APPLIED INFORMATION TECHNOLOGY AND COMPUTER SCIENCE
e-ISSN: 2773-5141

AITCS

Vol. 7 No. 1 (2026) p. 1125-1146
https://publisher.uthm.edu.my/periodicals/index.php/aitcs

Blockchain-Powered Electronic Product Authenticity Verification for Global Rigorous Aura Success (M) Sdn. Bhd.

Darven Danraj a/l Kathirveloo[1], Sofia Najwa Ramli[1]*

[1] *Fakulti Sains Komputer dan Teknologi Maklumat,*
Universiti Tun Hussein Onn Malaysia, Parit Raja, Batu Pahat, 86400, MALAYSIA

*Corresponding Author: sofianajwa@uthm.edu.my
DOI: https://doi.org/10.30880/aitcs.2026.07.01.061

Article Info

Received: 3 August 2025
Accepted: 18 June 2026
Available online: 30 June 2026

Keywords

Blockchain Technology, Ethereum, Supply chain, Product Verification, Smart Contract

Abstract

Counterfeit electronic products lead to significant financial losses, erode consumer trust, and severely damage the reputations of companies worldwide. Traditional authentication methods such as QR codes, RFID tags, and AI-driven systems fail to combat sophisticated counterfeiting techniques due to vulnerabilities such as cloning, high computational demands, and limited scalability. In response, this report proposes an innovative blockchain-based solution on the Ethereum network, designed for secure and reliable product verification across complex supply chains. The system utilizes tamper-proof smart contracts, secure QR code documentation, and user engagement features, implemented with tools like Solidity, Hardhat, React, Node.js, and ethers.js. Originally developed for Global Rigorous Aura Success (M) Sdn Bhd, this adaptable solution can also be applied to industries such as pharmaceuticals, luxury goods, and automotive parts. The implementation of the system resulted in a marked reduction in counterfeit products, enhanced consumer trust through real-time product authentication, and improved operational efficiency by enabling automated and paperless record management. This solution not only addresses current counterfeiting issues but also paves the way for future innovations in product authentication and supply chain management.

1. Introduction

Counterfeit products pose a significant challenge in the global market, particularly within the electronics sector, where they can severely damage brand reputation and consumer trust. Global Rigorous Aura Success (M) Sdn Bhd, an electronic product-based company, recognizes the urgent need to ensure that its customers receive authentic products. To combat this issue, the company plans to implement a blockchain-powered electronic product authenticity verification system. This innovative approach aims to leverage the strengths of blockchain technology by its nature of decentralized and immutable ledger to provide a secure and transparent method for b tracking the authenticity of electronic goods throughout their supply chain, from manufacturing to delivery to customers.

The current methods for verifying product authenticity, such as QR codes and AI-based systems, have notable limitations. QR codes can be easily replicated, while AI solutions often require extensive computational resources, making them impractical for widespread use. In contrast, the proposed blockchain solution offers a tamper-proof mechanism that records every transaction and movement of a product within the supply chain. Stakeholders that stakeholders can verify the legitimacy of products with confidence, thereby enhancing brand integrity and

customer satisfaction. The project is designed not only to protect Global Rigorous Aura Success (M) Sdn Bhd's reputation but also to foster long-term loyalty among its customer base by assuring them of product authenticity.

The objectives of this project include designing a blockchain platform that guarantees traceability and transparency of electronic products and testing the performance of the proposed system in terms of functional and security requirements through functionality testing, security testing, and user acceptance testing. By achieving these goals, the company aims to streamline its operations and reduce reliance on traditional record-keeping methods, ultimately leading to a more efficient and secure supply chain management process. The expected outcome is a significant reduction in counterfeit risks while promoting data integrity and trustworthiness across all transactions.

The significance of this blockchain-powered verification system extends beyond Global Rigorous Aura Success (M) Sdn Bhd; it has the potential to benefit various industries facing similar challenges with counterfeit goods. By establishing a reliable framework for product authentication, this initiative could enhance consumer trust across sectors such as luxury goods, pharmaceuticals, and automotive parts. While not a panacea for all counterfeiting issues, this project represents a meaningful step towards creating a more secure environment for product tracking and verification, ultimately benefiting businesses and communities alike. The remainder of this paper presents a review of related literature, outlines the development methodology, details the system analysis and implementation, and concludes with key findings and future work.

2. Literature Review

This section provides a literature review on topics related to product authenticity systems, definition, importance & challenges in supply chain management, current alternative approaches to combat product counterfeiting, overview of blockchain technology, characteristics of blockchain, smart contracts and study of existing system. The literature review aims to gain a comprehensive understanding and discuss the project background about the proposed system and blockchain.

2.1 Definition, Importance & Challenges in Supply Chain Management

Supply Chain Management (SCM) refers to the coordination of material, information, and financial flows among manufacturers, suppliers, transporters, warehouses, retailers, and customers [1]. Supply chain integration plays a vital role in managing and controlling these flows effectively [2]. However, the growing complexity of supply chains, combined with limited transparency, inadequate resources, and weak enforcement mechanisms, has made it increasingly difficult for manufacturers to verify the origin, identity, and authenticity of their products [3]. This has facilitated the entry of counterfeit goods, posing risks to product integrity, customer trust, and brand reputation. These challenges underscore the need for innovative approaches, such as blockchain-based authentication, to enhance product traceability, improve supply chain security, and mitigate the risks associated with counterfeiting.

2.2 Current Alternative Approaches to Combat Product Counterfeiting

Numerous anti-counterfeit solutions have been integrated into supply chains to combat counterfeit goods, including RFID, NFC, and QR codes, which provide product authentication but face limitations such as replication vulnerabilities, scalability challenges, and reliance on centralized architectures prone to single points of failure and service disruptions [4]. Recent advancements in AI, particularly Convolutional Neural Networks (CNNs), have enhanced security by enabling detailed analysis of product labels and packaging, but these methods alone are insufficient. Blockchain and Distributed Ledger Technology (DLT) offer decentralized, transparent, and immutable frameworks that address these limitations, enabling real-time tracking, reduced fraud, and increased reliability, making them critical enablers for secure, future-ready supply chains. The benefits and limitations of centralized and decentralized systems, as compared in Table 1, further highlights a comparison between centralized and decentralized networks across several key aspects. Centralized networks are managed by a single central server, making them vulnerable to data modification, single points of failure, and limited scalability. In contrast, decentralized networks operate through independent nodes, offering immutable data integrity, distributed control, and higher resilience. Verification in centralized systems is handled by a central authority, whereas decentralized systems rely on cryptographic methods, ensuring trustless and secure validation. Additionally, while centralized systems may require cryptographic authentication to be added separately, decentralized systems have it inherently integrated. This comparison highlights the advantages of decentralized frameworks like blockchain in enhancing security, scalability, and reliability, which are particularly valuable in securing complex systems such as supply chains.

Table 1 *Comparison between Centralized Network & Decentralized Network[5]*

Aspect	Centralized Network	Decentralized Network
Definition	Managed by a single central server	Operates via independent nodes
Data Integrity	Can be modified	Immutable
Control	Single point of control	Distributed, autonomous node operation
Single Point of Failure	High risk	Low risk
Scalability	Limited scalability	Highly scalable
Verification	Handled by the server	Ensured through cryptographic methods
Authentication	Cryptography added separately	Cryptographic authentication integrated.

2.3 Authentication Technique

The comparative analysis of existing anti-counterfeit platforms reveals varying degrees of effectiveness driven by the underlying technologies such as QR codes, NFC, AI, and blockchain each presenting distinct trade-offs in scalability, security, and cost. Firstly, QR codes, while cost-effective and easily deployable, remain highly susceptible to duplication and tampering due to their static nature and reliance on centralized verification backends. Their lack of cryptographic resilience undermines long-term integrity in open supply chain environments. NFC technologies offer enhanced hardware-based security through encrypted, proximity-based authentication. However, their higher production costs and dependency on specialized reading devices limit large-scale deployment, particularly in resource-constrained settings.AI-driven verification, employed in select platforms, introduces intelligent anomaly detection and pattern recognition capabilities. While promising in precision, such systems are typically centralized, resource-intensive, and opaque in their decision-making, raising concerns around interpretability, operational overhead, and systemic vulnerabilities. Blockchain-based architectures, as proposed in the Verifai system, provide a paradigm shift by enabling decentralized, tamper-evident data recording and smart contract automation. This eliminates single points of failure and ensures traceability across all transactional nodes. The immutable ledger, coupled with distributed consensus, enhances trust, reduces reliance on intermediaries, and supports scalability at reduced marginal costs. In contrast to traditional frameworks constrained by centralization and escalating costs, blockchain offers a resilient, transparent, and scalable infrastructure, better aligned with the dynamic requirements of modern anti-counterfeit strategies across complex global supply chains.

2.4 Characteristics of Blockchain Technology

Its key features such as immutability, decentralization, transparency, pseudonymity, and chronology—align perfectly with the requirements for electronic product authenticity verification. Immutability ensures that authentication records cannot be altered or deleted, preserving data integrity. Decentralization eliminates single points of failure, enhancing system resilience. Transparency provides a tamper-resistant record accessible to all stakeholders, fostering trust across the supply chain. Pseudonymity safeguards user privacy while enabling secure participation, and chronology ensures that all authentication records are traceable and verifiable. These characteristics collectively strengthen product authenticity and customer confidence, making blockchain an ideal framework for Global Rigorous Aura Success (M) Sdn Bhd's verification system [6].

2.5 Overview of Blockchain Technology

Blockchain technology, introduced by Nakamoto Satoshi in 2008, operates as a distributed ledger where consecutive blocks are linked through cryptographic hashes. Each block includes a timestamp and transaction data, ensuring secure, traceable, and immutable records [7][8]. The timestamp is valid only if it surpasses the median timestamp of the prior eleven blocks and the network-adjusted time, safeguarding against manipulation. The network relies on a consensus protocol among nodes for generating and validating new blocks, reinforcing its decentralized and secure nature [9]. This makes blockchain a robust solution for data integrity and security challenges.

2.6 Smart Contract

Smart contracts are self-executing programs with tamper-resistant properties that enhance security by automating processes and minimizing the need for trusted third parties [10]. Proposed by Nick Szabo in 1994, smart contracts aim to secure systems against malicious attacks and automate transactions through predefined conditions, with state variables changing based on executed functions [11][12]. When integrated with blockchain, smart contracts facilitate secure, peer-to-peer transactions in a tamper-proof, publicly trusted manner [13]. After

being signed by relevant parties, they are added to the blockchain, validated by nodes, and executed automatically when trigger conditions are met [14]. Smart contracts are central to Blockchain 2.0 and 3.0, supporting decentralized systems like DAOs, DACs, and DAS [15]. For Global Rigorous Aura Success (M) Sdn Bhd, smart contracts automate product authentication and supply chain verification, ensuring transparency, security, and fraud minimization within the decentralized framework, offering a secure, automated alternative to traditional verification processes.

