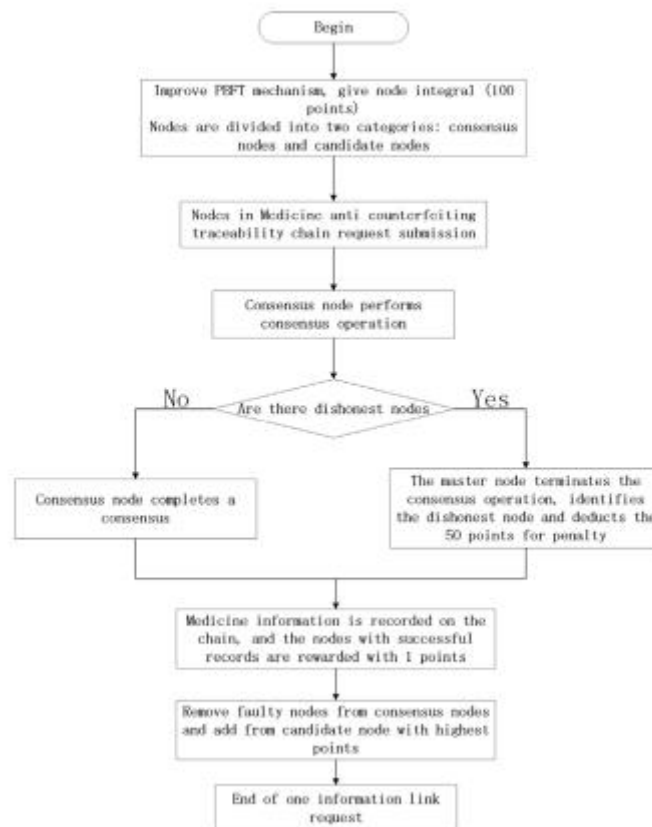


Figure 2.7: The improved PBFT consensus mechanism flowchart

The diagram is shown above to show more information on how the PBFT works in a flow chart. The system uses consensus mechanisms where the data is separated into different nodes, and each dishonest or suspicious node is identified and can be removed from the consensus process, leaving only trustworthy nodes to continue storing. The system uses the Practical Byzantine Fault Tolerance (PBFT) algorithm to improve the efficiency and effectiveness of the system, as it can tolerate up to $(N-1)/3$ of errors within the network, where N is the total number of nodes [8]. The honest nodes are rewarded and promoted while the dishonest or suspicious nodes get downgraded during the consensus process, and the continuous evaluation can ensure the integrity and reliability of the blockchain network.

The data security is then maintained through a model known as Role-Based Access Control (RBAC). The model allows the system to categorise users based on their roles and assign different permissions accordingly. This ensures the security of the sensitive information. The integration of the smart contracts with RBAC is essential during the process as it automates the process of granting and verifying access permissions. This combination can enhance the security, making the system more secure and flexible.

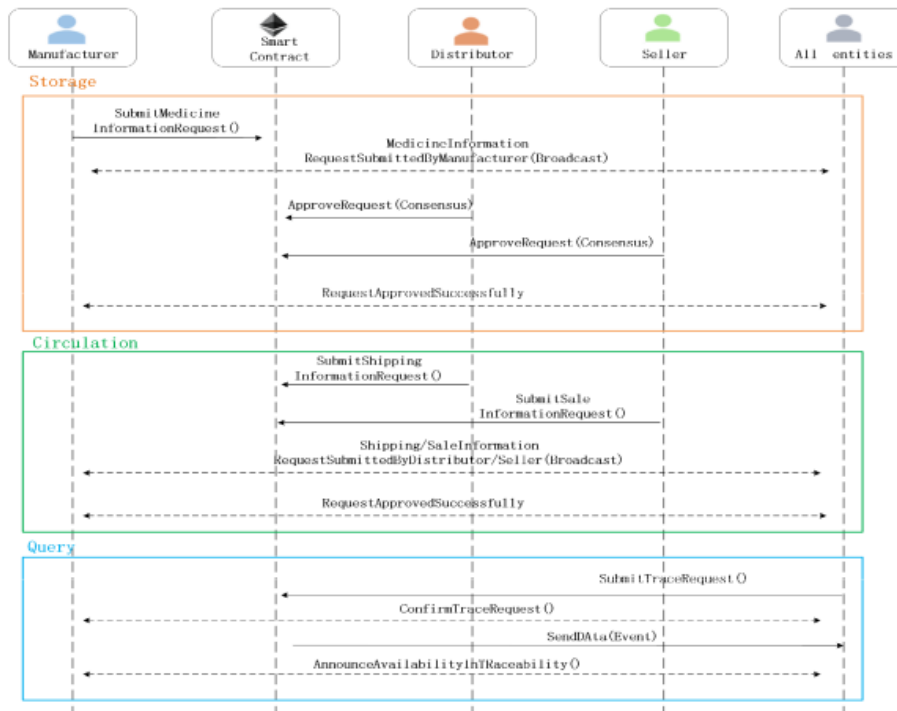


Figure 2.8: Sequence diagram of participating entities interacting with smart contract.

The diagram above shows the sequence of how data is recorded. The manufacturer is the first link in the supply chain the information will be updated and recorded as the first block. The order of information is then continuously updated and recorded as the nodes continue. If a manufacturer tries to modify the recorded medication information, the algorithm can compare the current block with the next one to prevent any alteration of data. The users can then trace the information of the medication based on the ID and address, allowing for easier product authentication.

2.1.4 Product authentication technology integrating blockchain and traceability structure

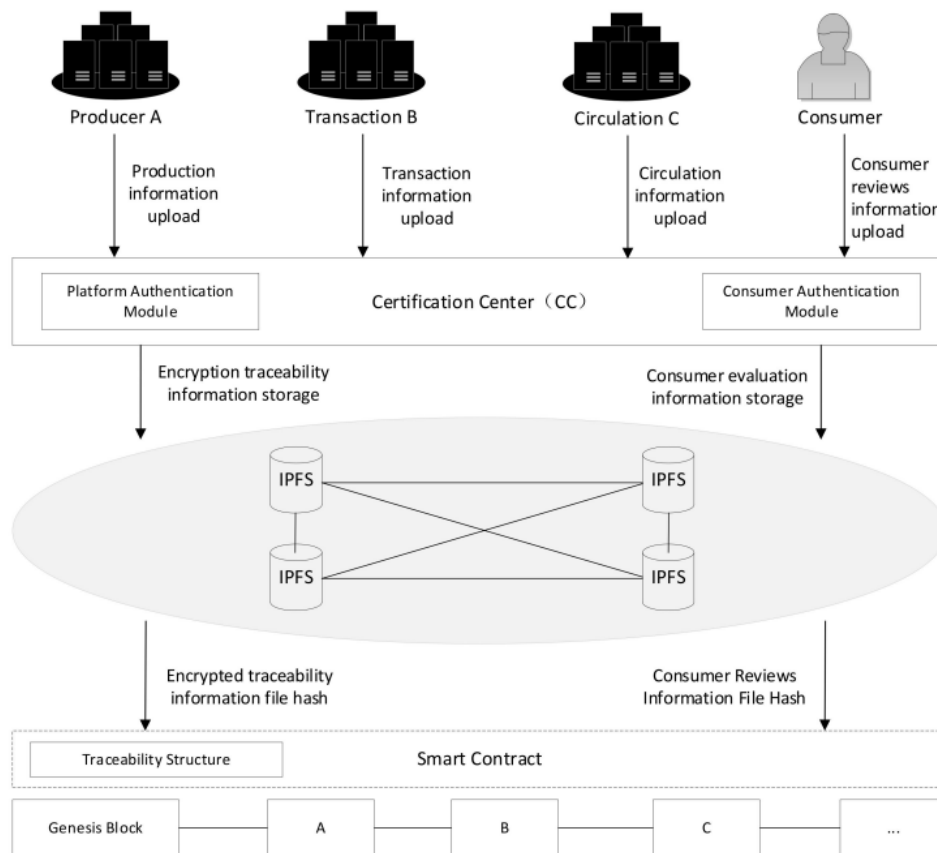


Figure 2.9: Product certification technology architecture

The studies in [11] shows that the ability to tackle the questioning of the products faced in e-commerce live streaming and low data integrity along the process. The model proposed by the author consists of 6 main parts, which are the producer, transaction party, circulation party, certification center, blockchain, and IPFS. Each of the parts are responsible for different tasks. For example, the producer, transaction party and circulation party are mainly responsible for uploading the relevant information about the product while the certification center is involved in two main sections which are consumer certification and platform certification. The blockchain element used is Ethereum to store the encrypted and traceable information in the InterPlanetary File System (IPFS). Figure 9 shown above is the structure proposed by the author which consists of 6 parts.

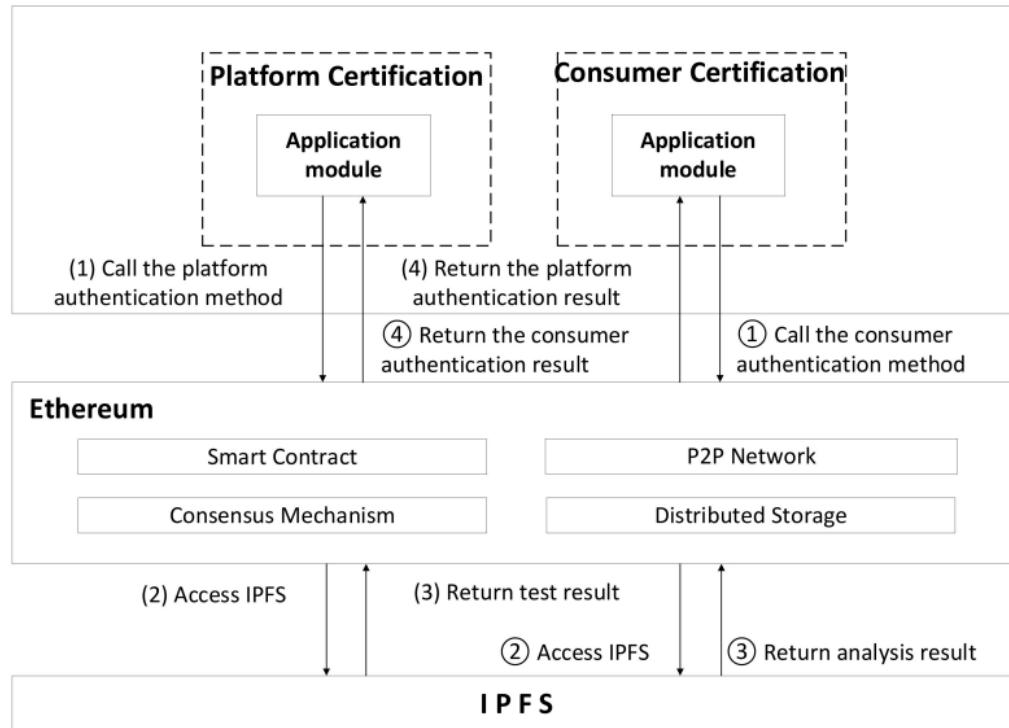


Figure 2.10: Product traceability certification architecture diagram

The figure above illustrates the realization of the product traceability certification method based on the supply chain, which is managed by the Certification Center (CC) and supported by the Ethereum platform and the InterPlanetary File System (IPFS). The Certification Center included the entire certification process, which is divided into two main parts, platform certification and consumer certification [12].

The Ethereum platform serves as the blockchain layer, securing the certification process by storing critical data such as the storage locations of consumer reviews and encrypted traceability information. Utilizing smart contracts and the Proof of Work (PoW) consensus algorithm, Ethereum ensures data integrity and security. IPFS provides a distributed storage system that securely handles large files, such as encrypted traceability data and consumer reviews, complementing the blockchain by offering efficient data retrieval.

Platform certification integrates smart contracts to verify the product's traceability information stored on IPFS. This process involves comparing the hash values of the product's production, transaction, and circulation information with the hash values stored on the blockchain. A match certifies the product as genuine, while a mismatch results in certification failure.

Consumer certification, on the other hand, analyses feedback stored on IPFS. This feedback is processed using a trained FastText model, which evaluates the proportion of positive (non-negative) reviews to reflect consumer satisfaction with the product, thus contributing to the final product certification.

The certification process is triggered when the number of consumer certification applications reaches a preset threshold or at a scheduled time. Once initiated, the platform certification and consumer certification run simultaneously. Platform certification verifies the integrity of traceability data, while consumer certification assesses authenticity based on consumer feedback.

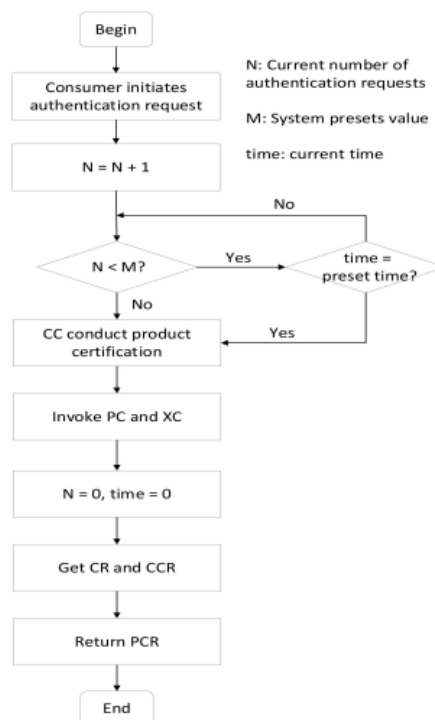


Figure 2.11: Authentication system process

The picture above shows the flowchart of the whole authentication system process for a clearer view. The final Product Certification Result (PCR) combines the outcomes of both processes. If the platform certification result (CR) is positive (True), the product is confirmed as authentic, supported by a percentage of consumers agreeing with this assessment. If the platform certification result is negative (False), the product may still be considered possibly genuine, depending on the consumer feedback received.

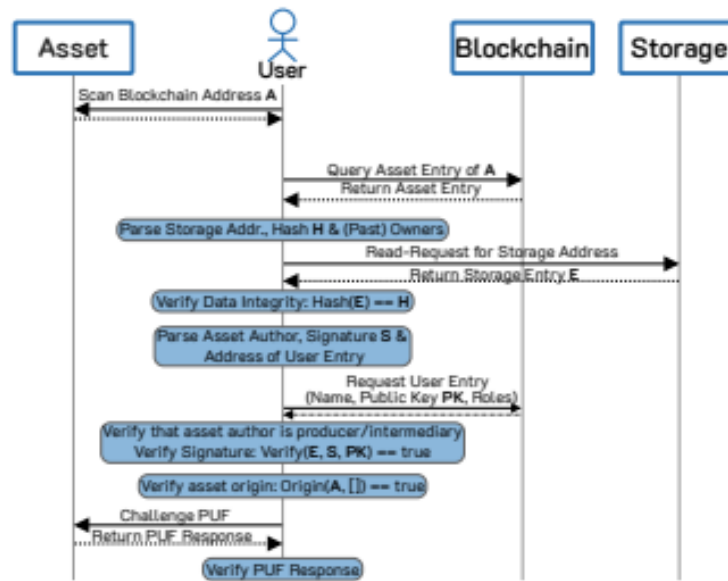


Figure 2.12: The Hyperledger Sawtooth to track the product lifecycle and authenticity

In the research paper [13], the author shows the sequence diagram explaining how a user verifies an asset's authenticity in the SPOQchain system by interacting with the asset using tools such as qr code or NFC tag. The first step is to scan a blockchain address A from the asset and then query the blockchain for the asset entry. For example, using Address A, the user queries the blockchain to get the Asset Entry. The blockchain returns the Asset Entry, which includes basic information like the storage address, past owners, and hash of the storage entry. The user parses the returned asset data which reads Storage Address, Hash (H), and Past Owners. Then, the system reads the actual Storage Entry I from the storage server (like IPFS or HTTP server).

In this way, the user verifies that the hash of the retrieved Storage Entry I matches the hash H stored on the blockchain. This step ensures no tampering has happened in the storage data. The user then parses the asset's author information: the digital Signature (S) and the address of the user entry and request user entry from the blockchain to verify the signature and role. After this, if the asset came from a batch, it verifies the asset's origin by tracing back through its parent batches.

2.1.5 Blockchain-based platforms for diamond authentication and certification in luxury supply chains

The study paper in [14] mentioned that the purchaser of the luxury items is concerned with authenticity, such as the quality of the diamond on the necklace and so forth. Thus, it is important to provide the details on the diamond origin so that customers interested in buying can see and learn the steps involved from start to end using the blockchain system. With the new technology of the blockchain, it uses platforms such as Everledger, which implement a digital thumbprint to create the diamond details from mining to certification and even including the weight, colour and the cutting. This digital thumbprint is implemented by a laser system that can scan the diamond and record its unique features, and the integration of blockchain can be used to handle and store all the details, such as the origin, manufacturing of the diamonds mined. By using the decentralised feature of the blockchain, consumers, sellers and even the certifiers can access the data anytime and ensuring full traceability and security of the diamond's journey.

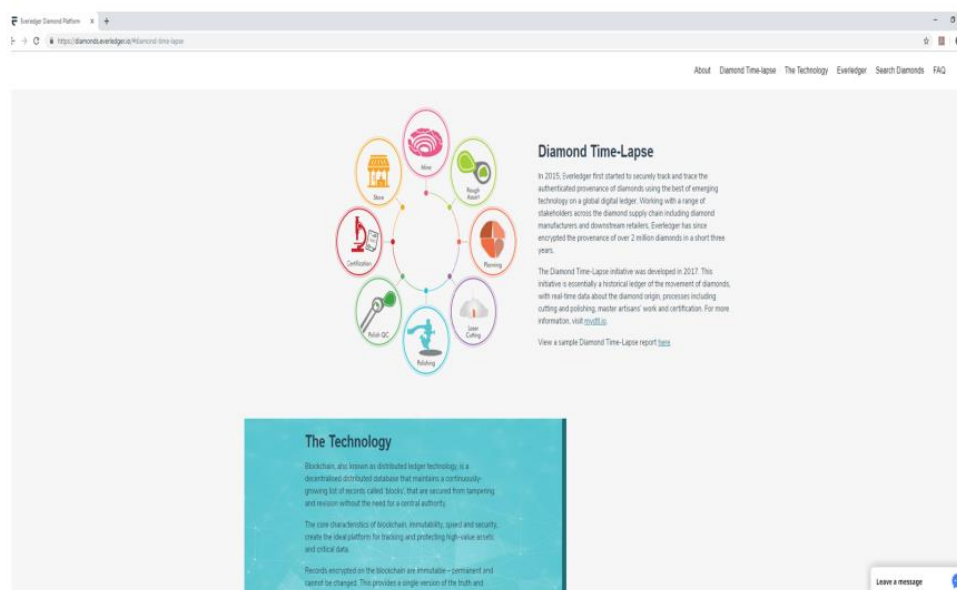


Figure 2.13: The diamond-lapse program by BTS platform Everledger

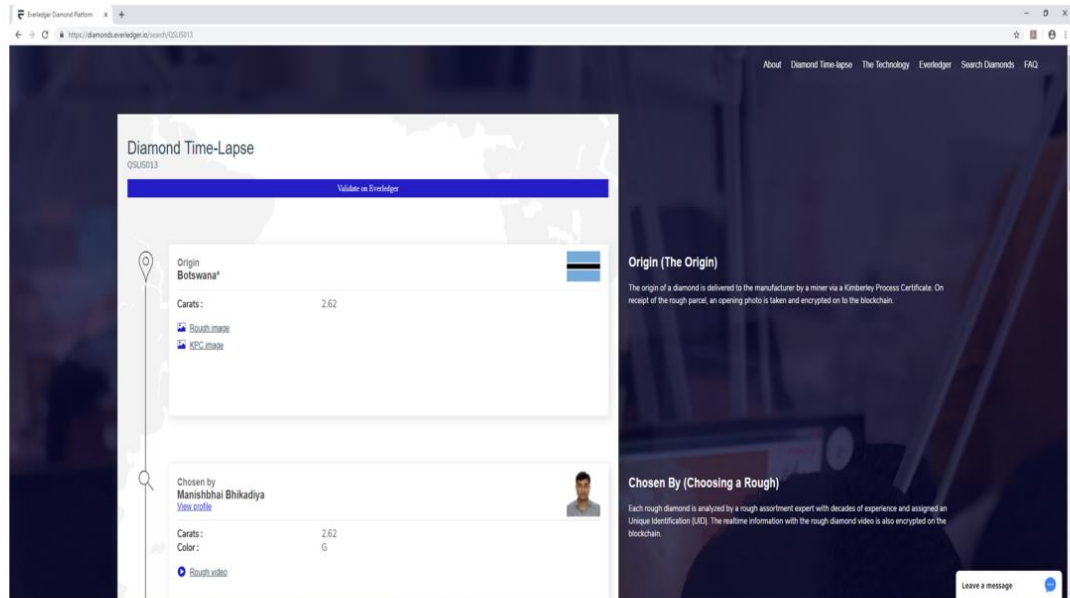


Fig. 2. The sample report on the origin, and people involved with the diamond under the “Diamond Time-Lapse” program by the BTS platform Everledger.

