



Faculty of Computer Science and Information Technology

Blockchain authentication technique for supply chain product management system

KALVIN YAW ZHI HONG

Bachelor of Computer Science with Honours (Software Engineering)

2025

Blockchain authentication technique for supply chain product management system

KALVIN YAW ZHI HONG

This project is submitted in partial fulfillment of the
requirements for the degree of
Bachelor of Computer Science with Honours
(Software Engineering)

Faculty of Computer Science and Information Technology

UNIVERSITI MALAYSIA SARAWAK

2025

ACKNOWLEDGEMENT

First of all, I would like to express my deepest gratitude to my course coordinator and supervisor, Prof. Dr. Wang Yin Cai, for his continuous guidance, support, and encouragement throughout the completion of my Final Year Project 1. He also allows me to explore something new such as working with communities. Next, my supervisor also provides me with useful advices which will enable me to improve myself to become a better student, specifically in completing this project. Besides, I would also like to thank my examiner, Associate Professor Dr Noor Alamshah B. Bolhassan which provides me with pieces of advice that allow me to improve my project. All of their contributions are sincerely appreciated because, without them, the project will never be completed.

TABLE OF CONTENT

ACKNOWLEDGEMENT	I
TABLE OF CONTENT	III
LIST OF FIGURES	VI
LIST OF TABLE	VIII
ABSTRACT	IV
CHAPTER 1 : INTRODUCTION.....	1
1.1 Introduction	1
1.2 Problem Statement	3
1.3 Objectives	4
1.4 Methodologies	5
1.5 Scope	7
1.6 Significant of Project	8
1.8 Expected Outcome	9
1.9 Project outline	9
1.10 Summary	11
CHAPTER 2 : LITERATURE REVIEW	12
2.1 Introduction	12
2.2 Reviews on Similar Existing Work / System	13
2.2.1 Review for transnational supply chain management system	13
2.2.2 Review for Corda-based Supply Chain System	15
2.2.3 Review for Hyperledger Fabric Supply Chain System	18
2.2.4 Review for proposed Supply Chain System	20
2.3 Comparison between three existing type systems and proposed system	22

2.4 Review on tools or technology used	25
2.5 Summary	27
CHAPTER 3 : REQUIREMENT ANALYSIS AND DESIGN	28
3.1 Introduction	28
3.2 Technology/Blockchain Integration Methods	30
3.2.1 Hybrid Blockchain Justification	33
3.2.2 Tokenization in Blockchain-Based Supply Chain Systems	35
3.3 System Work Methods	38
3.3.1 Use case diagram	38
3.3.2 Use case specification	40
3.3.3 Sequence diagram	48
3.4 System Mockup	57
3.5 Summary	61
CHAPTER 4 : IMPLEMENTATION	62
4.1 Login Page	63
4.2 Registration Page	64
4.3 Admin Dashboard	65
4.3.1 Admin dashboard - Manage User	66
4.3.1 Admin dashboard - Manage Livestock	67
4.4 Supplier Dashboard	69
4.4.1 Supplier dashboard - Livestock History	70
4.4.2 Supplier dashboard - Transfer Ownership	71
4.4.3 Supplier dashboard - Transfer History	73
4.5 Public Dashboard	75
CHAPTER 5 : TESTING	76

5.1 Introduction	76
5.2 Functionality Testing	76
5.2.1 Register	77
5.2.2 Login	78
5.2.3 Admin Module	79
5.2.4 Supplier Module	82
5.2.5 Public Module	85
5.3 Non-Functionality Testing	86
5.3.1 Usability Testing	86
5.3.2 Compatibility Testing	87
5.3.3 Reliability Testing	88
5.4 Summary	89
CHAPTER 6 : CONCLUSION AND FUTURE WORK	90
6.1 Introduction	90
6.2 Achivement	90
6.3 Limitations	90
6.4 Future Work	91
REFERENCE	92
APPENDIX	95

LIST OF FIGURES

Figure 1 : Traditional supply chain model(Kumar & Gupta, 2017)	14
Figure 2 : Cordas architecture (Wan, C.,2019).....	17
Figure 3 : Fabric high level transaction flow (Androulaki et al., 2018).....	19
Figure 4 : Ethereum blockchain architecture (Wan, C., 2019).....	21
Figure 5 : Comparison between traditional & digital SCM	22
Figure 6 : Comparison of Centralized and Distributed Ledger Systems.	28
Figure 7 : Architecture diagram.	32
Figure 8 : Tokenization process.	37
Figure 9 : Use case diagram.....	38
Figure 10 : Sequence diagram for register livestock flow	48
Figure 11 : Sequence diagram for check livestock status flow	49
Figure 12 : Sequence diagram for update livestock status flow	50
Figure 13 : Sequence diagram for transfer ownership flow	51
Figure 14 : Sequence diagram for deploy smart contract flow	52
Figure 15 : Sequence diagram for manage user account flow	53
Figure 16 : Sequence diagram for check livestock history flow	54
Figure 17 : Sequence diagram for login flow	55
Figure 18 : Sequence diagram for logout flow	56
Figure 19 : Login page.....	57
Figure 20 : Admin dashboard	58
Figure 21 : Supplier dashboard	58
Figure 22 : Public Dashboard	59
Figure 23 :The hierarchical diagram	62

Figure 24 : Login page	63
Figure 25 : Registration page	64
Figure 26 :Admin dashboard	65
Figure 27 :Manage user page	66
Figure 28 : Manage livestock page	67
Figure 29 :Supplier dashboard	69
Figure 30 :Livestock history page	70
Figure 31 : Transfer ownership page (step 1)	71
Figure 32 : Transfer ownership page (step 2)	71
Figure 33 : Transfer ownership page (step 3)	72
Figure 34 : Transfer history page	73
Figure 35 : Public dashboard	75
Figure 36 : Project schedule gantt chart	96

LIST OF TABLE

Table 1 : Comparison between three existing type systems and proposed system	23
Table 2 : Use case specification for register livestock	40
Table 3 : Use case specification for check livestock status	41
Table 4 : Use case specification for update livestock status	42
Table 5 : Use case specification for transfer ownership	43
Table 6 : Use case specification for deploy smart contract	44
Table 7 : Use case specification for manage user account	44
Table 8 : Use case specification for check livestock history	45
Table 9 : Use case specification for login	46
Table 10 : Use case specification for logout	47
Table 11 : Test case for registration module	77
Table 12 : Test case for login module	79
Table 13 : Test case for admin module overall	80
Table 14 : Test case for registration module(User Management)	81
Table 15 : Test case for registration module(Livestock Management)	81
Table 16 : Test case for supplier dashboard module(Add Livestock)	82
Table 17 : Test case for supplier dashboard module(Livestock History)	83
Table 18 : Test case for supplier dashboard module(Transfer Ownership)	84
Table 19 : Test case for supplier dashboard module public dashboard	85
Table 20 : Usability testing table	87
Table 21 : Compatibility testing table	87
Table 22 : Reliability testing table	89

Table 23 : Project Schedule 95

ABSTRACT

Customers today demand greater transparency and trust in the supply chain, especially in industries like livestock, where product authenticity and safety are critical. However, traditional supply chain systems often lack efficiency, security, and visibility, leading to challenges like fraud and operational delays. To address these issues, this project proposes a blockchain-based supply chain management system specifically designed for the livestock sector. The system leverages a hybrid blockchain model, combining private blockchains for secure storage of sensitive data and public blockchains for transparent, verifiable transaction records. The project also integrates innovative features such as tokenization, where each livestock is represented as a unique digital token for easy tracking and fraud prevention. Additionally, smart contracts automate processes like livestock registration, ownership transfer, and status updates. The system includes role-based access for suppliers, administrators, buyers, and retailers, ensuring streamlined operations. This project aims to revolutionize livestock supply chain management by offering a secure, transparent, and scalable solution powered by blockchain technology.

CHAPTER 1 : INTRODUCTION

1.1 Introduction

In today's globalized economy, supply chains span continents and involve a complex web of suppliers, manufacturers, logistics providers, and retailers. This complexity brings with it significant challenges. Ensuring that every product is authentic, traceable, and secure can be difficult, and any breakdown in these areas can lead to counterfeit goods entering the market, data discrepancies, and limited visibility across the network. These problems reduce trust among those involved and make the supply chain less efficient and reliable, impacting both businesses and customers.

Blockchain technology offers a transformative approach to solving these problems by providing a secure, decentralized platform for tracking and verifying every transaction or movement of a product (Vyas et al., 2019). Unlike traditional centralized systems, where a single authority controls the database, blockchain is a shared ledger that all authorized participants can access and update in real-time. This ledger is organized into blocks, each containing a batch of transaction data, which are cryptographically linked to previous blocks, creating a permanent and tamper-proof chain of information. Once data is added, it cannot be altered without changing all subsequent blocks, ensuring a high level of security and trust in the system (Rodeck & Curry, 2022).

By applying blockchain to supply chain management, this project aims to create an authentication technique that enhances transparency, traceability, and security at each stage of a product's journey. Imagine a system where every transaction from sourcing raw materials to delivering the final product to the consumer is recorded on an immutable ledger. Each

participant can verify the authenticity of the data without relying on third-party auditors, reducing costs and increasing efficiency. For example, a retailer can trace a product back through its manufacturing process, ensuring that it meets quality standards and regulatory requirements. Similarly, consumers can gain insights into the origin and authenticity of the products they buy, building trust and confidence in the brands they support.

This blockchain-based approach to supply chain management is not only about protecting data integrity; it also has the potential to fundamentally reshape how information flows between stakeholders. By making each step in the supply chain verifiable and transparent, blockchain can reduce the risk of counterfeiting, prevent data fraud, and enhance collaboration across the network. Such a system empowers businesses to respond to issues more quickly, reduces inefficiencies, and fosters an environment of trust and accountability.

1.2 Problem Statement

Food security and authenticity have become critical concerns in today's globalized economy, as consumers demand greater transparency and trust in the food supply chain. Traditional supply chain systems lack the necessary visibility, making it challenging for suppliers, regulators, and consumers to verify the origins, handling processes, and safety of food products. This opacity has led to increasing incidents of food fraud, contamination, and mistrust among stakeholders, posing significant risks to public health and brand integrity.

Blockchain technology presents a transformative solution by enabling a decentralized, secure, and tamper-proof system to document every transaction and movement of goods throughout the supply chain. Unlike traditional systems, blockchain ensures real-time verification of food provenance, handling practices, and compliance with safety standards, eliminating reliance on intermediaries (Kshetri, 2018).

This project aims to develop a blockchain-based supply chain management system that enhances transparency, strengthens food safety measures, and fosters consumer trust. By implementing traceability, ownership determination, and real-time product journey visibility, this system seeks to address the pressing challenges in the food industry and establish a secure and transparent supply chain ecosystem.

1.3 Objectives

- a) To study and identify the suitable blockchain authentication technique for product authentication
- b) To design and integrate an authentication technique into supply chain management system processes.
- c) To develop a prototype system to prove the transparency, security and traceability of the technique proposed.

1.4 Methodologies

The methodology for this blockchain authentication technique for supply chain product management system is designing a decentralized, transparent, and tamper-proof architecture, using HTML, CSS, Bootstrap, JavaScript (Ethers.js), Solidity, and Hardhat. The system architecture was designed to integrate blockchain to enhance traceability and transparency across the supply chain. This design is essential for ensuring that information on product origin, ownership history, and transaction records remains immutable and accessible to stakeholders. The choice of Solidity for smart contract development and Hardhat for testing and deploying these contracts was influenced by their reliability and widespread support within the Ethereum ecosystem, making them ideal for decentralized application development.

The front-end interface is developed using HTML, CSS, and Bootstrap to provide a responsive, user-friendly experience across devices. Bootstrap's grid system and components enable a seamless, adaptable UI, while Ethers.js facilitates interaction with the blockchain, allowing the frontend to communicate securely and efficiently with smart contracts. This choice was made to ensure a smooth user experience for supply chain interactions, such as viewing product history or transferring ownership, which are central to the system's functionality.

The smart contracts, written in Solidity, were developed to handle core functionalities, including tracking product ownership, recording transactions, and storing product history. Key data structures, such as Supplier, Livestock, and OwnershipHistory, were designed as structs to organize and store data efficiently, while mappings provide rapid access to data points. Events were also implemented to emit information about actions, such as adding suppliers or transferring ownership, enhancing transparency across the blockchain. Security

measures, including input validation, access control mechanisms, and adherence to Solidity best practices, were prioritized to protect against vulnerabilities, ensuring data integrity and secure contract interactions.

Hardhat was employed as the primary development and testing environment, allowing for a controlled, local blockchain setup. This facilitated rigorous testing through unit tests for individual functions and integration tests simulating real-world interactions. End-to-end testing ensured that front-end actions correctly triggered smart contract functions, while user acceptance testing (UAT) validated that the system met user needs and functioned seamlessly. Git version control was used throughout development to manage code changes, track iterations, and support collaborative debugging.

1.5 Scope

The scope of this blockchain authentication technique for supply chain product management system is creating a secure, transparent platform to track products as they progress through the supply chain, with a focus on the food and livestock sectors where traceability is crucial. The system aims to provide end-to-end product traceability, allowing stakeholders to verify product origins, ownership histories, and transit stages through a reliable, immutable blockchain record. Key functionalities include ownership tracking to prevent fraud, immutable data storage to ensure trust, and a responsive web interface that allows users, including suppliers, manufacturers, and customers, to access relevant product information. This interface will be developed with HTML, CSS, and Bootstrap, while smart contracts written in Solidity manage essential processes, such as adding products, transferring ownership, and recording product histories. Integrating the Ethereum blockchain network with Ethers.js will secure the interactions between the user interface and the blockchain, while Hardhat provides a test environment to validate system performance.

This project scope also incorporates robust testing, validation, and basic security measures, ensuring that only authorized parties can perform specific actions, such as ownership transfers. The system's design includes comprehensive testing on Ethereum test networks to ensure functionality and reliability before deployment on the blockchain. Additionally, user documentation will be provided to guide navigation and ease adoption. While the project delivers essential supply chain functions, it has limitations in scalability and transaction costs on the public blockchain. Future expansions could address these constraints by integrating IoT for real-time tracking, adding support for additional data types, or exploring private blockchain options to optimize transaction speed and cost-effectiveness.

1.6 Significant of Project

The significance of this blockchain authentication technique for supply chain product management system lies in its ability to enhance transparency, security, and trust within the supply chain, addressing key challenges in industries like food and livestock that demand rigorous monitoring. By providing an immutable record of product origin, ownership, and transaction history, the system helps combat fraud, reduces the risk of counterfeit products, and assures end-users of product authenticity. This level of transparency is crucial for businesses and consumers alike, as it provides verifiable information on product journey and quality, thereby strengthening consumer confidence and supporting brand integrity.

Additionally, the project contributes to efficiency and accountability across the supply chain. By decentralizing data storage, it removes reliance on intermediaries and centralized databases, which can be vulnerable to tampering and inefficiencies. The system's secure, automated tracking of product movement streamlines processes for stakeholders, reducing administrative overhead and the likelihood of disputes. This can foster more streamlined operations, reduce costs, and ultimately contribute to better resource management. Furthermore, the project demonstrates the practical application of blockchain technology beyond cryptocurrency, setting a foundation for future improvements, such as real-time tracking and integration with IoT, that could further revolutionize supply chain management practices.

1.8 Expected Outcome

The expected outcome of this blockchain authentication technique for supply chain product management system is creating a secure, transparent platform that enhances traceability, accountability, and efficiency across the supply chain. The system will provide an immutable record of product origin, ownership history, and transaction details, reducing fraud and ensuring data integrity. By enabling automated, secure ownership transfers, it will streamline interactions between stakeholders and reduce reliance on manual processes. Additionally, this project will lay a strong foundation for future expansions, such as integrating real-time IoT tracking or scaling to a private blockchain for optimized performance and lower costs. Ultimately, this system will demonstrate the practical benefits of blockchain in improving transparency, building trust, and modernizing supply chain management.

1.9 Project outline

Chapter 1: Introduction. This chapter provides an overview of the Blockchain Authentication Technique for Supply Chain Product Management System. It includes the problem statement, project objectives, methodology for developing the system, project scope, significance, project schedule, and the expected outcomes of the proposed system.

Chapter 2: Background Study/Literature Review. This chapter presents a review of existing blockchain authentication techniques and their applications in supply chain management. It includes a comparison of methods such as Ethereum-based smart contracts and other authentication approaches, highlighting their strengths and limitations. The proposed solution is analyzed against these existing techniques.

Chapter 3: Requirement Analysis and Design. This chapter outlines the system architecture and design, focusing on the decentralized blockchain infrastructure. It also includes software and hardware requirements, stakeholder inputs, and the detailed design of the prototype, including user interface considerations for stakeholder accessibility.

Chapter 4: Implementation. This chapter covers the implementation of the blockchain system. It includes the development of smart contracts using Solidity, the integration of Ethereum blockchain with the system, and detailed explanations of features such as product traceability and ownership tracking.

Chapter 5: Testing and Evaluation. This chapter explains the testing procedures applied to validate the functionality of the blockchain system. It includes unit testing of smart contracts, integration testing between the frontend and backend.

Chapter 6: Conclusion and Future Work. This chapter summarizes the findings and achievements of the project, including the developed prototype's contribution to supply chain transparency and security. It also discusses the limitations of the project and provides recommendations for future enhancements, such as IoT integration and the adoption of private blockchains.

1.10 Summary

This chapter introduces the Blockchain Authentication Technique for Supply Chain Product Management System, addressing challenges like counterfeit goods, inefficiencies, and lack of transparency in global supply chains. It highlights blockchain as a secure, decentralized solution to enhance traceability and trust among stakeholders. The project's objectives include studying blockchain authentication techniques, designing their integration into supply chain processes, and developing a prototype to demonstrate transparency, security, and traceability. The chapter also outlines the project's scope in the food and livestock industries, its significance in building trust and efficiency, and its potential for future innovations like IoT integration and private blockchains.