2.7 Study of existing system

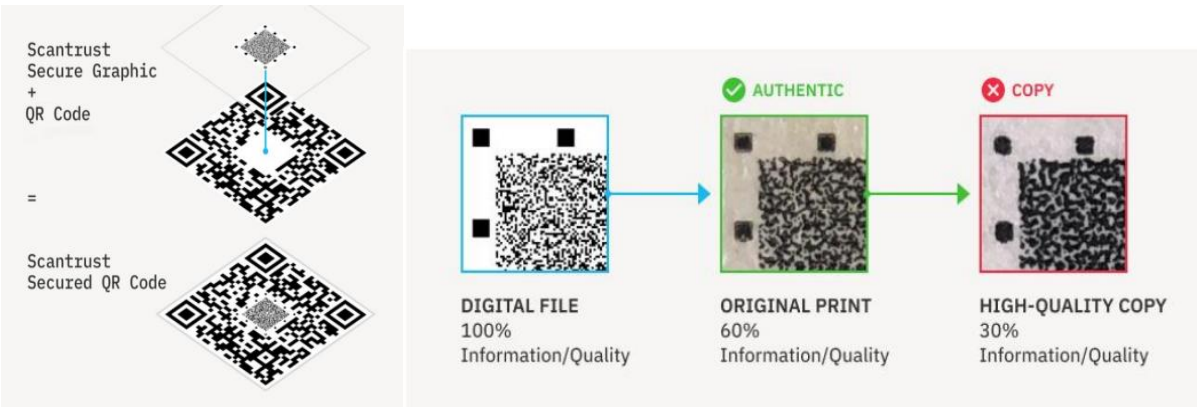
To develop a blockchain-powered electronic product authenticity verification system, it is essential to analyze existing market solutions addressing counterfeiting. This section reviews current anti-counterfeiting systems across various sectors to understand prevailing methodologies. While numerous studies have proposed blockchain-based authentication models, few have progressed to working prototypes, as blockchain remains an emerging technology in this domain.

2.7.1 Authentic Hyde

Authentic Hyde is an anti-counterfeit product identification system that uses QR code technology for real-time product authentication. However, it lacks blockchain integration, making it vulnerable to attacks as it relies on a central server for data integrity. While cost-effective for small configurations, the system's price increases with scale due to cloud storage and maintenance costs. Centralization creates a single point of failure, posing a security risk.

2.7.2 Scantrust

Scantrust offers an anti-counterfeit solution using QR codes with a copy detection pattern, which embeds a security image that loses key information when copied, signaling counterfeit products. However, the system's reliance on cloud infrastructure increases operational costs, particularly for large-scale deployments. The lack of decentralization makes the central database vulnerable to compromise, affecting the system's credibility. Additionally, the reliance on detecting information loss can lead to inaccuracies, and QR code verification at scale requires significant computational resources, making it less accessible for smaller businesses. Furthermore, printed QR codes are prone to issues like fading and UV damage. Fig. 1 shows the QR technology being used by scantrust and the verification method being implemented [16].



(a) **(b)**

Fig. 1 *QR technology used by Scantrust [16]: (a) QR Code design technology being used; (b) Verification process.*

2.7.3 EON Brand protection

EON Brand Protection offers a multi-layered authentication system using high-security identifiers like QR codes and NFC chips, along with AI-powered intelligence for product verification. While effective in protecting against counterfeiting and fraud, its reliance on centralized systems raises concerns about integrity and transparency. The complexity and cost of using NFC tags and AI for counterfeit detection may be impractical for smaller businesses, and without a distributed ledger, the risk of tampering with authentication records remains. Additionally, the system's need for significant computational resources can limit accessibility for growing brands [17][18].

2.8 Comparison of Existing System

Table 2 compares existing anti-counterfeit systems and underscores the need for a more robust solution, such as the proposed blockchain-based approach. While platforms like Authentic Hyde, Scantrust, DigiTrax by Authentix, and EON Brand Protection offer essential counterfeit detection features, they lack comprehensive supply chain coverage. Their reliance on centralized infrastructures leads to issues such as limited scalability, single points of failure, and high operational costs. Although some integrate secure QR codes, AI, and proprietary algorithms, they often lack decentralized verification and immutable data storage, leaving them vulnerable to manipulation and inefficiencies. Moreover, their cost-intensive nature limits accessibility for SMEs. In contrast, the proposed blockchain solution leverages smart contracts, distributed validation, and an immutable ledger to enhance security, transparency, and scalability at reduced cost. This decentralized architecture addresses the key limitations of current systems, offering a more reliable and scalable framework for end-to-end product authentication.

Table 2 *Comparison of Existing System*

Feature	Authentic Hyde	Scantrust	EON Brand Protection	Verify (Proposed System)
Technology Used	Secure QR Codes, Cloud-Base, No blockchain	Secure QR Codes, Connected Packaging No blockchain	High-Security, Identifiers, AI-Powered, No blockchain	QR Codes, Smart contract, Blockchain-Integration
Drawbacks	Centralized system, scalability issues	Centralized database, high costs	High costs, High computational power	Decentralized system, Lower cost, easily scalable
Vulnerability	Centralized database, single points of failure	Data tampering risk, no immutable ledger	Lacks decentralized verification	Authentication method

3. Methodology

This section outlines the adopted methodology, focusing on the selected software development model. It provides detailed insights into each phase of the chosen model, presented in a structured and systematic manner. The selected model is the Object-Oriented Software Development (OOSD) Model, from Fig. 2, which uses a structured and formal approach that integrates Object-Oriented Programming (OOP) principles throughout the software development life cycle [19].

Fig. 2 illustrates the lifecycle of the Object-Oriented Software Development Model, a structured methodology that followed for the system by following Object-Oriented Programming (OOP) principles throughout the software development life cycle. This model emphasizes modularity, code reusability, maintainability, scalability, and real-world modeling, aligning seamlessly with the system's requirements. By representing key entities such as products, manufacturers, distributors, customers, and blockchain-related data, this approach ensures flexibility and long-term system viability. The iterative nature of OOP facilitates continuous refinement, while tools like Unified Modeling Language (UML) provide a clear architectural blueprint during analysis and design phases, resulting in a robust, scalable solution.

Object-oriented requirement gathering is a phase that involves gathering some key entities and functionalities for the Blockchain-Powered Electronic Product Authenticity Verification system involved a verbal interview with the chairman of Global Rigorous Aura Success (M) Sdn Bhd to analyze the current verification processes and define the core functionalities of the system. The primary objective was to understand how to verify product authenticity effectively and establish clear requirements that align with business expectations. The system was developed using a JavaScript-based technology stack, with ReactJS as the frontend framework. The hardware requirements for successful implementation include a processor such as an Intel Core i5 or AMD Ryzen 5, at least 8 GB of RAM, and 256 GB of solid-state storage (SSD). In terms of software, the development environment includes Visual Studio Code, which serves as the integrated development environment (IDE) for writing, editing, and debugging code. Solidity was used as the smart contract programming language, while Hardhat was employed as the framework to test, deploy, and manage smart contracts. The frontend was developed using React, and Node.js was used as the runtime environment for backend development and API management. Ethers.js served as the blockchain library for interacting with the Ethereum blockchain. PostgreSQL was used for managing the system's database. Additionally, design tools such as Canva and Lucidchart were utilized to create flowcharts and UML diagrams during the design phase

Object-Oriented Analysis (OOA) focuses on understanding system requirements from both technical and user perspectives. For the Blockchain-Powered Verification System, this involved analysing integration with verification sources and ensuring authenticity standards. UML diagrams were used to model system behaviour use case diagrams illustrated user interactions, sequence diagrams mapped message flows, activity diagrams showed workflow logic, and class diagrams defined entities and relationships ensuring accurate representation of the system.

The Object-Oriented Design (OOD) phase for Global Rigorous Aura Success (M) Sdn Bhd focused on building the application's architecture, user interfaces, and database. User-friendly UI designed to ensure intuitive navigation, while a normalized database was developed to maintain data consistency, eliminate redundancy, and enhance retrieval efficiency. A data dictionary was created to document system entities, attributes, and metadata, ensuring clarity and robust data management.

The implementation phase is where blueprints and developed prototypes are transformed into functional code. The system was built using core object-oriented principles such as encapsulation, inheritance, and polymorphism. JavaScript and ReactJS were used for the frontend, while Solidity was employed to develop smart contracts. JSX enabled the creation of modular UI components, and Hardhat supported contract testing and deployment. All elements were integrated using Visual Studio Code, resulting in a cohesive, web-based verification system.

The testing phase employed a systematic approach to ensure system reliability and performance. Unit tests validated individual methods, integration testing ensured seamless module interaction, and acceptance testing confirmed the system met user requirements. User Acceptance Testing (UAT) verified functionality and user-friendliness, delivering a robust solution tailored to the needs of Global Rigorous Aura Success (M) Sdn Bhd.

Post-deployment, is important for Product Authenticity Verification System as ongoing support ensures system stability and flexibility through regular maintenance, including bug fixes, performance updates, and security enhancements. A robust backup and recovery system safeguards blockchain and database integrity, while regular monitoring addresses potential issues promptly. Periodic updates to encryption and Multi-Factor Authentication (MFA) ensure continued security and reliability.

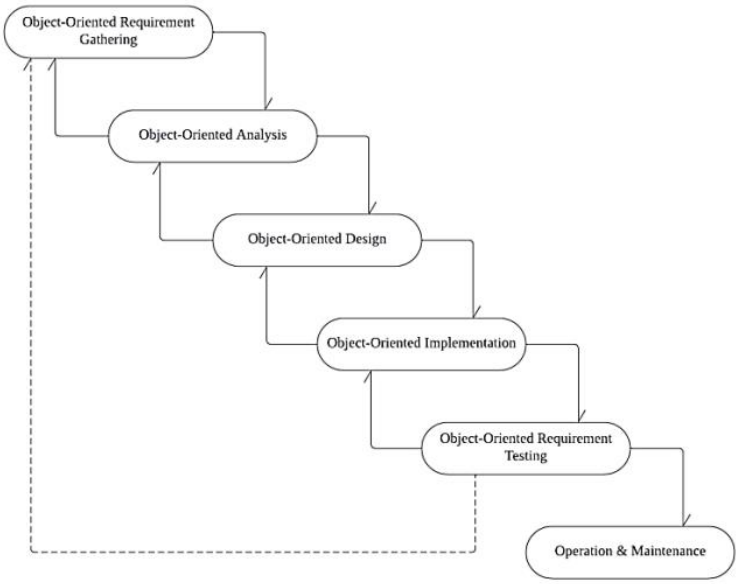


Fig 2 *Cycle of Object-Oriented Software Development [19]*

4. System Analysis and Design

This section examines the systematic analysis and design of the Product Verification System, aiming to define the system requirements and create a structured design that integrates blockchain technology.

4.1 System Development Workflow

Table 3 outlines the object-oriented system development workflow, covering phases from requirements to maintenance. Each stage ranging from UML modeling and wireframing to implementation and testing contributed essential outputs, ensuring a reliable, user-centric, and secure solution.