Figure 2.14: The sample report of the origin and people involved in diamond retailing

The paper compares three different business models using game-theoretic and mathematical analysis. The first one is Model R, which is the Traditional Jewellery Retail that uses no blockchain. These products are sold through retail stores. The consequences of this are that the retailers can provide shopping convenience, but they struggle to give the proper authenticity when the consumers ask for it. The second method is through the Blockchain-Based Selling Platform, which is called the PL model. In this method, the manufacturer sells directly to customers through a BTS (Blockchain Technology Supported) platform. During this process, the sellers can ensure full product authenticity and traceability. Although there are no physical stores involved, which means less convenience but higher trust. The last model is known as Model BCR, which is the Hybrid Model that includes both blockchain and retail. Although this method combines the traditional retailers with blockchain certification, and the customers enjoy both shopping convenience and authenticity, this model adds more intermediaries, which may reduce efficiency.

The paper results show that the BTS selling platform is better than traditional retail when customers value authenticity more than convenience. The hybrid Model BCR can

outperform both traditional and blockchain-only models if the convenience value is moderate. This shows that in real business, the implementation of blockchain can be beneficial to secure the product authenticity, together with increasing the profit of the business.

2.2 Limitations of Previous Studies

The limitations of each study will be compared in the table below. Each numbering represents a different subsection, and the tick represents the current available features in each study while the blank space corresponds to each limitation or improvement that can be made

2.1.1 Limitations of product authenticity verification using blockchain and OTP

This system primarily focuses on enhancing product authenticity using OTP verification at various stages in the supply chain, but it has certain limitations:

- **Manual Process for OTP Verification:** The requirement for users to manually log into a website and receive OTP codes for every stage or checkpoint adds friction to the verification process. This may not be user-friendly, especially for consumers or smaller suppliers, leading to inefficiencies in supply chain operations.
- **Dependence on Human Checking:** The need for Quality Control Officers (QCOs) to perform spot checks introduces a potential for human error or corruption. This reliance on manual intervention undermines the automation and transparency that blockchain promises.
- **Lack of Scalability:** While OTP provides a secure means of verifying authenticity, the scaling of this method for larger supply chains may be very difficult, especially when managing thousands of OTP verifications in real-time.

2.1.2 Limitations of blockchain-based traceability systems for agricultural products

The system in this study has a strong focus on ensuring transparency and traceability in agriculture, but there are some key limitations:

- **Limited Focus on Privacy Protection:** Although the system uses encryption to secure private information, it lacks secure and comprehensive privacy features for consumers. Sensitive data may still be vulnerable to unauthorised access, accidentally, especially if not adequately encrypted at every stage of the process.
- **Dependence on AES and ECC for Security:** While the system uses advanced encryption techniques like AES and ECC, it doesn't mention the ability to dynamically update encryption algorithms or key management strategies, which could be critical since the breaching tools are improving fast nowadays.
- **Challenges in Data Processing:** The system separates data into multiple layers (storage, service, interface, and application). However, the complexity of handling real-time data at scale could lead to performance bottlenecks, especially in large-scale agricultural operations where continuous data streams are common and larger costs might be charged to ensure the operating smoothness.

2.1.3 Limitations of blockchain-based solution for medication anti-counterfeiting and traceability

The system designed for the pharmaceutical supply chain has significant strengths but also faces several limitations:

- **Complexity of PBFT Consensus:** The PBFT consensus mechanism is suitable for small networks but can become inefficient in larger networks due to its high communication overhead. This limits scalability when applied to global pharmaceutical supply chains involving thousands of nodes.
- **RBAC Model Dependency:** While the Role-Based Access Control (RBAC) model ensures strict data access control, it may not be flexible enough for

dynamic environments where user roles change frequently, making the system less reliable and exposed to danger in the outside world.

- **No Mention of Consumer Feedback Integration:** Unlike some other studies, this system does not integrate consumer feedback into the traceability or authentication process, which limits the extent of verification beyond the supply chain. This makes the consumer's comment not accessible to the development and technology team

2.1.4 Limitations of product authentication technology integrating blockchain and traceability structure

This system integrates multiple layers of security and traceability, but several limitations exist:

- **Reliance on Ethereum's Proof of Work (PoW):** The use of Ethereum's PoW consensus mechanism can be resource-intensive, resulting in slower transaction times and higher energy consumption, especially in comparison to more efficient mechanisms like Proof of Stake (PoS) or PBFT.
- **Limited Focus on Product Feedback:** While this study introduces a FastText model for consumer feedback analysis, the method relies on reaching a predefined threshold for feedback collection, potentially delaying product certification. This may slow down the certification process for products with fewer consumer reviews.
- **IPFS Vulnerability to Data Availability Issues:** The use of IPFS provides a decentralised way to store large data files, but it introduces challenges related to data availability and retrieval. Since IPFS nodes may not always be available, this could impact the reliability of the traceability information.

2.1.5 Limitations of blockchain-based platforms for diamond authentication and certification in luxury supply chains

This system integrates multiple layers of security and traceability, but several limitations exist:

- **Lack of empirical validation:** In the research paper, this study is purely analytical and theoretical, using stylish models. There is no empirical data or real-world case studies to validate the proposed models or assumptions, which limits the practical applicability of the findings.
- **Assumption of homogeneous retailers and consumers:** The models assume uniform distribution of consumer preferences and homogeneous retailers, which may not reflect the diversity in real-world consumer behaviour, retailer capabilities, and market dynamics. While the paper introduces the BDAC (Blockchain-technology-based Diamond Authentication and Certification) cost, it treats it as a static value without considering potential variability over time or between regions, technologies, or platforms.
- **Exclusion of Competitive Dynamics:** The paper assumes monopolistic or controlled settings for different models, such as the BTS platform vs traditional retail, but does not consider competition among existing retailers or platforms, which could significantly influence outcomes.

2.2.6 Summary

2.1.1: Product authenticity verification using blockchain and OTP

2.1.2: Blockchain-based traceability system for agricultural products

2.1.3: Blockchain-based solution for medication on anti-counterfeiting and traceability

2.1.4: Product authentication technology integrating blockchain and traceability structure

CHAPTER 2

2.1.5: Blockchain-based platforms for diamond authentication and certification in luxury supply chains

| Limitations of Previous study | 2.1.1 | 2.1.2 | 2.1.3 | 2.1.4 | 2.1.5 | Proposed System |
|---|-------|-------|-------|-------|-------|-----------------|
| Make use of a unique code | | ✓ | | ✓ | | ✓ |
| Privacy protection | | ✓ | ✓ | ✓ | ✓ | ✓ |
| Traceability of supply chain | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| Public and private data encryption | | ✓ | ✓ | ✓ | ✓ | ✓ |
| Use of smart contracts | | ✓ | ✓ | ✓ | ✓ | ✓ |
| Product certification | | | | ✓ | ✓ | ✓ |
| Trace product from manufacturer to consumer | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| Use of Elliptic Curve Cryptography (ECC) | | ✓ | | | | ✓ |

Table 2.1: The summary of limitations in each study

2.3 Proposed Solutions

Product authentication and counterfeit prevention challenges are significant, especially in industries such as healthcare, food manufacturing, and luxury goods, where counterfeit products can pose serious risks to consumers. Counterfeit products harm businesses financially and compromise public health, safety, and trust in the real, genuine manufacturer. To address these challenges, this project aims to propose a solution that integrates blockchain technology to enhance the security, reliability, and transparency of product authentication systems. By leveraging blockchain, the system will ensure that the entire supply chain is decentralised and secure, and it can provide a robust mechanism for verifying product authenticity from the manufacturer to the end consumer.

One of the key features of the proposed system is blockchain-based traceability. This allows stakeholders across the supply chain to trace a product's entire lifecycle, ensuring the process's integrity at every stage. By using blockchain's decentralised nature, data stored about each product becomes immutable, making it impossible to alter or manipulate the information. This feature will be particularly effective in preventing counterfeit products from leaking into the market, as each product's history can be verified and validated transparently. This level of traceability ensures that both businesses and consumers have confidence in the authenticity of the products they handle or purchase, since they can accurately tell the source of origin of the particular product.

The system will also implement smart contracts, which play a crucial role in automating and enforcing agreements between parties. Smart contracts eliminate the need for intermediaries, reducing human error and increasing transparency in business transactions. This automation ensures that agreements related to product handling, authentication, and verification are executed reliably, with less room for errors and mistakes. Smart contracts ensure that all parties follow the rules that exist, further strengthening the integrity of the supply chain.

To protect sensitive information and safeguard privacy, the proposed solution will implement both public and private data encryption methods. Public encryption will be used for data that needs to be accessible to all stakeholders, such as product details and history, while private encryption will be employed to protect sensitive information that

should remain confidential, such as transaction details. This dual encryption approach ensures that privacy is maintained while still allowing the necessary traceability of products within the system.

Additionally, the system will incorporate with the Metamask wallet. The user is required to log in to the system using their account. By securing the account information in system using Metamask, the risk of data manipulation or leaking is reduced, as no single party or individual can control or alter the data once it is entered into the blockchain. This distributed approach also makes the data more accessible to stakeholders, ensuring transparency while enhancing security.

To further enhance the security of the system, Elliptic Curve Cryptography (ECC) will be employed for data encryption. As discussed in [9], ECC is a highly secure encryption method that provides strong cryptographic protection with minimal computational overhead. This makes ECC ideal for systems because both efficiency and security are reliable and ensured. By implementing ECC, the proposed system will ensure that encryption is both lightweight and robust, balancing security with system performance. ECC is beneficial and has the advantage due to its smaller key size and faster operations, as it requires fewer resources for encryption and decryption, leading to quicker execution times—an essential feature for real-time Iot applications and mobile devices. Additionally, ECC offers superior energy efficiency and scalability, making it well-suited for protecting against unauthorised access and facilitating user authentication or verification of products.

Another key feature of the solution is the integration one-time unique code generation technology. The authentication will be used to verify the identities of stakeholders and consumers, ensuring that only authorised parties can access or modify product data. On the other hand, the unique code on the product can allow consumers to easily retrieve information about the products they purchase. By providing the unique code for consumers, consumers can verify the authenticity of a product, view its history, and ensure that it has not been modified or changed from the start of manufacturing until the delivery phase. Moreover, since the code can only be used once, the replication of codes to fool the public will not work. This integration of the unique code simplifies the verification process, making it accessible to consumers and businesses alike.

CHAPTER 2

In short, the proposed solution addresses the challenges of product authentication and counterfeit prevention by integrating blockchain technology, smart contracts, encryption methods, and unique code verification. These technologies together create a decentralised, transparent, and secure supply chain that allows for the authentication of products at every stage. By making it harder for counterfeit products to enter the market, the system enhances public trust, protects businesses, and ensures the safety of consumers.

CHAPTER 3 SYSTEM METHODOLOGY/APPROACH

3.1 System Requirements

3.1.1 Hardware

The hardware used in this project plays a critical role in setting up and maintaining the blockchain infrastructure, ensuring secure data handling, and enabling user interaction through mobile devices. Additionally, IoT hardware is integrated for real-time monitoring, providing a seamless interface with the blockchain system.

- **Computer:** The desktop serves as the backbone for the blockchain node setup, smart contract deployment, and overall system testing. It ensures that the blockchain ledger operates efficiently, records product data immutably, and processes smart contract transactions.
- **Mobile Device:** The Redmi K60 is crucial for developing and testing the unique code generation. Consumers interact with the app on this device to see the unique code generated, retrieve product data, and verify product authenticity securely in real time.

| Description | Specifications |
|------------------|-------------------------------|
| Monitor | AOC 27G4 |
| Processor | Ryzen 5 3600 |
| Operating System | Windows 11 |
| Graphic | NVIDIA GeForce GTX 1660 Super |
| Memory | 16GB DDR4 RAM |
| Storage | 1TB SSD Kingston |

Table 3.1: Specifications of Desktop

| Description | Specifications |
|-------------|-----------------------|
| Model Name | Redmi K60 |
| Processor | Snapdragon 8gen1 plus |
| RAM | 16GB |
| Battery | 5500mAh |
| Storage | 256GB |
| Version | Android 14, HyperOS |

Table 3.2: Specifications of Smartphone

3.1.2 Software

The software used in this project consists of several components, each serving specific functions, such as blockchain setup, smart contract deployment, encryption, distributed storage, and mobile application development.

| Software Component | Description | Purpose/Usage |
|----------------------------------|---|--|
| Blockchain Framework | Ethereum / Hyperledger Fabric | To create and maintain the decentralised blockchain ledger for product traceability. |
| Smart Contract Development Tools | Solidity / Truffle | Used for writing and deploying smart contracts that automate agreements and ensure compliance. |
| Encryption Libraries | OpenSSL / Java Cryptography Extension (JCE) | To implement Elliptic Curve Cryptography (ECC) for secure data encryption and decryption processes. |
| Metamask | Online blockchain | Provides the functionality for the user to log in to the blockchain account on the browser |
| Backend Integration Tools | Node.js | To integrate unique code generation functionalities, and communication between the blockchain and the app. |
| Front end | React | To let the user and manufacturer view and modify the product using the blockchain smart contract. |

Table 3.3: Software Components and Corresponding Purpose

3.2 Project Costing

The estimated costing for the blockchain-based product authenticity verification system is divided into several key categories, which are mainly hardware, software, cloud services (optional for deployment), and other costs (such as testing and contingency expenses). The project mainly utilises open-source tools to minimise expenses, but certain costs may still be incurred for a complete deployment in real-world applications.

3.2.1 Hardware Costs

| Hardware | Description | Estimated Cost (RM) |
|------------------------|---|---------------------|
| Desktop Computer | Already available, used for development and testing | RM 0 |
| Smartphone | Redmi K60 (for unique code verification testing) | RM 0 |
| IoT Devices (optional) | NFC Tags / Sensors (if expanded for the future) | RM 100–200 |

Table 3.4: Hardware Costs for the Project

Subtotal (Hardware): RM 0 (already available). If IoT integration is needed, the cost might add RM 100 to RM 200 in the future.

3.2.2 Software Costs

| Software | Description | Estimated Cost (RM) |
|---|---|---------------------|
| React, Node.js, Truffle, Ganache | Open-source development frameworks | RM 0 |
| Visual Studio Code | Open-source IDE for coding | RM 0 |
| IPFS | Free to use for decentralised storage (local setup) | RM 0 |
| SSL Certificate (optional for production) | If hosted online for real-world use | RM 150–250/year |

Table 3.5: Software Costs for the Project

Subtotal (Software): RM 0. If the SSL certificate is needed in the future, the estimated cost will increase by RM 150 to RM 250.

3.2.3 Cloud Hosting Costs (Optional for Public Deployment)

| Cloud Service | Description | Estimated Cost (RM) |
|----------------------|---|---------------------|
| AWS EC2 Instance | Hosting blockchain nodes/web app for 1 year | RM 200–400 |
| IPFS Hosting Service | Optional for better IPFS uptime | RM 50–100 |

Table 3.6: Cloud Hosting Costs for the Project

Subtotal (Cloud Hosting): Around RM 200 to RM 500. The cloud hosting is optional and can only be used if we decide to test the project online and make it publicly available.

3.2.4 Other Costs

| Others | Description | Estimated Cost (RM) |
|-------------------------|--|---------------------|
| Testing and Maintenance | Cost for data plans, additional devices, backups | RM 50–100 |
| Contingency | Unexpected expenses | RM 100 |

Table 3.7: Other Costs for the Project

Subtotal of the others' cost is around RM 150 to RM 200, leaving some money for some unexpected cases.

3.2.5 Summary of Cost Estimation

| Category | Estimated Cost (RM) |
|--------------------------|--------------------------|
| Hardware | RM 0 (already available) |
| Software | RM 0 |
| Cloud Hosting (optional) | RM 200–500 |
| Miscellaneous | RM 150–200 |

Table 3.8: Summary Costs for the Project

The Total Estimated Cost (minimum) for local testing is around **RM 150 to RM 200**, but if with online deployment, the cost will be increased to around **RM 350–700** or more.