CHAPTER 2 : LITERATURE REVIEW

2.1 Introduction

Supply chain management has long struggled with issues like counterfeiting, inefficient data handling, and a lack of transparency throughout the product journey. These challenges make it difficult to ensure the authenticity and quality of goods, often leading to inefficiencies and a loss of trust among stakeholders. Blockchain technology offers a potential solution, providing a secure and decentralized platform that can improve traceability, authentication, and data integrity. In this chapter, we will explore three type of existing supply chain systems to see how they address these issues and how they compare to the proposed blockchain solution. These systems include the traditional supply chain model, the Corda-based supply chain system, and the Hyperledger Fabric supply chain system. By examining the features, functions, and characteristics of each, we will highlight their strengths and weaknesses, as well as the proposed blockchain solution can bring to supply chain management.

2.2 Reviews on Similar Existing Work / System

2.2.1 Review for transnational supply chain management system

The traditional supply chain system is the backbone of many industries, it heavily relying on manual processes and centralized databases. While this framework offers a familiar and straightforward approach for stakeholders, its inherent inefficiencies have become increasingly obvious as supply chains have grown more complex. One key characteristic of traditional supply chains is their reliance on paper-based documentation for tasks like inventory management, order tracking, and shipment documentation. This manual approach often leads to inefficiencies and errors, as data is stored in centralized databases controlled by individual stakeholders such as manufacturers, suppliers, and distributors. However, these databases are typically not interconnected, resulting in data silos and communication gaps that hinder real-time data sharing and collaborative decision-making.

Additionally, the traditional system lacks automation in critical areas, including product journey tracking, fraud detection, and authenticity verification, which increases the likelihood of delays and mistakes. Communication among stakeholders, often conducted via phone, email, or fax, further slows down operations and elevates the risk of errors (Kumar & Gupta, 2017). Despite these shortcomings, the traditional supply chain system has notable strengths. Its established presence and simplicity make it easier for businesses, especially smaller ones with limited resources, to operate without the need for significant technological investments. The low initial cost and adaptability to diverse industries also contribute to its appeal. However, the system's drawbacks are increasingly significant. Manual processes are time-consuming and prone to errors, while the lack of real-time information sharing often leads to discrepancies in data, shipment delays, and miscommunications (Chandran et al.,

2014).

Moreover, the limited visibility across the supply chain exacerbates the risk of counterfeiting and fraud, as there are no robust mechanisms to verify product authenticity (Kumar & Gupta, 2017). Additionally, the reliance on human intervention and outdated communication methods slows response times, making the system less capable of adapting to disruptions or changes in demand. Overall, while the traditional supply chain system has served industries well historically, its inefficiencies, lack of transparency, and vulnerability to fraud make it increasingly inadequate for addressing the challenges of modern supply chains.

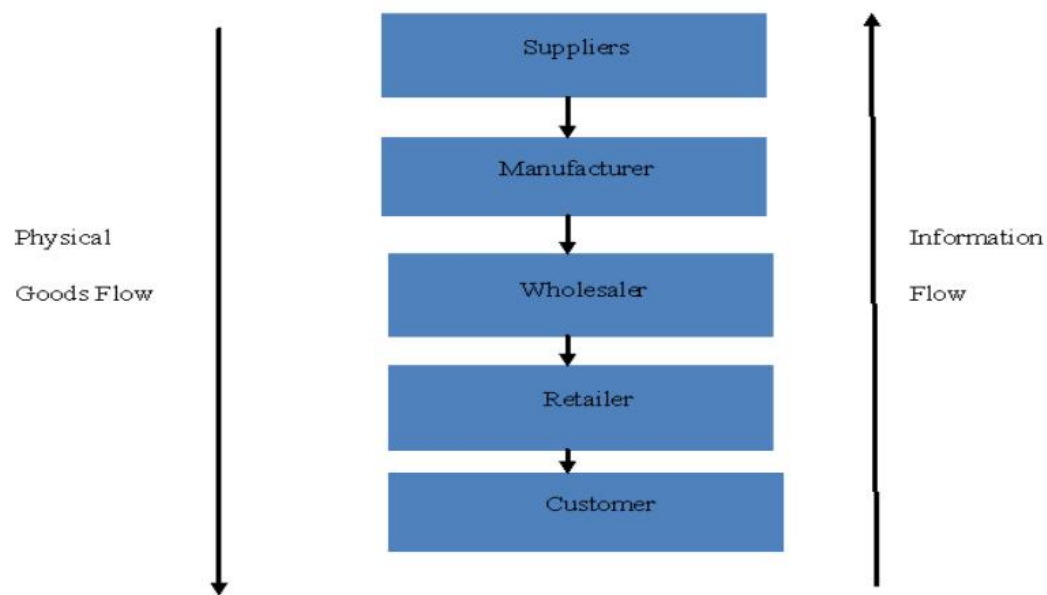


Figure 1 : *Traditional supply chain model*(Kumar & Gupta, 2017)

2.2.2 Review for Corda-based Supply Chain System

A Corda-based Supply Chain System leverages Corda's distributed ledger technology, designed to offer unique advantages for enterprise applications, particularly in supply chains that demand high levels of privacy, efficiency, and interoperability. One of the primary strengths of Corda is its privacy and confidentiality model. Unlike public blockchains, where transaction data is visible to all participants, Corda allows participants to maintain private, shared ledgers. Only the parties involved in a transaction have access to the relevant data, ensuring that sensitive business information, such as pricing, contracts, and logistics data, remains confidential. This privacy feature is critical in supply chains, where different parties—such as suppliers, manufacturers, and distributors—may not want to reveal their proprietary data to competitors.

Additionally, interoperability is another key strength of Corda. Unlike many blockchain solutions, Corda is designed to allow different organizations, industries, and even countries to interact seamlessly. It enables various supply chain partners to integrate their existing systems, whether they are legacy systems or other blockchain networks, ensuring smooth and efficient communication across the entire supply chain. This integration capability helps to create a more connected and streamlined ecosystem, improving the overall efficiency of the supply chain. Corda also supports smart contracts, which are self-executing contracts with the terms of the agreement directly written into code. These smart contracts automate complex processes such as tracking inventory, processing payments, and ensuring regulatory compliance. This automation reduces human errors, minimizes the risk of fraud, and speeds up transaction processing, ultimately leading to lower operational costs and improved trust among supply chain participants (Vukolic, 2016).

However, despite its strengths, a Corda-based supply chain system has several weaknesses. One of the main challenges is the complexity in setup and maintenance. Setting up a Corda network requires significant technical expertise, especially in configuring the nodes, defining the consensus mechanisms, and developing smart contracts tailored to the specific needs of the supply chain. Furthermore, maintaining the network involves continuous updates and monitoring, which may require a dedicated team of blockchain experts. This complexity can be resource-intensive, particularly for smaller companies that may lack the expertise or budget to manage the system effectively (He & Xu, 2019).

Another weakness is scalability. While Corda is designed to handle a large volume of transactions, it can still face performance bottlenecks as the network grows in size. The more participants there are in the system, the more complex and resource-demanding the consensus process becomes, potentially limiting the platform's ability to scale effectively in large global supply chains. This could lead to slower transaction times and higher operational costs as the number of nodes and transactions increases (Vukolic, 2016). Additionally, the adoption and integration barriers pose a significant challenge. Many organizations in the supply chain industry still rely on traditional, centralized systems and are resistant to adopting blockchain technology due to concerns about the upfront cost, the need to train staff, or the disruption to existing processes. Furthermore, integrating Corda with legacy systems can be challenging, requiring both technical and organizational changes, which may be a significant barrier for some firms (He & Xu, 2019).

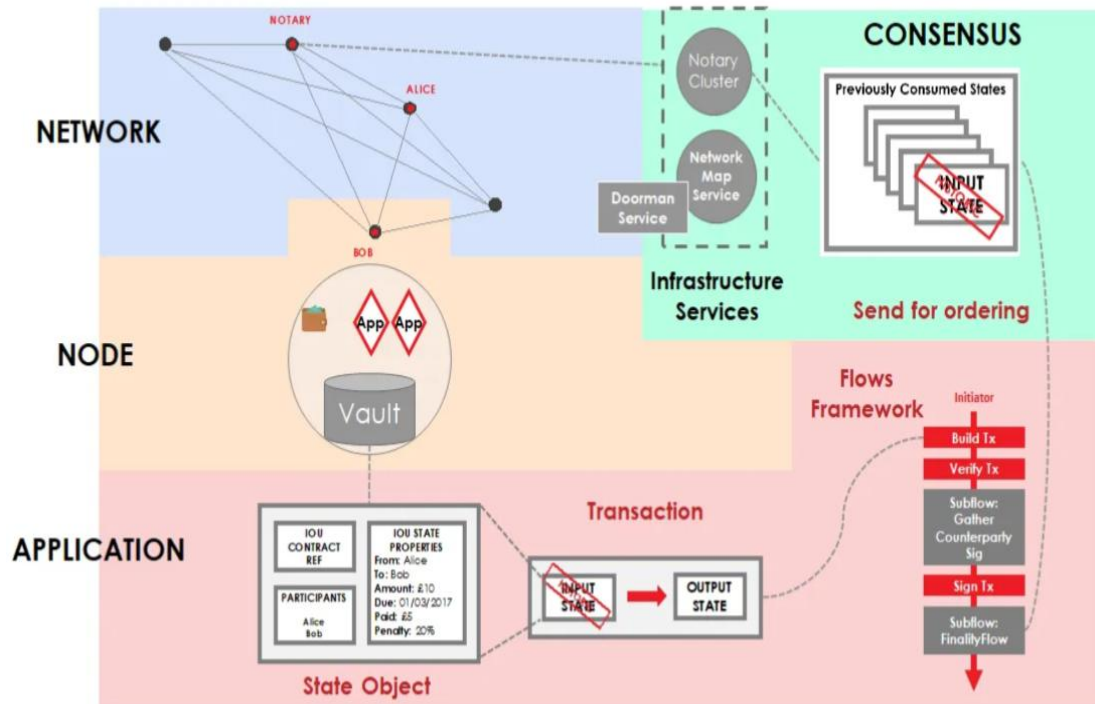


Figure 2 : Corda architecture (Wan, C.,2019).

2.2.3 Review for Hyperledger Fabric Supply Chain System

Hyperledger Fabric is a modular blockchain framework specifically designed to support the development of decentralized applications and solutions in enterprise environments, including supply chain management. One of its key strengths is its permissioned nature, which allows for fine-grained access control and enhanced privacy by ensuring that only authorized participants can access certain data. This is particularly important in a supply chain context, where sensitive information, such as product origins and ownership, needs to be protected (Androulaki et al., 2018).

Hyperledger Fabric uses a pluggable consensus mechanism, meaning that organizations can choose the appropriate consensus algorithm depending on their needs. This flexibility is beneficial for supply chains with varying requirements regarding trust and performance. Another strength of Hyperledger Fabric is its support for smart contracts, called "chaincode," which can be written in languages such as Go, JavaScript, and Java, allowing developers to use familiar tools for automating business logic within the supply chain (Ranjan & Thomas, 2019).

The modular architecture of Hyperledger Fabric enables scalability and high throughput, making it well-suited for large-scale supply chain operations with numerous participants and transactions. Additionally, the system's capability to support private transactions, where only specific participants are privy to transaction details, enhances confidentiality—a vital feature in competitive supply chain environments (Luu et al., 2019).

However, there are some weaknesses associated with Hyperledger Fabric. The complexity of its setup and the requirement for specialized knowledge in blockchain

development can be a barrier to entry for organizations without in-house expertise. Additionally, while its modular approach provides flexibility, it can also lead to increased overhead in terms of system maintenance and integration. Furthermore, Hyperledger Fabric's reliance on a permissioned network limits its applicability for scenarios that require complete decentralization, such as those found in public blockchain networks (Androulaki et al., 2018).

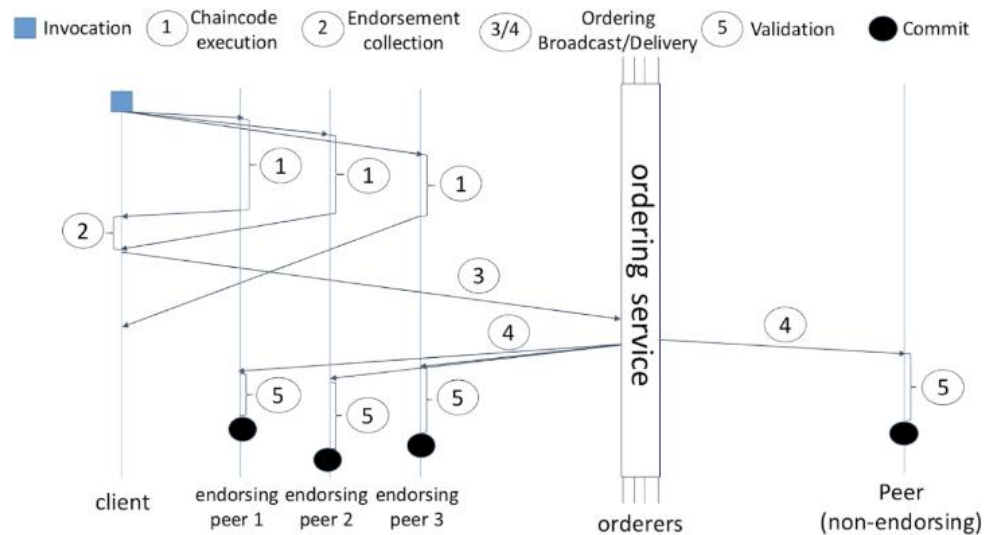


Figure 3: Fabric high level transaction flow (Androulaki et al., 2018).

2.2.4 Review for proposed Supply Chain System

The proposed Supply Chain System utilizing Ethereum, Hardhat, and Solidity aims to enhance transparency, traceability, and security within industries like livestock management. Ethereum's decentralized blockchain architecture ensures that data is immutable and tamper-proof, addressing major concerns in food safety and authenticity. Smart contracts, written in Solidity, automate critical processes such as ownership transfers and product tracking, eliminating intermediaries and reducing associated costs. By integrating Internet of Things (IoT) devices with the blockchain, real-time data about livestock health, movement, and conditions can be recorded and verified, providing a transparent and reliable history of the product's lifecycle (Narayanan et al., 2016). This transparency not only helps combat fraud but also allows all stakeholders to access verified information, building trust across the supply chain.

Despite its strengths, the system faces several challenges. Ethereum's current scalability limitations, particularly concerning transaction throughput, could hinder the system's ability to handle high volumes of data and transactions. Although Ethereum 2.0's transition to a Proof-of-Stake (PoS) consensus mechanism is expected to improve scalability and reduce energy consumption, the full implementation of this upgrade is still in progress (Buterin, 2020). Additionally, the cost of executing smart contracts, known as "gas fees," could become prohibitive during periods of high network congestion (Buterin, 2017). The complexity of developing and maintaining secure, efficient smart contracts is another challenge, especially for smaller producers lacking blockchain expertise, potentially leading to higher costs for implementation and ongoing maintenance (Mougayar, 2016).

Moreover, regulatory compliance remains a significant concern. While blockchain offers enhanced transparency, aligning the system with global regulations, especially those concerning data privacy, such as the General Data Protection Regulation (GDPR) in Europe, could pose legal challenges (Tapscott & Tapscott, 2016). The transparency inherent in blockchain may conflict with privacy requirements in some jurisdictions. Despite these challenges, the integration of blockchain in supply chain management has the potential to revolutionize industries by offering improved traceability, greater efficiency, and stronger safeguards against fraud. As blockchain technology continues to evolve and scaling solutions like Layer 2 protocols become more common, the system's benefits will likely outweigh its current limitations.

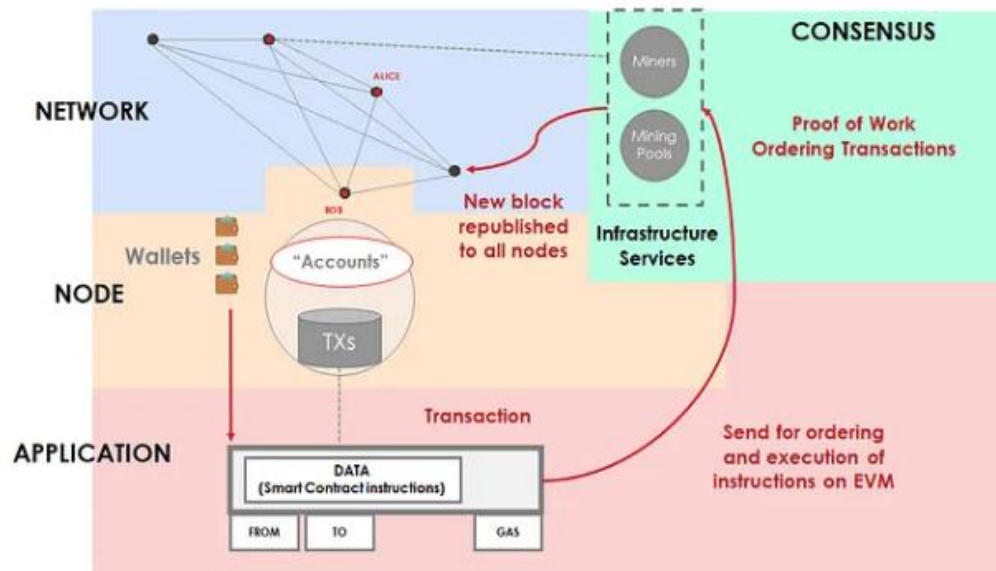


Figure 4 : *Ethereum blockchain architecture* (Wan, C., 2019).