Applied Information Technology and Computer Science Vol. 7 No. 1 (2026) p. 1125-1146

Table 3 *System Development Workflow*

Phases	Task	Output
Object-Oriented Requirement	Sent interview form	Collected insights from chairperson
Object-Oriented Analysis	Created class diagram, sequence diagram, activity diagram, system flow	Comprehensive blueprint and clear understanding achieved
Object-Oriented Design	Created wireframes, system architecture	Visualize designs, clear implementation route
Object-Oriented Implementation	Developed backend, frontend, smart contract	User friendly functional interface
Object-Oriented Requirement Testing	Conduct unit testing for seamless interaction of modules	Verified system compliance
Operation and Maintenance	Regularly analyze & resolve bugs	Reliable secure system performance

4.2 System Requirements

System requirements provide a comprehensive outline of the necessary functionalities a system must have to satisfy user expectations. Table 4 lists six functional requirements, Table 5 presents five categories of non-functional requirements.

Table 4 *Functional requirements*

Function	Requirements	User
Login	The system should allow users to login using their respective account.	Admin, Manufacturer, Supplier, Retailer, Customer
Register account	The system should allow admin to add account / manage account	Admin
Wallet interaction	The system should allow users with privilege of admin, manufacturer, retailer, to interact with MetaMask wallet	Admin, Manufacturer, Supplier, Retailer
Generate QR	The system should generate QR code upon Manufacturer add in product details.	Manufacturer
Scan QR	The system must enable users to scan a QR code previously generated by the manufacturer, which will then update the blockchain transaction history appropriately.	Retailer, Supplier, Customer
View Transaction	The system must provide customers with access to the transaction history and display whether the product is authentic or potentially counterfeit.	Customer

Table 5 *Non-Functional requirements*

Property	Measure
Security	The system needs to ensure security by preserving the integrity of the product at any cost.
Speed	The system should be able to process functions faster for normal accessibility
Ease of use	The user able to see details and operate easier as it is designed to be friendly
Reliability	The system must be accessible 24/7 without service disruptions, ensuring uninterrupted access for Global Rigorous Aura Success (M) Sdn Bhd clients. The transaction history feature should maintain information integrity, building trust in product authenticity.
Usability	The system is accessible from anywhere, but only admins/employees with privileges can use admin features.

4.3 System Analysis

This section presents the system analysis through various diagrams, including the Use Case Diagram, Sequence Diagram, Activity Diagram, and Class Diagram, to model system components, interactions, and workflows.

4.3.1 Use Case Diagram

The use-case diagrams in Fig. 3 illustrate the interactions of various actors with the Blockchain-Powered Electronic Product Authenticity Verification System. The admin manages accounts, able to view system dashboard, maintains system security by login and able to register new manufacturer/supplier/retailer. Manufacturers log in, link their MetaMask wallets, input product details, generate unique QR codes, and monitor dashboards. Retailers authenticate products by scanning QR codes, updating product details, and ensuring data accuracy and being able to check their profile. Suppliers verify and authenticate products by updating details and manage transactions through blockchain-integrated features. Meanwhile, customer able to scan the QR of the product and view transaction history along with product authenticity results. These interactions streamline product verification and ensure transparency across the supply chain.

Fig. 3 *Use Case Diagram*

4.3.2 Sequence Diagram

The sequence diagrams, from Fig. 4 to 8 illustrate step-by-step interactions in the Blockchain-Powered Electronic Product Authenticity Verification System for various user roles. The admin manages user accounts through authentication, account registration, and updates via the user interface. The manufacturer logs in, connects to a MetaMask wallet, inputs product specifications, and generates blockchain-linked QR codes for product tracking. Suppliers and retailers authenticate via MetaMask wallets, scan QR codes to verify and update product details, and ensure accurate blockchain records, with retailers updating product status upon receipt or sale. Customers scan QR codes to verify product authenticity and access blockchain transaction history, ensuring transparency and trust.