3.3 System Architecture Diagram and Explanation

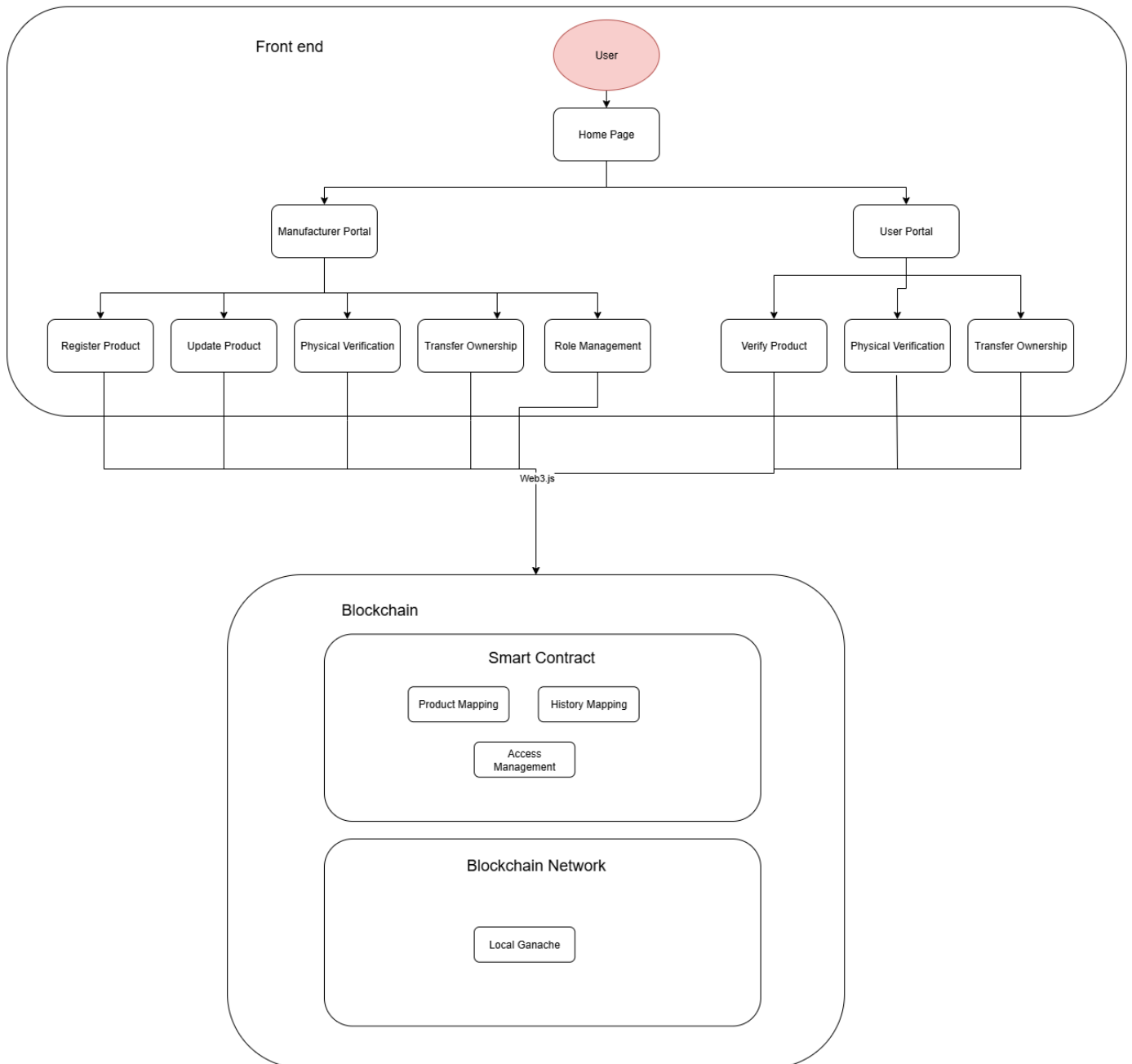


Figure 3.1: The system architecture diagram of the project

The system architecture for the blockchain-based product verification platform is built based on a React-powered frontend for user interaction and a blockchain backend that ensures secure, decentralised data management. The frontend provides two main access points through the Manufacturer Portal and the User Portal. Within the Manufacturer

Portal, authorised users can perform several key functions, including registering new products, updating product details, conducting physical verification of items, transferring product ownership, and managing roles to control access within the system.

The User Portal, meanwhile, is designed for consumers and includes functions such as verifying product authenticity, performing physical verification, and initiating or viewing product ownership transfers. Together, these functions enable both manufacturers and consumers to interact with the system transparently and securely, while maintaining a simple and intuitive interface.

Both portals communicate with the blockchain through Web3.js, a JavaScript library that acts as the middleware between the React interface and the underlying blockchain network. When an action such as product registration, verification, or ownership transfer is initiated, Web3.js forwards the corresponding transaction request to the smart contract deployed on the blockchain. The smart contract is central to the system and incorporates three key modules: Product Mapping, which stores core product information such as identifiers, current ownership, and status; History Mapping, which maintains a complete record of all past updates and ownership transfers; and Access Management, which enforces role-based permissions to ensure that only authorised parties can perform sensitive operations like registration and updates.

The blockchain network, powered by Ganache in this prototype, processes and permanently records each transaction. Although the Ganache is only available in the local testing environment, this ensures that every operation is transparent, immutable, and traceable throughout the product's lifecycle. For consumers, verification is carried out by submitting a product identifier through the User Portal, which triggers a query to the blockchain via Web3.js. The smart contract then retrieves the authenticity details and ownership history in real time, allowing users to confirm the product's legitimacy with confidence.

Overall, this architecture integrates React's user-friendly interface with the security and transparency of blockchain technology. By combining decentralised storage, tamper-proof records, and strict role-based access control, the system provides an efficient and reliable solution for product authenticity verification. It not only strengthens consumer trust but also offers businesses a robust framework to safeguard brand integrity and reduce the risks associated with counterfeit goods.

3.4 Use Case Diagram and Explanation

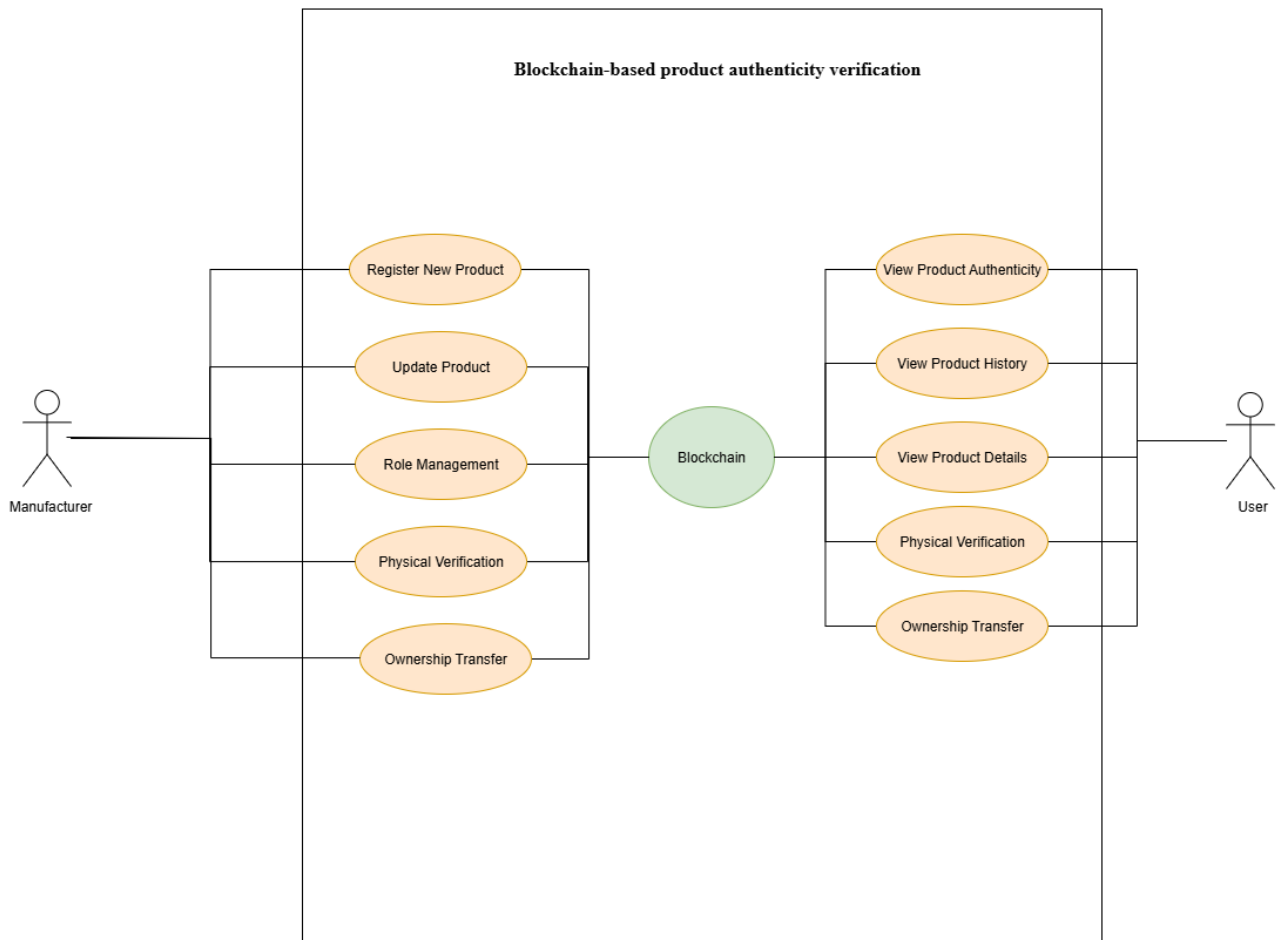


Figure 3.2: User Case Diagram

The activity diagram for the blockchain-based product authenticity verification system illustrates the flow of activities between the Manufacturer, the User, and the blockchain network. It captures both the sequential and conditional steps required to complete the main use cases, reflecting how the system operates in practice.

The process begins with the **Manufacturer** logging into the Manufacturer Portal. From here, the manufacturer can initiate several actions, such as registering a new product, updating product details, performing physical verification, transferring ownership, or managing user roles. For example, when registering a product, the manufacturer enters product details including the serial number and submits them through the portal. This request is processed by the Web3.js interface and recorded on the blockchain via the smart contract, which updates the **Product Mapping** and logs the transaction in the

History Mapping. Similarly, updates to product details or ownership transfers follow the same pathway, ensuring that all modifications are permanently stored and auditable. The **Role Management** function ensures that only authorised manufacturers or registrars can execute these write operations.

On the **User** side, the activity diagram shows how consumers interact with the system through the User Portal. A user may verify product authenticity, view product details, check historical records, perform physical verification, or initiate an ownership transfer. When verifying authenticity, the user provides a product identifier, which triggers a query to the blockchain. The smart contract retrieves real-time information from the Product and History mappings, returning authenticity status, details, and transaction history to the user. This ensures transparency and allows the consumer to confirm that the product is genuine before purchase or use.

The diagram also highlights the **role-based access control** integrated into the system. Manufacturers have written permissions, enabling them to add, update, or manage product records, while users are limited to read-only operations and ownership-related actions. Every activity eventually leads to an interaction with the blockchain network (Ganache), which processes and immutably stores the results.

Overall, the activity diagram demonstrates how each actor interacts with the system in a step-by-step manner. It reflects the integration of blockchain features—immutability, transparency, and decentralisation into the routine operations of registering, verifying, updating, and transferring products. By visualising these workflows, the diagram clarifies how the system enforces security, maintains an auditable record, and builds trust between manufacturers and consumers.

3.5 Project Timeline

| Phase/ Duration (FYP1) | Task Description |
|------------------------|--|
| Week 1–2 | Project Topic Confirmation & Supervisor Assignment |
| Week 2–3 | Initial Research: Blockchain basics, product authentication needs, study similar systems |
| Week 3–5 | Literature Review: Summarise existing solutions, find gaps |
| Week 6–7 | Finalise Project Scope, Objectives, Contributions |
| Week 7–8 | System Design (Architecture Diagram, Use Case Diagram) |
| Week 8–9 | Select Tools (React, Solidity, Truffle, Ganache) |
| Week 9–10 | Start Coding: Solidity Smart Contracts (Product registration, update, verify) |
| Week 10–11 | Set up Ganache, Truffle, Deploy initial contracts |
| Week 11–12 | Develop Basic Frontend UI (React: Manufacturer & User Portal) and system integration |
| Week 12–13 | Prepare Proposal Report + Submit FYP I Proposal (with system design, partial prototype) |
| Week 13–14 | Prepare Presentation Slides |

Table 3.9: Timeline for FYP 1

| Phase / Duration FYP II (Next Semester) | Task Description |
|--|--|
| Week 1–2 | Revise based on FYP I feedback (if needed) |
| Week 2–4 | Continuing Full System Development: Enhance React frontend |
| Week 4–6 | Add Encryption (ECC implementation) and IPFS Integration |

| | |
|------------|---|
| Week 6–8 | Testing Smart Contract functions (register, update, verify products) |
| Week 8–10 | Security Testing: Attack simulations (e.g., invalid QR codes, fake updates) |
| Week 10–11 | User Acceptance Testing (ask friends/family to test the system) |
| Week 11–12 | System Refinement (UI/UX improvements, bug fixes) |
| Week 12–13 | Final Report Writing: Introduction, Background, Literature Review, Methodology, Results, Conclusion and prepare Presentation Slides |
| Week 13–14 | Submit Final Report and Demonstrate System (Final Presentation) |

Table 3.10: Timeline for FYP 2

CHAPTER 4 SYSTEM DESIGN

4.1 Setting Up

Before starting to develop the system, multiple software packages need to be installed and downloaded on my laptop:

1. React version 18.3.1 deduped
2. Microsoft Visual Studio Code version 1.99.3
3. Truffle v5.11.5 (core: 5.11.5)
4. Ganache v7.9.1
5. Solidity — 0.8.13 (solc-js)
6. Node v20.16.0
7. Web3.js v1.10.0

4.2 System Design

4.2.1 Truffle

Truffle needs to be installed on the computer using the command prompt in the directory of our files by using the command **npm install -g truffle** at the command prompt.

```

The system cannot find message text for message number 0x2350 in the message file for Application.

(c) Microsoft Corporation. All rights reserved.

Z:\Desktop\Qirui\yeu\utar\degree year 3\product-verification-structure>npm install -g truffle
npm warn deprecated testrpc@0.0.1: testrpc has been renamed to ganache-cli, please use this package from now on.
npm warn deprecated inflight@1.0.6: This module is not supported, and leaks memory. Do not use it. Check out lru-cache if
you want a good and tested way to coalesce async requests by a key value, which is much more comprehensive and powerfu
l.
npm warn deprecated @truffle/source-map-utils@1.3.119: Package no longer supported. Contact Support at https://www.npmjs
.com/support for more info.
npm warn deprecated mkdirp-promise@5.0.1: This package is broken and no longer maintained. 'mkdirp' itself supports prom
ises now, please switch to that.
npm warn deprecated rimraf@2.7.1: Rimraf versions prior to v4 are no longer supported
npm warn deprecated level-concat-iterator@3.1.0: Superseded by abstract-level (https://github.com/Level/community#faq)
npm warn deprecated @truffle/db-loader@0.2.36: Package no longer supported. Contact Support at https://www.npmjs.com/sup
port for more info.
npm warn deprecated @truffle/promise-tracker@0.1.7: Package no longer supported. Contact Support at https://www.npmjs.co
m/support for more info.
npm warn deprecated har-validator@5.1.5: this library is no longer supported
npm warn deprecated @truffle/error@0.2.2: Package no longer supported. Contact Support at https://www.npmjs.com/support
for more info.
npm warn deprecated glob@7.2.0: Glob versions prior to v9 are no longer supported
npm warn deprecated memdown@1.4.1: Superseded by memory-level (https://github.com/Level/community#faq)
npm warn deprecated apollo-datasource@3.3.2: The 'apollo-datasource' package is part of Apollo Server v2 and v3, which a
re now end-of-life (as of October 22nd 2023 and October 22nd 2024, respectively). See https://www.apollographql.com/docs
/apollo-server/previous-versions/ for more details.
npm warn deprecated apollo-server-errors@3.3.1: The 'apollo-server-errors' package is part of Apollo Server v2 and v3, w
hich are now end-of-life (as of October 22nd 2023 and October 22nd 2024, respectively). This package's functionality is
now found in the 'apollo-server' package. See https://www.apollographql.com/docs/apollo-server/previous-versions/ for m

```

Figure 4.1: The screenshot of the truffle installation

```

Z:\Desktop\QiRui\yeu\utar\degree year 3\fyp\product-verification-structure>truffle compile

Compiling your contracts...
=====
> Compiling .\contracts\Migrations.sol
> Compiling .\contracts\ProductVerification.sol
> Compilation warnings encountered:

    Warning: SPDX license identifier not provided in source file. Before publishing, consider adding a comment containin
g "SPDX-License-Identifier: <SPDX-License>" to each source file. Use "SPDX-License-Identifier: UNLICENSED" for non-open-
source code. Please see https://spdx.org for more information.
--> project:/contracts/Migrations.sol

> Artifacts written to Z:\Desktop\QiRui\yeu\utar\degree year 3\fyp\product-verification-structure\build\contracts
> Compiled successfully using:
   - solc: 0.8.13+commit.abaa5c0e.Emscripten.clang

```

Figure 4.2: The screenshot of the compilation of contracts

The truffle is essentially a modular and extensible architecture, designed to simplify blockchain development. Its system design consists of the following key layers that can help to manage project configurations, compilations, and deployments. It can act as a smart contract compiler and integrate with Solidity and Vyper to compile contracts into bytecode. Managing the contract deployments across different networks allows interaction with contracts via Truffle Console and custom scripts. To start using this, we just need to type **truffle init** as a command in the root directory of the project to start creating the necessary file structure that includes folders such as contracts, migrations, tests, and truffle-config.js that can help us to manage and edit the smart contracts deployed. After that, the contracts can be deployed in the created folders, and after successfully creating the contracts, the command **truffle compile** is needed to check for any errors that occur when compiling. After that, the command **truffle migrate** is used to migrate the contracts to the blockchain network to test the smart contract.