2.3 Comparison between three existing type systems and proposed system

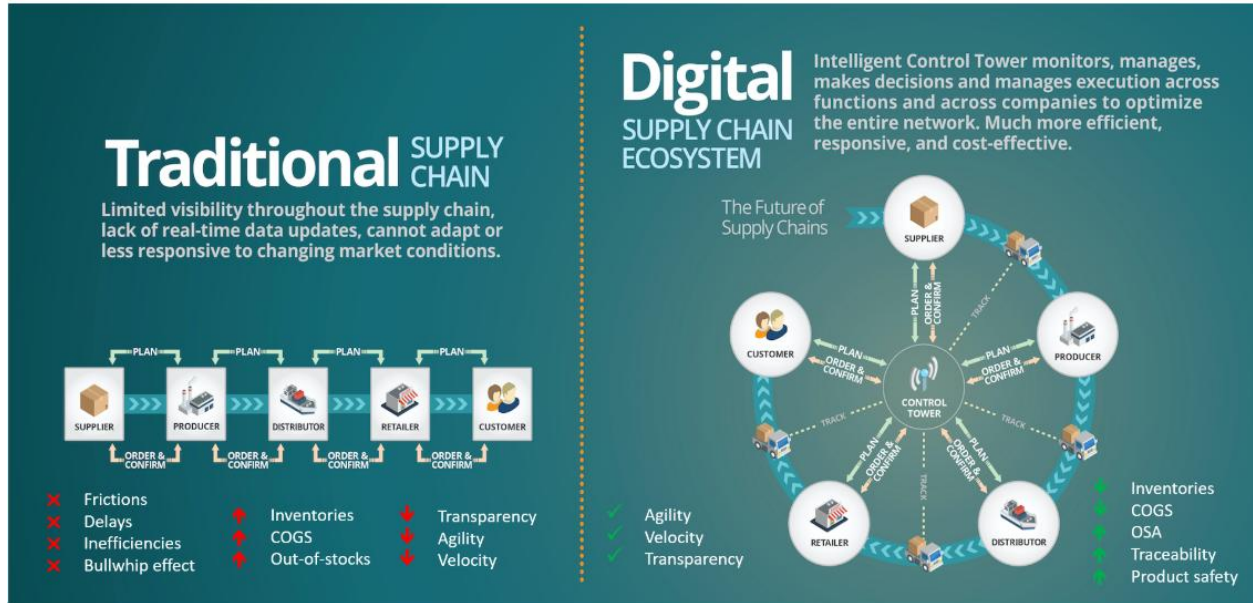


Figure 5 : Comparison between traditional & digital SCM

The traditional supply chain system, despite its simplicity and low cost, is increasingly inadequate due to inefficiencies, lack of automation, data silos, and vulnerability to fraud. The Corda-based system, designed for privacy and interoperability, offers significant advantages in these areas but faces challenges with scalability, complexity, and adoption. Similarly, the Hyperledger Fabric system offers permissioned access and modularity, making it suitable for enterprise-level operations, but it also has barriers in setup, integration, and scalability. The proposed Ethereum-based system, leveraging decentralized blockchain technology, aims to tackle issues like transparency, fraud prevention, and traceability but encounters challenges in scalability, gas fees, and regulatory compliance. Below is a detailed comparison of the existing systems and the proposed Ethereum-based system:

Table 1 : Comparison between three existing type systems and proposed system

Criteria	Traditional Supply Chain	Corda-based Supply Chain	Hyperledger Fabric Supply Chain	Proposed Ethereum-based Supply Chain
Privacy	Relies on centralized databases	High privacy with shared ledgers for participants involved only.	Permissioned network with fine-grained access control.	Public ledger, but privacy ensured via smart contracts and encryption.
Interoperability	Limited due to data silos and manual processes.	Strong interoperability between organizations, industries, and countries.	Good interoperability, especially for enterprise systems.	Requires integration with IoT devices and third-party systems for full interoperability.
Smart contract life cycle	none	Requires node-level administrative operations to deploy/update.	Requires elaborate process to deploy/change. Stored off-chain.	Immutable. Easy to deploy. Stored on-chain.

Scalability	Limited by manual processes and centralization.	Can face bottlenecks with a large number of participants.	Scalable but dependent on network setup and modular components.	Limited scalability due to Ethereum's transaction throughput
Transparency	Poor transparency; reliance on paper-based documentation.	data visible only to involved parties, limiting transparency.	High transparency for permissioned participants, with private transactions.	High transparency; data on the blockchain is immutable and accessible to all stakeholders.
Security & Fraud Prevention	Vulnerable to fraud, errors, and counterfeiting due to lack of automation.	High security with smart contracts and private ledgers.	High security with private transactions and chaincode.	High security with immutable blockchain records
Cost	Low initial cost but high operational costs due to inefficiencies.	High setup and maintenance cost due to technical complexity.	Moderate cost, but the modular approach can increase overhead.	High setup cost due to Ethereum's complexity and potential gas fees during congestion.

Adoption and Integration				Adoption could
				be challenging
	Easy adoption	Complex setup	Easier adoption	for non-
	for smaller and		for enterprises,	technical
	companies but	integration,	but requires	stakeholders;
	limited	especially for	blockchain	integration with
	scalability.	smaller firms.	expertise.	existing
				systems is
				complex.

2.4 Review on tools or technology used

The proposed system leverages several key tools and technologies that enhance its functionality and performance. Solidity is the programming language of choice for writing smart contracts on the Ethereum blockchain. It supports advanced features like inheritance and modifiers, which allow for secure and efficient contract management. Its ability to handle complex data structures makes it ideal for managing supply chain records, ensuring data integrity and transparency.

Hardhat is used as the development and testing framework for the Ethereum-based blockchain system. It offers a comprehensive suite of testing tools that allow developers to simulate real-world interactions and debug the system efficiently. The framework provides valuable features such as automated contract testing, network simulation, and advanced debugging capabilities, ensuring that all system components function as expected before deployment.

To facilitate interaction between the frontend and the Ethereum blockchain, Ethers.js is used. This lightweight JavaScript library allows seamless integration with the Ethereum network, enabling secure transactions between the smart contracts and the user interface. It supports features such as wallet integration, contract interaction, and event handling, all of which are crucial for the smooth functioning of the system.

For the frontend development, Bootstrap is utilized to create a responsive and user-friendly interface. Its predefined components and grid system ensure that the application is adaptable to various devices, enhancing accessibility for users across different platforms. Bootstrap speeds up the development process while maintaining design consistency, making it an essential tool for building an intuitive and engaging user experience.

Together, these technologies form the backbone of the proposed blockchain authentication system, ensuring a secure, scalable, and user-friendly solution for supply chain management. Compared to other tools, such as Truffle and Web3.js, the selected tools offer a balanced combination of simplicity, performance, and flexibility, ensuring the system meets the diverse needs of its stakeholders.

2.5 Summary

This chapter addresses the challenges in supply chain management, such as counterfeiting, inefficient data handling, and poor visibility, and explores how blockchain technology can provide solutions through secure and transparent tracking of goods. It compares the traditional supply chain system, which relies on paper and centralized databases, with two blockchain systems: Corda, a permissioned blockchain offering privacy and scalability, and Hyperledger Fabric, a modular, enterprise-focused blockchain that requires technical expertise. The proposed blockchain solution combines the strengths of both Corda and Hyperledger Fabric, offering enhanced transparency, security, and scalability to address the weaknesses of traditional systems, ultimately improving the efficiency and traceability of supply chains.

CHAPTER 3 : REQUIREMENT ANALYSIS AND DESIGN

3.1 Introduction

Blockchain technology has emerged as a groundbreaking innovation that underpins numerous applications beyond its initial use in cryptocurrencies. At its core, blockchain is a decentralized and distributed ledger that ensures transparency, security, and immutability. It organizes data into blocks, each containing a set of transactions. These blocks are linked cryptographically, forming a chain where each block depends on the hash of the previous one. This architecture ensures that data integrity is preserved, as altering any block requires consensus from the entire network. Additionally, blockchain's decentralized nature eliminates reliance on a central authority, fostering trust among participants by allowing real-time access to a shared ledger.

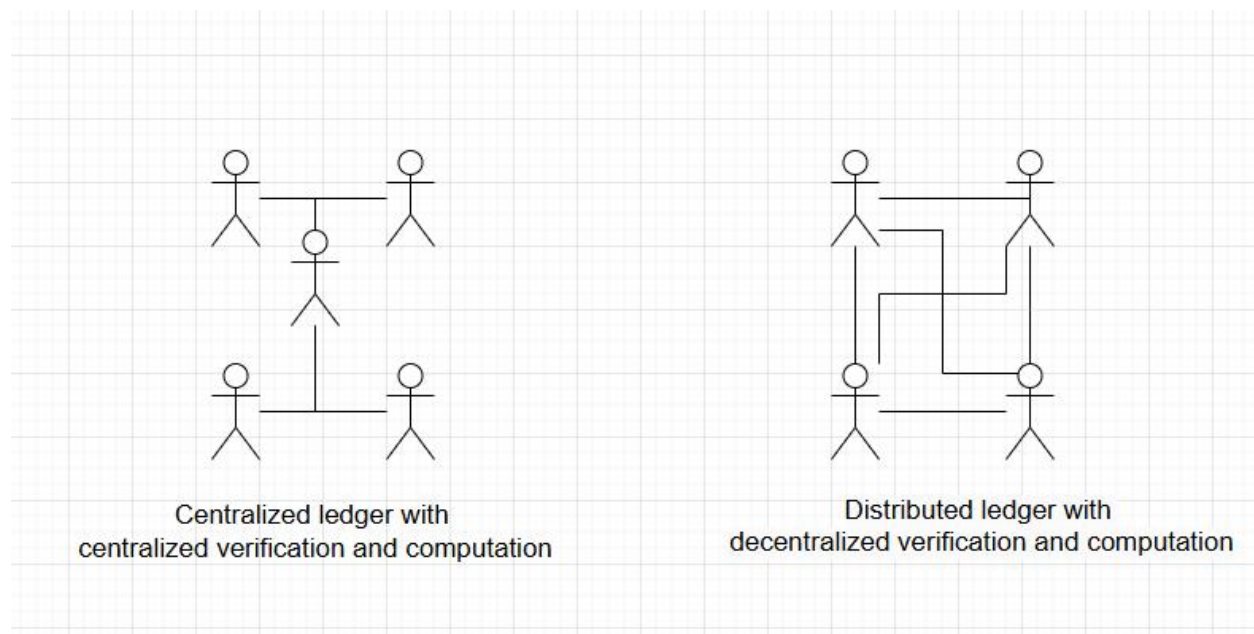


Figure 6 : Comparison of Centralized and Distributed Ledger Systems.

The transformative potential of blockchain is particularly evident in supply chain management, where issues such as fraud, inefficiencies, and lack of transparency prevail. By implementing blockchain, each transaction in the supply chain can be recorded immutably, enabling stakeholders to verify the authenticity and origin of products without intermediaries. Smart contracts further enhance this capability by automating processes such as ownership transfers and regulatory compliance, reducing human error and operational costs.

Building upon this foundation, the objective of this project is to develop a blockchain-based authentication technique tailored for supply chain management, with a focus on the food and livestock industries. As outlined in Chapter 1, the project aims to address critical challenges in food safety and authenticity by ensuring traceability, security, and transparency. Specifically, it seeks to:

- a) **Study and Identify Suitable Blockchain Authentication Techniques:** Explore various blockchain approaches to determine the most effective method for supply chain product authentication.
- b) **Design and Integrate Blockchain Authentication into Supply Chain Processes:** Develop a system that incorporates blockchain to enhance traceability and security across all stages of the supply chain.
- c) **Develop a Prototype:** Demonstrate the practicality of the proposed technique through a functional prototype that highlights its transparency and reliability.

The scope of the project includes designing a decentralized platform that tracks products throughout the supply chain, capturing essential data such as origin, ownership,

and transit details. By leveraging Ethereum's smart contract capabilities, the system ensures data immutability and accessibility. The focus on the food and livestock sectors underscores the importance of rigorous monitoring and trust in these industries. The anticipated outcome is a system that not only strengthens supply chain transparency and security but also sets the stage for future enhancements, such as IoT integration and private blockchain adoption.

This chapter delves into the detailed requirements and design considerations necessary to achieve these objectives, paving the way for a robust blockchain-based solution.

3.2 Technology/Blockchain Integration Methods

In this section, we explore the methodologies, tools, and platforms utilized for integrating blockchain technology into the proposed system. The system is designed to enhance traceability, transparency, and security in the livestock supply chain. Blockchain integration will ensure that all transactions and data points are immutable, verifiable, and accessible to relevant stakeholders in real time.

The proposed integration leverages a hybrid blockchain model. This approach combines a private blockchain, which is used for storing sensitive data such as ownership history, livestock details, and supplier information, ensuring secure access control for stakeholders, and a public blockchain, which is used for transparency in supply chain transactions, enabling customers and external parties to verify the authenticity of data without compromising sensitive information.

To achieve this integration, several steps are followed. Firstly, smart contracts are developed in Solidity to handle key functionalities such as livestock registration, transfer of ownership, and data validation. These contracts are deployed on the Ethereum blockchain using Hardhat for testing and deployment. Additionally, the system incorporates tokenization, where each livestock unit is represented as a unique digital token on the blockchain. These tokens serve as digital twins of physical assets, enabling efficient tracking, trading, and ownership management. Token lifecycle events, including creation, transfer, and metadata updates, are seamlessly managed by the smart contracts. Secondly, the front-end development utilizes HTML, CSS, and JavaScript with Bootstrap to create an interactive and user-friendly interface for stakeholders, with blockchain functionality integrated through Metamask to enable wallet-based authentication and interaction. Thirdly, middleware services are developed using Ethers.js to bridge the front-end application and blockchain network. Ethers.js is employed for seamless communication between the front-end and blockchain, enabling functionalities such as reading data from and writing data to the blockchain. Additionally, Hardhat is utilized for local blockchain testing and debugging smart contracts. Finally, monitoring of the deployed smart contracts is performed using Etherscan and custom logging mechanisms.

A variety of tools and platforms are utilized for this integration. Ethereum serves as the blockchain platform for deploying smart contracts, while Hardhat provides the development environment for building and testing these contracts. MetaMask facilitates wallet integration for user authentication, and HTML, CSS, and JavaScript with Bootstrap are used for front-end development. On the backend, Ethers.js handles API and blockchain interaction, and Etherscan aids in monitoring smart contract activity.

The system architecture diagram illustrates the integration flow. The user frontend, built with HTML, CSS, JavaScript, and Bootstrap, interacts with the blockchain node through Ethers.js, which communicates with the smart contracts deployed on Ethereum. The middleware API bridges the front-end application, blockchain network, and decentralized storage system like IPFS. This ensures that user interactions are recorded immutably on the blockchain while additional metadata is stored off-chain for efficiency.

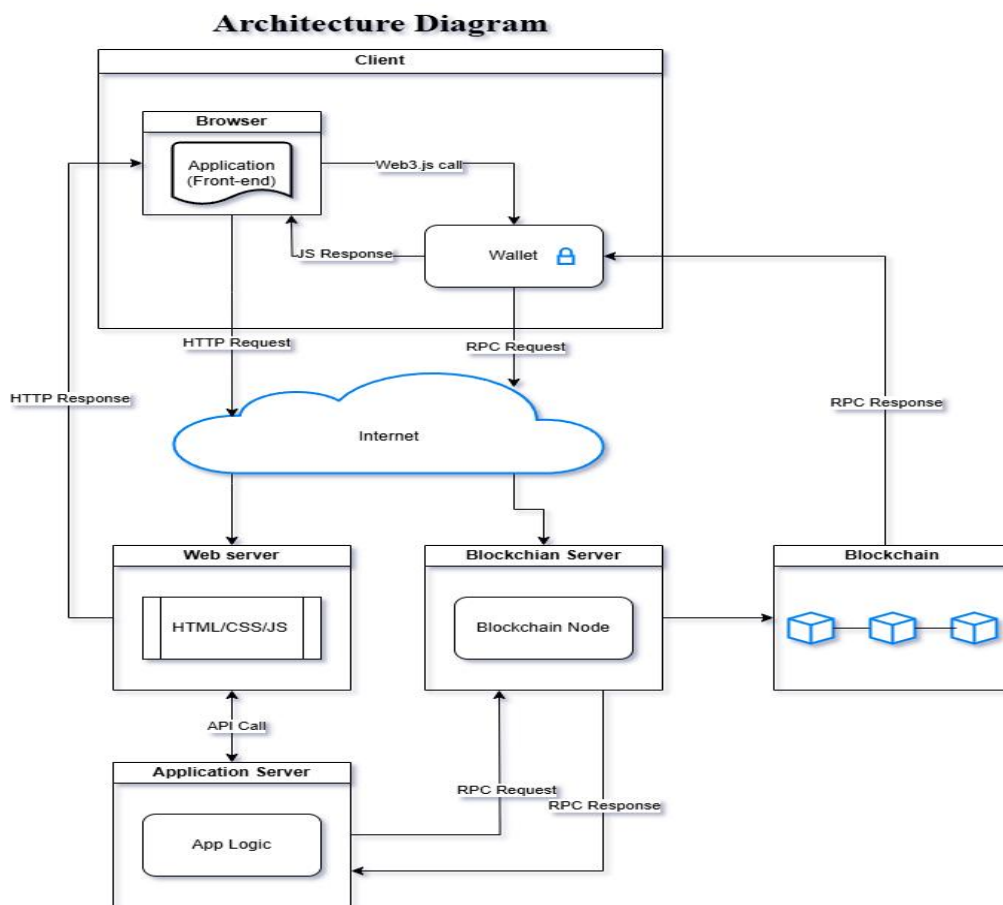


Figure 7 : Architecture diagram.

This integration offers several benefits. Enhanced transparency allows customers to verify the journey and authenticity of livestock products, while improved traceability ensures that all stages of the supply chain are recorded on an immutable ledger. Data security is maintained through private blockchain and access control, scalability is achieved through the hybrid model, and interoperability allows for potential integration with other platforms and blockchain networks in the future.

By leveraging these integration methods and tools, the proposed system will set a benchmark in blockchain-based supply chain management systems, ensuring trust and transparency for all stakeholders.

3.2.1 Hybrid Blockchain Justification

The hybrid blockchain model is increasingly recognized for its ability to merge the strengths of both public and private blockchains while addressing their inherent limitations. Public blockchains offer transparency and verifiability, which are essential for external stakeholders like consumers and regulators to independently verify supply chain transactions. This transparency builds trust but can lead to challenges such as slower transaction speeds and higher costs due to less efficient consensus mechanisms (Liu et al., 2022). On the other hand, private blockchains enhance data confidentiality and improve transaction speeds by restricting access to authorized participants, which is crucial for managing sensitive information like supplier contracts and proprietary methods. However, this restricted access can raise concerns regarding data integrity and external trust.