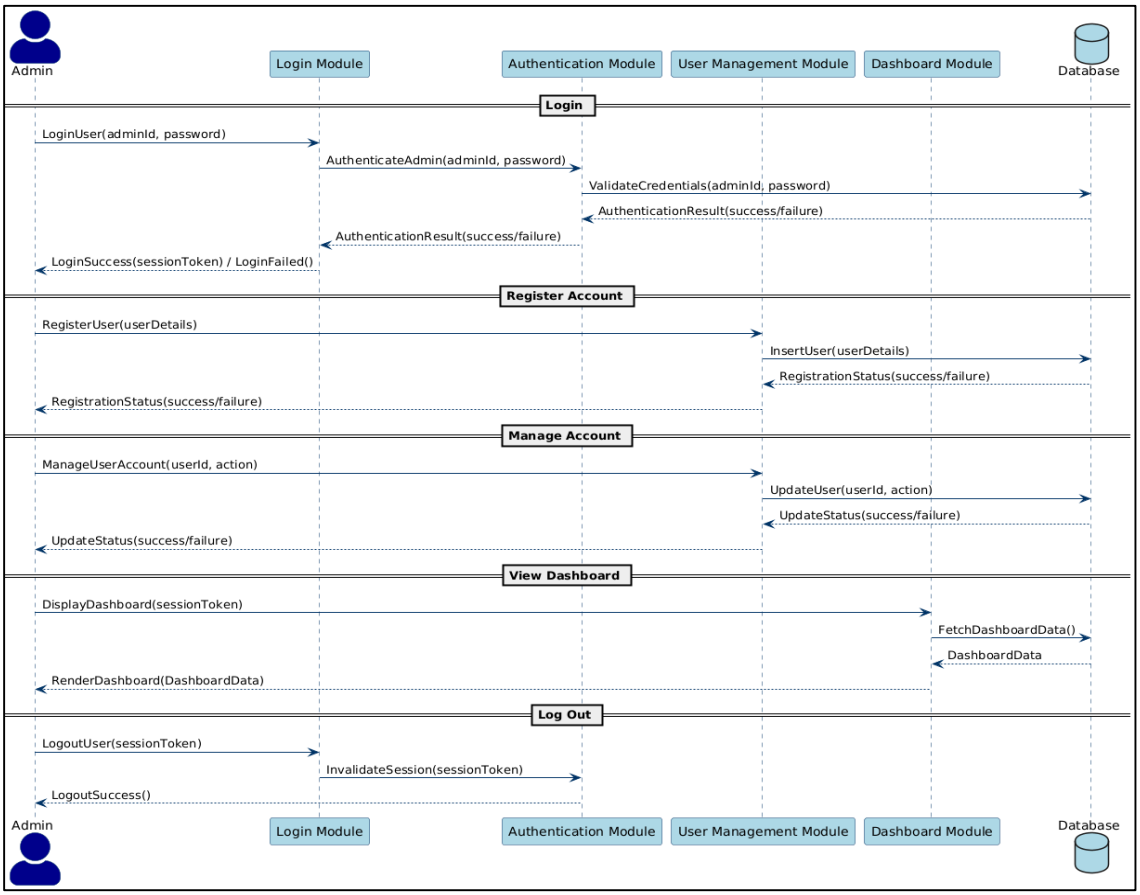


Fig. 4 *Sequence Diagram for Admin*

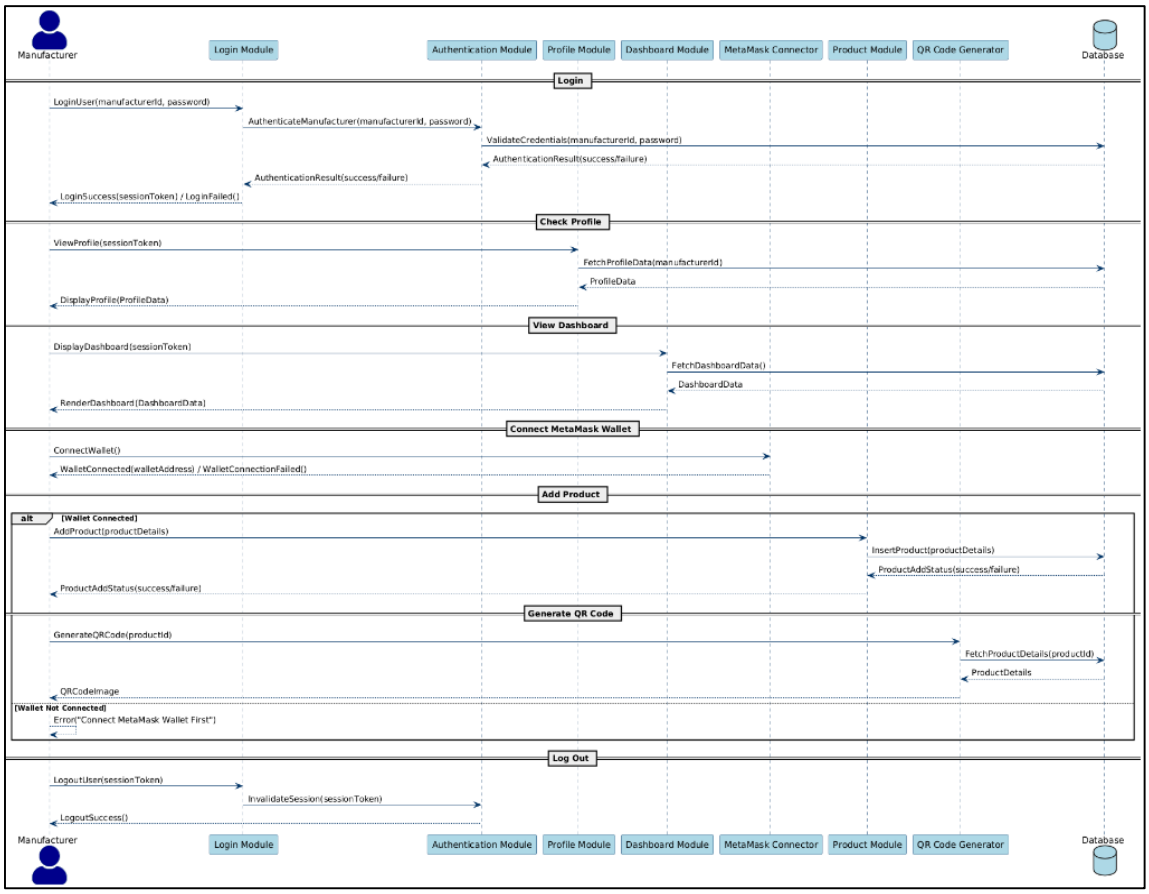


Fig. 5 *Sequence Diagram for Manufacturer*

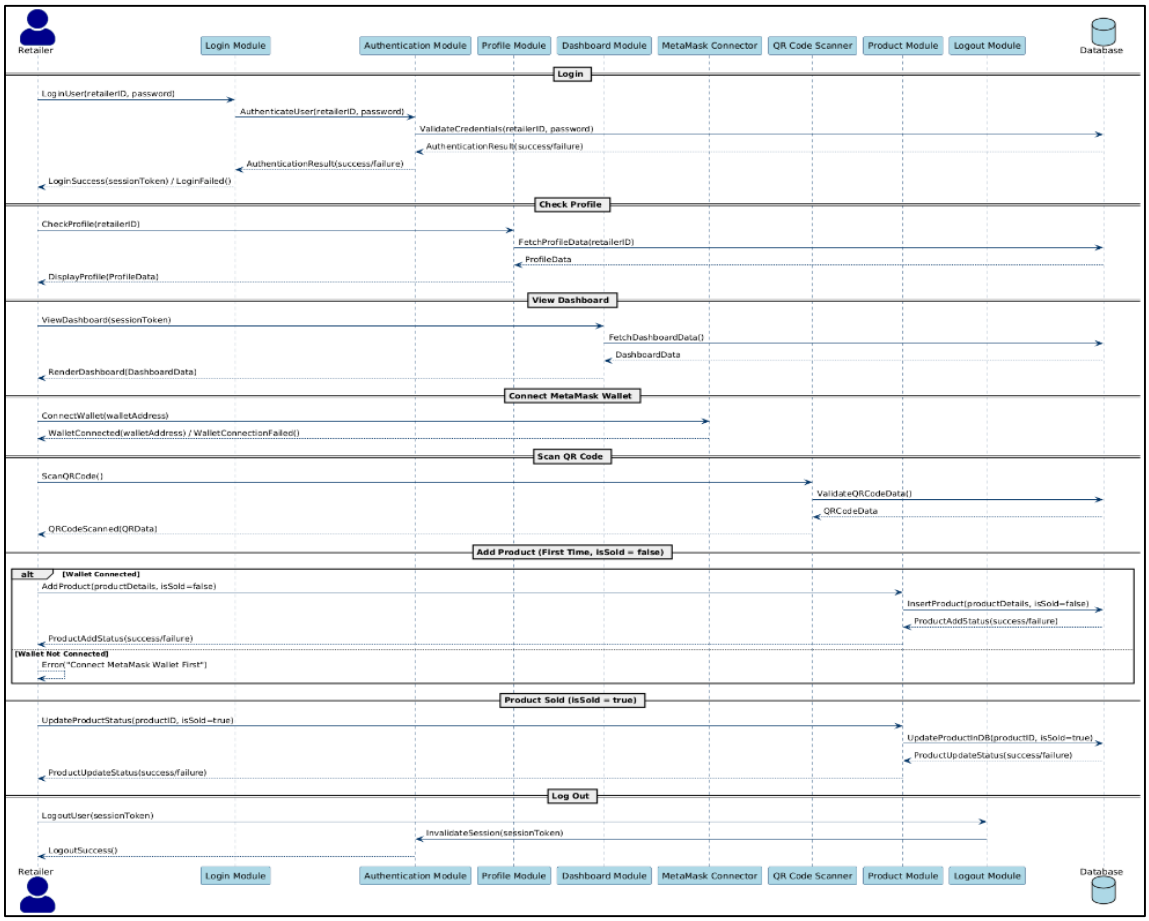


Fig. 6 *Sequence Diagram for Retailer*

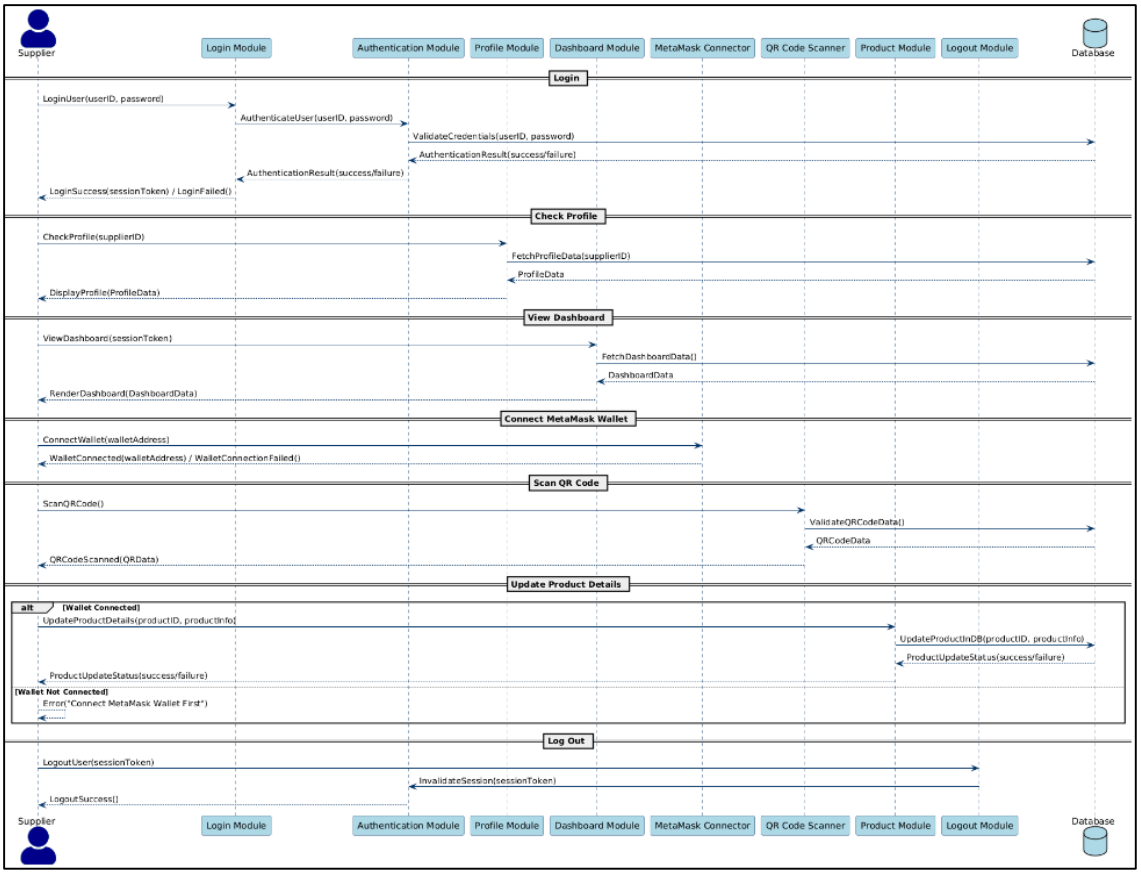


Fig. 7 *Sequence Diagram for Supplier*

Applied Information Technology and Computer Science Vol. 7 No. 1 (2026) p. 1125-1146

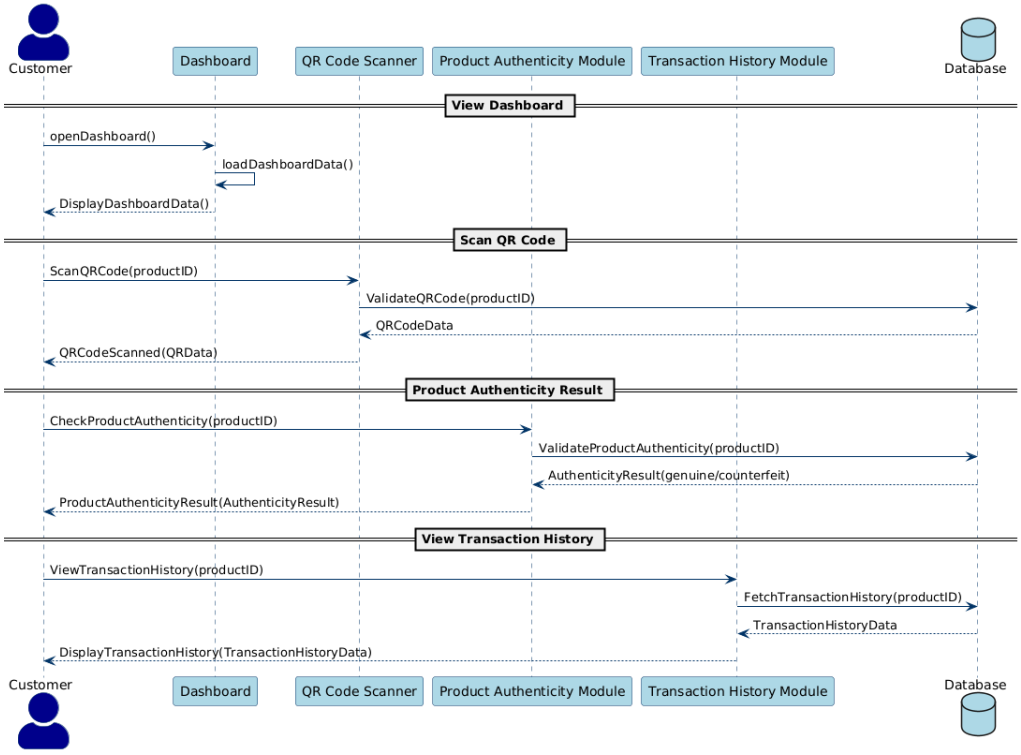
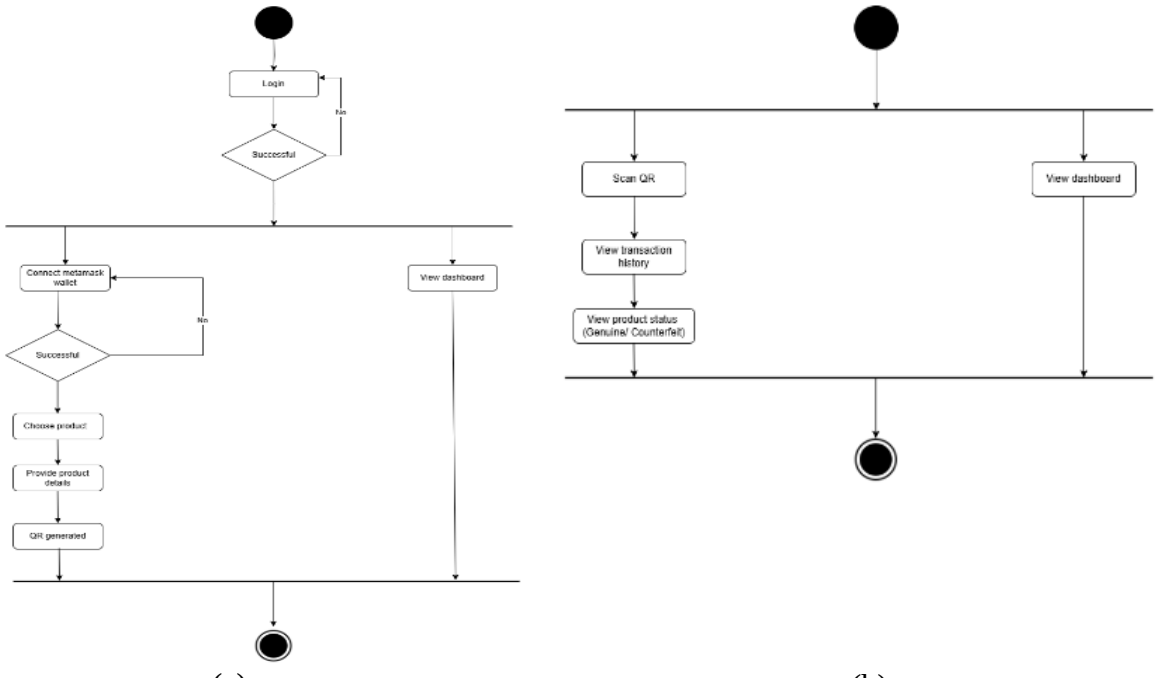