CHAPTER 4

4.2.2 Solidity

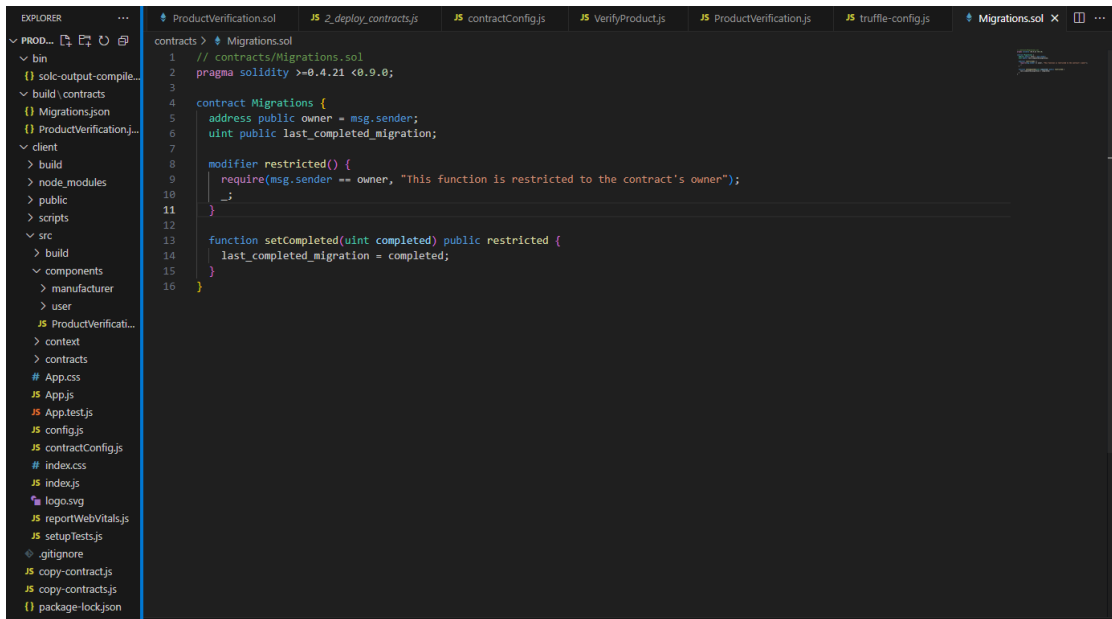


Figure 4.3: The screenshot of the written contract using Solidity

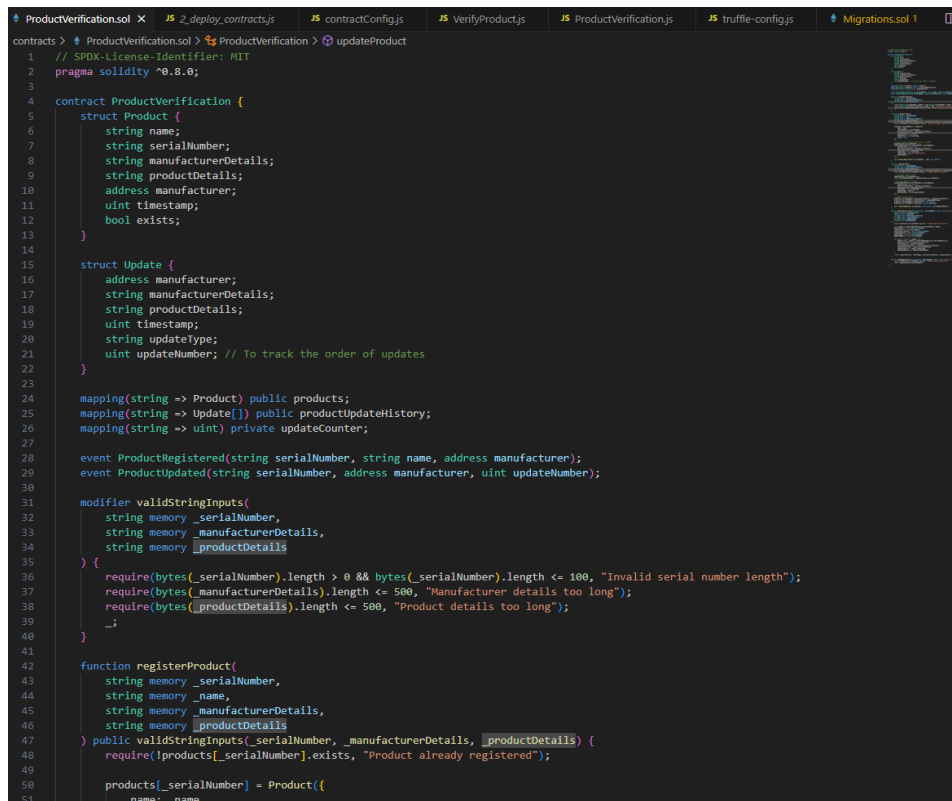


Figure 4.4: The solidity code to record the product and save it into the blockchain

Solidity is a high-level, contract-oriented programming language designed to build secure, tamper-proof applications on blockchain platforms like Ethereum. Its syntax, inspired by JavaScript and C++, lets developers define complex business logic in a familiar format while leveraging the inherent immutability and decentralization of the blockchain. In this project, we used Solidity to implement a Product Verification smart contract that records every stage of a product's lifecycle—registration, updates, and history retrieval—directly on-chain.

The core of the solution is the ProductVerification contract, which begins by allowing manufacturers to register new products using a unique serial number, descriptive name, and detailed manufacturer information. As soon as a product is registered, the contract emits a ProductRegistered event, making it easy for users to check for the new items or products registered every time. Each product record is stored in a mapping from serial numbers to a Product struct, making sure that lookups are efficient, and that data cannot be overwritten or changed by unauthorised parties.

After registration, manufacturers who own a product record can submit updates, such as warranty extensions, quality checks, or shipping status changes, through a dedicated function. Each update is captured in its Update struct, which includes the manufacturer's address, the new details, a timestamp, and a sequential update number. These updates are then added to a separate mapping that maintains the full update history for each product for the user to view. To guard against the danger of hackers, a validStringInputs modifier that creates a length limit on serial numbers, manufacturer details, and product descriptions is employed to reduce the risk of storage abuses.

Finally, users and auditors can retrieve a product's complete history with its original registration and every subsequent update in a single call, providing end-to-end traceability and proof of authenticity. By storing all critical data on the blockchain, this system eliminates the possibility of backdated edits or hidden modifications, helping to combat counterfeiting and build trust between manufacturers, retailers, and consumers. In short, the Solidity smart contract transforms the product verification process into an open, automated, and verifiable workflow, giving every stakeholder confidence in the integrity of the supply chain.

4.2.3 Ganache

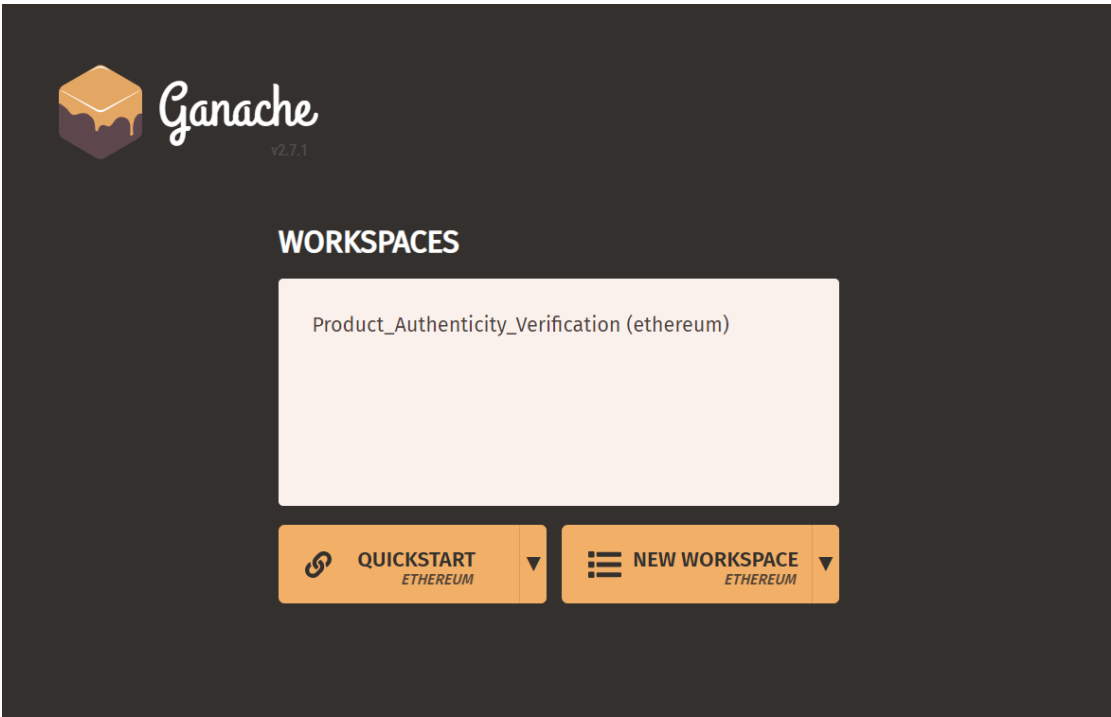


Figure 4.5: The ganache workspace

| | | | | | | | | | |
|---|--------------------------|----------------------|-------------------|--------------------|-------------------------------------|---------------------------------------|--|--------|---|
| Ganache | | | | | | | | | |
| ACCOUNTS | BLOCKS | TRANSACTIONS | CONTRACTS | EVENTS | LOGS | SEARCH FOR BLOCK NUMBERS OR TX HASHES | | | |
| CURRENT BLOCK 473 | GAS PRICE 20000000000 | GAS LIMIT 6721975 | HARDFORK MERGE | NETWORK ID 5777 | RPC SERVER HTTP://127.0.0.1:7545 | MINING STATUS AUTOMINING | WORKSPACE PRODUCT_AUTHENTICITY_VERIFICATION | SWITCH | ⚙ |
| MNEMONIC ⓘ ivory kid decade appear blush load exist large gossip yard explain lava | | | | | HD PATH m44'60'0'0'account_index | | | | |
| ADDRESS 0x92AD65AB5d491ffaFCA7B6A995256B9Ea1182A57 | BALANCE 98.39 ETH | TX COUNT 473 | INDEX 0 | 🔑 | | | | | |
| ADDRESS 0x9db467F99D08E173A6725D4041617E0Fbb0b549a | BALANCE 100.00 ETH | TX COUNT 0 | INDEX 1 | 🔑 | | | | | |
| ADDRESS 0xEC829Fc4FF2121909dE6B0738D617Dc90672F977 | BALANCE 100.00 ETH | TX COUNT 0 | INDEX 2 | 🔑 | | | | | |
| ADDRESS 0x230C71C5e24f4F4eA548d2a01d53987Fe24765f0 | BALANCE 100.00 ETH | TX COUNT 0 | INDEX 3 | 🔑 | | | | | |
| ADDRESS 0xAD1FF668Cc4eA6f5516aB692f8BE2DDaCaBCbAf0 | BALANCE 100.00 ETH | TX COUNT 0 | INDEX 4 | 🔑 | | | | | |
| ADDRESS 0x00dEDd7e534bc561CFb28B33A87EBfB5e320590f | BALANCE 100.00 ETH | TX COUNT 0 | INDEX 5 | 🔑 | | | | | |
| ADDRESS 0x6c36F436D695F420Cdc4F9970EDc3acD25354f95 | BALANCE 100.00 ETH | TX COUNT 0 | INDEX 6 | 🔑 | | | | | |

Figure 4.6: The accounts that existed in the local ganache

ACCOUNTS

BLOCKS

TRANSACTIONS

CONTRACTS

EVENTS

LOGS

SEARCH FOR BLOCK NUMBERS OR TX HASHES

CURRENT BLOCK
473

GAS PRICE
20000000000

GAS LIMIT
6721975

HARDFORK
MERGE

NETWORK ID
5777

RPC SERVER
HTTP://127.0.0.1:7545

MINING STATUS
AUTOMINING

WORKSPACE
PRODUCT_AUTHENTICITY_VERIFICATION

SWITCH

TX HASH

0xbc1add9c482795fc10ef8c13936af4f712aa00d25b9f983ce92f59e7c4a11e83

CONTRACT CALL

FROM ADDRESS

0x92AD65AB5d491ffAFCA7B6A995256B9Ea1182A57

TO CONTRACT ADDRESS

0xf388aeE62Eb228637c36C256d403504747cF9e41

GAS USED

25790

VALUE

0

TX HASH

0xca5732685d174d0428151dfc2118f6b9597907682bc2d8343f9f3f373feb1f6d

CONTRACT CREATION

FROM ADDRESS

0x92AD65AB5d491ffAFCA7B6A995256B9Ea1182A57

CREATED CONTRACT ADDRESS

0x87655Ea3996AeB1114cC96c5a4fA4c85E28e066e

GAS USED

740270

VALUE

0

TX HASH

0xddae54e61d633f1b7e0afeed002bcfc6ded054471d18799ad407c0aca3bc52c2

CONTRACT CALL

FROM ADDRESS

0x92AD65AB5d491ffAFCA7B6A995256B9Ea1182A57

TO CONTRACT ADDRESS

0xf388aeE62Eb228637c36C256d403504747cF9e41

GAS USED

25790

VALUE

0

TX HASH

0xbe3751b280e0e9248b91c0e6f786698d57796175d9ae3bc019238f4645ba2add

CONTRACT CREATION

FROM ADDRESS

0x92AD65AB5d491ffAFCA7B6A995256B9Ea1182A57

CREATED CONTRACT ADDRESS

0xE8494c99341D6DA09C912dACbFb85F3f90f44060

GAS USED

740270

VALUE

0

TX HASH

0x4ddcbe91e017c1135f231564b5d8a4ec5dab7f9d5303143cb5d1f7eacd0cb8dc

CONTRACT CALL

Figure 4.7: The transaction details after the smart contracts are deployed

Ganache is a personal, local Ethereum blockchain designed to facilitate rapid development and testing of smart contracts without the latency, cost, or security risks associated with deploying to a public test net. In Figure 4.5, we see the Ganache workspace titled `Product_Authenticity_Verification` (Ethereum). This workspace encapsulates a private blockchain instance running on the developer's machine, with its own chain ID (5777), RPC endpoint (HTTP://127.0.0.1:7545), and mining behaviour set to "automining" for instant transaction confirmation.

Figure 4.6 displays the Accounts tab within Ganache, listing ten automatically generated Ethereum addresses, each prefunded with 100 ETH except the first one as it is being used to test the project prototype. Developers can use any of these accounts to deploy contracts, send transactions, or test account-specific logic such as access controls in the `ProductVerification` contract.

Once contracts are deployed, Ganache records each interaction in its Transactions tab (Figure 4.7). Here, every transaction, whether a contract creation (tagged `CONTRACT CREATION`) or a contract method call (`CONTRACT CALL`), is logged with its hash, sender and recipient addresses, gas used, and value transferred.

CHAPTER 4

By providing real-time visibility into accounts, blocks, and transactions, Ganache allows developers to iterate quickly on smart contract logic, debug state changes, and validate event emissions (such as our `ProductRegistered` and `ProductUpdated` events) before moving to the public environments. The simplicity of the Ganache functionality allows users to reproduce test cases and share development setups with minimal configuration overhead.

4.2.4 Microsoft Visual Code (React)

```
+-- @emotion/react@11.14.0
| +-- @emotion/use-insertion-effect-with-fallbacks@1.2.0
| | '-- react@18.3.1 deduped
| '-- react@18.3.1 deduped
+-- @emotion/styled@11.14.0
| '-- react@18.3.1 deduped
+-- @mui/icons-material@5.17.1
| '-- react@18.3.1 deduped
+-- @mui/material@5.17.1
+-- @mui/system@5.17.1
| +-- @mui/private-theming@5.17.1
| | '-- react@18.3.1 deduped
| +-- @mui/styled-engine@5.16.14
| | '-- react@18.3.1 deduped
| '-- react@18.3.1 deduped
+-- @mui/utils@5.17.1
| '-- react@18.3.1 deduped
+-- react-transition-group@4.4.5
| '-- react@18.3.1 deduped
| '-- react@18.3.1 deduped
+-- @testing-library/react@13.4.0
| '-- react@18.3.1 deduped
+-- qrcode.react@3.2.0
| '-- react@18.3.1 deduped
+-- react-dom@18.3.1
| '-- react@18.3.1 deduped
+-- react-router-dom@7.5.2
```

Figure 4.8: Installing React using the command prompt

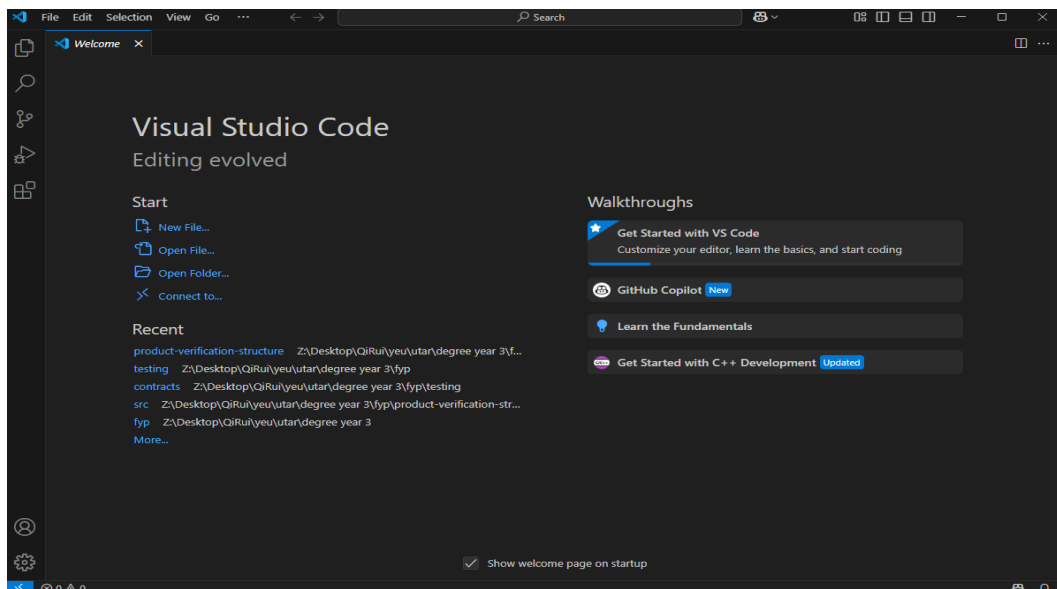


Figure 4.9: The Visual Studio Code used to debug and write code

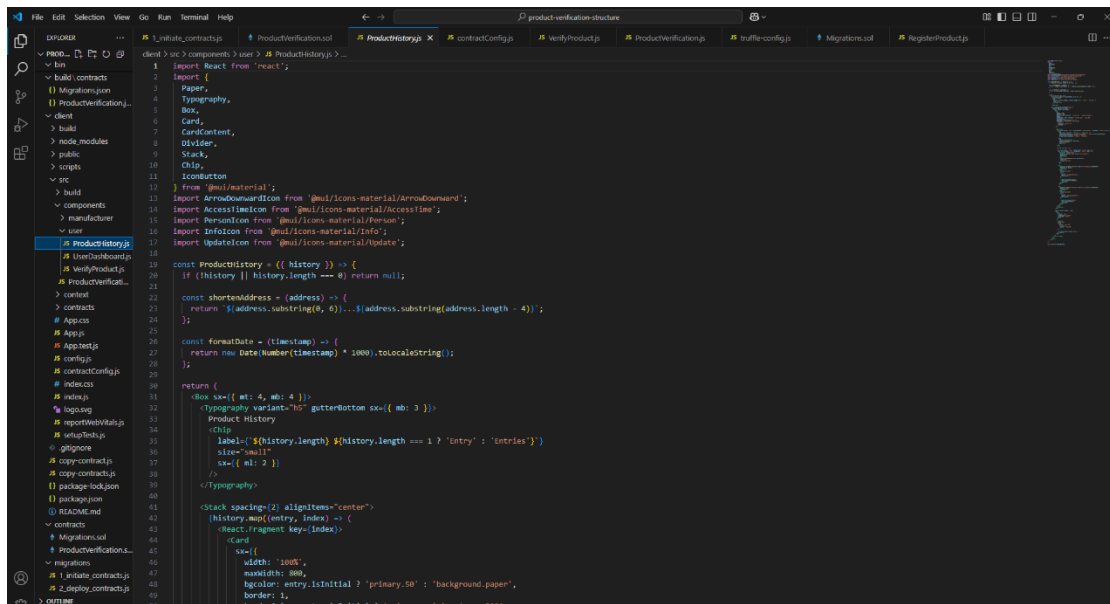


Figure 4.10: The user details to verify and view the previous history

In order to build the web-based user interface for our Product Verification system, React, a popular JavaScript library for building dynamic, component-driven applications and Visual Studio Code as our integrated development environment (IDE) are used. As shown in Figure 4.8, React and its dependencies are installed via the command line using `npx create-react-app`, `npm install react-router-dom`, and additional UI libraries such as Material UI (`@mui/material`, `@mui/icons-material`) to accelerate development with pre-built, theme components.