By integrating these two approaches, the hybrid blockchain model effectively utilizes

private blockchains to securely store sensitive data and execute transactions efficiently, while public blockchains are employed to record verifiable transaction proofs, thereby ensuring transparency and trustworthiness. This dual approach helps mitigate the trade-offs between transparency and privacy, security and efficiency, as well as control versus decentralization. Sensitive operations are protected within the private blockchain, while essential information is selectively shared on the public blockchain to maintain external accountability. Additionally, this model supports scalability and interoperability with other platforms, making it a robust choice for applications such as the livestock supply chain. This integration provides stakeholders with the necessary visibility to verify authenticity and ensures the overall integrity, reliability, and performance of the system (Ge et al., 2022).

In summary, the hybrid blockchain model represents a balanced solution that leverages the advantages of both public and private blockchains, addressing their limitations while enhancing trust and efficiency in supply chain management.

3.2.2 Tokenization in Blockchain-Based Supply Chain Systems

Tokenization is a transformative concept in blockchain technology, particularly suited for supply chain management. It represents physical or intangible assets as digital tokens on a blockchain, ensuring efficient tracking, secure ownership transfer, and enhanced transparency throughout the supply chain. Each token acts as a unique identifier tied to an asset, enabling immutable recording of critical information such as origin, ownership history, quality metrics, and transactional details. By embedding these attributes into a token, supply chain stakeholders can access real-time, verifiable data, facilitating a more transparent and efficient process.

Moreover, it also simplifies ownership transfer through the use of blockchain's smart contracts. These contracts automate token issuance and transactions, reducing the reliance on intermediaries and manual processes, which are prone to errors and fraud. For example, in the context of livestock management, each livestock unit can be tokenized, and the corresponding token would include metadata such as vaccination records, weight, origin, and transportation history. As livestock moves through the supply chain—from farm to processor to retailer—the token is updated to reflect changes in ownership and condition. This real-time update ensures the authenticity and traceability of the livestock, building trust among stakeholders.

One of the significant advantages of tokenization is fraud prevention. The immutable nature of blockchain ensures that tokens cannot be tampered with, making

counterfeit goods and fraudulent transactions easily detectable. Additionally, tokenized systems offer a high degree of automation. By leveraging smart contracts, tokens can be automatically updated or transferred when predefined conditions are met, streamlining operations and reducing administrative overhead. Moreover, tokenization enhances consumer confidence by providing end-users with a transparent view of the product journey. For instance, consumers can scan a QR code linked to a token and verify the product's authenticity, origin, and quality attributes directly on the blockchain. This capability is especially crucial in industries like food and livestock, where consumer trust heavily depends on the assurance of product authenticity and safety.

Tokenization Process

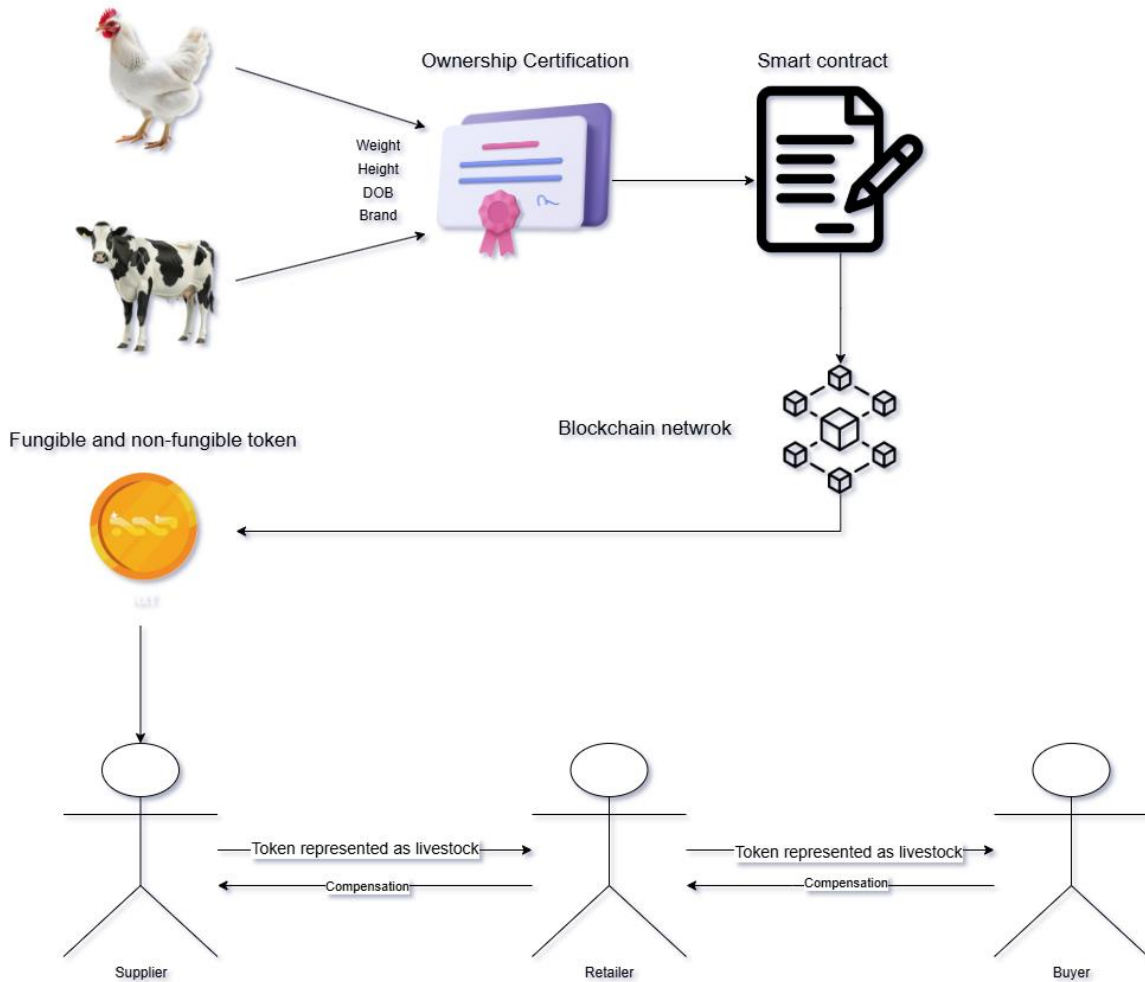


Figure 8 : Tokenization process.

3.3 System Work Methods

3.3.1 Use case diagram

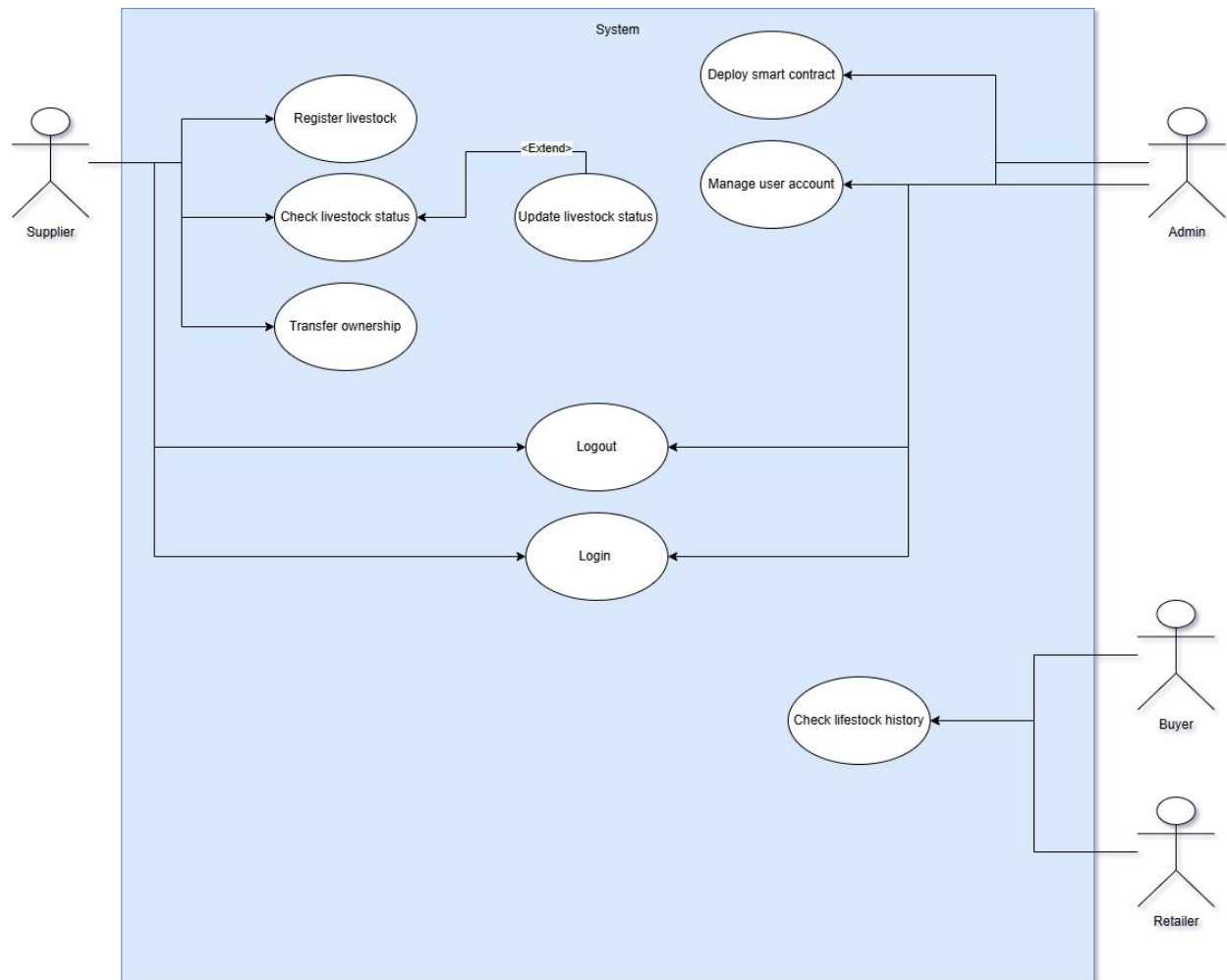


Figure 9 : Use case diagram.

This system is a comprehensive platform designed to facilitate the tracking and management of livestock through various stages of the supply chain, utilizing blockchain technology. The system accommodates four distinct user roles: Suppliers, Admin, Buyers,

and Retailers, each with specific permissions and functionalities tailored to their responsibilities within the ecosystem.

Suppliers, who are primary users of the system, have access to several crucial functions. They can register new livestock in the system, which is entering detailed information such as breed, age, health status, and other relevant metrics. Suppliers can also monitor the status of their registered livestock through the check livestock status feature, which can be extended to update the livestock's information as needed through the update livestock status function. This extend relationship indicates that while checking status is a standalone function, it can naturally lead to updates when necessary. Additionally, suppliers have the ability to transfer ownership of livestock, likely when sales or transfers occur within the supply chain. Like all users, suppliers must authenticate themselves through the login function and can safely exit the system using the logout function.

Administrators play a crucial oversight role in the system. They are responsible for deploying smart contracts, which suggests that the system operates on blockchain technology to ensure transparency and immutability of livestock records. Administrators also manage user accounts, which likely includes creating new accounts, modifying access permissions, and handling any user-related issues that arise within the system.

Buyers and Retailers, who represent the end users in the supply chain, have more focused access to the system. Their primary function is to check livestock history, which presumably allows them to verify the authenticity and trace the journey of livestock

products through the supply chain. This feature is particularly important for ensuring transparency and maintaining quality standards in the livestock industry.

3.3.2 Use case specification

Table 2 : Use case specification for register livestock

Use Case	Register Livestock
Actor	Supplier
Description	Allows suppliers to register new livestock in the system
Pre-conditions	1. Supplier is logged into the system. 2. Supplier has the required livestock information.
Basic Flow	1. Supplier selects "Register Livestock" option. 2. System displays registration form. 3. Supplier enters livestock details (ID, breed, age, health status, etc.) 4. System validates the entered information. 5. System alert successful registration message.
Alternative Flow	4a. Invalid information entered: 1. System displays error message. 2. Returns to step 3.

	5a. System fails to store information: 1. System displays error message. 2. Prompts to retry.
Post-conditions	1. New livestock is registered in the system. 2. Supplier can view the registered livestock

Table 3 : Use case specification for check livestock status

Use Case	Check Livestock Status
Actor	Supplier
Description	Enables suppliers to view current status of registered livestock
Pre-conditions	1. Supplier is logged into the system. 2. Livestock exists in the system
Basic Flow	1. Supplier selects "Check Status" option. 2. System displays livestock interface 3. Supplier enters livestock ID. 4. System retrieves and displays livestock information
Alternative Flow	3a. Invalid livestock ID: 1. System displays error message 2. Returns to step 2
Post-conditions	1. Livestock status information is displayed

Table 4 : Use case specification for update livestock status

Use Case	Update Livestock Status
Actor	Supplier
Description	Allows suppliers to update the status of existing livestock
Pre-conditions	1. Supplier is logged into the system. 2. Livestock exists in the system. 3. Supplier has checked livestock status
Basic Flow	1. Supplier selects "Update Status" option. 2. System displays update form. 3. Supplier enters new status information. 4. System validates the updates. 5. System alert successful registration message.
Alternative Flow	4a. Invalid update information: 1. System displays error message 2. Returns to step 3
Post-conditions	1. Livestock status is updated in the system 2. System logs the update transaction

Table 5 : Use case specification for transfer ownership

Use Case	Transfer Ownership
Actor	Supplier
Description	Enables transfer of livestock ownership to another party
Pre-conditions	1. Supplier is logged into the system. 2. Supplier owns the livestock 3. Recipient account exists in the system
Basic Flow	1. Supplier initiates transfer process 2. System displays transfer form 3. Supplier enters recipient information and transfer details 4. System validates transfer request 5. System executes ownership transfer 6. Both parties receive confirmation
Alternative Flow	4a. Invalid recipient information: 1. System displays error message 2. Returns to step 3
Post-conditions	1. Ownership is updated in the system. 2. Transfer is recorded in blockchain

Table 6 : Use case specification for deploy smart contract

Use Case	Deploy Smart Contract
Actor	Admin
Description	Allows administrators to deploy new smart contracts for system functions
Pre-conditions	1. Admin is logged into the system 2. Smart contract code is verified and tested
Basic Flow	1. Admin initiates contract deployment 2. System validates contract code 3. Admin confirms deployment 4. System deploys contract to blockchain 5. System verifies deployment success
Alternative Flow	2a. Contract validation fails: 1. System displays error details 2. Returns to step 1
Post-conditions	1. New smart contract is deployed 2. Contract is operational on blockchain

Table 7 : Use case specification for manage user account

Use Case	Manage User Account
Actor	Admin
Description	Enables administration of user accounts and permissions

Pre-conditions	1. Admin is logged into the system
Basic Flow	1. Admin accesses user management interface 2. System displays user accounts 3. Admin selects account to manage 4. Admin makes necessary changes 5. System saves changes and responded to the blockchain
Alternative Flow	4a. Invalid changes: 1. System displays error message 2. Returns to step 3
Post-conditions	1. User account changes are saved 2. Changes are logged into blockchian

Table 8 : Use case specification for check livestock history

Use Case	Check Livestock History
Actor	Buyer, Retailer
Description	Allows viewing of livestock's complete historical record
Pre-conditions	1. User/Retailer is logged into the system 2. Livestock exists in the system
Basic Flow	1. User/Retailer enters livestock identifier 2. System retrieves historical data 3. System displays complete livestock

	history
Alternative Flow	1a. Invalid identifier: 1. System displays error message 2. Returns to step 1
Post-conditions	1. Livestock history is displayed to user

Table 9 : Use case specification for login

Use Case	Login
Actor	Admin,Supplier
Description	Authenticates Admin/Supplier to access the system
Pre-conditions	1. Admin/Supplier has valid account 2. Admin/Supplier is not already logged in
Basic Flow	1. Admin/Supplier enters credentials 2. System validates credentials 3. Admin/Supplier access to platform
Alternative Flow	2a. Invalid credentials: 1. System displays error message 2. Returns to step 1
Post-conditions	1. Admin/Supplier is logged into system 2. Admin/Supplier session is created

Table 10 : Use case specification for logout

Use Case	Logout
Actor	Admin,Supplier
Description	Allows Admin/Supplier to safely exit the system
Pre-conditions	1. Admin/Supplier is logged into the system
Basic Flow	1. Admin/Supplier selects logout option 2. System ends Admin/Supplier session 3. System returns to login page
Alternative Flow	2a. Session end fails: 1. System forces session termination 2. Continues to step 3
Post-conditions	1. Admin/Supplier is logged out 2. Session is terminated

3.3.3 Sequence diagram

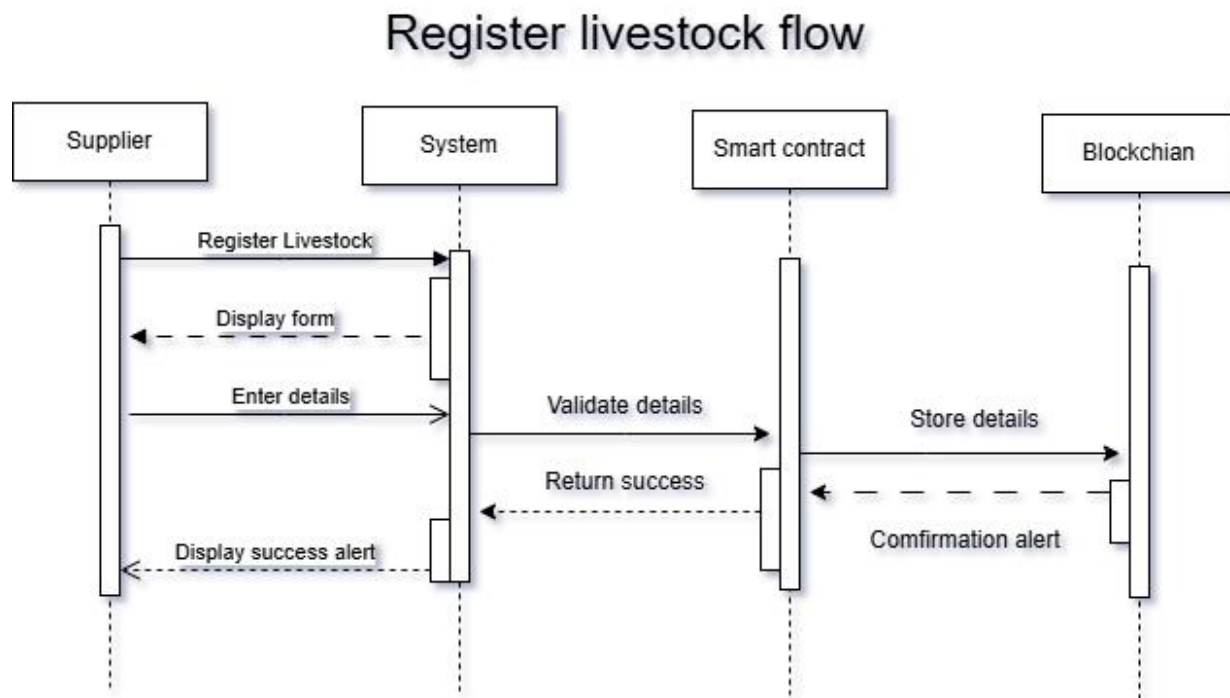


Figure 10 : Sequence diagram for register livestock flow

The "Register livestock flow" diagram shows how a supplier can register new livestock in the system. The process begins with the supplier registering livestock details through a form, which the system validates before storing the information via a smart contract on the blockchain. The system confirms the successful registration back to the supplier.