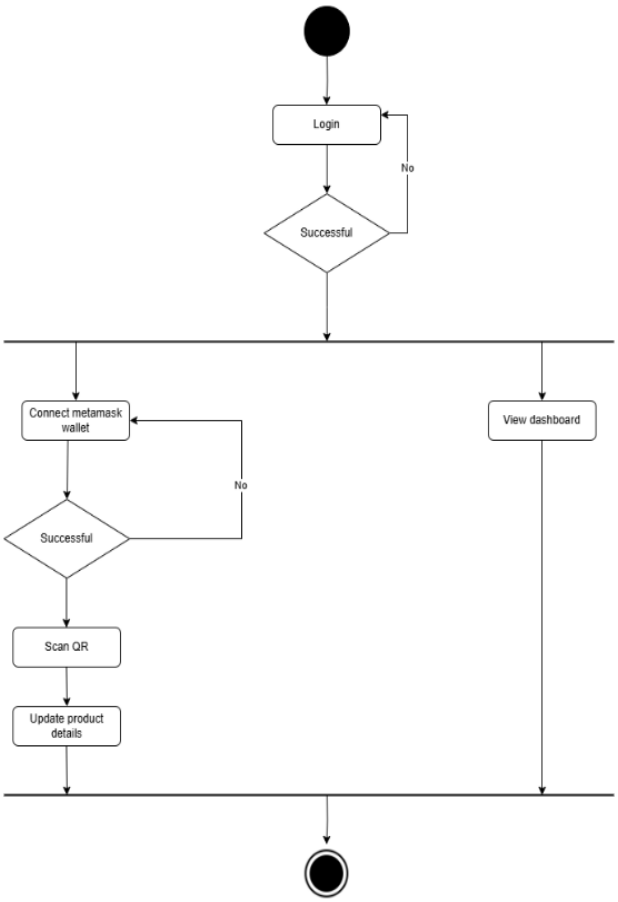
Fig. 8 *Sequence Diagram for Customer*

4.3.3 Activity Diagram

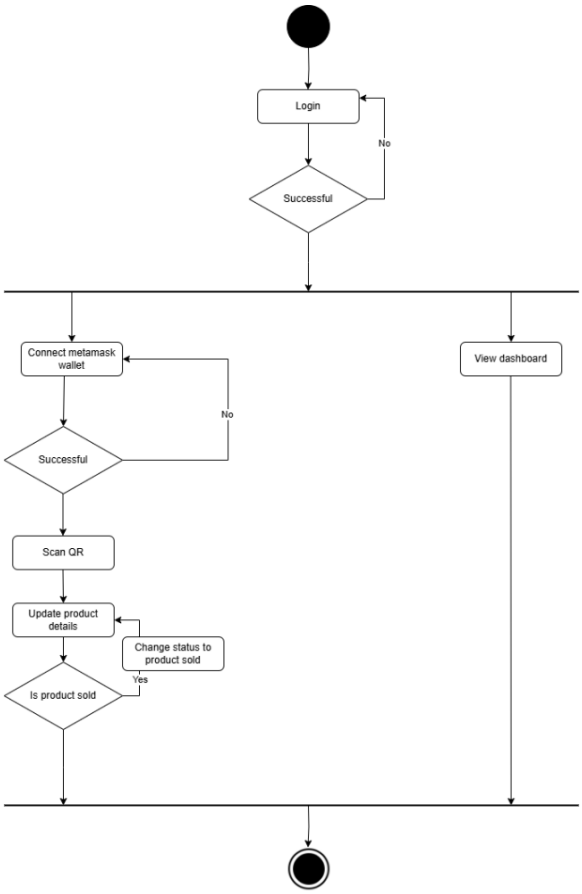
The sequence diagrams from Fig. 9 illustrate step-by-step interactions in the Blockchain-Powered Electronic Product Authenticity Verification System for various user roles. The admin manages user accounts through authentication, account registration, and updates via the user interface. The manufacturer logs in, connects to a MetaMask wallet, inputs product specifications, and generates blockchain-linked QR codes for product tracking. Suppliers and retailers authenticate via MetaMask wallets, scan QR codes to verify and update product details, and ensure accurate blockchain records, with retailers updating product status upon receipt or sale. Customers scan QR codes to verify product authenticity and access blockchain transaction history, ensuring transparency and trust.



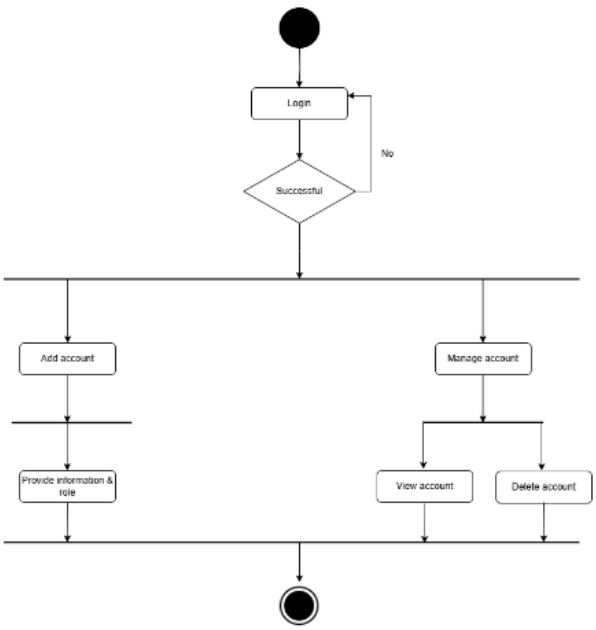
(a) (b)



(c)



(d)



(e)

Fig. 9 *Activity Diagram (a) Manufacturer; (b) Customer; (c) Supplier; (d) Retailer; (e) Administration*

4.3.4 Class Diagram

Referring to Fig. 10, the class diagram depicts key system classes, including Product, Manufacturer, Consumer, Transaction, Smart Contract, and Blockchain, along with their attributes and relationships.

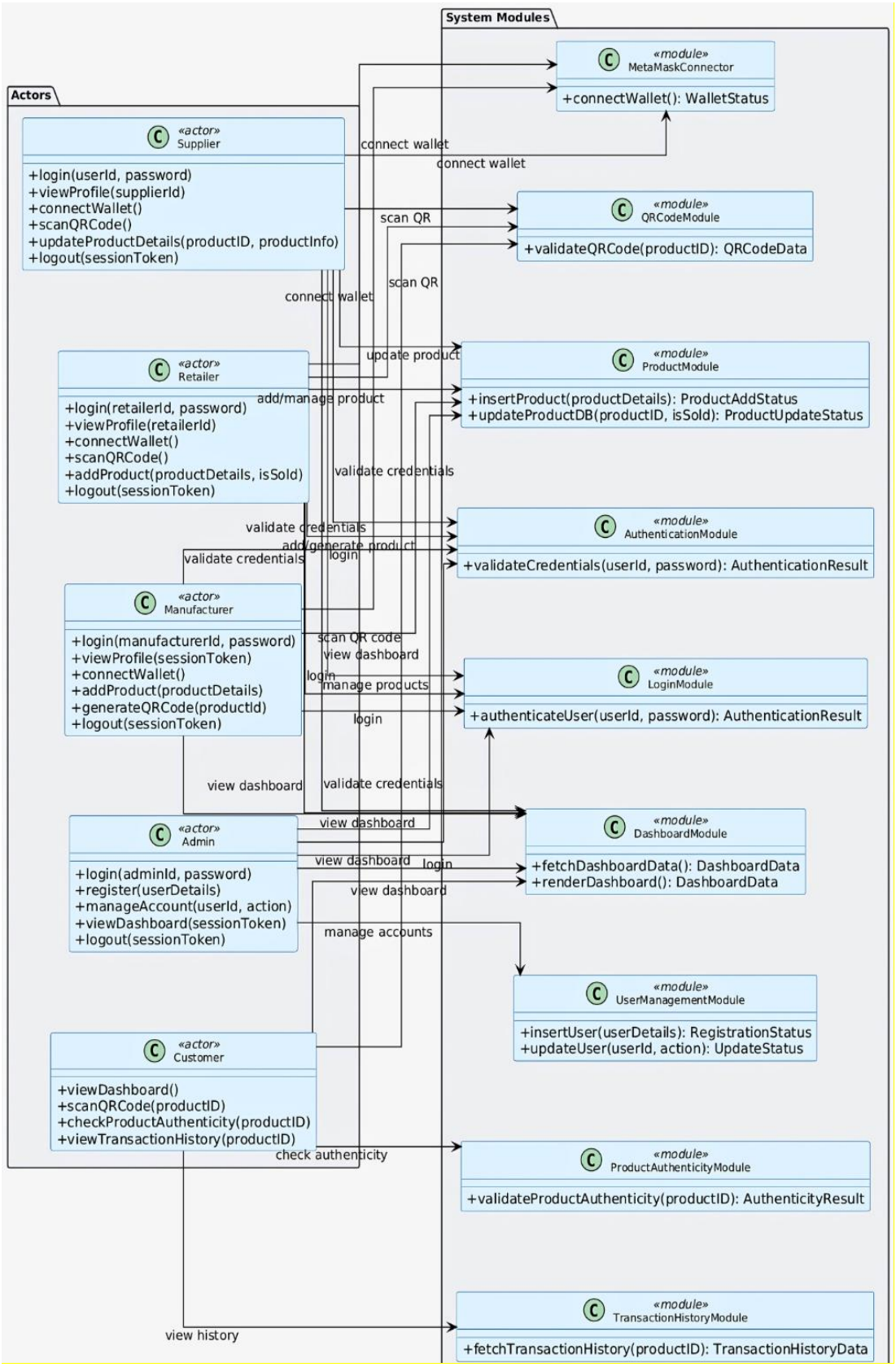


Fig. 10 *Class Diagram for Product Verification System*

4.4 System Design

The system design encompasses the system architecture, relational schema, and user interface design, providing a comprehensive framework for the development and functionality of the Product Verification System.

4.4.1 System Architecture

The system architecture of the Product Verification System, highlighting its components and overall structure. It serves as a visual representation of the system's structure and functionality, as shown in Fig. 11. The proposed system uses blockchain for product authentication and supply chain management, involving five key roles: Administrator, Manufacturer, Supplier, Retailer, and Customer. Each actor has a secure user account managed by the Administrator and must successfully connect to MetaMask wallet to perform any action except for customers. Manufacturers upload product details and generate a unique QR code linked to a blockchain record for authenticity. Suppliers and retailers scan the QR code to verify authenticity and update product status on the blockchain. Customers scan the QR code to retrieve product details and transaction history, with the system flagging counterfeit products. The blockchain ensures transparency, secure communication, and end-to-end traceability, preventing counterfeits and building customer trust.