Visual Studio Code's rich ecosystem of extensions provides real-time syntax highlighting, linting feedback, and in-editor error reporting, making it straightforward to catch mistakes early and enforce consistent coding styles (Figure 4.9). Within the IDE, the code is organised into logical directories—components, scripts, and contracts that can keep smart-contract interactions (via `contractConfig.js` and `truffle-config.js`) separate from UI code.

Debugging is seamless in VS Code's built-in debugger can attach to the React development server, allowing breakpoints in JavaScript code, inspection of component state, and live reloading on file saves.

CHAPTER 5 SYSTEM IMPLEMENTATION

5.1 Settings and Configuration

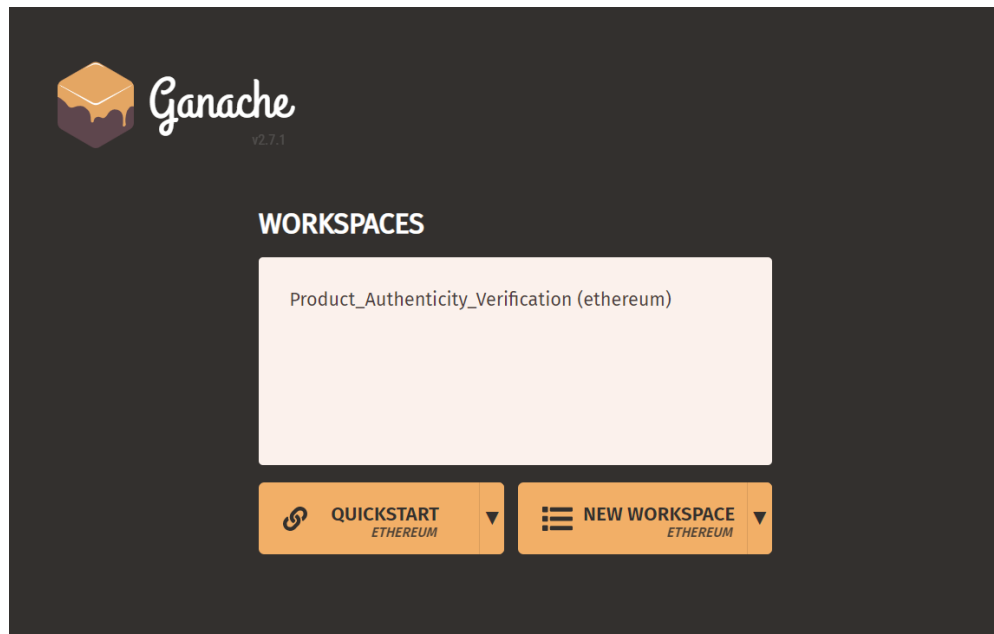


Figure 5.1: Ganache local workspace

In this chapter, the ganache application shown in the figure above is compulsory, and after it is downloaded and ready to be used, the new workspace should be filled with the title, and the application is required to be running in the background for the smart contracts to be deployed and used successfully.

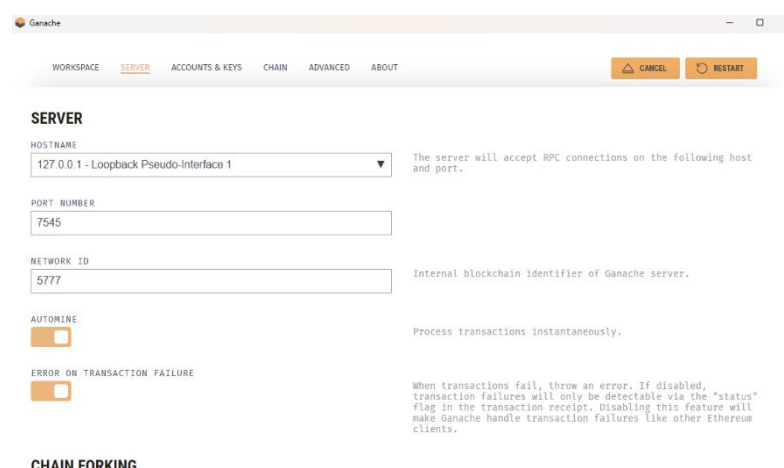
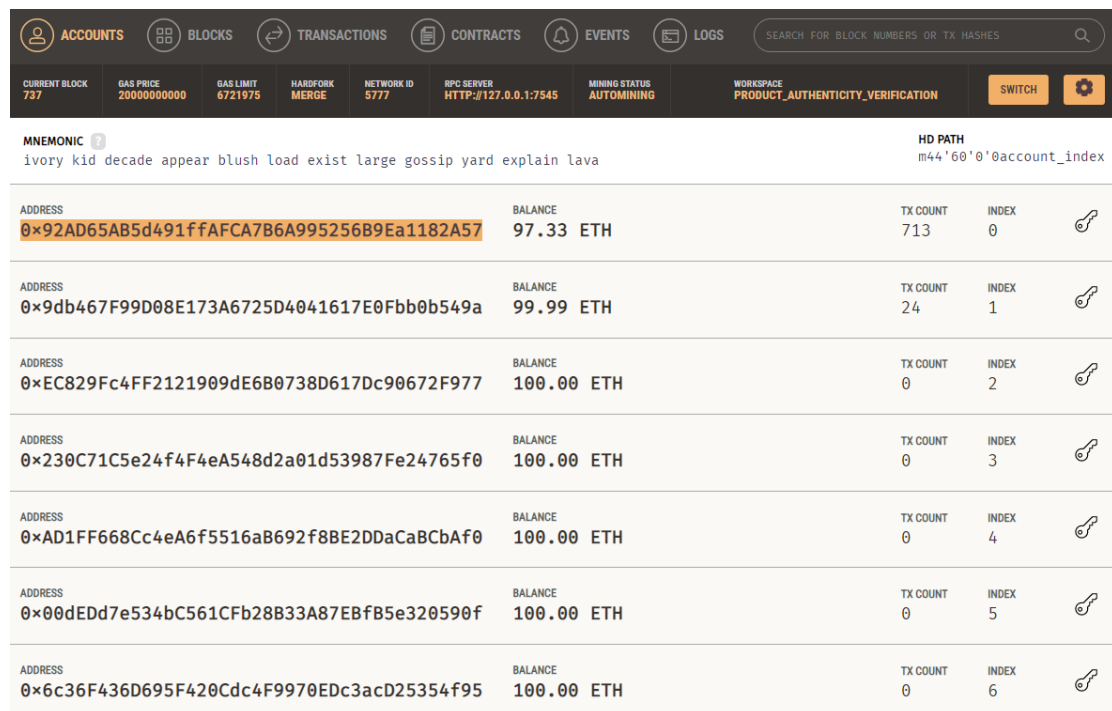


Figure 5.2: The ganache server details

CHAPTER 5

The details of the ganache are shown on the figure above, such as the hostname and port number, and the network ID, are configured by default, and these are needed to connect the frontend to the local ganache to see the transactions of the blockchain blocks.



The screenshot shows the Ganache application interface. At the top, there is a navigation bar with icons for ACCOUNTS, BLOCKS, TRANSACTIONS, CONTRACTS, EVENTS, and LOGS. Below this is a status bar with various metrics: CURRENT BLOCK (737), GAS PRICE (2000000000), GAS LIMIT (6721975), HARDFORK (MERGE), NETWORK ID (5777), RPC SERVER (HTTP://127.0.0.1:7545), MINING STATUS (AUTOMINING), and WORKSPACE (PRODUCT_AUTHENTICITY_VERIFICATION). A search bar is also present. The main section displays the MNEMONIC (ivory kid decade appear blush load exist large gossip yard explain lava) and the HD PATH (m44'60'0'0account_index). Below this is a table of accounts.

| ADDRESS | BALANCE | TX COUNT | INDEX | |
|--|------------|----------|-------|--|
| 0x92AD65AB5d491ffAFCA7B6A995256B9Ea1182A57 | 97.33 ETH | 713 | 0 | |
| 0x9db467F99D08E173A6725D4041617E0Fbb0b549a | 99.99 ETH | 24 | 1 | |
| 0xEC829Fc4FF2121909dE6B0738D617Dc90672F977 | 100.00 ETH | 0 | 2 | |
| 0x230C71C5e24f4F4eA548d2a01d53987Fe24765f0 | 100.00 ETH | 0 | 3 | |
| 0xAD1FF668Cc4eA6f5516aB692f8BE2DDaCa8CbAf0 | 100.00 ETH | 0 | 4 | |
| 0x00dEdD7e534bC561CFb28B33A87EBfB5e320590f | 100.00 ETH | 0 | 5 | |
| 0x6c36F436D695F420Cdc4F9970EDc3acD25354f95 | 100.00 ETH | 0 | 6 | |

Figure 5.3: The accounts on the local Ganache application

After clicking into the local workspace on the Ganache, the details of the account are shown, and there is a total of 10 accounts provided with 100eth each for us to carry out the local testing and configuration of the system. Each account is also linked with its associated private keys to be used for encryption purposes when developing the projects.


```

Z:\Desktop\QiRui\yeu\utar\degree year 3\fyf\product-verification-structure>truffle migrate --reset

Compiling your contracts...
=====
> Compiling .\contracts\Migrations.sol
> Compiling .\contracts\ProductVerification.sol
> Artifacts written to Z:\Desktop\QiRui\yeu\utar\degree year 3\fyf\product-verification-structure\build\contracts
> Compiled successfully using:
   - solc: 0.8.13+commit.abaa5c0e.Emscripten.clang

Starting migrations...
=====
> Network name:      'development'
> Network id:        5777
> Block gas limit:   6721975 (0x6691b7)

1_initiate_contracts.js
=====

Replacing 'Migrations'
-----
> transaction hash:  0xe602c9ad4c4039fd99436db16545dc1f45d84d1d83e6594b40730d4e1e6446ff
> Blocks: 0         Seconds: 0
> contract address: 0x692AE1D708Ca76Ef357a2EAEB5AFE4A94c824689
> block number:     705
> block timestamp:  1757772823
> account:          0x92AD65AB5d491ffAFCA7B6A995256B9Ea1182A57
> balance:          97.396982616949971448
> gas used:         130690 (0x1fe82)
> gas price:        20 gwei
> value sent:       0 ETH
> total cost:       0.0026138 ETH

> Saving migration to chain.
> Saving artifacts
-----
> Total cost:       0.0026138 ETH

```

Figure 5.4: The output at cmd after migrations of contracts

To deploy the smart contract into the blockchain, the Solidity code was first compiled using the `truffle compile` command, which generated the bytecode and the Application Binary Interface (ABI). The bytecode contains the low-level instructions that run on the Ethereum Virtual Machine (EVM), while the ABI defines the functions and events so the frontend can interact with the contract. After compilation, the command `truffle migrate --reset` was executed to deploy the contract, uploading the bytecode to the blockchain and assigning it a unique address. The `--reset` option ensures that the contract is redeployed, replacing any previous version. For the system to work correctly, the deployed contract's ABI and address must match those used in the frontend. Therefore, after each redeployment, the new ABI file and contract address were copied into the React frontend configuration to synchronise the blockchain backend with the user interface.

```

Z:\Desktop\QiRui\yeu\utar\degree year 3\fyf\product-verification-structure\client>npm start

> client@0.1.0 prestart
> node scripts/copy-contracts.js

Contracts copied successfully!

> client@0.1.0 start
> react-scripts start

(node:72788) [DEP_WEBPACK_DEV_SERVER_ON_AFTER_SETUP_MIDDLEWARE] DeprecationWarning: 'onAfterSetupMiddleware' option is deprecated. Please use the 'setupMiddlewares' option.
(Use 'node --trace-deprecation ...' to show where the warning was created)
(node:72788) [DEP_WEBPACK_DEV_SERVER_ON_BEFORE_SETUP_MIDDLEWARE] DeprecationWarning: 'onBeforeSetupMiddleware' option is deprecated. Please use the 'setupMiddlewares' option.
Starting the development server...
Compiled with warnings.

[eslint]
src\components\manufacturer\RoleManagement.js
  Line 37:6:  React Hook React.useEffect has a missing dependency: 'loadContractState'. Either include it or remove the dependency array  react-hooks/exhaustive-deps

Search for the keywords to learn more about each warning.
To ignore, add // eslint-disable-next-line to the line before.

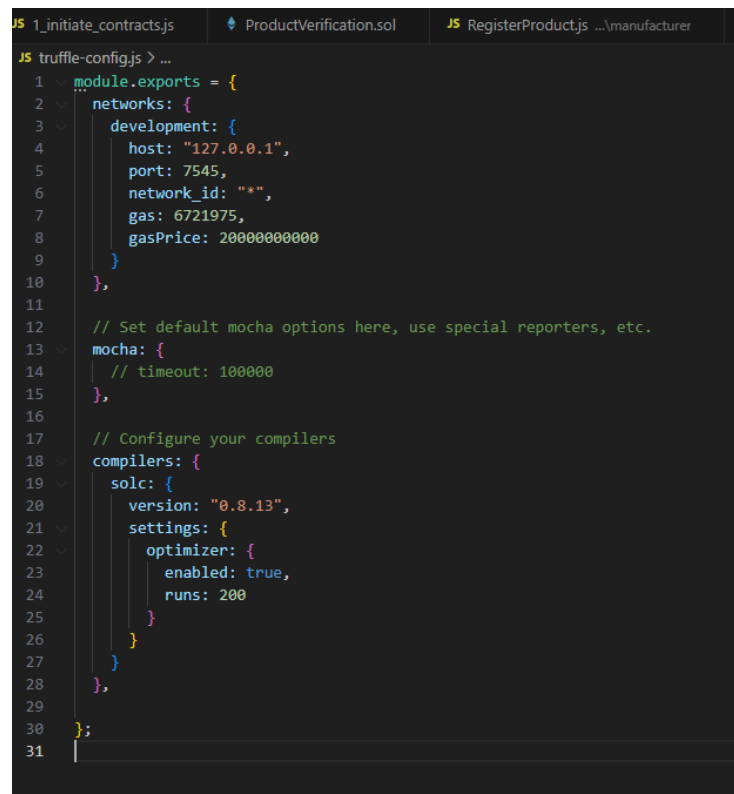
WARNING in [eslint]
src\components\manufacturer\RoleManagement.js
  Line 37:6:  React Hook React.useEffect has a missing dependency: 'loadContractState'. Either include it or remove the dependency array  react-hooks/exhaustive-deps

webpack compiled with 1 warning
The system cannot find message text for message number 0x237b in the message file for Application.

```

Figure 5.5: The output at cmd after launching React

After configuring both the frontend and backend logic using JavaScript and Solidity, the project was launched by running the **npm start** command in the React environment. This initialised the development server and allowed the frontend interface to run locally in the browser. Once the smart contract had been deployed and properly linked with the frontend through the ABI and contract address, any further changes made to the frontend code did not require redeploying the contract. Only when modifications were made to the core Solidity functions that interact with the blockchain was it necessary to recompile and redeploy the contract using Truffle. This streamlined workflow ensured that frontend updates could be tested immediately, while blockchain-related changes required a full redeployment process.



```

1  module.exports = {
2    networks: {
3      development: {
4        host: "127.0.0.1",
5        port: 7545,
6        network_id: "*",
7        gas: 6721975,
8        gasPrice: 20000000000
9      }
10   },
11
12   // Set default mocha options here, use special reporters, etc.
13   mocha: {
14     // timeout: 100000
15   },
16
17   // Configure your compilers
18   compilers: {
19     solc: {
20       version: "0.8.13",
21       settings: {
22         optimizer: {
23           enabled: true,
24           runs: 200
25         }
26       }
27     }
28   },
29 };
30
31

```

Figure 5.6: The truffle-config settings in the Visual Studio Code

The truffle-config.js file figure on above defines the development environment and compiler settings for deploying and testing smart contracts in Truffle. In this configuration, the development network is set up to connect to the local blockchain (Ganache) at host 127.0.0.1 and port 7545, with a specific network ID. The gas limit and the gas price are configured to manage transaction costs during contract deployment. The file also specifies the Solidity compiler version 0.8.1. Additionally, the optimiser is enabled with 200 runs, which improves contract execution efficiency by reducing gas consumption. This configuration ensures consistent contract compilation, efficient gas usage, and smooth interaction between the local blockchain and Truffle during development and deployment.

5.2 System Operation

Manufacturer portal

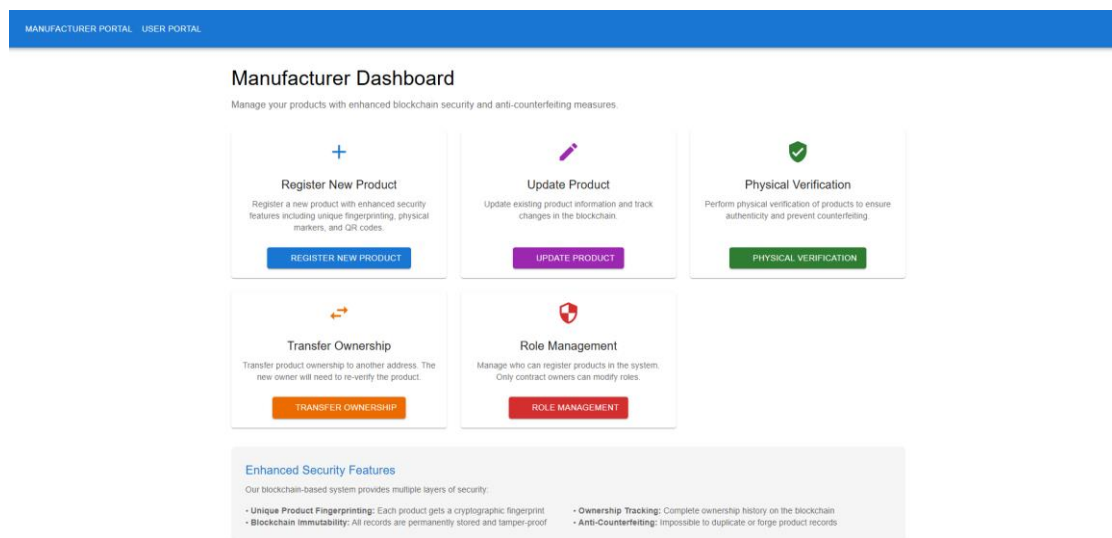


Figure 5.7: The manufacturer portal frontend using React

The figure above shows the home page for the manufacturer portal, which contains 5 main functions: Register New Product, Update Product, Physical Verification, Transfer Ownership, and Role Management. Each function is differentiated with different colours to have better visual effects.

The screenshot shows the 'Register New Product' form. It has a title 'Register New Product' at the top. The form contains three input fields:

- Serial Number ***: A text input field with the value 'ipad123' and a 'Max: 100 characters' label below it.
- Product Name ***: A text input field with the value 'Ipad Air 2025' and a 'Max: 100 characters' label below it.
- Manufacturer Details**: A text input field with the value 'US Ipad' and a '7/500 characters' label below it.