Check livestock status flow

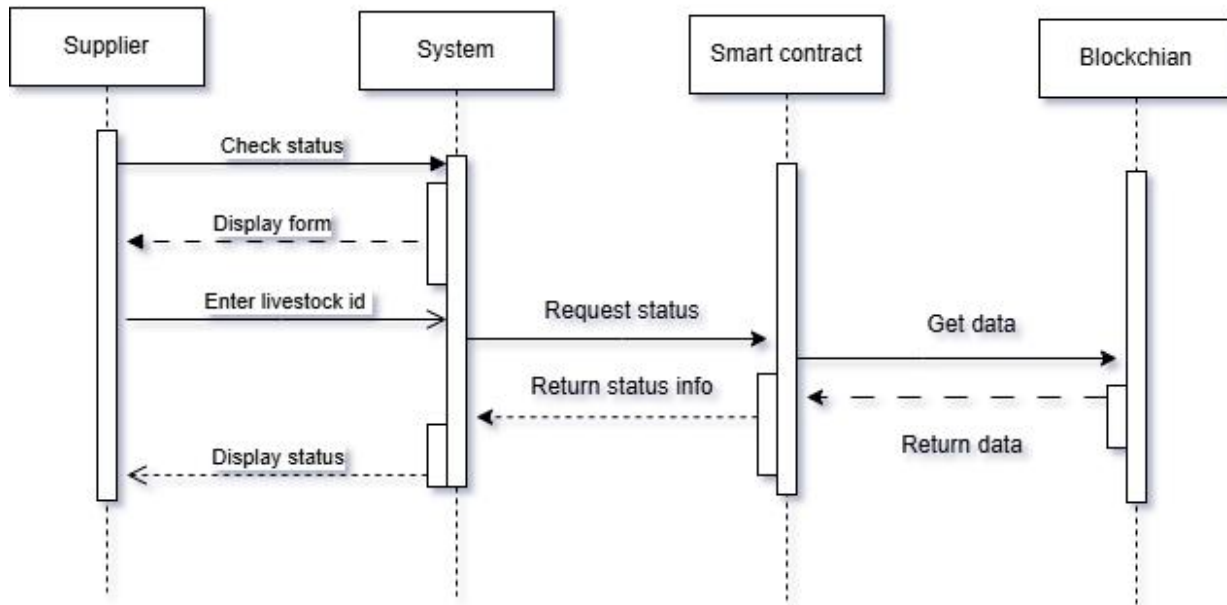


Figure 11 : Sequence diagram for check livestock status flow

The "Check livestock status flow" demonstrates how users can retrieve livestock information. By entering a livestock ID, the system queries the smart contract to fetch the relevant data from the blockchain, which is then displayed to the user.

Update livestock status flow

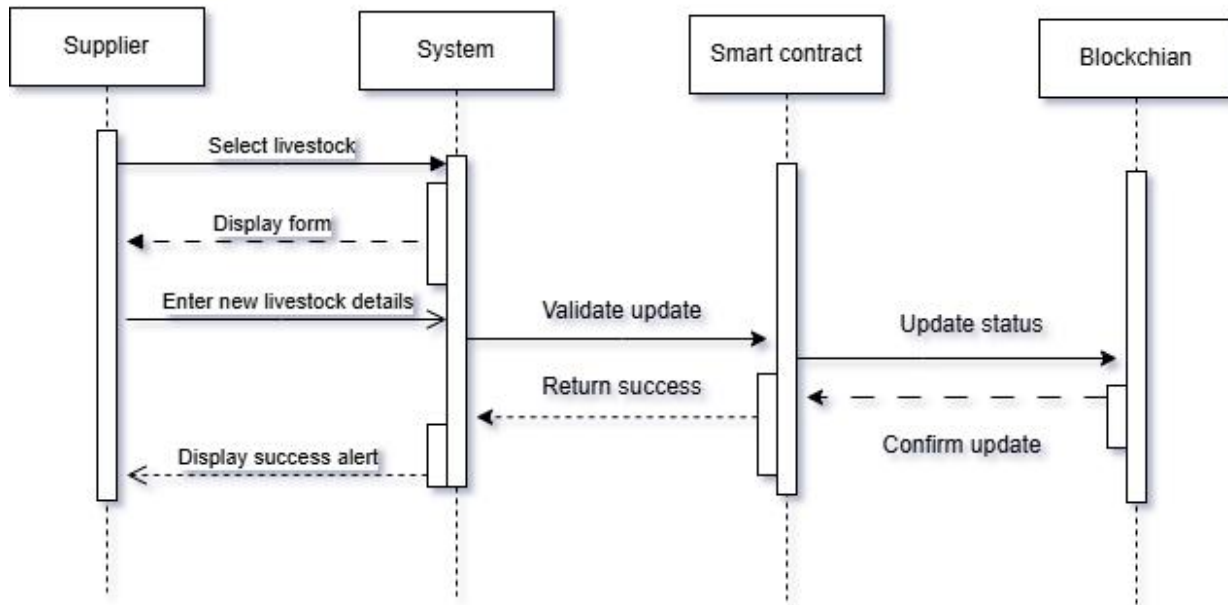


Figure 12 : Sequence diagram for update livestock status flow

The "Update livestock status flow" illustrates how suppliers can modify existing livestock information. They select specific livestock and enter new details, which go through system validation before being updated on the blockchain through the smart contract. The system then displays a success message to confirm the update.

Transfer ownership flow

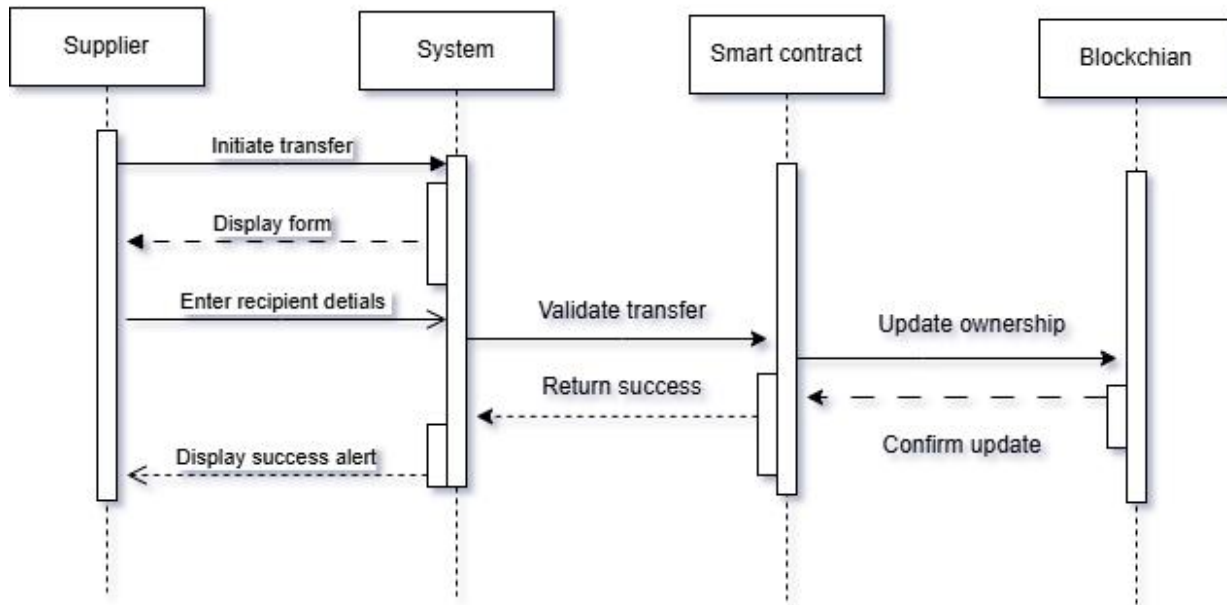


Figure 13 : Sequence diagram for transfer ownership flow

The "Transfer ownership flow" depicts the process of transferring livestock ownership. The supplier initiates the transfer by entering recipient details, which the system validates before updating ownership records on the blockchain via the smart contract. A success message confirms the transfer completion.

Deploy smart contract flow

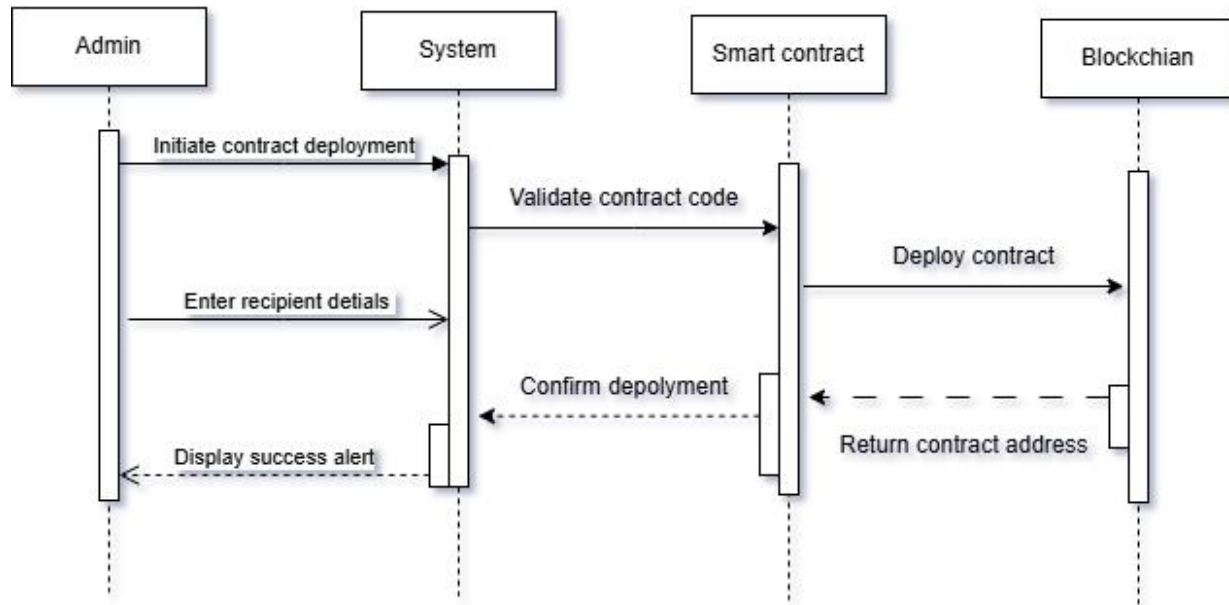


Figure 14 : Sequence diagram for deploy smart contract flow

The "Deploy smart contract flow" shows how an administrator can deploy new smart contracts to the blockchain. The admin initiates deployment and enters necessary details, the system validates the contract code, and upon successful deployment, receives and displays the contract address from the blockchain.

Manage user account flow

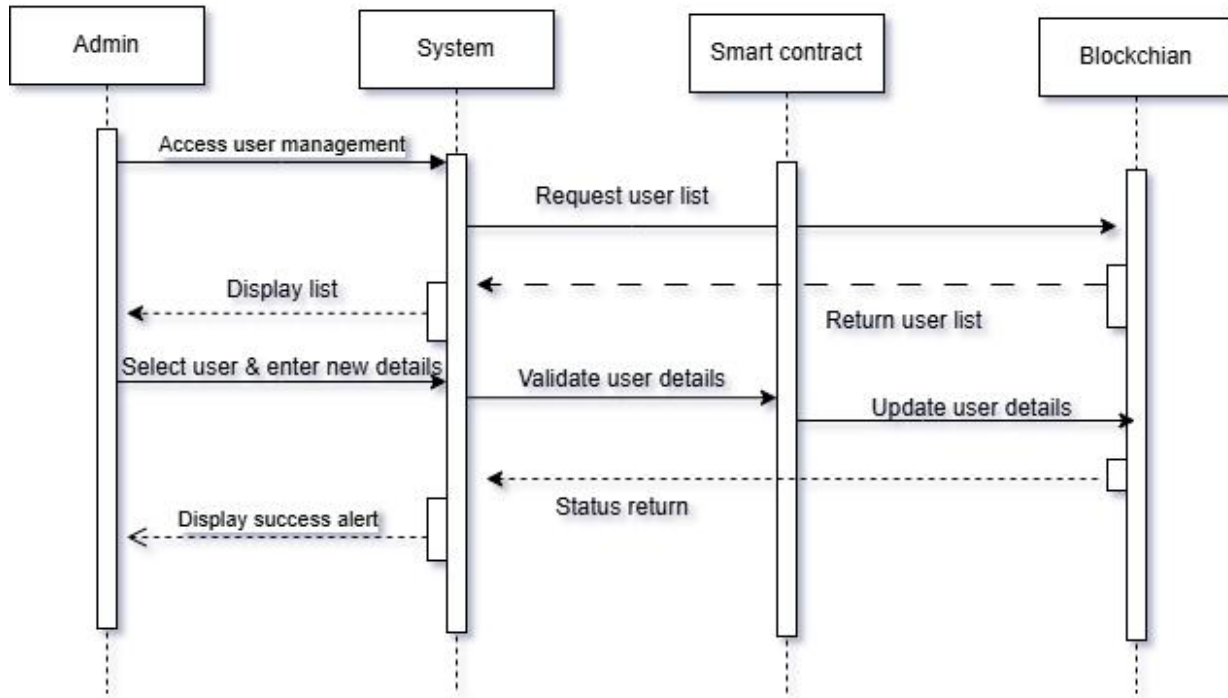


Figure 15 : Sequence diagram for manage user account flow

The "Manage user account flow" illustrates the user management process. Administrators can access user management, view existing users, and modify user details. The system validates these changes before updating them on the blockchain through the smart contract.

Check livestock history flow

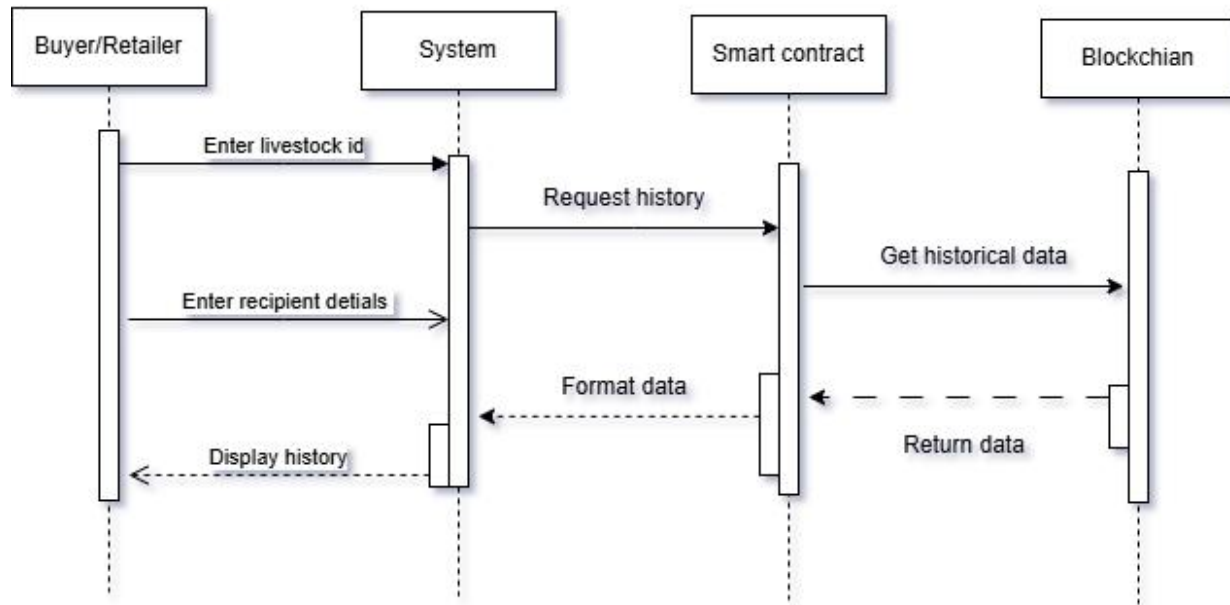


Figure 16 : Sequence diagram for check livestock history flow

The "Check livestock history flow" allows buyers or retailers to view the historical data of specific livestock. They enter the livestock ID and recipient details, and the system retrieves and formats the historical data from the blockchain before displaying it.