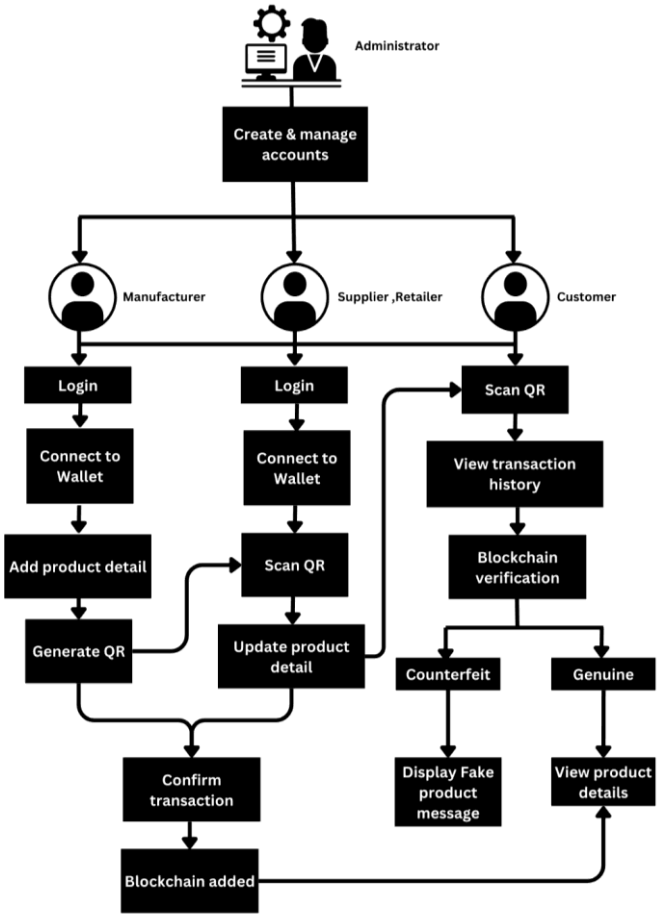


Fig. 11: *System Architecture of Blockchain-Powered Electronic Product Authenticity Verification*

4.4.2 Data Dictionary

The following data dictionaries define the attributes, data types, and constraints for key tables within the system, including authentication, profile, and product information. Authentication Schema from Table 6 manages user credentials and access control. It defines username and id as a composite primary key, ensuring unique identification, while password and role enforce access constraints. The unique index on username_id further optimizes secure user lookups. Profile Schema from Table 7 stores extended user information such as name, description, location, and website, linked via the username attribute. The id serves as the primary key, supporting distinct profile identification for system operations. Product Schema from Table 8 captures product-specific metadata, including name, brand, and a system-generated serialnumber as the primary key. The use of DEFAULT nextval ensures automatic serial number assignment, enabling consistent product traceability.

Table 6 *Data Dictionary of Authentication*

Attribute Name	Data Type	Constraint
username	varchar (50)	NOT NULL, PRIMARY KEY (id,username)
password	varchar (50)	NOT NULL
id	serial	PRIMARY KEY (id,username)
role	varchar (50)	NOT NULL Unique Index: username_id

Table 7 *Data Dictionary of profile*

Attribute Name	Data Type	Constraint
name	varchar (50)	
description	varchar (500)	
username	varchar (50)	NOT NULL

Table 7 *Data Dictionary of profile (cont)*

Attribute Name	Data Type	Constraint
website	varchar	
location	varchar (50)	
image	varchar (50)	
role	varchar (50)	
id	serial	PRIMARY KEY

Table 8 *Data Dictionary of product*

Attribute Name	Data Type	Constraint
name	varchar (50)	
serialnumber	varchar (50)	DEFAULT nextval ('product_product id_seq'::regclass), NOT NULL, PRIMARY KEY
brand	varchar (50)	

4.4.3 UI design

The Blockchain-Powered Electronic Product Authenticity Verification System has been designed to offer a seamless, user-friendly interface with distinct roles for admin, manufacturer, supplier, retailer, and customer. Refer Fig. 12, to view UI design that designed with priority to be clarity and user-friendliness.

(a) (b)

Fig. 12 *UI wireframe design (a)Homepage; (b) Product Result page*

5. Implementation and Testing

This section outlines the development and evaluation of a secure product authenticity verification system that integrates traditional web technologies with blockchain. The backend system is developed using Node.js, and PostgreSQL to manage user authentication and product data. Moreover, JSON Web Tokens (JWTs) are utilized to handle secure session management. To ensure data integrity and prevent tampering, blockchain technology is employed using Ethereum smart contracts, while MetaMask facilitates seamless interaction between users and the blockchain network. The following sections describe the system architecture, core technologies, and implementation details.

5.1 System Architecture and Core Technologies

The system architecture integrates both centralized and decentralized technologies to enhance scalability, security, and data integrity. The backend, built with Node.js, manages routing, session control, and API interactions between the client, database, and blockchain. PostgreSQL serves as the relational database for structured data such as user credentials and product metadata, offering efficient querying and data consistency. Authentication is handled via JSON Web Tokens (JWT), enabling stateless and secure client-server communication. Product authenticity is verified through Ethereum-based smart contracts written in Solidity, ensuring immutable and transparent records. Users interact with the blockchain via MetaMask, allowing secure submission and verification of product data. This hybrid design combines the flexibility of conventional systems with the tamper-proof nature of blockchain, providing a robust foundation for real-time product authentication.

5.2 Security Implementation

The security module of the VERIFAI backend system was designed and implemented to ensure robust protection of user data, system integrity, and secure file operations. Built using Node.js and integrated with a PostgreSQL database, the backend incorporates multiple layers of security to guard against unauthorized access and common web vulnerabilities such as SQL injection, brute-force attacks, cross-site scripting (XSS), cross-site request forgery (CSRF), path traversal, insecure file uploads, and improper session handling and etc.

5.2.1 Environment and Configuration Security

All sensitive data, including database credentials and JWT secrets, are stored securely in .env files. Cross-Origin Resource Sharing (CORS) policies are enforced to limit unauthorized external requests.

```
require('dotenv').config();
const client = new Client({
    host: "localhost",
    user: "postgres",
    port: 5432,
    password: process.env.DB_PASSWORD,
    database: "postgres"
});
client.connect();
```

(a)

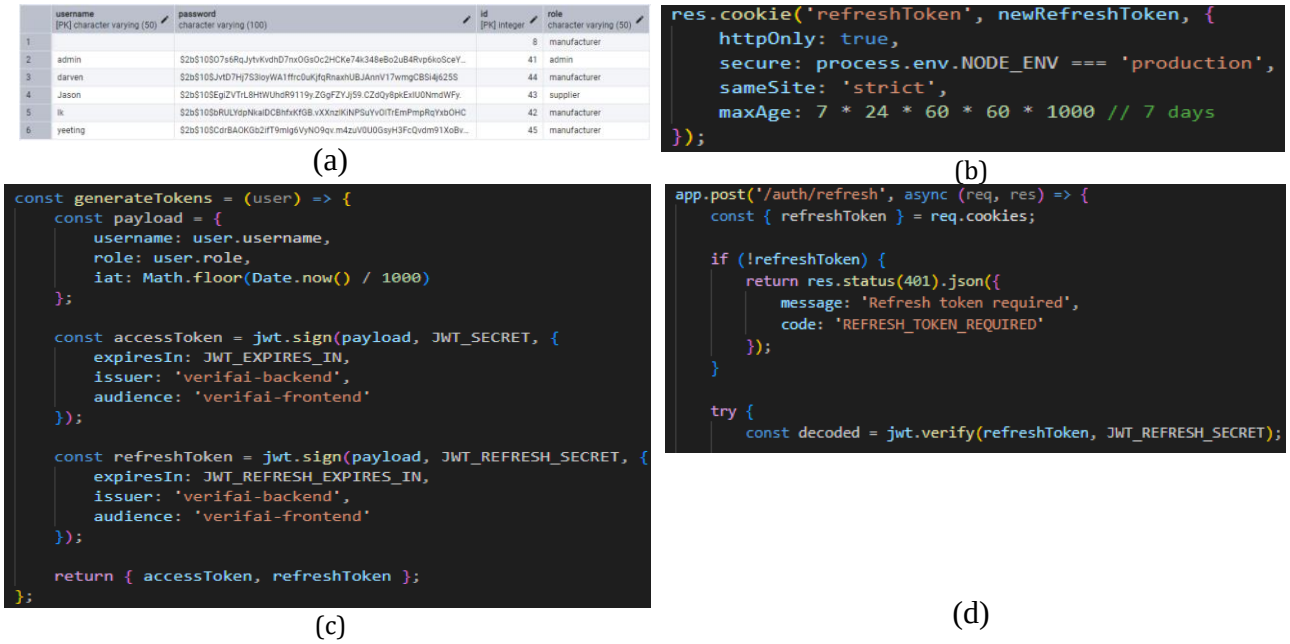
```
app.use(cors({
    origin: function (origin, callback) {

        if (!origin) return callback(null, true);

        const allowedOrigins = [
            'http://localhost:5173',
            'http://localhost:3000',
            'http://127.0.0.1:5173',
            'http://127.0.0.1:3000',
            process.env.FRONTEND_URL
        ].filter(Boolean);

        if (allowedOrigins.indexOf(origin) !== -1) {
            callback(null, true);
        } else {
            console.warn(`CORS blocked origin: ${origin}`);
            callback(new Error('Not allowed by CORS'));
        }
    },
    credentials: true,
    methods: ['GET', 'POST', 'PUT', 'DELETE', 'OPTIONS', 'PATCH'],
    allowedHeaders: [
        'Content-Type',
        'Authorization',
        'Cookie',
        'X-Requested-With',
        'Accept',
        'Origin'
    ],
    exposedHeaders: ['Set-Cookie'],
    optionsSuccessStatus: 200
}));
```

(b)

Fig. 12 *Environment and Configuration Security (a) Secure PostgreSQL Connection Setup in Node.js Using dotenv (b) CORS policies implemented*

5.2.2 Authentication and Session Security

User authentication is secured by hashing passwords with the Bcrypt library before storage, ensuring that plain-text credentials are never saved. The system uses JSON Web Tokens (JWT) for managing user sessions, generating access and refresh tokens upon successful login. These tokens are securely stored in httpOnly cookies, with the secure and sameSite attributes enabled to mitigate risks associated with cross-site scripting (XSS) and cross-site request forgery (CSRF) attacks.



	username [PK] character varying (50)	password character varying (100)	id [PK] integer	role character varying (50)
1			8	manufacturer
2	admin	$2b$10$O7s6RqJytvKvdhD7nxOGsOc2HCKe74k348eBo2uB4Rvp6koSceY..	41	admin
3	darven	$2b$10$JvtD7Hj7S3loyWA1ffrc0uKjfqRnaxhUBJAnnV17wmgCBSl4j625S	44	manufacturer
4	Jason	$2b$10$EgiZVTrL8HtWUhdR9119y.ZGgFZYJj59.CZdQy8pkExlU0NmdWFy.	43	supplier
5	lk	$2b$10$bRULYdpNkaIDCBnfxKfGB.vXXnziKiNPSuYvOiTrEmPmpRqYxbOHC	42	manufacturer
6	yeeting	$2b$10$CdrBAOKGb2ifT9mlg6VyNO9qv.m4zuV0U0GsyH3FcQvdm91XoBv...	45	manufacturer