At the bottom of the form, there are two buttons: 'REGISTER PRODUCT' (blue) and 'BACK TO DASHBOARD' (light blue). A green circular logo with a white 'G' is visible on the right side of the form.

Figure 5.8: The Register New Product function

Register New Product

Serial Number *

Max: 100 characters

Product Name *

Max: 100 characters

Manufacturer Details

0/500 characters

REGISTER PRODUCT

BACK TO DASHBOARD

✔

Product registered successfully! Unique Code: Generated

Figure 5.9: The Register New Product function after completion

In this function, the manufacturer is required to enter essential product information such as the serial number, product name, and product details in the designated input fields. Once the information is provided, the manufacturer can click on the “Register Product” button, which triggers the smart contract to record the product on the blockchain. This process ensures that the product information is securely stored in an immutable ledger, enabling traceability and verification in later stages of the system.

CONTRACT

CONTRACT

ProductVerification

ADDRESS

0x7aff3cAAe367a9BA9332f9b3599E0B0Ec8AC8587

FUNCTION

registerProduct(_serialNumber: string, _name: string, _manufacturerDetails: string)

INPUTS

ipad123, Ipad Air 2025, US ipad

EVENTS

EVENT NAME

ProductRegistered

CONTRACT

ProductVerification

TX HASH

0xf95e1754lead3ff4e4d26a94c3f66a09a78b29d406c149cd3ca6b4019d903d7

LOG INDEX

0

BLOCK TIME

2025-09-16 22:02:52

Figure 5.10: The Register New Product function after completion (local ganache)

The figure above shows the Register New Product function after it has been successfully executed. Once a product is registered, its details are also recorded in the local Ganache transaction log. This allows every contract interaction to be clearly tracked, including important information such as the transaction hash (tx hash), the

CHAPTER 5

completion time of the contract, and the input provided by the manufacturer. By storing this information on the blockchain, the system ensures transparency, immutability, and verifiable proof of product registration.

Register New Product

Serial Number *

ipad123

Max 100 characters

Product Name *

Ipad Air 2025

Max 100 characters

Manufacturer Details

US ipad

7/500 characters

REGISTER PRODUCT

BACK TO DASHBOARD

❗

Error registering product: Product already registered.

Figure 5.11: The Register New Product when duplicate registration occurs

Register New Product

Serial Number *

k90

Max 100 characters

Product Name *

k90

Max 100 characters

Manufacturer Details

gt

2/500 characters

REGISTER PRODUCT

BACK TO DASHBOARD

❗

Error registering product: Do not have permission to register products.

Figure 5.12: The Register New Product when the manufacturer does not have access

The figures above show that if the product with the same batch number is registered again, the error message will be shown, and if the user does not have permission to register the product, it will block the user from doing it.

MANUFACTURER PORTAL USER PORTAL

Update Product Details

Serial Number *

ipad123

Max: 100 characters

Manufacturer Details

Malaysia retailer

17/500 characters

Product Details

ipad Air 2025

13/500 characters

UPDATE PRODUCT

BACK TO DASHBOARD

Figure 5.13: The Update Product Details

MANUFACTURER PORTAL USER PORTAL

Update Product Details

Serial Number *

Max: 100 characters

Manufacturer Details

0/500 characters

Product Details

0/500 characters

UPDATE PRODUCT

BACK TO DASHBOARD

✔ Product updated successfully! Re-verification required.

Figure 5.14: The Update Product Details when completed

MANUFACTURER PORTAL USER PORTAL

Update Product Details

Serial Number *

ipad123

Max: 100 characters

Manufacturer Details

ipad air

0/500 characters

Product Details

malaysia retailer

17/500 characters

UPDATE PRODUCT

BACK TO DASHBOARD

⚠ Only the manufacturer who registered this product can update it

Figure 5.15: The Update Product Details when the user is not actual manufacturer

The registered product can then be updated using this function, and only the manufacturer of this product can do this update. After the update is completed, the system will respond with a message indicating that it has been successfully updated. And if the user is not the manufacturer of the product, they are unable to update the information to avoid any duplication or unauthentic behaviour.

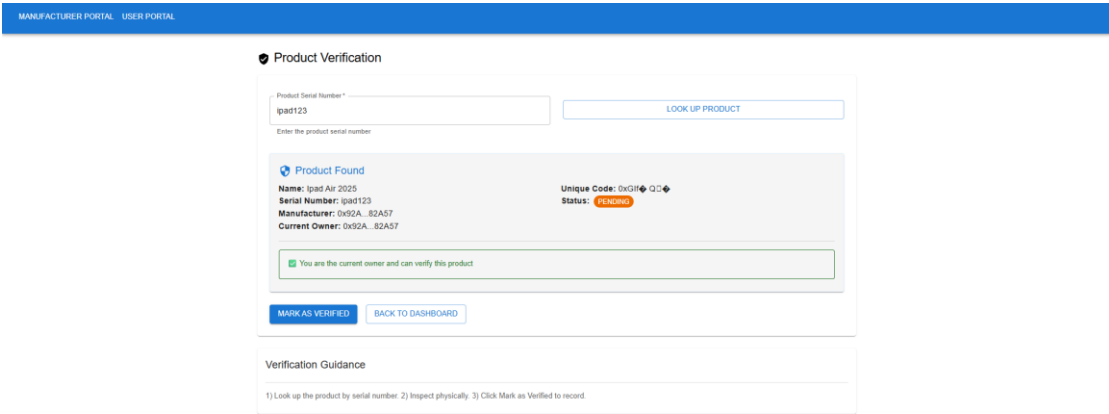


Figure 5.16: The Product Verification function

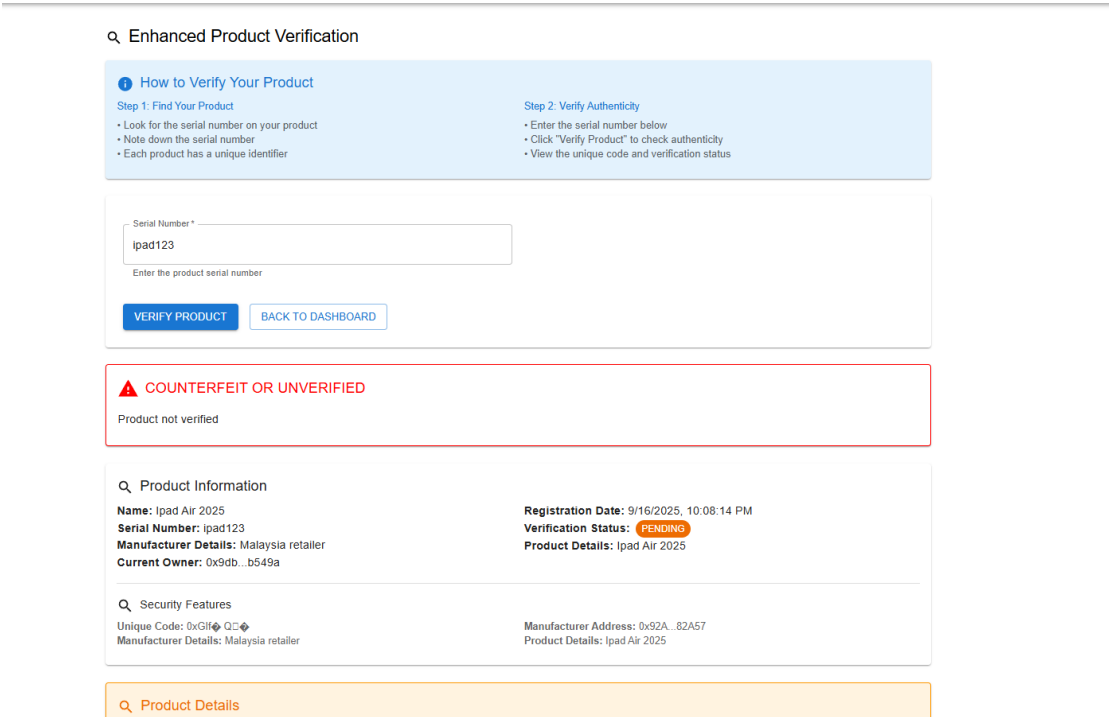


Figure 5.17: The Verify Product function before verification

CHAPTER 5

The figures above show the function of Product Verification. Before the manufacturer or the owner of the product verifies it, the product will appear as counterfeit or unverified when searched in the user portal.

Product Verification

Product Serial Number *

ipad123

Enter the product serial number

LOOK UP PRODUCT

Product Found

Name: Ipad Air 2025

Serial Number: ipad123

Manufacturer: 0x92A...82A57

Current Owner: 0x9db...b549a

Unique Code: 0xGIf...QE

Status: VERIFIED

✓ You are the current owner and can verify this product

MARK AS VERIFIED

BACK TO DASHBOARD

✓ Verification submitted! The product status has been updated on-chain.

Figure 5.18: The Product Verification function after verification

Enhanced Product Verification

How to Verify Your Product

Step 1: Find Your Product

• Look for the serial number on your product

• Note down the serial number

• Each product has a unique identifier

Step 2: Verify Authenticity

• Enter the serial number below

• Click "Verify Product" to check authenticity

• View the unique code and verification status

Serial Number *

ipad123

Enter the product serial number

VERIFY PRODUCT

BACK TO DASHBOARD

✓ AUTHENTIC PRODUCT

Product verified

Product Information

Name: Ipad Air 2025

Serial Number: ipad123

Manufacturer Details: Malaysia retailer

Current Owner: 0x9db...b549a

Registration Date: 9/16/2025, 10:08:14 PM

Verification Status: VERIFIED

Product Details: Ipad Air 2025

Security Features

Unique Code: 0xGIf...QE

Manufacturer Details: Malaysia retailer

Manufacturer Address: 0x92A...82A57

Product Details: Ipad Air 2025

Product Details

Figure 5.19: The Verify Product function after verification

Bachelor of Information Technology (Honours) Communications and Networking
Faculty of Information and Communication Technology (Kampar Campus), UTAR

60

CHAPTER 5

The figures above show the function of Product Verification upon completion. After the manufacturer or the owner of the product verifies it, the product will appear as an authentic product when searched in the user portal. The colour is also bright green to indicate it is the real product

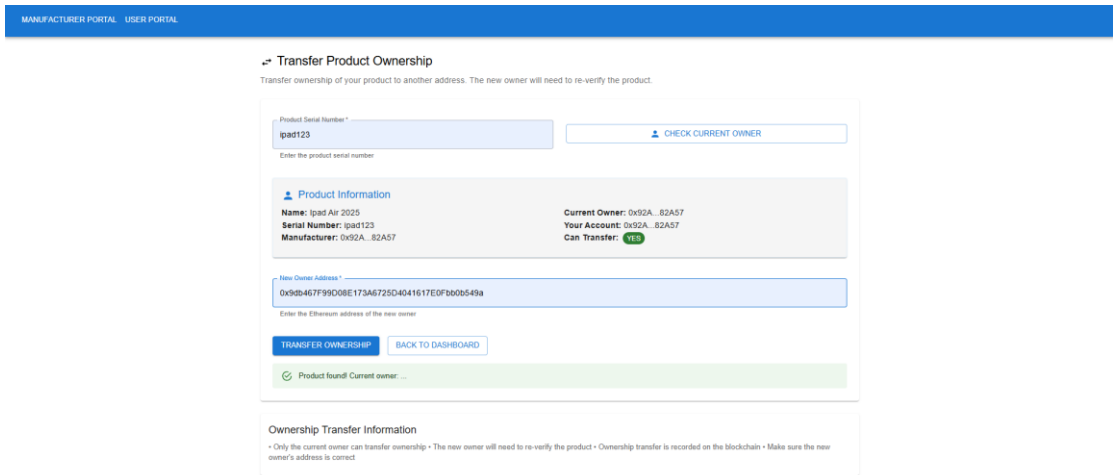


Figure 5.20: The Transfer Product Ownership function (can transfer)

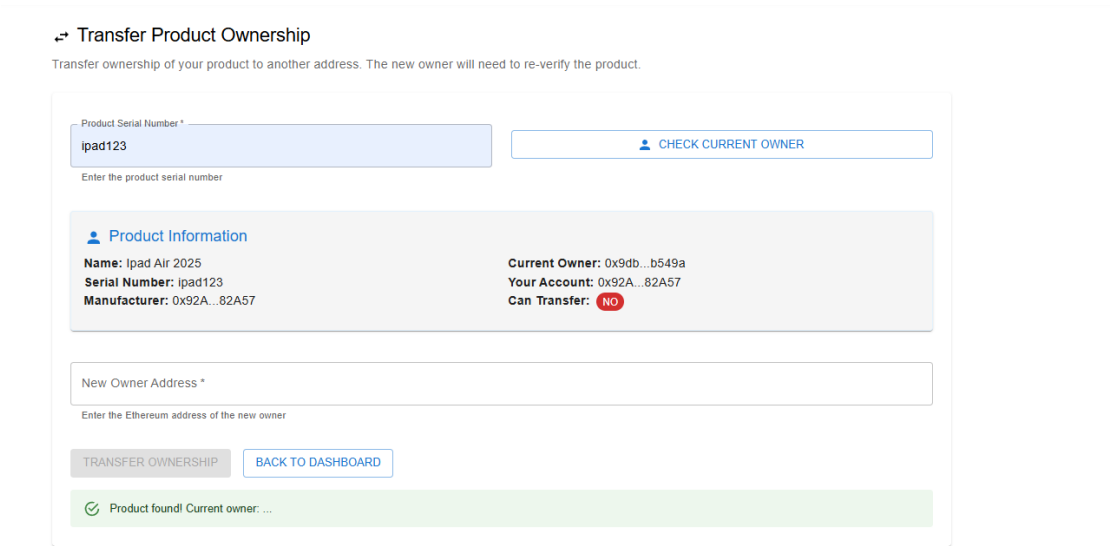


Figure 5.21: The Transfer Product Ownership function (can’t transfer)

➔ Transfer Product Ownership

Transfer ownership of your product to another address. The new owner will need to re-verify the product.

Product Serial Number *

ipad123

Enter the product serial number

CHECK CURRENT OWNER

Product Information

Name: Ipad Air 2025
Serial Number: ipad123
Manufacturer: 0x92A...82A57

Current Owner: 0x9db...b549a
Your Account: 0x9db...b549a
Can Transfer: YES

New Owner Address *

Enter the Ethereum address of the new owner

TRANSFER OWNERSHIPBACK TO DASHBOARD

✔ Product found! Current owner: ...

Ownership Transfer Information

- Only the current owner can transfer ownership
- The new owner will need to re-verify the product
- Ownership transfer is recorded on the blockchain
- Make sure the new owner's address is correct

Figure 5.22: The Transfer Product Ownership function after completion

Update #2

9/16/2025, 10:08:14 PM

Account: 0x92A...82A57

Details: —

Product Status: Ownership transferred

↓

Update #3

9/16/2025, 10:36:06 PM

Account: 0x9db...b549a

Details: —

Product Status: Ownership transferred

Figure 5.23: The details of the transfer show on the product history

The 4 figures above show the process of transferring product ownership. At first, the user or the manufacturer needs to check the current owner of the product to verify and make sure they are the current owner, and if they are the current owner, they can type the new blockchain account in the address column and pass the product to them after

CHAPTER 5

pressing the transfer ownership. The local ganache contract will be updated, and the new contract call transfer ownership can be seen on the product search as well.

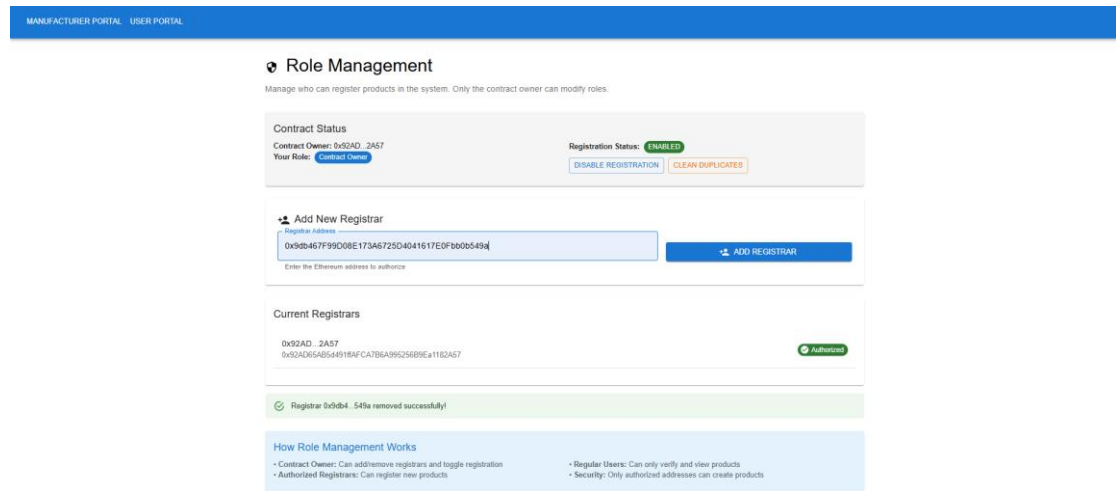


Figure 5.24: The Role Management function

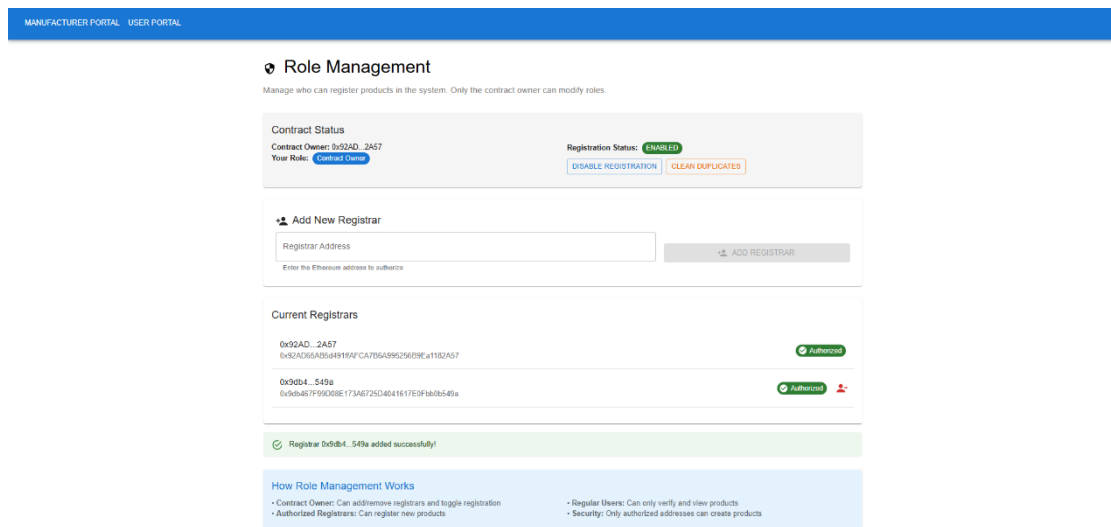


Figure 5.25: The Role Management function after completion

The figures above show the role management function on the manufacturer portal. Only the account that is added to the role management function can perform actions such as registering and updating. This is to prevent users from randomly updating the wrong information about the product.