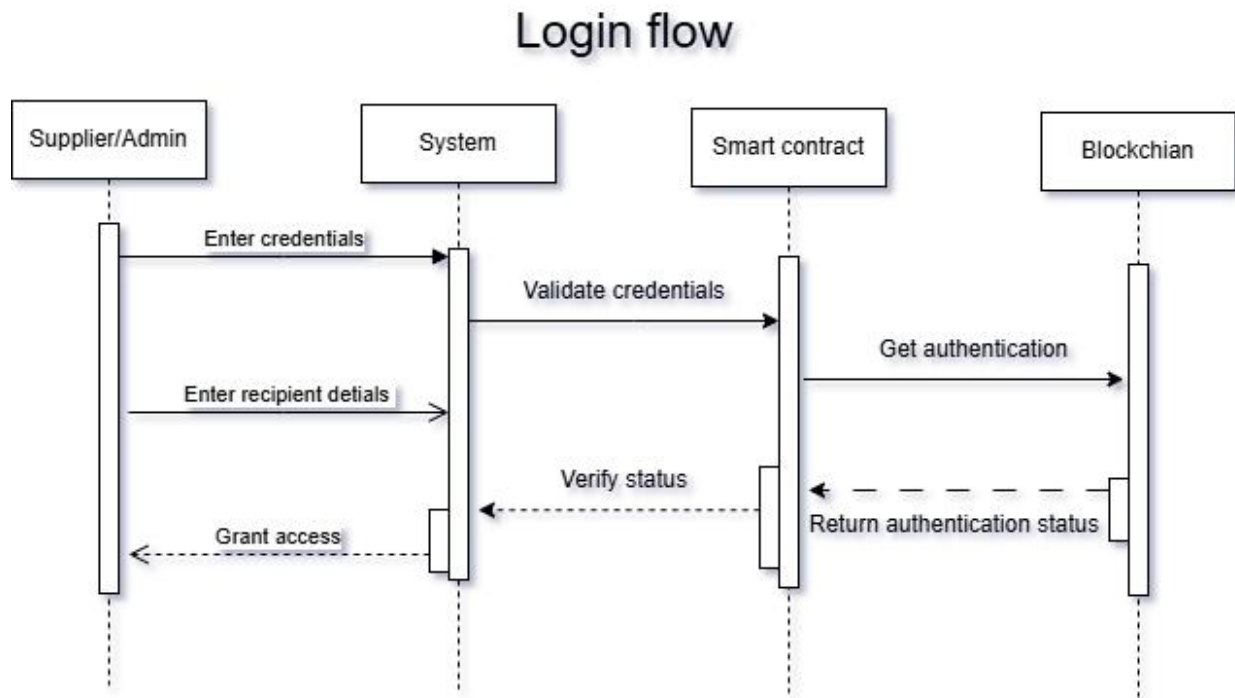


Figure 17 : Sequence diagram for login flow

The "Login flow" shows the authentication process for suppliers and administrators. Users enter their credentials, which the system validates through the smart contract. Upon successful authentication from the blockchain, the system grants access.

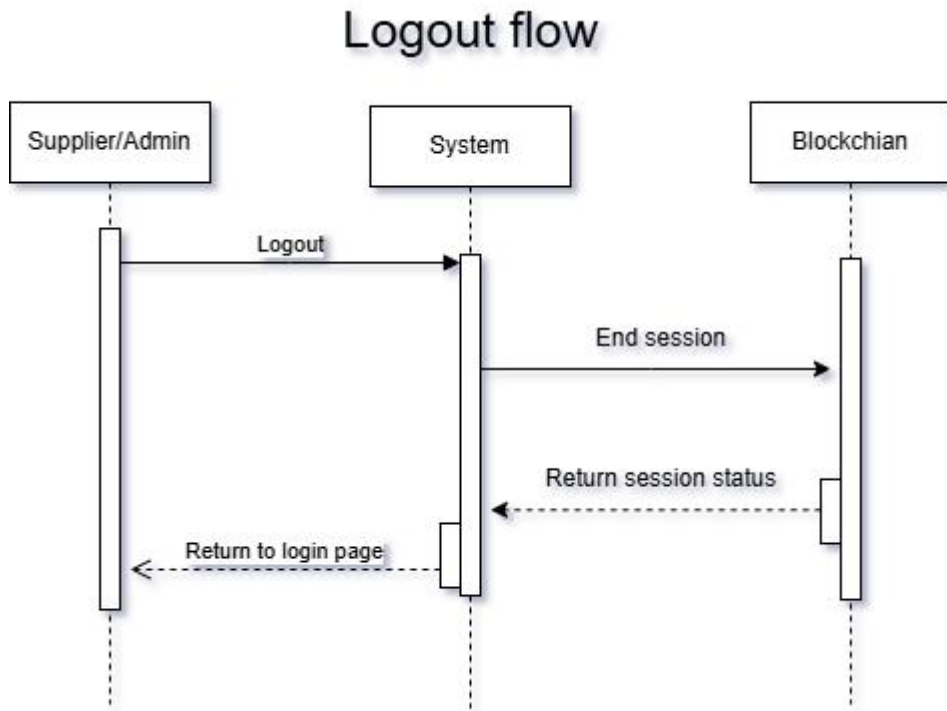


Figure 18 : Sequence diagram for logout flow

The "Logout flow" demonstrates the simple process of logging out, where the user initiates logout, the system ends the session on the blockchain, and returns the user to the login page after receiving confirmation of the session end

3.4 System Mockup

This section will focus on the system mockup of the proposed application. The wireframe of the proposed application is done using Figma. Figma is a collaborative interface design tool that operates in the browser, enabling teams to work together in real-time on user interface design.

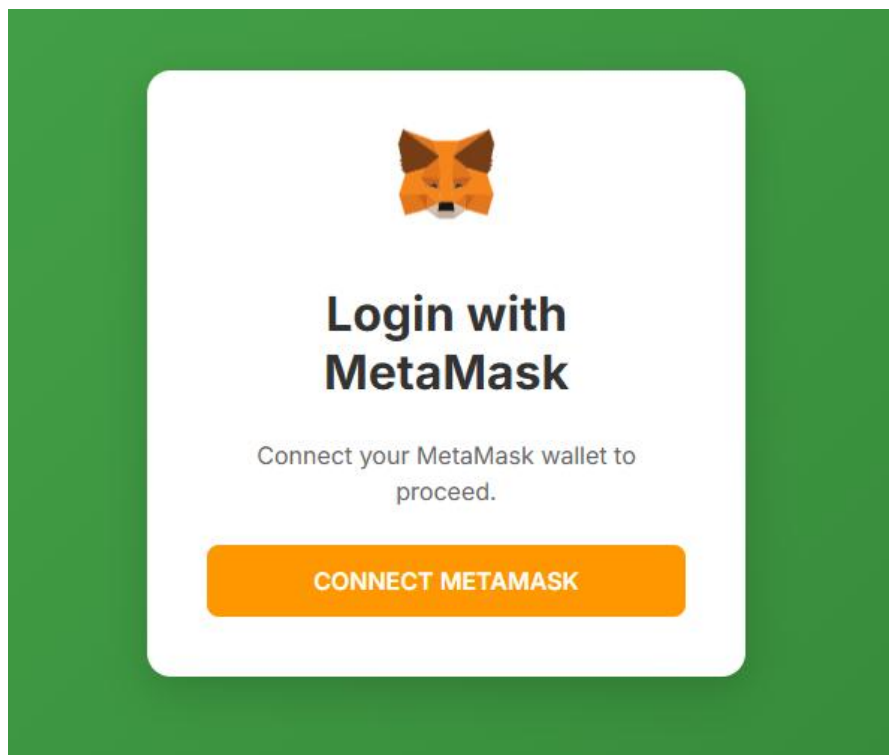


Figure 19 : Login page

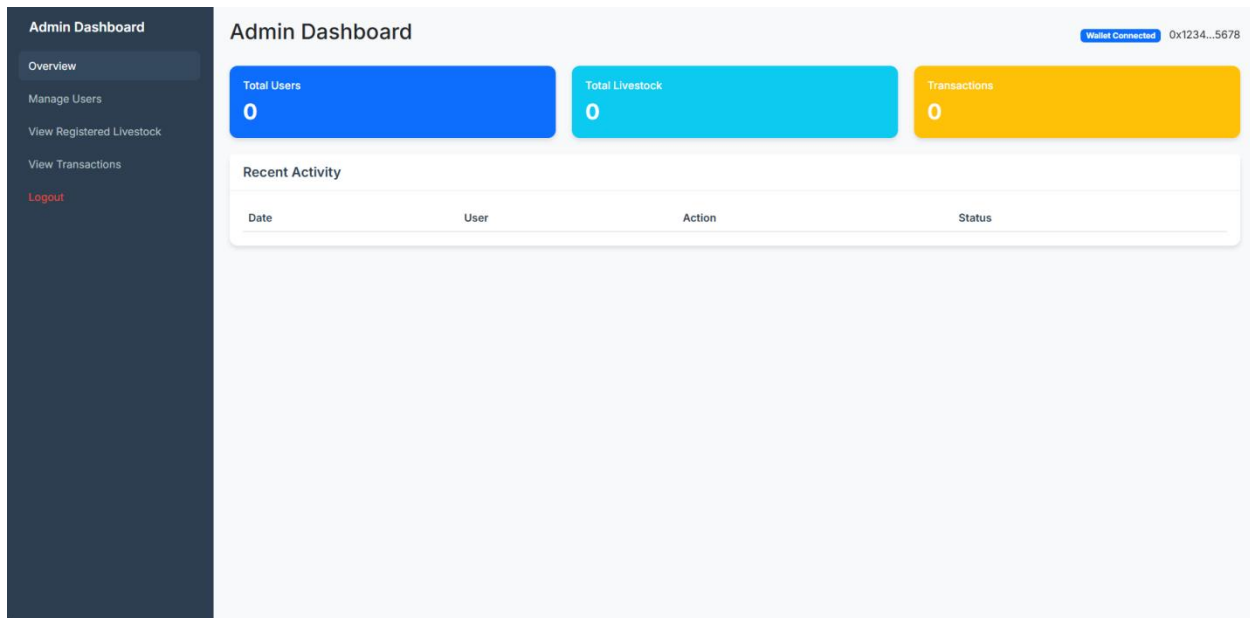


Figure 20 : Admin dashboard

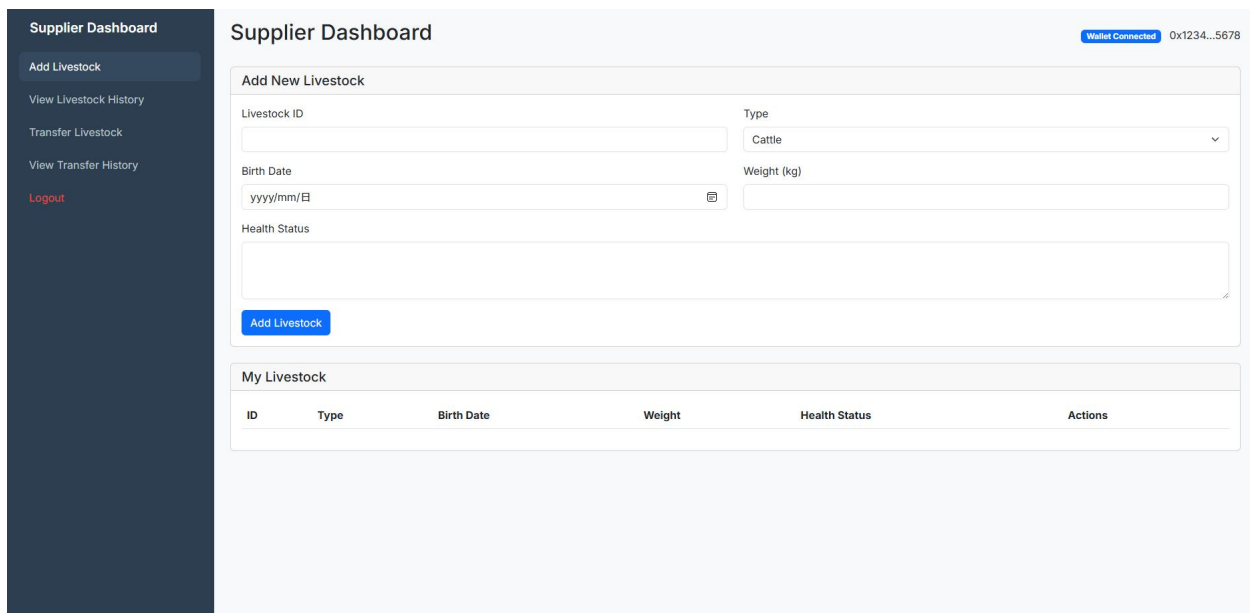


Figure 21 : Supplier dashboard

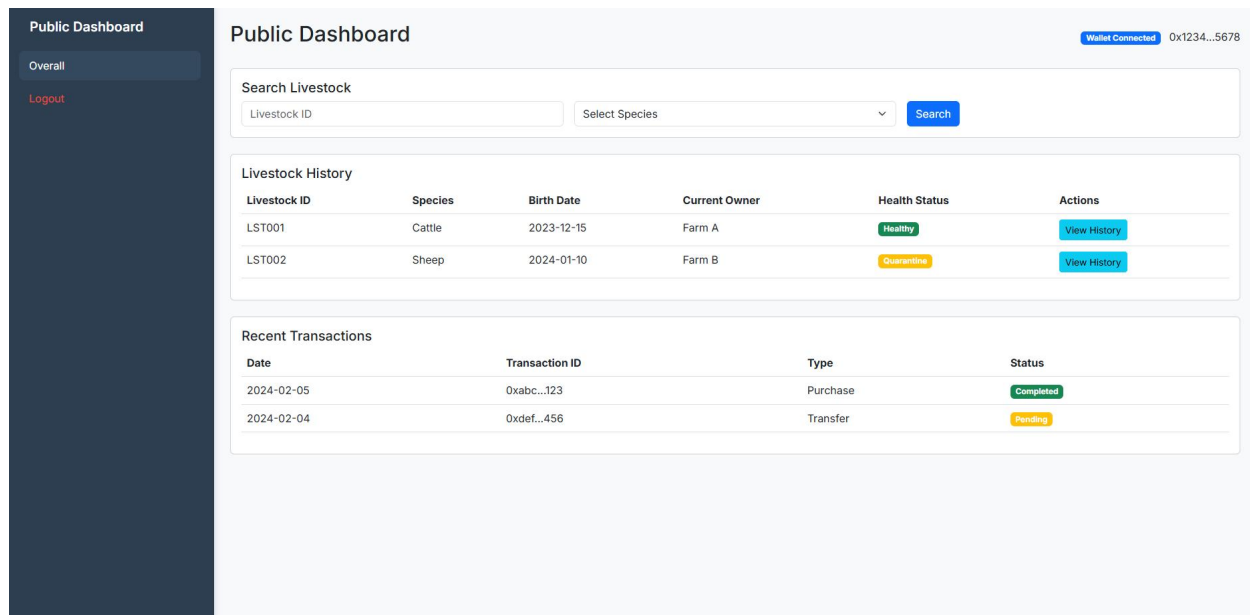


Figure 22 : Public Dashboard

Figure 20 shows the login page for the proposed application. The user need to make sure that the meta-mask e-wallet need to be installed in the browser before the login process. After the system connected to the meta-mask e-wallet it will verify the user and redirect them to different dashboard. If the user is the type of admin , then the redirection will be on the admin dashboard, based on figure 21 the admin dashboard shows the admin dashboard provides a centralized interface for managing livestock-related blockchain activities, featuring key metrics for users, livestock, and transactions in the main view. The left sidebar offers navigation options for user management, view livestock registrations, and view transaction records, while the recent activity table tracks all system actions. Besides, if the user is the type of supplier, then the redirection will be on the supplier dashboard, based on figure 22 the supplier dashboard shows a dedicated interface for

livestock suppliers to manage their inventory, featuring a form to add new livestock with details like ID, type, birth date, weight, and health status. The dashboard includes a data table showing all registered livestock and navigation options for viewing history and transferring livestock, with integrated blockchain wallet connectivity. The left sidebar offers navigation options for add livestock, view livestock history, transfer livestock and view transfer history. Lastly, if the user type is not registered under the system then will be redirected to the public dashboard, based on figure 23 the public dashboard shows the livestock information and livestock transactions history for tracking the product origin.

3.5 Summary

This chapter has covered the requirement analysis and application design for the proposed application, which is a blockchain-based supply chain management system for the livestock sector. First, this chapter specifies the methodology used for the development of this project, which is a hybrid blockchain approach. This includes details on how private and public blockchains are combined to balance security and transparency, along with the tools and techniques needed for integration, such as Solidity for smart contracts and Ethers.js for blockchain communication. Besides, all the requirements for the proposed system have been specified in detail, covering user roles (suppliers, admins, buyers, and retailers), functional requirements (like livestock registration, ownership transfer, and status updates), and non-functional requirements (such as security, scalability, and user-friendliness). The results of tokenization, which creates digital representations of livestock assets, have also been visualized in this chapter. Finally, the design for the system has been specified. The Unified Modeling Language (UML) methods, such as use case diagrams for system functionalities, and sequence diagrams for workflow steps, have been used to illustrate the design of the application.

CHAPTER 4 : IMPLEMENTATION

In this chapter, the system is implemented to meet the objectives and requirements of the supply chain management system that was proposed in the previous chapter. There are 3 types of users who will be using this system, which are the administrator of the system, the supplier and the public. The hierarchical diagram shown below shows the general modules that are being implemented in this system.