(a)

```
res.cookie('refreshToken', newRefreshToken, {
    httpOnly: true,
    secure: process.env.NODE_ENV === 'production',
    sameSite: 'strict',
    maxAge: 7 * 24 * 60 * 60 * 1000 // 7 days
});
```

(b)

```
const generateTokens = (user) => {
    const payload = {
        username: user.username,
        role: user.role,
        iat: Math.floor(Date.now() / 1000)
    };

    const accessToken = jwt.sign(payload, JWT_SECRET, {
        expiresIn: JWT_EXPIRES_IN,
        issuer: 'verifai-backend',
        audience: 'verifai-frontend'
    });

    const refreshToken = jwt.sign(payload, JWT_REFRESH_SECRET, {
        expiresIn: JWT_REFRESH_EXPIRES_IN,
        issuer: 'verifai-backend',
        audience: 'verifai-frontend'
    });

    return { accessToken, refreshToken };
};
```

(c)

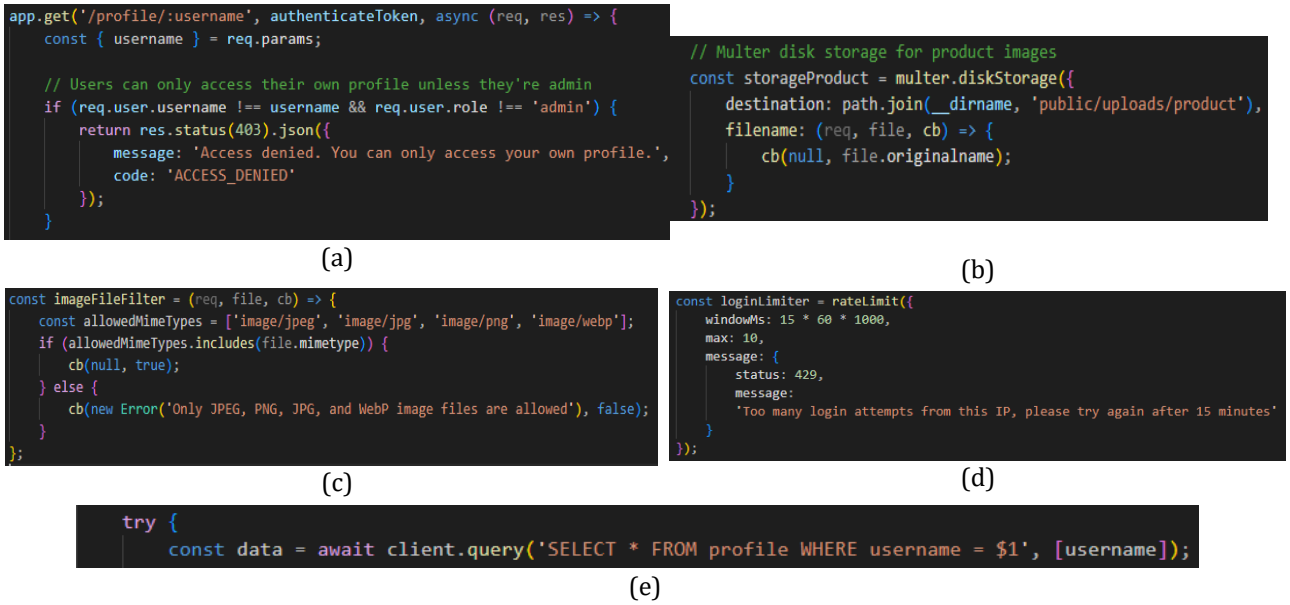
```
app.post('/auth/refresh', async (req, res) => {
    const { refreshToken } = req.cookies;

    if (!refreshToken) {
        return res.status(401).json({
            message: 'Refresh token required',
            code: 'REFRESH_TOKEN_REQUIRED'
        });
    }

    try {
        const decoded = jwt.verify(refreshToken, JWT_REFRESH_SECRET);
```

(d)

Fig. 13 *Authentication and Session Security (a) Hashed Password using Bcrypt; (b) Securely Setting a refresh token Cookie; (c) Function to Generate JWT Access and Refresh Tokens; (d) Node.js Express Route Handling Refresh Token Validation*

5.2.3 Access Control and Input Validation

A simple Role-Based Access Control (RBAC) mechanism restricts resource access based on user roles, such as admin or standard users. To enhance security, rate limiting is applied to authentication endpoints, mitigating brute-force login attempts. Additionally, parameterized SQL queries safeguard against SQL injection, while MIME-type filtering via Multer middleware prevents malicious file uploads.



```
app.get('/profile/:username', authenticateToken, async (req, res) => {
    const { username } = req.params;

    // Users can only access their own profile unless they're admin
    if (req.user.username !== username && req.user.role !== 'admin') {
        return res.status(403).json({
            message: 'Access denied. You can only access your own profile.',
            code: 'ACCESS_DENIED'
        });
    }
```

(a)

```
// Multer disk storage for product images
const storageProduct = multer.diskStorage({
    destination: path.join(__dirname, 'public/uploads/product'),
    filename: (req, file, cb) => {
        cb(null, file.originalname);
    }
});
```

(b)

```
const imageFileFilter = (req, file, cb) => {
    const allowedMimeTypes = ['image/jpeg', 'image/jpg', 'image/png', 'image/webp'];
    if (allowedMimeTypes.includes(file.mimetype)) {
        cb(null, true);
    } else {
        cb(new Error('Only JPEG, PNG, JPG, and WebP image files are allowed'), false);
    }
};
```

(c)

```
const loginLimiter = rateLimit({
    windowMs: 15 * 60 * 1000,
    max: 10,
    message: {
        status: 429,
        message:
        'Too many login attempts from this IP, please try again after 15 minutes'
    }
});
```

(d)

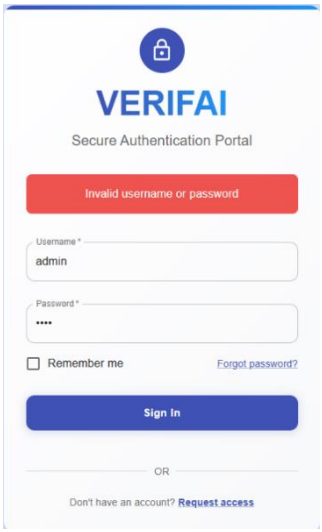
```
    try {
        const data = await client.query('SELECT * FROM profile WHERE username = $1', [username]);
```

(e)

Fig. 14 *Access Control and Input Validation (a) Restricting Profile Access Based on User Roles; (b) Rate Limit Implementation; (c) File Upload Sanitization Implementation; (d) Secure Static File Handling for Products; (e) Parameterise Query*

5.2.4 Error and Exception Handling

Custom error responses are implemented to prevent the exposure of stack traces or sensitive system details. JWT-related issues, such as token expiration or signature mismatches, are managed with clear yet non-revealing messages. Additionally, a global 404 handler is in place to gracefully handle and reject requests to unknown routes.



(a)

```
app.use('*', (req, res) => {
    console.log(` 404 - Route not found: ${req.method} ${req.originalUrl}`);
    res.status(404).json({
        message: 'Route not found',
        method: req.method,
        url: req.originalUrl,
        timestamp: new Date().toISOString()
    });
});
```

(b)

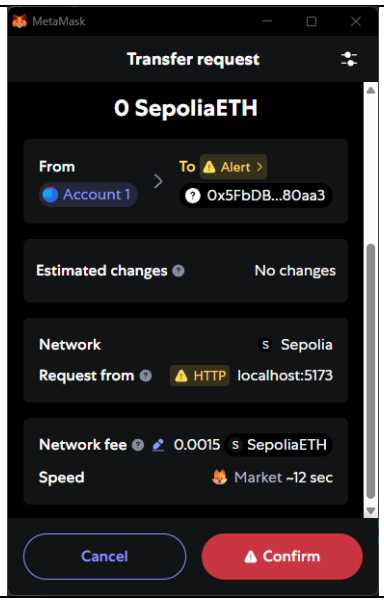
```
jwt.verify(token, JWT_SECRET, (err, user) => {
    if (err) {
        if (err.name === 'TokenExpiredError') {
            return res.status(401).json({
                message: 'Access token expired',
                code: 'TOKEN_EXPIRED'
            });
        }
        return res.status(403).json({
            message: 'Invalid access token',
            code: 'TOKEN_INVALID'
        });
    }
    req.user = user;
    next();
});
```

(c)

Fig. 15 *Error and Exception Handling (a) Secure Error Message at Login Page; (b) Custom 404 Handler for Undefined Routes; (c) JWT Verification with Explicit Error Responses*

5.3 Blockchain Integration and Testing

To enable decentralized product verification in VERIFAI, an Ethereum-compatible blockchain environment was established. MetaMask, a browser wallet extension, was integrated to handle user authentication through the window.ethereum API. It verifies wallet connection via eth_accounts and prompts login using eth_requestAccounts, ensuring that manufacturer, supplier, and retailer dashboards can securely identify users via their wallet addresses. The smart contract, developed in Solidity and deployed using the Hardhat framework, facilitates secure product registration and traceability. It supports functions such as registerProduct for recording product metadata and addProductHistory for appending supply chain events (e.g., ownership transfers, timestamps, and sale status). React-based frontends interact with the blockchain using Ethers.js, and all contract interactions are authorized through MetaMask, guaranteeing authenticity and immutability. This implementation forms the cryptographic backbone of VERIFAI's transparent and tamper-proof supply chain model.



(a)

```
pragma solidity ^0.8.17;

import "hardhat/console.sol";

contract Verifai{
    address public owner;

    struct Product {
        string name;
        string serialNumber;
        string description;
        string brand;
        string image;
        mapping(uint => ProductHistory) history;
        uint historySize;
    }

    mapping(string => Product) products;
    mapping(uint => ProductHistory) history;

    struct ProductHistory {
        uint id;
        string actor;
        string location;
        string timestamp;
        bool isSold;
    }
```

(b)

```
Contract deployment: Verifai
Contract address:    0x9ac91029691a8d483f32672a0261d4e8708351d7
Transaction:         0x1d6c8b13591802979d9cdd9867ee6cf51835f061c9c1a5b0ba7e81f3fe51e348
From:                0xf39fd6e51aad88f6f4ce6ab8827279cfffb92266
Value:               0 ETH
Gas used:            1438983 of 1438983
Block #22619876:     0x26b292cac7a456d079719bb864d3e6cebb83e87498cb9a070797d2fede4f9c87
```

(c)

```
function registerProduct(string memory _name, string memory _brand, string memory _serialNumber, string memory _description, string memory _image,
string memory _actor, string memory _location, string memory _timestamp) public {
    Product storage p = products[_serialNumber];

    p.name = _name;
    p.brand = _brand;
    p.serialNumber = _serialNumber;
    p.description = _description;
    p.image = _image;
    p.historySize = 0;

    addProductHistory(_serialNumber,_actor, _location, _timestamp, false);
}

function addProductHistory(string memory _serialNumber, string memory _actor,
string memory _location, string memory _timestamp, bool _isSold) public {
    Product storage p = products[_serialNumber];
    p.historySize++;
    p.history[p.historySize] = ProductHistory(p.historySize, _actor, _location, _timestamp, _isSold);

    console.log("i1: %s", p.historySize);
    console.log("Product History added: %s", p.history[p.historySize].actor);
    console.log("Product : %s", p.name);
}

function getProduct(string memory _serialNumber) public view returns (string memory, string memory,
string memory, string memory, string memory, ProductHistory[] memory) {
    ProductHistory[] memory pHistory = new ProductHistory[](products[_serialNumber].historySize);

    for (uint i = 0; i < products[_serialNumber].historySize ; i++) {
        pHistory[i] = products[_serialNumber].history[i+1];

    }

    return (products[_serialNumber].serialNumber, products[_serialNumber].name, products[_serialNumber].brand,products[_serialNumber].description, products[_serialNumber].image, pHistory);
}
```

(d)

Fig. 16 *Blockchain Integration and Testing(a) Example of a MetaMask Wallet Transaction Prompt; (b) Smart Contract Initialization and Ownership Assignment; (c) Blockchain Transaction Metadata for Contract Interaction; (d) Code Segment of Deployment of Smart Contract*

5.4 System Testing and Validation

To validate the Blockchain-Powered Electronic Product Authenticity Verification system, two main categories of evaluation were performed which is functional testing and user acceptance testing.