User portal

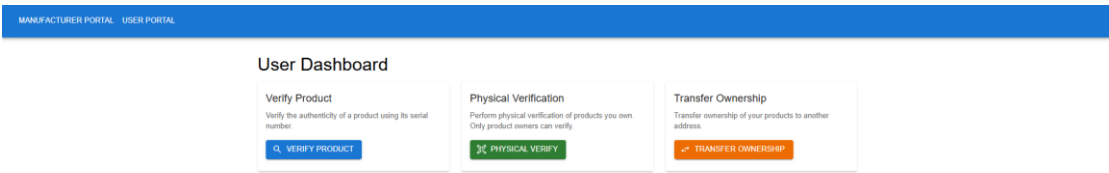
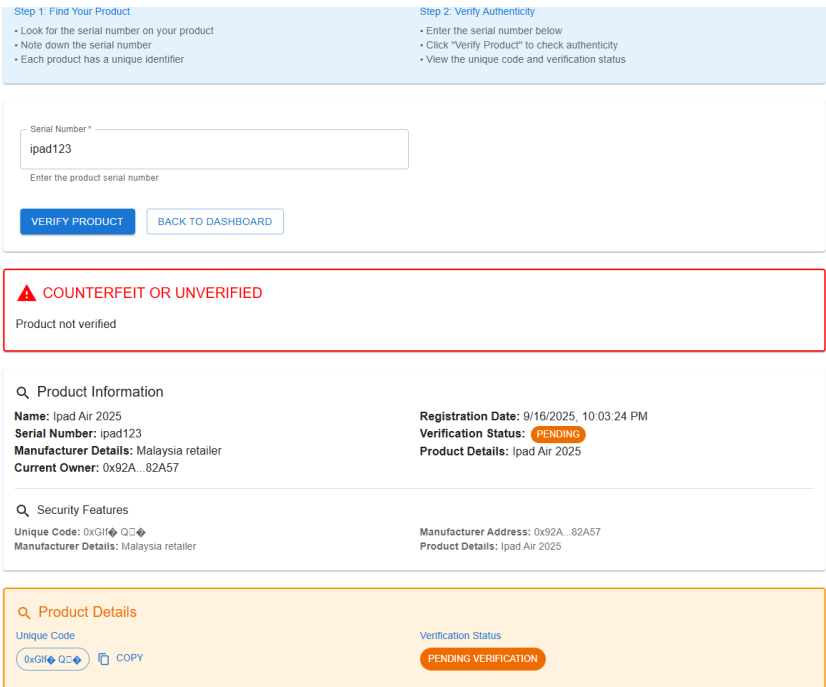


Figure 5.26: The User Dashboard overview

The figure above shows the overview of the user dashboard, which consists of 3 main functions which are mainly verifying product, physical verification and transferring ownership.



CHAPTER 5

Product History 4 Entries

The screenshot displays a 'Product History' section with 4 entries. Each entry is a card showing a timestamp, an account ID (0x92A...82A57), and product details. The first entry, 'Initial Registration' (9/16/2025, 10:02:52 PM), shows 'US Ipad' and 'Registered' status. The second entry, 'Update #1' (9/16/2025, 10:03:24 PM), shows 'Malaysia retailer' and 'Ipad Air 2025' status. The third entry, 'Update #2' (9/16/2025, 10:08:14 PM), is partially visible. Downward arrows indicate the sequence of updates.

Figure 5.27: The Verify Product function overview

After entering the serial number, if the product already exists on the blockchain, the system will automatically display its details, including the serial number, manufacturer information, product description, and other relevant attributes in the first section. The second section focuses on the product's history mapping, which provides a transparent record of all updates and interactions associated with the product. This includes critical functions such as transfer of ownership and other modifications, ensuring that every change is securely documented and traceable on the blockchain.

MANUFACTURER PORTAL USER PORTAL

Transfer Ownership

Check Product Ownership

Enter the product serial number to check current ownership and transfer to another address.

Serial Number

ipad123

CHECK OWNER

✔

You are the current owner. You can transfer ownership to another address.

Ownership Information

Current Owner:

0x92AD...2A57

✔

You are the current owner of this product. You can transfer ownership to another address.

Transfer Ownership

Enter the new owner's address to transfer ownership of this product.

New Owner Address

TRANSFER

Product Information

Product Name: Ipad Air 2025

Registration Date: Invalid Date

Serial Number:

Verification Count:

Manufacturer: N/A

Current Owner: 0x92AD...2A57

Figure 5.28: The Physical Verification

MANUFACTURER PORTAL USER PORTAL

Physical Verification

Verify Product Authenticity

Enter the product serial number to perform physical verification. Only the current owner can verify the product.

Serial Number

ipad123

CHECK OWNER

VERIFY

❗

You are not the current owner of this product. Current owner: 0x92AD...2A57

Verification Result

✔ Status:

NOT VERIFIED

Ownership:

NOT YOURS

⚠

You are not the current owner of this product. Current owner: 0x92AD...2A57

Ownership Status

⚠

You are not the current owner. Current owner: 0x92AD...2A57

Figure 5.29: The Physical Verification when user is not the product owner

Product Verification

Product Serial Number *

ipad123

Enter the product serial number

LOOK UP PRODUCT


Product Found

Name: Ipad Air 2025

Serial Number: ipad123

Manufacturer: 0x92A...82A57

Current Owner: 0x92A...82A57

Unique Code: 0xY  RQw

Status: VERIFIED

☒ You are the current owner and can verify this product

MARK AS VERIFIED

BACK TO DASHBOARD


 Verification submitted! The product status has been updated on-chain.

Figure 5.30: The Physical Verification after completion

Enhanced Product Verification

How to Verify Your Product

Step 1: Find Your Product

- Look for the serial number on your product
- Note down the serial number
- Each product has a unique identifier

Step 2: Verify Authenticity

- Enter the serial number below
- Click "Verify Product" to check authenticity
- View the unique code and verification status


Serial Number *

ipad123

Enter the product serial number

VERIFY PRODUCT

BACK TO DASHBOARD

 AUTHENTIC PRODUCT

Product verified

Product Information

Name: Ipad Air 2025

Serial Number: ipad123

Manufacturer Details: asdfasdf


Current Owner: 0x92A...82A57

Registration Date: 9/18/2025, 2:24:24 AM

Verification Status: VERIFIED

Product Details:

Security Features

Unique Code: 0xY  RQw

Manufacturer Address: 0x92A...82A57

Product Details

Figure 5.30: The Verify Product after verification

Bachelor of Information Technology (Honours) Communications and Networking
Faculty of Information and Communication Technology (Kampar Campus), UTAR

67

The figures above illustrate the process of physical verification. Before proceeding, the system checks whether the user is the rightful account owner; if not, the verification attempt will fail. Once confirmed as the owner, the user can mark the product as verified after physically receiving and confirming it in real life. The system then updates the product status to “verified,” assuring authenticity. This verification status also benefits the user in future transactions, such as when reselling the product second-hand, as it strengthens trust and credibility for the next buyer.

↔ Transfer Ownership

Check Product Ownership

Enter the product serial number to check current ownership and transfer to another address.

You are the current owner. You can transfer ownership to another address.

Ownership Information

Current Owner: 0x92AD...2A57

You are the current owner of this product. You can transfer ownership to another address.

Transfer Ownership

Enter the new owner's address to transfer ownership of this product.

Product Information

| | |
|------------------------------------|--|
| Product Name: Ipad Air 2025 | Registration Date: Invalid Date |
| Serial Number: | Verification Count: |
| Manufacturer: N/A | Current Owner: 0x92AD...2A57 |

Figure 5.32: The Transfer Ownership function (product owner)

↔ Transfer Ownership

Check Product Ownership

Enter the product serial number to check current ownership and transfer to another address.

Serial Number

ipad123

CHECK OWNER

⚠ You are not the current owner of this product. Current owner: 0x92AD...2A57

Ownership Information

👤

Current Owner: 0x92AD...2A57

⚠ You are not the current owner of this product. Current owner: 0x92AD...2A57

Product Information

| | |
|------------------------------------|--|
| Product Name: Ipad Air 2025 | Registration Date: Invalid Date |
| Serial Number: | Verification Count: |
| Manufacturer: N/A | Current Owner: 0x92AD...2A57 |

Figure 5.33: The Transfer Ownership function (not product owner)

Ownership transferred successfully! New owner: 0x9db4...549a

Figure 5.34: The Transfer Ownership function after completion

Update #2 9/16/2025, 10:08:14 PM

Account: 0x92A...82A57

Details:

—

Product Status:

Ownership transferred

↓

Update #3 9/16/2025, 10:36:06 PM

Account: 0x9db...b549a

Details:

—

Product Status:

Ownership transferred

Figure 5.35: The Verify Product after Transfer Ownership function

The figures above present the Transfer Ownership function on the user portal. This feature ensures that only the actual product owner, logged in with the correct blockchain account, can access the product details and initiate a transfer. By enforcing this ownership validation, the system guarantees that transfers are performed only by legitimate owners, thereby preserving product authenticity. When a user chooses to sell the product, they can securely transfer ownership to another account, and the updated ownership record on the blockchain provides transparent proof of authenticity for the new buyer.

CHAPTER 5

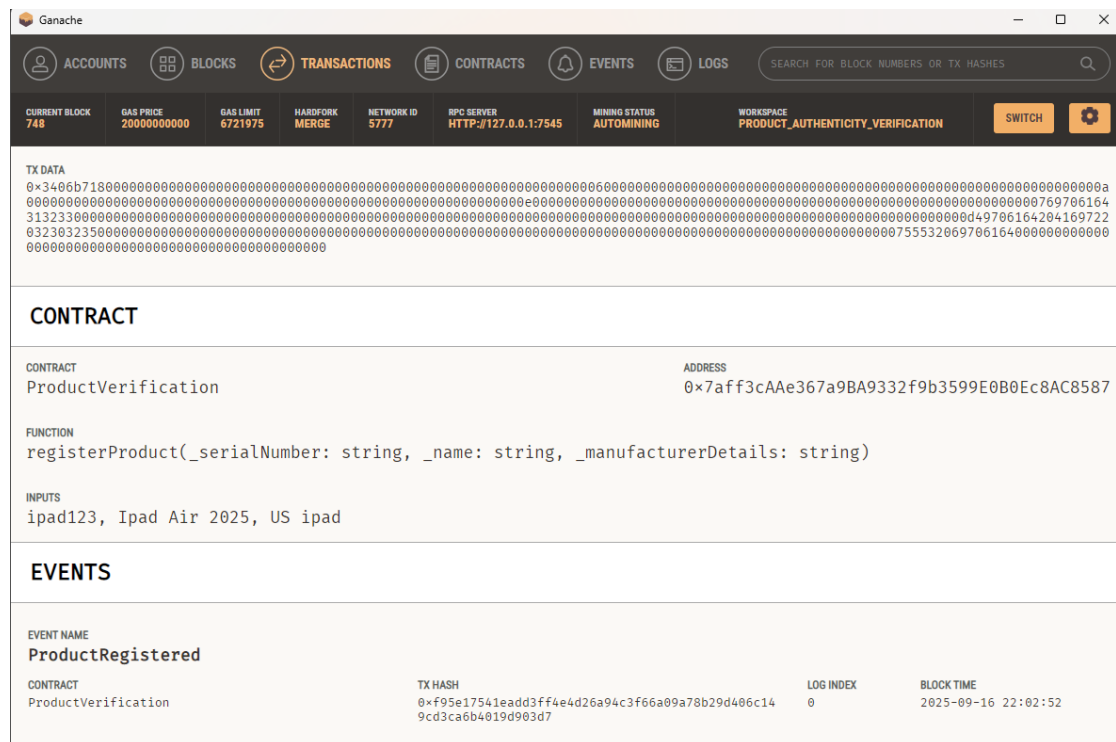


Figure 5.36: The transaction blocks on the local ganache after registering product

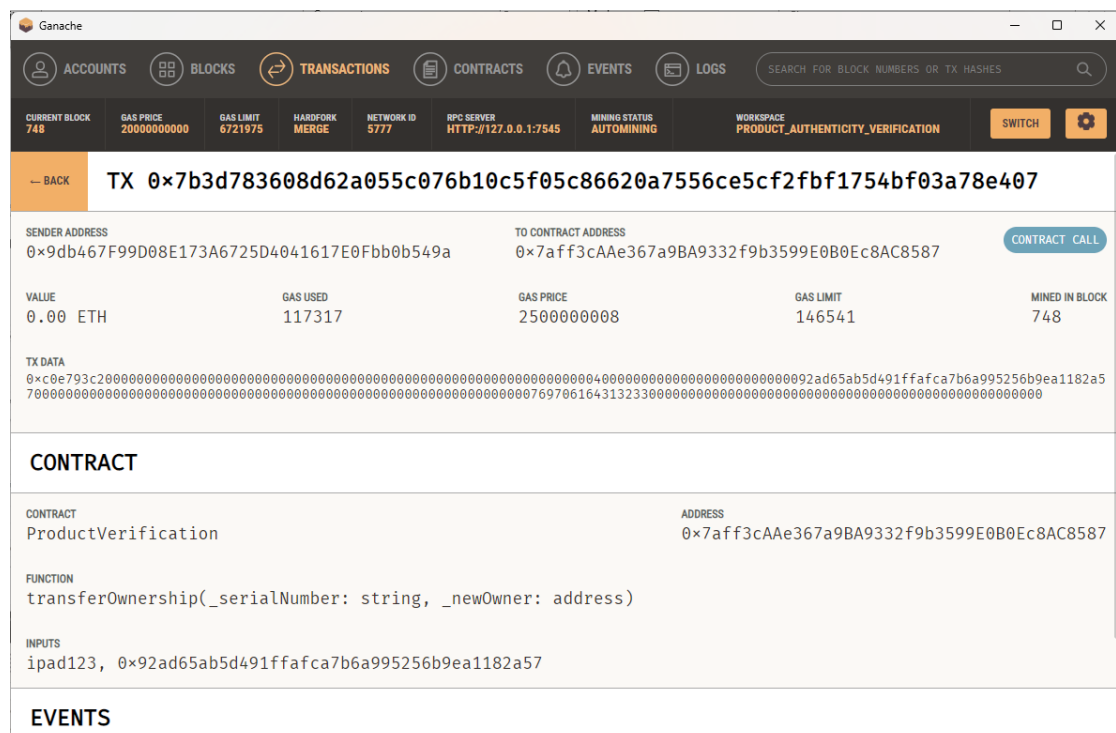


Figure 5.37: The transaction blocks on the local ganache after transferring product

CHAPTER 5

Figures 5.35 and 5.36 illustrate the transaction blocks recorded in the local Ganache account after performing product registration and product transfer, respectively. These records confirm that the smart contract operations were successfully executed and stored on the blockchain. Each block contains detailed information such as the transaction hash, contract address, and execution status, ensuring that every interaction—whether registering a new product or transferring ownership—is transparently documented and verifiable within the local blockchain environment.

5.3 Implementation Issues and Challenges

During the development of the system, several implementation challenges were encountered and resolved. One major issue was the **ABI mismatch and parameter decoding errors**, which occurred when the smart contract was updated, but the front end continued to use an outdated ABI or contract address. This was solved by following a consistent workflow of compiling the contract, migrating it, copying the updated ABI into 'client/src/contracts/', and then restarting the React application. Another challenge involved **Ganache resets and data loss**, where redeploying the contract with the '—reset' option cleared all existing data. To address this, roles had to be re-added and products re-registered after each reset.

In addition, out-of-gas errors arose during ownership transfer operations. These were mitigated by the description of the contract functions and adding gas estimation on the front end with a 20% buffer. There were also issues with the **MetaMask network and account alignment**, where transactions would fail if the chain ID did not match Ganache or if the user account was not properly configured. This was resolved by ensuring network consistency and clarifying the distinction between the contract owner and the transaction sender.

On the front end, challenges included **maintaining UI state consistency**, where crashes occurred due to undefined values. This was fixed by implementing safer functions such as shortenAddress to handle null cases, correcting malformed try/catch blocks that caused build failures, and describing product details when the ABI returned arrays or tuples. Furthermore, the project required **addressing registrar listing and duplication issues**, which were solved by implementing an on-chain registrarList, adding a getAllRegistrars() function, and enforcing duplicate prevention with cleanup mechanisms.

Lastly, **access control clarity** was critical to ensure system security. The solution was to strictly restrict registration functions to description registrars, allow only manufacturers to update product details, and ensure that verification and transfer functions could only be executed by the current owner. These access rules were also reflected in the user interface, where actions were carefully guarded and supported with

clear errors or success messages. Together, these measures ensured stability, security, and usability of the system.

5.4 Concluding Remarks

In short, the system implementation successfully demonstrated the integration of blockchain-based smart contracts with a React frontend to achieve secure product registration, verification, and ownership transfer. Through careful configuration of the development environment, deployment of contracts, and testing via the manufacturer and user portals, the system achieved its intended functionalities while ensuring transparency, traceability, and authenticity of products. Although several challenges were encountered—such as ABI mismatches, Ganache resets, gas-related issues, and frontend state inconsistencies—each was resolved through systematic debugging and design improvements. These solutions not only strengthened the reliability and stability of the system but also enhanced its usability and security. Overall, the completed implementation validates the feasibility of using blockchain technology as a robust solution for product authentication and supply chain transparency.

CHAPTER 6 SYSTEM EVALUATION AND DISCUSSION

In this chapter, the evaluation of the system is presented through a series of test cases designed to verify whether the objectives of the project have been successfully achieved. Each test case focuses on the core functionalities of the system, ensuring that the implementation aligns with the intended goals of transparency, security, and traceability. Furthermore, this chapter discusses the significance of integrating blockchain technology within the system, performing its role in ensuring immutability, decentralisation, and trust. A comparative discussion is also provided to illustrate the potential risks, vulnerabilities, and limitations that would arise if blockchain were not applied, thereby reinforcing the necessity of blockchain as the backbone of this project.