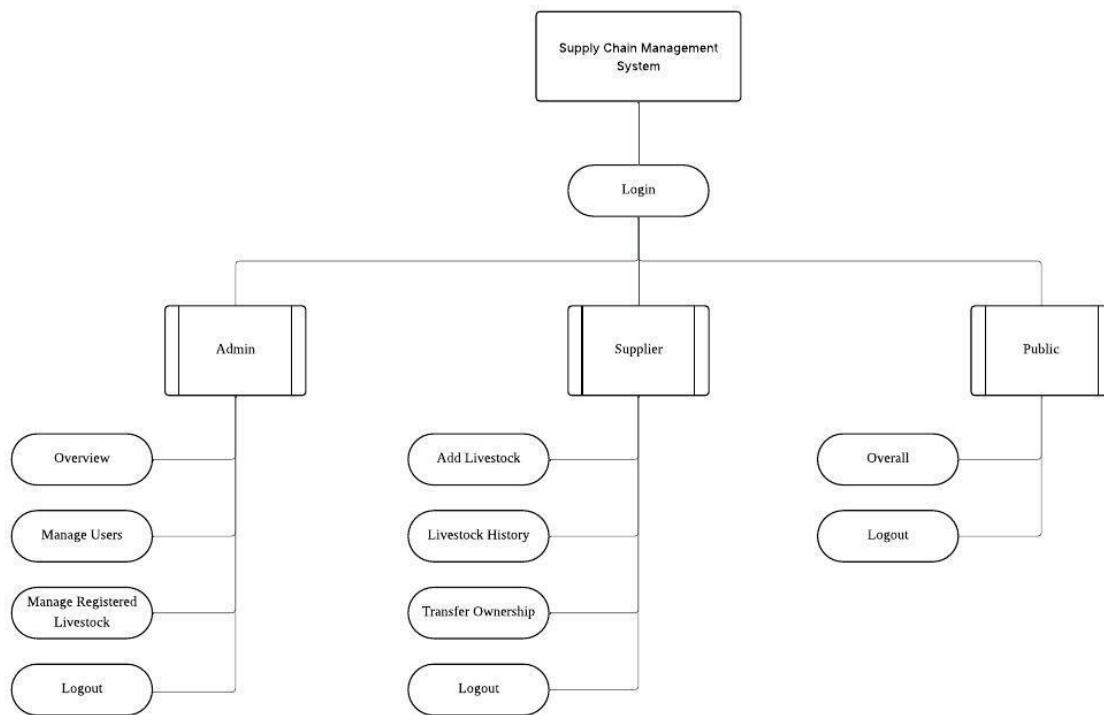


Figure 23: The hierarchical diagram

4.1 Login Page

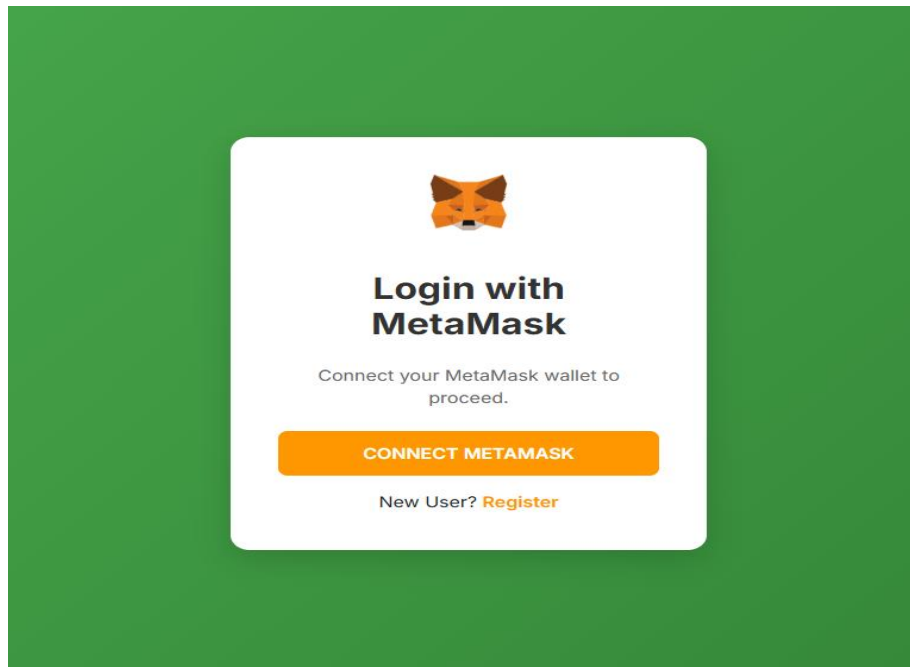
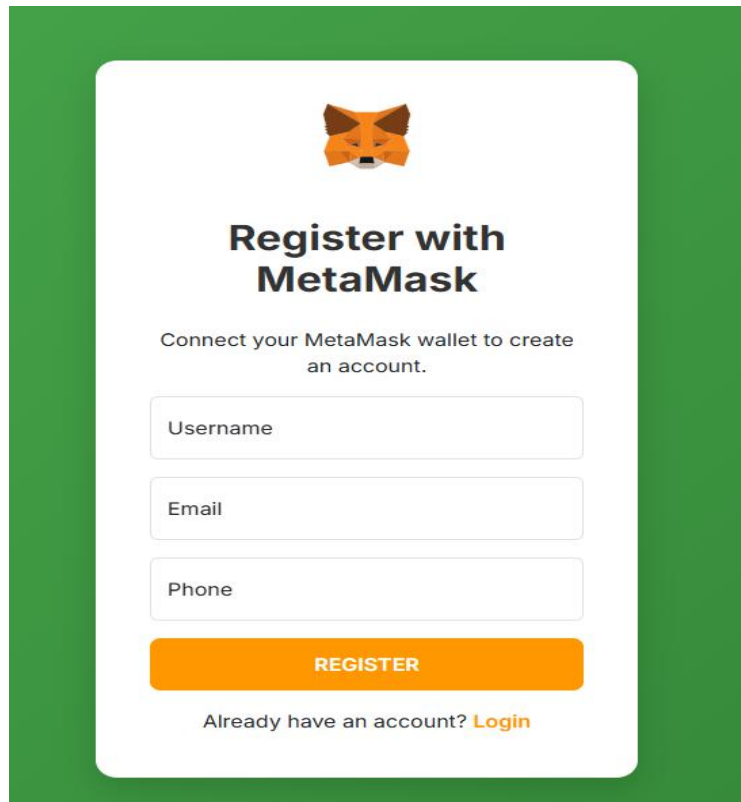


Figure 24: Login page

Figure 25 above shows the login page of the system, which authenticates users through MetaMask wallet integration. This login page tracks the user's Ethereum address to determine their role which are admin, supplier (user), and public buyer. The system will redirect them to the appropriate dashboard. When a user connects their wallet, the system checks their address against the smart contract data. If the address matches the contract owner's address, the user is identified as an admin and redirected to the admin dashboard for system management. Registered suppliers are verified for approval status; approved suppliers gain access to the supplier dashboard, while pending approvals are directed to the buyer dashboard. Public users, who are neither admins nor registered suppliers, are automatically redirected to the buyer dashboard.

4.2 Registration Page



The image shows a registration page with a green background. At the top center is the MetaMask fox logo. Below it, the text "Register with MetaMask" is displayed in bold. Underneath, a smaller text says "Connect your MetaMask wallet to create an account." There are three input fields: "Username", "Email", and "Phone". Below these fields is an orange "REGISTER" button. At the bottom, there is a link that says "Already have an account? Login".

Figure 25: Registration page

Figure 26 above shows the registration page of the system, which allows users to create accounts through MetaMask wallet authentication. This registration page collects essential user information including username, email, and phone number, while automatically capturing the user's Ethereum address through wallet connection. When a user submits the registration form, the system first verifies if the connected wallet is already registered by checking against the smart contract. Unregistered users are then added to the system as pending suppliers through a blockchain transaction. Successful registration triggers an automatic redirection to the login page, where users can authenticate while waiting for admin approval. The page includes validation for all

required fields and provides real-time feedback on registration status, ensuring a smooth user experience while maintaining security through blockchain verification. Conversely, the public buyers who don't need supplier privileges can simply use the system through wallet authentication without registration.

4.3 Admin Dashboard

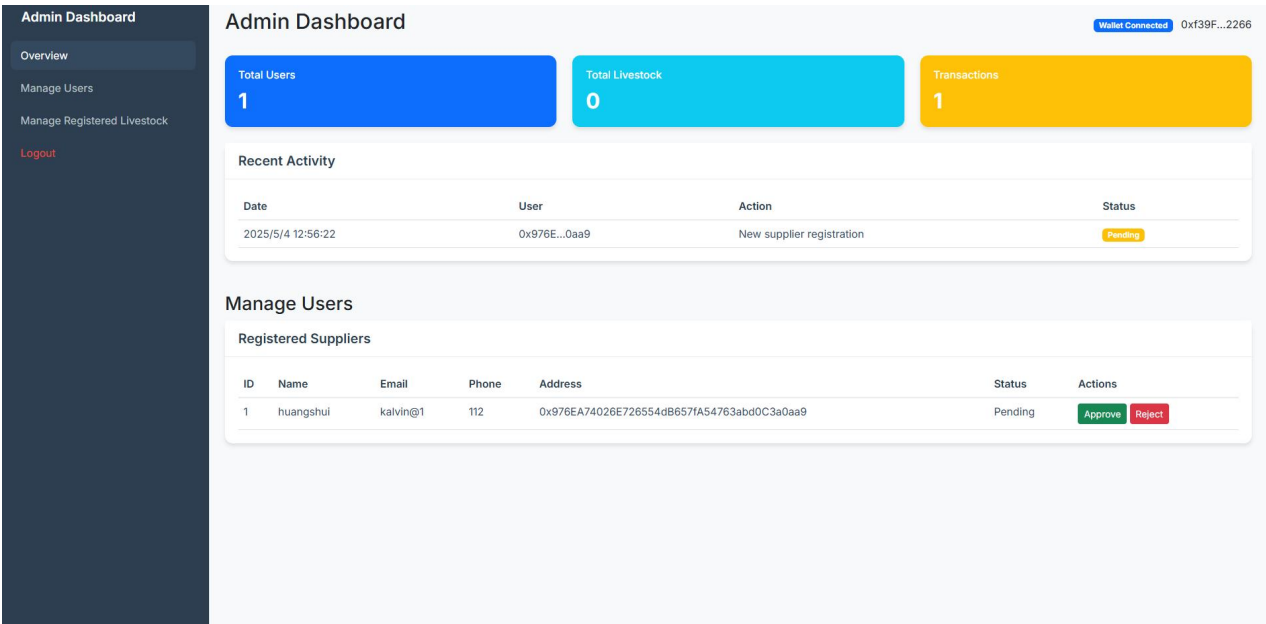


Figure 26:Admin dashboard

Figure 27 above shows the admin dashboard of the supply chain management system. The dashboard features a sidebar navigation with access to key sections including overview, user management, and livestock management. Upon login, the system verifies the administrator's credentials by checking their Ethereum address against the contract owner's address stored on the blockchain. The main dashboard displays real-time statistics including total users, registered livestock, and system transactions, all pulled directly from

the smart contract. A dedicated user management section allows administrators to review, approve, or reject supplier registrations through blockchain transactions. The interface also includes an activity log that tracks all system events chronologically, providing transparency for administrative actions. Security is maintained through continuous wallet connection monitoring, with automatic logout triggered by account changes.

4.3.1 Admin dashboard - Manage User

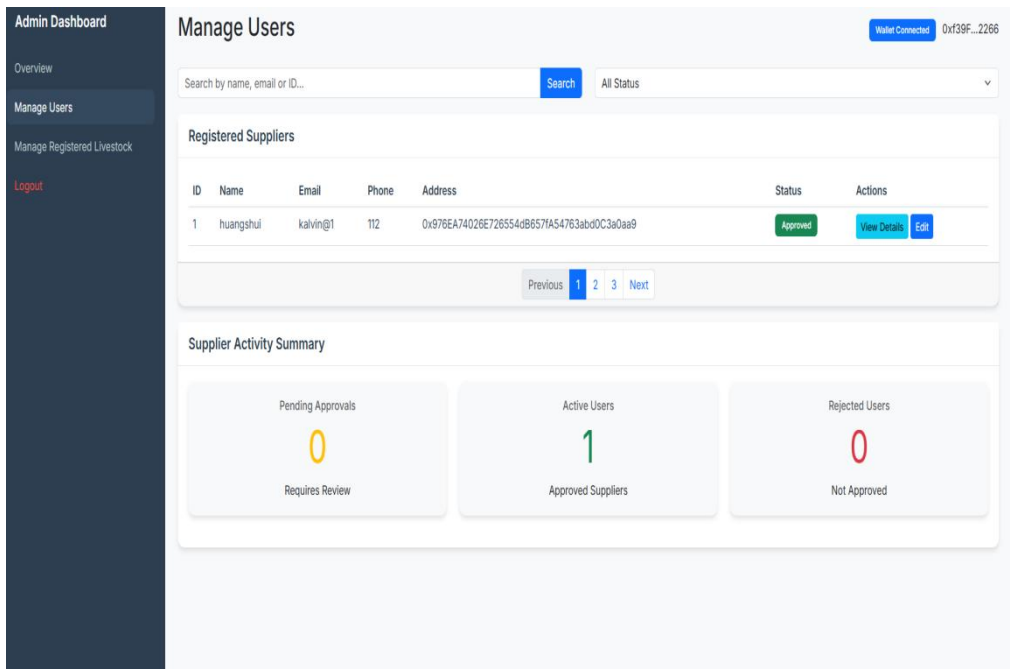


Figure 27:Manage user page

Figure 28 above shows the user management interface of the supply chain system. This page provides comprehensive tools for managing supplier accounts through a filterable and searchable table that displays all registered users. The system automatically verifies admin privileges by comparing the connected wallet address with the contract owner's address stored on the blockchain. Administrators can review pending applications, approve

or reject suppliers, and update user details through blockchain transactions that modify the smart contract state. The interface includes real-time counters for pending, approved, and rejected suppliers, along with detailed modals for viewing and editing user information. All changes trigger immediate updates to both the blockchain and the user interface, ensuring data consistency. The page maintains security through continuous wallet connection monitoring and implements responsive design for efficient user management across devices.

4.3.1 Admin dashboard - Manage Livestock

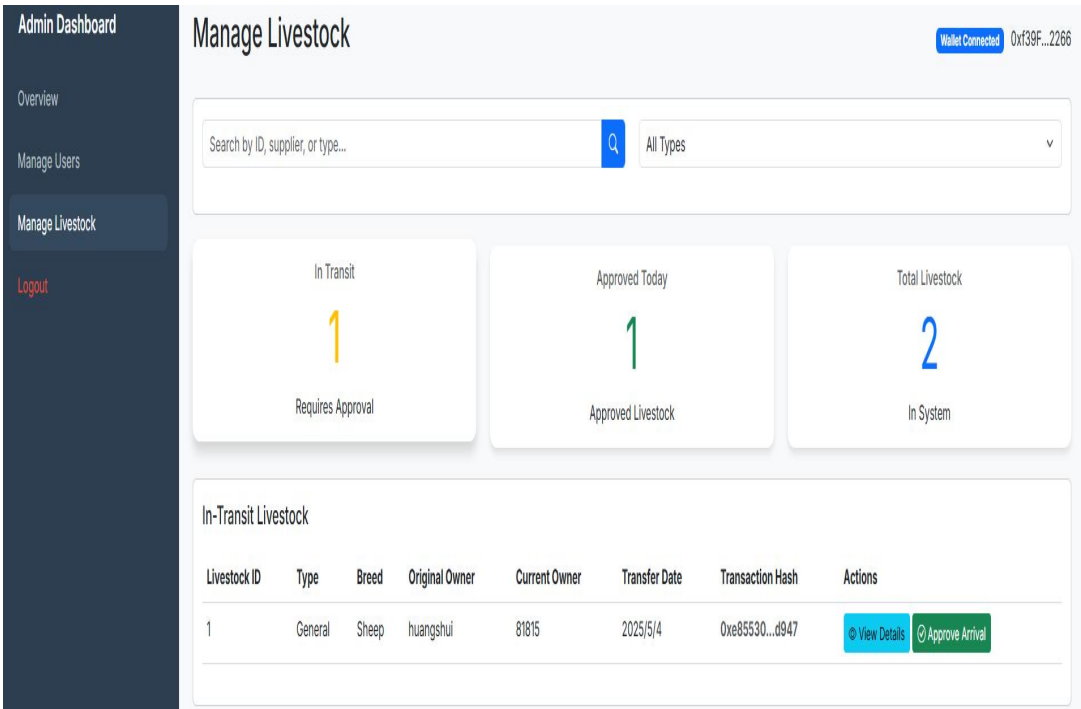


Figure 28: Manage livestock page

Figure 29 above shows the livestock management interface of the supply chain system, designed for administrator oversight of animal transfers. This page features a comprehensive dashboard that tracks livestock in transit, approved animals, and total

system inventory through real-time blockchain data. The interface automatically verifies admin privileges by comparing the connected wallet address with the smart contract owner address. Administrators can filter livestock by type (cattle, goat, sheep) and search by ID, or breed. Each entry displays complete transfer details including origin, destination, and blockchain transaction hashes for full transparency. The system provides detailed modals for viewing animal specifics and tools for approving arrivals, with all changes recorded as immutable transactions on the blockchain. Security is maintained through continuous wallet monitoring, ensuring only authorized administrators can validate livestock movements.