5.4.1 Functional Testing

The system was assessed using a structured test plan and a security checklist. Functional tests ensured that core features like user authentication, MetaMask integration, role-based access, product registration, QR generation, and blockchain-based history tracking worked as intended. The smart contract reliably stored and retrieved data, and role-based dashboards (admin, manufacturer, supplier, retailer, customer) were validated. Table 8 summarizes core test cases, all of which passed the expected results. Security testing included password hashing, role-based access control (RBAC), CORS enforcement, SQL injection prevention, brute-force attack mitigation, and secure file handling. All measures met their security criteria (Table 9).

Table 8 *System Functional Test Plan Results*

Test Area	Features Included	Actual Result
Authentication & Error Handling	Login (valid/invalid), Logout, Unauthorized access	Pass/~~Fail~~
User & Account Management	Registering Other Users (Admin), View/Edit accounts, View profile	Pass/~~Fail~~
Blockchain & Wallet Integration	Connect MetaMask wallet, Add/Update product, View transaction history	Pass/~~Fail~~
QR & Product Handling	Generate QR, Scan QR, Product authenticity result	Pass/~~Fail~~
UI & Navigation	View role-based dashboard, Navigate system features	Pass/~~Fail~~

Table 9 *Security Check List Results*

Security Area	Checklist Items Included	Actual Result
Authentication & Error Handling	Generic error messages (e.g., "Invalid credentials") without disclosing username/password correctness	Pass/~~Fail~~
Password Policy Enforcement	Enforces complexity (uppercase, lowercase, numbers, special characters)	Pass/~~Fail~~
Password Policy Enforcement	Enforces minimum password length (≥8 characters)	Pass/~~Fail~~
User Interface (UI) Security	Password input obscured (e.g., masked with •••••)	Pass/~~Fail~~
Database Security	Protection from SQL injection via parameterized queries	Pass/~~Fail~~
Password Storage	Passwords hashed using Bcrypt with salting	Pass/~~Fail~~
Access Control	Role-Based Access Control (RBAC) enforced with appropriate access denial messages	Pass/~~Fail~~
File Upload Security	Only image MIME types allowed; executable files blocked	Pass/~~Fail~~
Brute-Force Protection	Express-rate-limit middleware restricts excessive login attempts	Pass/~~Fail~~
Secure File Handling	Static files served through controlled directories to prevent path traversal	Pass/~~Fail~~
CORS Policy	CORS headers properly configured with restricted origins and credential support	Pass/~~Fail~~
Error Handling	Custom error pages do not expose internal server details or stack traces	Pass/~~Fail~~

5.4.2 User Acceptance Testing

The User Acceptance Testing (UAT) involved 25 participants, with five users representing each key role in the system: Administrator, Manufacturer, Supplier, Retailer, and Customer. Participants included real stakeholders such as office administrators (e.g., Darvin Raj), manufacturers (e.g., Hikvision, TP-Link), suppliers (e.g., Dahua, Ezviz, SunElectric), retailers, and customers (e.g., Jie Sung, Hardrockcafe patrons). The form was distributed by Global Rigorous Aura Success (M) Sdn Bhd to both internal employees and external partners, ensuring relevant and practical feedback from actual users interacting with the blockchain-powered product verification system.

Table 10 *User Acceptance Test Summary by Role*

Role	Core feature Tested	Login Success	Invalid Login Handling	Dashboard Redirect	Logout success	User-Friendly
Admin	Register + manage users	5/ 5	5/ 5	5/ 5	5/ 5	5/ 5
Manufacturer	Add product + QR	5/ 5	5/ 5	5/ 5	5/ 5	5/ 5
Supplier	Scan + Update product	5/ 5	5/ 5	5/ 5	5/ 5	5/ 5
Retailer	Update product as sold	5/ 5	5/ 5	5/ 5	5/ 5	5/ 5
Customer (User)	View product journey + authenticity	-	-	-	-	5/ 5

6. Conclusion and Future Work

The development of the Blockchain-Powered Electronic Product Authenticity Verification System for Global Rigorous Aura Success (M) Sdn Bhd marks a significant step forward in addressing the issue of counterfeit products through secure and transparent verification. By leveraging blockchain technology, the system ensures that product data remains tamper-proof, while features like JWT-based authentication, strong password enforcement, and secure file validation contribute to a robust security framework. The system also promotes decentralized trust and allows users to trace a product's lifecycle from source to sale, enhancing consumer confidence. Despite its strengths, the system currently operates only in a local environment, requires manual data input, and lacks alert notifications and mobile optimization. Looking ahead, several enhancements are planned to overcome these limitations. These include migrating from Testnet to Mainnet to enable real blockchain transactions, deploying the system on a Web3 platform for broader access, introducing a product catalogue for

easier management, and incorporating notification features and mobile-friendly designs. In summary, the project not only fulfills its initial goals but also lays a strong foundation for future improvements. It demonstrates how blockchain technology can be used to build trust in product authenticity and strengthen supply chain security.

Acknowledgement

The authors would like to thank the Faculty of Computer Science and Information Technology, Universiti Tun Hussein Onn Malaysia for its support.

Conflict of Interest

Authors declare that there is no conflict of interest regarding the publication of the paper.

Author Contribution

The authors confirm contribution to the paper as follows: **study conception and design**: *D. D. Kathirveloo, S. N. Ramli*; **data collection**: *D. D. Kathirveloo, S. N. Ramli*; **analysis and interpretation of results**: *D. D. Kathirveloo, S. N. Ramli*; **draft manuscript preparation**: *D. D. Kathirveloo, S. N. Ramli*; All authors reviewed the results and approved the final version of the manuscript.

References

[1] B. M. Beamon, "Designing the green supply chain," *Logistics Information Management*, vol. 12, no. 4, pp. 332–342, Jan. 1999, doi: 10.1108/09576059910284159.

[2] A. Gunasekaran, C. Patel, and R. E. McGaughey, "A framework for supply chain performance measurement," *Int J Prod Econ*, vol. 87, no. 3, pp. 333–347, Feb. 2004, doi: 10.1016/J.IJPE.2003.08.003.

[3] "Blockchain-Technology-for-Product-Authentication-and-Sustainability".

[4] N. C. K. Yiu, "Article toward blockchain-enabled supply chain anti-counterfeiting and traceability," *Future Internet*, vol. 13, no. 4, Apr. 2021, doi: 10.3390/fi13040086.

[5] "Comparison: Centralized, Decentralized, and Distributed Systems." Accessed: Dec. 29, 2024. [Online]. Available: https://www.geeksforgeeks.org/comparison-centralized-decentralized-and-distributed-systems/

[6] B. Faber, G. Michelet, N. Weidmann, R. R. Mukkamala, and R. Vatrapu, "BPDIMS:A blockchain-based personal data and identity management system," in *Proceedings of the Annual Hawaii International Conference on System Sciences*, IEEE Computer Society, 2019, pp. 6855–6864. doi: 10.24251/hicss.2019.821.

[7] S. Nakamoto, "Bitcoin: A Peer-to-Peer Electronic Cash System." [Online]. Available: www.bitcoin.org

[8] R. Huo *et al.*, "A Comprehensive Survey on Blockchain in Industrial Internet of Things: Motivations, Research Progresses, and Future Challenges," *IEEE Communications Surveys and Tutorials*, vol. 24, no. 1, pp. 88–122, 2022, doi: 10.1109/COMST.2022.3141490

[9] T. R. Gadekallu et al., "Blockchain for the Metaverse: A Review," Mar. 2022, [Online]. Available: http://arxiv.org/abs/2203.09738

[10] H. Taherdoost, "Smart Contracts in Blockchain Technology: A Critical Review," Feb. 01, 2023, *MDPI*. doi: 10.3390/info14020117.

[11] N. Szabo, "Formalizing and Securing Relationships on Public Networks."

[12] A. Bahga and V. K. Madisetti, "Blockchain Platform for Industrial Internet of Things," *Journal of Software Engineering and Applications*, vol. 09, no. 10, pp. 533–546, 2016, doi: 10.4236/jsea.2016.910036.

[13] B. K. Mohanta, S. S. Panda, and D. Jena, "An Overview of Smart Contract and Use Cases in Blockchain Technology," in *2018 9th International Conference on Computing, Communication and Networking Technologies, ICCCNT 2018*, Institute of Electrical and Electronics Engineers Inc., Oct. 2018. doi: 10.1109/ICCCNT.2018.8494045.

[14] K. Delmolino, M. Arnett, A. Kosba, A. Miller, and E. Shi, "Step by Step Towards Creating a Safe Smart Contract: Lessons and Insights from a Cryptocurrency Lab," 2015.

[15] S. Wang, Y. Yuan, X. Wang, J. Li, R. Qin, and F.-Y. Wang, "An overview of smart contract: Architecture, applications, and future trends," in Proc. IEEE Intell. Veh. Symp. (IVS), Changshu/Suzhou, China, Jun. 2018, pp. 108–113.

[16] "Review: Secure QR codes for anti-counterfeiting, with examples." Accessed: Dec. 28, 2024. [Online]. Available: https://www.scantrust.com/secure-qr-code-anti-counterfeiting-solutions/

[17] "Brand Protection and Product Authentication | EON." Accessed: Dec. 29, 2024. [Online]. Available: https://glitchmagazine.xyz/eon-digital-ids-reshaping-the-future-of-fashion-and-the-planet-2/

[18] Amber Weir, "EON Digital ID's Reshaping The Future of Fashion And The Planet," GLITCH Magazine.

[19] R. Agarwal and P. Lago, "PATHOS—a paradigmatic approach to high-level object-oriented software development," *ACM SIGSOFT Software Engineering Notes*, vol. 20, no. 2, pp. 36–41, Apr. 1995, doi: 10.1145/224155.224156.