6.1 System Testing and Performance Metrics

| Test Case | Test Case Description | Testing Steps | Expected Results | Pass/Fail |
|-----------|--|---|--|-----------|
| 1 | Open the application and check the Web3 connection | <ol style="list-style-type: none"> 1. Start Ganache 2. Open React app 3. Check console for connection logs | <ul style="list-style-type: none"> • Web3 initialised successfully” • “Contract initialised successfully” • No error messages | Pass |
| 2 | Product Registration (Authorised User) | <ol style="list-style-type: none"> 1. Go to Manufacturer Portal → Register Product 2. Fill: Serial Number: “TEST001”, Name: “Test | <ul style="list-style-type: none"> • Success message: “Product registered successfully” • Product appears in verification • Unique code generated | Pass |

| | | | | |
|---|--|--|--|------|
| | | Product”, Manufacturer: “Test Company” 3. Click Register | | |
| 3 | Product Registration (Unauthorised User) | 1. Switch to a non- authorised MetaMask account 2. Try to register the product | <ul style="list-style-type: none"> Error: “You are not authorised to register products” Registration fails | Pass |
| 4 | Duplicate Serial Number Prevention | 1. Register product with serial “TEST001” 2. Try to register another product with same serial “TEST001” | <ul style="list-style-type: none"> Error: “Serial number already exists” Second registration fails | Pass |
| 5 | Product Verification (Public) | 1. Go to User Portal → Verify Product 2. Enter serial number “TEST001” 3. Click Verify | <ul style="list-style-type: none"> Product details displayed Shows authentic status Shows manufacturer info | Pass |

| | | | | |
|---|------------------------------------|--|--|------|
| 6 | Physical Verification (Owner Only) | <ol style="list-style-type: none"> 1. Go to User Portal → Physical Verification 2. Enter serial “TEST001” 3. Click “Check Owner” (should show you’re owner) 4. Click “Verify” | <ul style="list-style-type: none"> • “Ownership confirmed” message • “Physical verification completed successfully” • Verification counts increases | Pass |
| 7 | Physical Verification (Non-Owner) | <ol style="list-style-type: none"> 1. Switch to a different MetaMask account 2. Try physical verification for “TEST001” | <ul style="list-style-type: none"> • “You are not the current owner” message • Shows current owner’s address • Verification fails | Pass |
| 8 | Ownership Transfer (Owner) | <ol style="list-style-type: none"> 1. Go to User Portal → Transfer Ownership 2. Enter serial “TEST001” 3. Click “Check Owner” (should show you’re owner) 4. Enter the new owner’s address 5. Click “Transfer” | <ul style="list-style-type: none"> • “Ownership transferred successfully” message • Snackbar confirmation • The new owner can now verify | Pass |

| | | | | |
|----|----------------------------------|--|---|------|
| | | | | |
| 9 | Ownership Transfer (Non-Owner) | <ol style="list-style-type: none"> 1. Switch to non-owner account 2. Try to transfer ownership of "TEST001" | <ul style="list-style-type: none"> • "You are not the current owner" message • Transfer fails | Pass |
| 10 | Role Management (Contract Owner) | <ol style="list-style-type: none"> 1. Go to Manufacturer Portal → Role Management 2. Add new registrar address 3. Remove registrar address 4. Toggle registration on/off | <ul style="list-style-type: none"> • "Registrar added successfully" • "Registrar removed successfully" • Registration status changes | Pass |
| 11 | Role Management (Non-Owner) | <ol style="list-style-type: none"> 1. Switch to a non-owner account 2. Try to access Role Management | <ul style="list-style-type: none"> • Access denied or redirected • Cannot modify roles | Pass |
| 12 | Product Update (Manufacturer) | <ol style="list-style-type: none"> 1. Go to Manufacturer Portal → Update Product 2. Enter serial "TEST001" 3. Update product status/details | <ul style="list-style-type: none"> • "Product updated successfully" • Changes reflected in verification | Pass |

| | | | | |
|----|---|---|---|------|
| | | | | |
| 13 | Product History Tracking | <ol style="list-style-type: none"> 1. Register product 2. Update product 3. Transfer ownership 4. Check history in User Portal | <ul style="list-style-type: none"> • All actions logged with timestamps • Performer addresses shown • Action description are clear | Pass |
| 14 | Gas Estimation and Transaction Handling | <ol style="list-style-type: none"> 1. Perform various transactions (register, transfer, verify) 2. Check gas estimates in the console | <ul style="list-style-type: none"> • Gas estimates calculated • 20% buffer applied • No “out of gas” errors | Pass |
| 15 | Error Handling and Edge Cases | <ol style="list-style-type: none"> 1. Enter invalid serial numbers 2. Try operations without MetaMask 3. Test with empty fields | <ul style="list-style-type: none"> • Clear error messages • No crashes • Graceful degradation | Pass |
| 16 | Unique Code Generation and Validation | <ol style="list-style-type: none"> 1. Register product with serial “ABC123” 2. Register product with serial “XYZ789” 3. Check unique codes are different | <ul style="list-style-type: none"> • Different serials = different unique codes • Same serial = same unique code • Code contains serial, timestamp, block number | Pass |

| | | | | |
|----|----------------------------------|---|---|------|
| | | 4. Verify same serial generates the same cod | | |
| 17 | Manufacturer Details Consistency | <ol style="list-style-type: none"> 1. Register product as Manufacturer A 2. Update product as Manufacturer A 3. Check the manufacturer field unchanged | <ul style="list-style-type: none"> • Original manufacturer preserved • Updates don't change the manufacturer • History shows the correct performer | Pass |
| 18 | Verification Count Tracking | <ol style="list-style-type: none"> 1. Register product (count = 0) 2. Perform physical verification (count = 1) 3. Perform another verification (count = 2) | <ul style="list-style-type: none"> • Count increments with each verification • Count persists across page refreshes • Count displayed in product details | Pass |
| 19 | Ownership Chain Validation | <ol style="list-style-type: none"> 1. Manufacturer registers product (owner = manufacturer) 2. Transfer to Distributor (owner = distributor) 3. Transfer to Retailer | <ul style="list-style-type: none"> • Each transfer updates owner correctly • Only the current owner can transfer • Complete chain visible in history | Pass |

| | | | | |
|----|--|---|--|------|
| | | (owner = retailer) 4. Transfer to Customer (owner = customer) | | |
| 20 | Registrar Authorisation Workflow | <ol style="list-style-type: none"> 1. Contract owner adds new registrar 2. New registrar registers product 3. Contract owner removes registrar 4. Removed registrar tries to register | <ul style="list-style-type: none"> • A new registrar can register • Removed registrar cannot register • Authorization changes take effect immediately | Pass |
| 21 | Contract Owner Privileges | <ol style="list-style-type: none"> 1. Contract owner adds themselves as registrar 2. Contract owner removes other registrars 3. Contract owner transfers contract ownership | <ul style="list-style-type: none"> • Owner can manage all registrars • Owner cannot remove themselves • Contract ownership transfer works | Pass |
| 22 | Cross-Portal Functionality | <ol style="list-style-type: none"> 1. Register product in Manufacturer Portal 2. Verify product in User Portal | <ul style="list-style-type: none"> • Data consistent across portals • All features are accessible where expected | Pass |

| | | | | |
|--|--|--|---|--|
| | | 3. Transfer ownership in User Portal 4. Update product in Manufacturer Portal | <ul style="list-style-type: none"> No portal-specific data isolation | |
|--|--|--|---|--|

Table 5.1: The system testing and results

6.2 Objectives Evaluation

1. Enhanced Product Authentication

- The system successfully records product information on the blockchain, ensuring transparency and immutability.
- Consumers and stakeholders can verify product authenticity at each stage of the supply chain without depending on a central authority.
- The blockchain prevents tampering and fraudulent claims, guaranteeing a reliable authentication mechanism.

2. Improved Traceability

- Every transaction and product movement is permanently stored on the blockchain ledger.
- Stakeholders can trace the complete product lifecycle, from manufacturing to delivery.
- In case of recalls, disputes, or contamination issues, the immutable record provides clear accountability.

3. Prevention of Counterfeiting

- Smart contracts are used to restrict unauthorised entries, ensuring that only legitimate products are registered.
- Advanced encryption techniques safeguard the data, making it extremely difficult for counterfeit products to be inserted.
- This reduces the risk of fake goods entering sensitive sectors such as healthcare and food supply chains.

4. Data Security and Privacy

- Sensitive product and stakeholder data is protected through encryption methods such as public/private keys and elliptic curve cryptography (ECC).
- Transparency is maintained for verification, while private data remains accessible only to authorised parties.
- This balance ensures that the system maintains both traceability and confidentiality.

5. User-Friendly Verification

- A mobile/web application was developed to allow consumers to easily check product authenticity.
- The verification process is simplified, requiring only account-based blockchain details, with ECC providing secure validation.
- This improves user experience and encourages adoption, as consumers do not need technical knowledge of blockchain.

6. Efficient Distributed Storage

- Product data is stored securely on the blockchain, eliminating reliance on centralised databases.
- Distributed storage ensures data integrity and availability, even if some nodes fail.
- This makes the system resistant to tampering, hacking, or single points of failure.

6.3 Concluding Remarks

The evaluation results demonstrate that the system has successfully achieved its primary objectives of enhancing product authentication, improving traceability, preventing counterfeiting, and ensuring data integrity through blockchain integration. The test cases confirm that the system is functional, reliable, and capable of providing a transparent and tamper-proof product verification mechanism. Furthermore, the objectives evaluation highlights that the system not only addresses the core issues of counterfeit prevention and supply chain transparency but also delivers a user-friendly verification experience for consumers.

However, the analysis also revealed some limitations, such as the absence of advanced encryption mechanisms, mobile application support, and large-scale distributed storage integration. These limitations do not undermine the overall effectiveness of the system but instead present opportunities for future enhancements.

In conclusion, the application of blockchain technology has proven essential in achieving the desired goals of decentralisation, transparency, and trust within the supply chain. Without blockchain, the system would face risks of data manipulation, centralised control, and reduced consumer confidence. Therefore, blockchain serves as the foundation that ensures the security, accountability, and resilience of the proposed product authentication system.

CHAPTER 7

CONCLUSION AND RECOMMENDATIONS

7.1 Conclusion

This project's goal is to deal with the persistent and growing issue of counterfeit products in supply chains by developing a blockchain-based product authenticity verification system. Counterfeiting poses significant risks to both consumers and businesses, including financial losses, health hazards, and loss of trust. Traditional centralised authentication systems are often vulnerable to manipulation, lack transparency, and fail to provide consumers with a reliable method of verification. In response to these challenges, this project demonstrated the potential of blockchain technology to provide a decentralised, immutable, and transparent solution for product authentication.

The system designed and implemented in this study leverages blockchain smart contracts, deployed in a local Ganache environment, to manage critical operations such as product registration, product updates, role-based access control, and ownership transfer. These core functionalities ensure that once product details are recorded, they cannot be altered or falsified, thereby safeguarding authenticity at every stage of the supply chain. The React-based web application provided two distinct portals—one for manufacturers to manage product information and another for users to verify authenticity and ownership history. By integrating these components, the prototype successfully demonstrated the feasibility of using blockchain as the foundation for a secure and tamper-proof verification system.

The testing phase confirmed that the system performed reliably across all implemented functions. Manufacturers were able to register new products with unique serial numbers, update details where necessary, and transfer ownership securely. Users, on the other hand, could easily query the blockchain to confirm authenticity and view an immutable history of product transactions. The role management feature further ensured that only authorised parties could register or update product information, enhancing system integrity and reducing opportunities for misuse.

One of the key strengths of this project lies in its emphasis on usability and adaptability. The web-based interface was intentionally designed to be user-friendly, lowering the barrier for adoption by both businesses and consumers. Furthermore, while the

prototype was implemented in a controlled development environment, its modular structure and reliance on standard blockchain tools (Solidity, Truffle, React, Ganache) ensure that it can be extended or migrated to larger-scale blockchain networks in the future.

From an academic perspective, this project validates the hypothesis that blockchain can serve as a practical technological foundation for tackling product counterfeiting. It not only demonstrates the theoretical advantages of decentralisation, immutability, and transparency but also provides a working system that operationalises these principles in a supply chain context. The project contributes to ongoing research and industrial discussions by showing that blockchain technology, even at a prototype stage, can directly strengthen consumer confidence and protect businesses from reputational and financial harm caused by counterfeit goods.

In summary, the project has met its stated objectives by successfully building and testing a functional blockchain-based product authenticity verification system. It shows that blockchain has the capacity to fundamentally transform how authenticity and trust are managed in supply chains. While improvements and enhancements remain possible, the system developed represents an important step towards building more transparent, reliable, and secure verification frameworks that can be adapted across industries.

7.2 Recommendations

Although the project successfully achieved its objectives, there are several areas in which the system can be further improved and extended. One important direction for future work is scalability testing. The current prototype was deployed on a local Ganache blockchain for development and testing purposes, which provided a safe and efficient environment for proof-of-concept validation. However, to determine the system's effectiveness in real-world conditions, it would be valuable to migrate the implementation to a public test network such as Ethereum Sepolia or to a consortium blockchain. Such testing would provide insights into how the system performs under higher transaction volumes, with multiple distributed participants, and in environments where transaction costs and network delays may have practical implications.

Another recommendation concerns improving the overall user experience. While the current React-based web interface provides a functional and intuitive environment for both manufacturers and consumers, there is potential to enhance usability by extending the platform to mobile devices. Developing a mobile application would allow consumers to verify product authenticity more conveniently, particularly at the point of purchase. Furthermore, features such as multilingual support, simplified navigation, and accessibility options would broaden the reach of the system and make it more inclusive for a diverse range of users.

The system could also benefit from greater automation in product verification. At present, consumers rely on manually entering serial numbers to check product authenticity. In future versions, integrating identifiers such as barcodes, RFID tags, or NFC chips could enable automatic retrieval of product information. This would not only improve efficiency but also reduce the possibility of user error, making the verification process faster and more reliable.

From a security perspective, further enhancements could be done. Although role-based access control has been implemented to restrict registration and update functions to authorised users, additional mechanisms such as multi-factor authentication and periodic security audits could strengthen system integrity. In addition, exploring alternative or hybrid consensus mechanisms beyond those currently used in Ethereum development environments may improve both efficiency and resilience against emerging security threats.

Another area worth exploring is regulatory alignment and industry-specific adaptation. Since different industries, such as pharmaceuticals, food, and luxury goods, face unique challenges in combating counterfeit products, the system should be extended with compliance modules tailored to these sectors. For example, pharmaceutical products may require integration with Good Manufacturing Practice (GMP) standards, while agricultural supply chains might benefit from compliance with HACCP or food safety standards. Aligning the system with such regulations would not only strengthen its practical applicability but also encourage broader adoption by industry stakeholders.

In addition, while this project stored essential product information directly on the blockchain, it did not incorporate larger or more complex datasets such as product certificates, manufacturing documents, or images. For future iterations, integrating

decentralised file storage solutions such as the InterPlanetary File System (IPFS) would allow larger files to be stored securely and referenced via the blockchain. This would create a more comprehensive system that preserves both product metadata and supporting documentation in a tamper-proof and accessible manner.

In summary, while the current prototype demonstrates the feasibility of blockchain for product authenticity verification, these recommendations outline several pathways to strengthen, expand, and adapt the system for real-world deployment. Addressing scalability, usability, security, regulatory compliance, and integration with other technologies will ensure that the system can evolve into a robust and widely applicable solution capable of tackling the global challenge of counterfeit products.

REFERENCES

- [1] Synopsys, "What Is Blockchain and How Does It Work? | Synopsys," *www.synopsys.com*, 2022.
<https://www.synopsys.com/glossary/what-is-blockchain.html>
- [2] S. Nakamoto, "Bitcoin: a Peer-to-Peer Electronic Cash System," 2008.
Available: <https://bitcoin.org/bitcoin.pdf>
- [3] "NST Online, 'Haagen-Dazs vanilla ice cream products recalled over presence of carcinogen,' Aug. 09, 2022. [Online]. Available: <https://www.nst.com.my/news/nation/2022/08/821467/haagen-dazs-vanilla-ice-cream-products-recalled-over-presence-carcinogen>. [Accessed: Aug. 24, 2024]."
- [4] O. P. Choudhary, Priyanka, I. Singh, T. A. Mohammed, and A. J. Rodriguez-Morales, "Fake COVID-19 vaccines: scams hampering the vaccination drive in India and possibly other countries," *Human Vaccines & Immunotherapeutics*, vol. 17, no. 12, pp. 4117-4118, Aug. 2021, doi: 10.1080/21645515.2021.1960770.
- [5] "The Secret of Fake Sneakers Workshop Capital - Putian," China Sourcing Agent, Aug. 01, 2020. <https://mysourcify.com/the-secret-of-fake-sneakers-workshop-capital-putian>

REFERENCES

- [6] S. Singh, G. Choudhary, S. K. Shandilya, V. Sihag, and A. Choudhary, "Counterfeited Product Identification in a Supply Chain using Blockchain Technology," *Embedded Systems Engineering Department of Applied Mathematics and Computer Science, VIT Bhopal University, Sardar Patel University*, 2024. [Online]. Available: <https://orbit.dtu.dk/en/publications/counterfeited-product-identification-in-a-supply-chain-using-bloc>.
- [7] X. Yang, M. Li, H. Yu, M. Wang, D. Xu, and C. Sun, "A Trusted Blockchain-Based Traceability System for Fruit and Vegetable Agricultural Products," *IEEE Access*, vol. 9, pp. 36282-36293, 2021, doi: 10.1109/ACCESS.2021.3062845.
- [8] N. Koblitz and A. Menezes, "Elliptic curve cryptography: The serpentine course of a paradigm shift," *Journal of Mathematical Cryptology*, vol. 9, no. 1, pp. 1–15, 2015.
- [9] M. Castro and B. Liskov, "Practical Byzantine Fault Tolerance," in *Proc. 3rd USENIX Symposium on Operating Systems Design and Implementation (OSDI)*, 1999, pp. 173-186.
- [10] P. Zhu, J. Hu, Y. Zhang, and X. Li, "A Blockchain Based Solution for Medication Anti-Counterfeiting and Traceability," *IEEE Access*, vol. 8, pp. 161515-161525, 2020, doi: 10.1109/ACCESS.2020.3014839.

REFERENCES


- [11] X. Gao, W. Zhang, B. Zhao, J. Zhang, J. Wang, and Y. Gao, "Product Authentication Technology Integrating Blockchain and Traceability Structure," **Electronics**, vol. 11, no. 20, pp. 3314, Oct. 2022, doi: 10.3390/electronics11203314.

- [12] J. Benet, "IPFS - Content Addressed, Versioned, P2P File System," arXiv preprint arXiv:1407.3561, 2014

- [13] Hyperledger Foundation, "An Introduction to Hyperledger," Linux Foundation, 2020. [Online]. Available: <https://www.hyperledger.org>.

- [14] T.-M. Choi, "Blockchain-technology-supported platforms for diamond authentication and certification in luxury supply chains," *Transportation Research Part E: Logistics and Transportation Review*, vol. 128, pp. 17–29, Aug. 2019, doi: <https://doi.org/10.1016/j.tre.2019.05.011>.


Poster



UTAR
UNIVERSITI TUNKU ABDUL RAHMAN

FACULTY OF INFORMATION
COMMUNICATIONS AND TECHNOLOGY

BLOCKCHAIN-BASED PRODUCT AUTHENTICITY VERIFICATION



INTRODUCTION

A blockchain-powered system to fight counterfeit products by securing product data immutably, ensuring transparency and trust from production to delivery.

OBJECTIVE


To provide a secure, transparent, and user-friendly platform for product verification and supply chain integrity.


DISCUSSION

- Full traceability from manufacturer to end-user.
- Harder for fake products to enter supply chain.
- Easy verification via mobile/web.
- Protects consumers and businesses.

CONCLUSION

- Blockchain technology ensures product authenticity.
- Enhances trust, security, and transparency.
- Flexible and scalable across industries.





Project Developer: Yeu Qi Rui

Project Supervisor: Ts Dr Gan Ming Lee