4.4 Supplier Dashboard

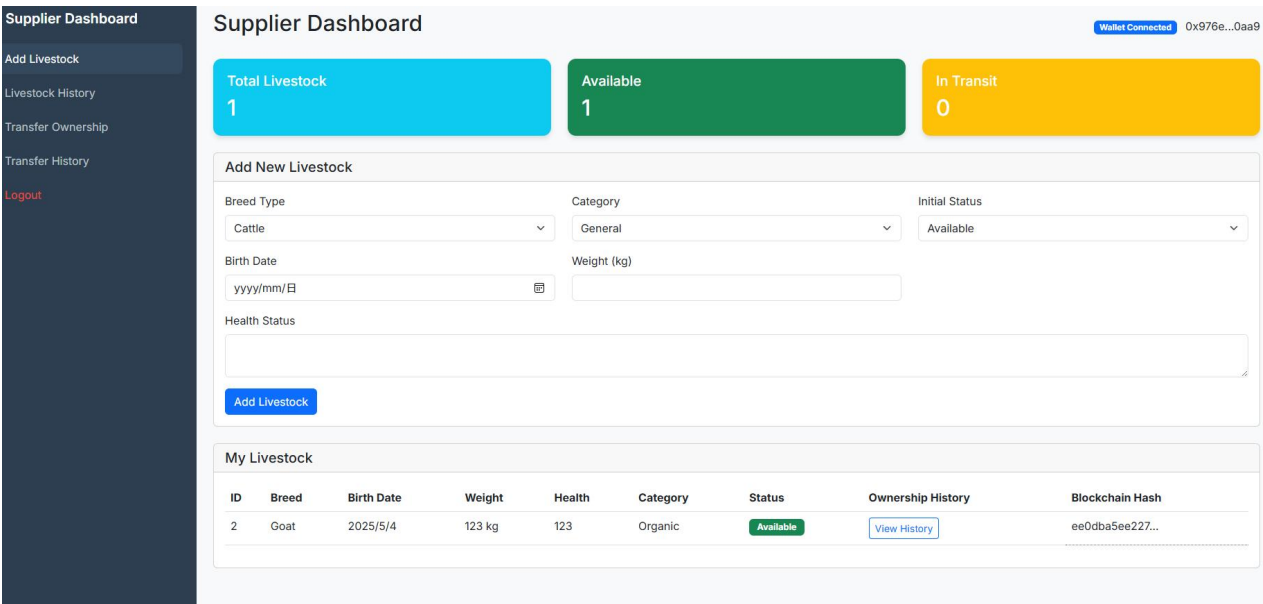


Figure 29:Supplier dashboard

Figure 30 above shows the supplier dashboard interface of the livestock supply chain system, designed for authorized suppliers to manage their livestock inventory. The dashboard features a comprehensive overview with statistics tracking total livestock, available animals, and those in transit. Suppliers can register new livestock through a secure form that records breed type, health status, and other vital information directly on the blockchain. Each animal entry displays complete details including ownership history and unique blockchain transaction hashes for verification. The system automatically verifies supplier authorization by checking the connected wallet address against approved suppliers in the smart contract. Real-time updates ensure inventory statistics remain accurate, while cryptographic hash generation creates tamper-proof records for each animal. The interface includes tools for viewing complete ownership transfer history, providing full transparency throughout the supply chain. This dashboard serves as the primary hub for suppliers to manage their livestock.

4.4.1 Supplier dashboard - Livestock History

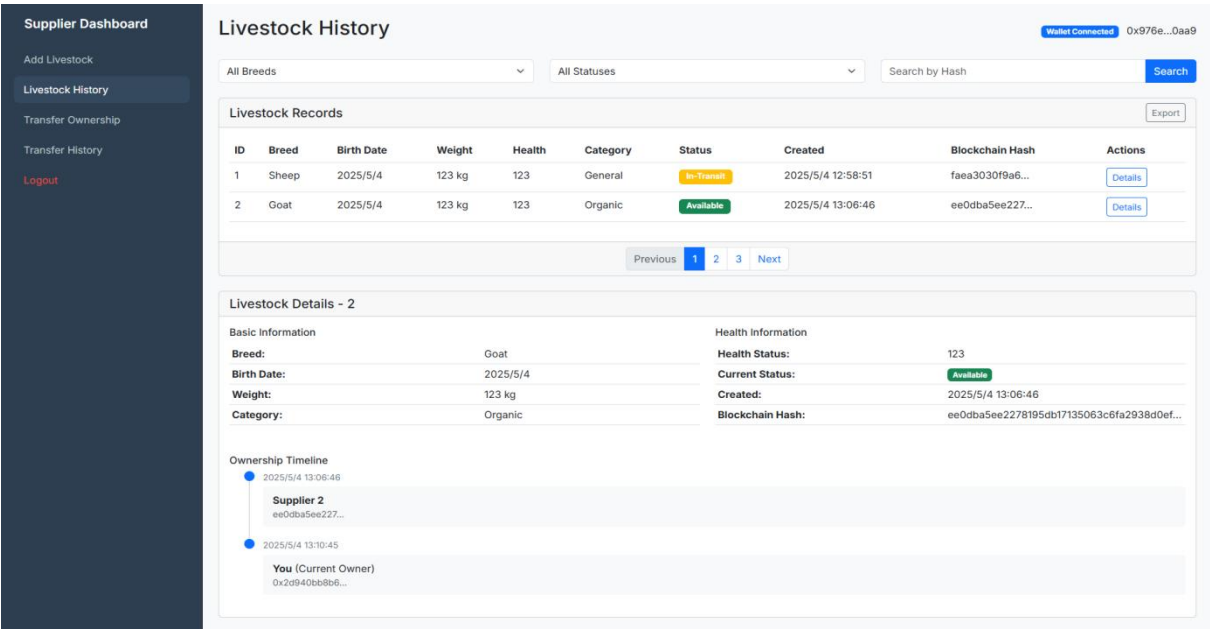


Figure 30:Livestock history page

Figure 31 above shows the Livestock History interface of the supply chain system, designed to provide suppliers with comprehensive tracking of their animals. This page features an advanced filtering system that allows suppliers to sort livestock by breed type, current status, or blockchain transaction hashes. The interface displays detailed records including health information, weight tracking, and complete ownership timelines for each animal. All data is pulled directly from the blockchain, with each entry showing its unique transaction hash for verification. The system includes export functionality, enabling suppliers to download their livestock records for offline analysis. A visual timeline displays the complete custody chain for each animal, showing when ownership changed hands. The page maintains security by verifying the supplier's authorization status through their connected wallet address before displaying any information.

4.4.2 Supplier dashboard - Transfer Ownership

Supplier Dashboard

- Add Livestock
- Livestock History
- Transfer Ownership**
- Transfer History
- Logout

Transfer Ownership

Wallet Connected 0x976e...0aa9

1 Select Livestock 2 Choose Recipient 3 Confirm Transfer

Select Livestock to Transfer

Search by ID, breed or health...

All Breeds Available Only

ID	Breed	Birth Date	Weight	Health	Status
☐ 2	Goat	2025/5/4	123 kg	123	Available

Figure 31: Transfer ownership page (step 1)

Supplier Dashboard

- Add Livestock
- Livestock History
- Transfer Ownership**
- Transfer History
- Logout

Transfer Ownership

Wallet Connected 0x976e...0aa9

1 Select Livestock 2 Choose Recipient 3 Confirm Transfer

Choose Recipient Supplier

Selected Livestock

Goat (ID: 2)
Birth: 2025/5/4
Health: 123

Weight: 123 kg

Search Registered Suppliers

81815 (ID: 2)

81815
ID: 2 • xixing2000@gmail.com

Transfer Notes

asdsd

Figure 32: Transfer ownership page (step 2)

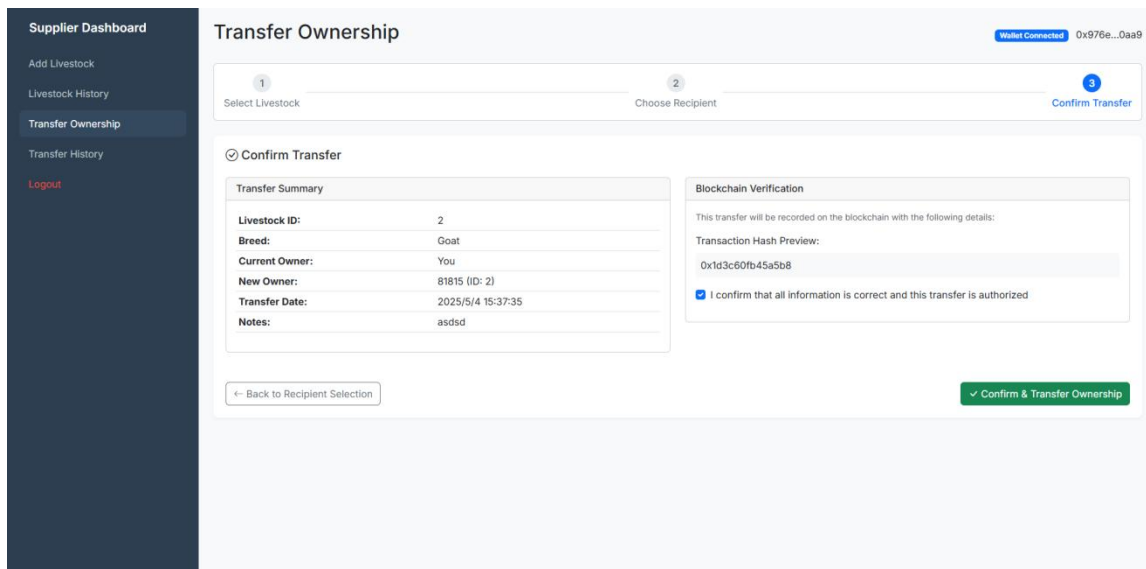


Figure 33: Transfer ownership page (step 3)

Figure 32,33,34 above shows the Ownership Transfer interface of the system, featuring a three-step workflow for secure livestock transfers between suppliers. The process begins with livestock selection from the supplier's available inventory, verified against blockchain records to ensure only valid animals can be transferred. The second step provides a comprehensive recipient selection tool with real-time supplier search functionality, displaying only approved suppliers from the smart contract. The final confirmation screen generates a preview of the blockchain transaction and requires explicit authorization before executing the transfer. Each step includes verification checks against the connected wallet address to maintain security. The interface will automatically updates livestock status to "In Transit" during transfers and provides visual feedback through a progress indicator. Upon completion, the system generates an immutable record on the blockchain, including the transaction hash, transfer timestamp, and optional notes.

4.4.3 Supplier dashboard - Transfer History

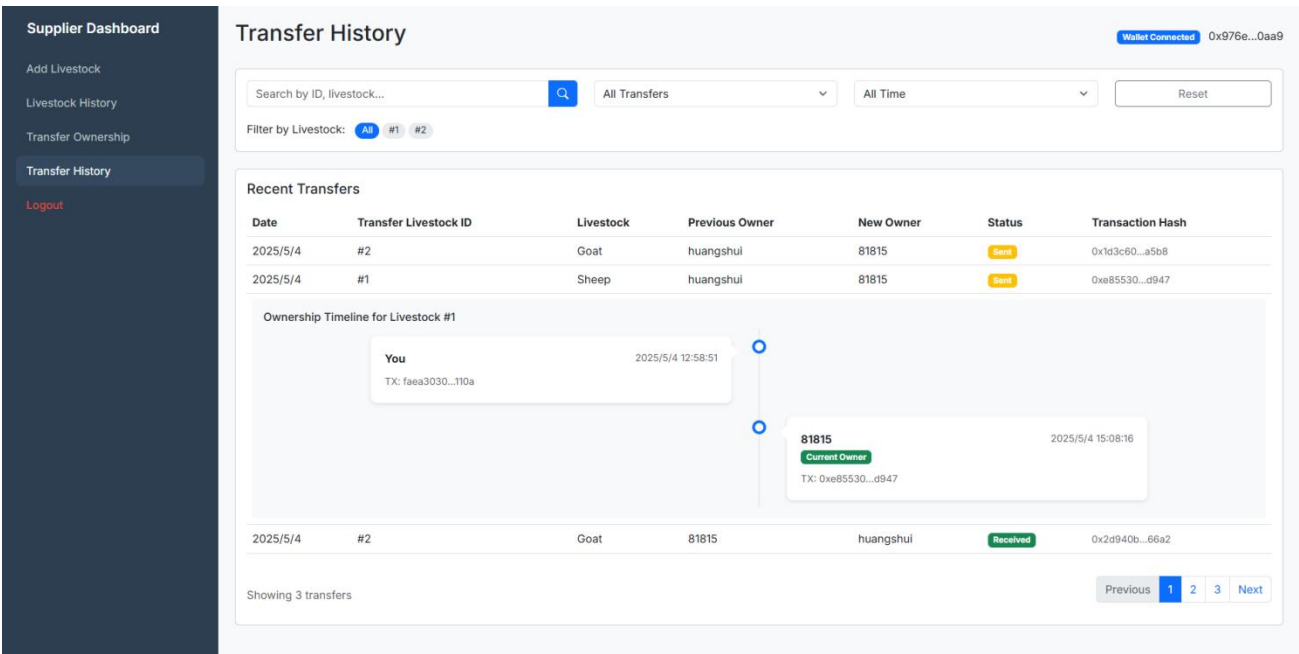


Figure 34: Transfer history page

Figure 35 above shows the Transfer History interface of the livestock supply chain system, providing suppliers with a comprehensive audit trail of all ownership transfers. This page displays a chronological record of livestock movements, including detailed information about senders, recipients, and blockchain transaction hashes for each transfer. The interface features advanced filtering capabilities, allowing users to sort transfers by direction (sent/received), time period, and specific livestock IDs. Each entry in the transfer history includes interactive elements that reveal additional details when clicked, such as complete ownership timelines and health information. The system generates visual indicators to distinguish between incoming and outgoing transfers, with color-coded badges for quick identification. All data is pulled directly from blockchain records, ensuring tamper-proof verification of each transaction. The page maintains security by verifying the

supplier's authorization status through their connected wallet address before displaying any information.

4.5 Public Dashboard

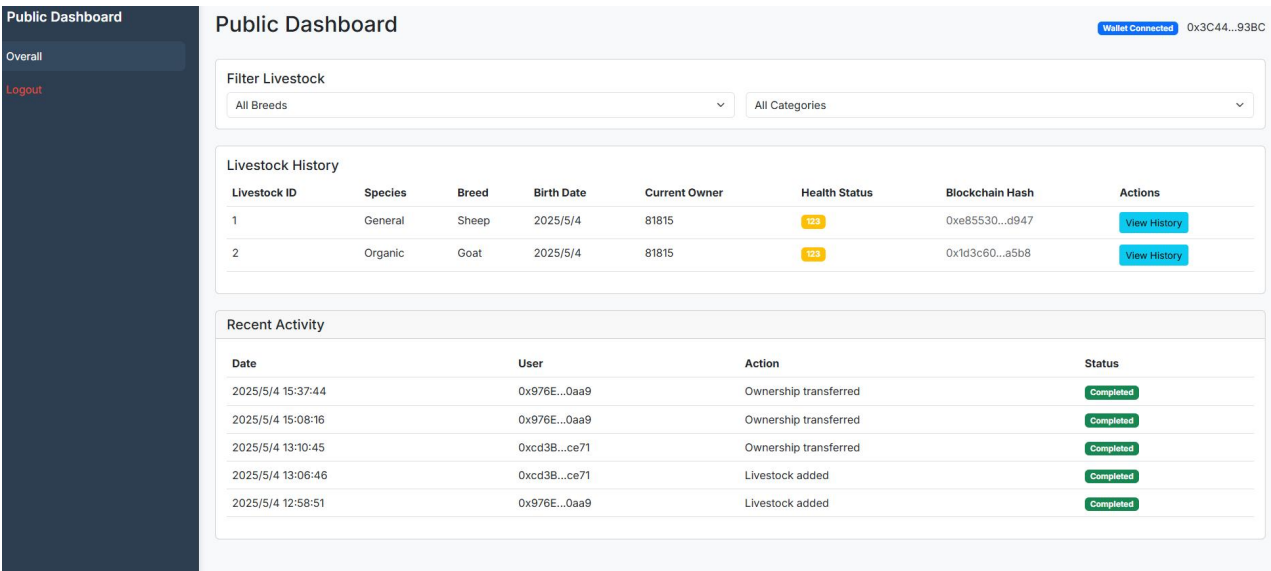


Figure 35: Public dashboard

Figure 13 above shows the Public Dashboard interface of the livestock supply chain system, designed to provide transparent access to livestock information for all users. This page features a comprehensive view of registered livestock, including breed details, ownership history, and health status - all verified through blockchain records. The interface includes advanced filtering capabilities allowing users to search by breed type and livestock category. Each livestock entry displays its complete custody chain with timestamps and blockchain transaction hashes for verification. The dashboard also shows recent system activity, creating full transparency for all supply chain operations. Security is maintained through wallet authentication, though with limited permissions compared to supplier/admin accounts.

CHAPTER 5 : TESTING

5.1 Introduction

This chapter emphasizes the testing phase of the proposed system. Testing the project is crucial to make sure all the functionalities are workable in order to avoid the sudden change or modification at the end of the development. This phase also can ensure the users' satisfaction in which the features are function well and the objectives will be met. Hence, this testing phase will be divided into two parts which are the functional testing and nonfunctional testing. It will be carried out from time to time to make sure all the functionalities are working well. Two parts of testing phase which are functional testing and non-functional testing will be carried out in this session.

5.2 Functionality Testing

The functionality testing is done by testing each of the modules by trying to input the invalid and valid information. For each module, there are a few test case with its test scenario. The results from the functional testing are tabulated in Table 5.2.1 to Table 5.2.5. The component for the testing table are the test description, test steps, test input, expected result, and test status.

5.2.1 Register

Module Name: Registration – MetaMask Integration

Test Case Description: To verify the registration functionality of the platform using MetaMask wallet.

Test Case ID	Description	Test Steps	Test Input	Expected Results	Status
1	MetaMask not installed.	1. Open register page. 2. MetaMask not installed.	-	Button is disabled. Message: "MetaMask is not installed. Please install it..."	Pass
2	MetaMask not connected.	1. Fill in all fields. 2. Click "Register". 3. MetaMask not connected.	Valid user input	MetaMask prompts for account connection.	Pass
3	Already Registered Wallet	1. Fill all fields. 2. User already registered wallet.	Registered wallet	Message: "This wallet is already registered."	Pass
4	Successful Registration	1. Fill all fields. 2. Use new wallet. 3. Complete registration via MetaMask.	Valid input + new address	Message: "Registration successful! Redirecting to login page..." Redirect.	Pass

Table 11: Test case for registration module

5.2.2 Login

Module Name: Login – MetaMask Integration

Test Case Description: To verify the login functionality of the platform using MetaMask wallet.

Test Case ID	Description	Test Steps	Test Input	Expected Results	Status
1	MetaMask Connection Prompt	1. Open login page. 2. Click on "Connect MetaMask".		MetaMask connection prompt appears.	Pass
2	Admin Address Login	1. Connect using admin address.	Admin Address	Redirect to admin-dashboard.html	Pass
3	Unregistered Address Login	1. Connect using unregistered address.	Public Address	Redirect to buyer-dashboard.html	Pass
4	Approved Supplier Login	1. Connect using supplier address (Approved).	Supplier Address	Redirect to supplier-dashboard.html	Pass
5	Pending Supplier Login	1. Connect using supplier address (Pending).	Supplier Address	Redirect to buyer-dashboard.html with pending message.	Pass

6	MetaMask Not Installed	1. Open page without MetaMask installed.	-	Message: "MetaMask is not installed." Button is disabled.	Pass
---	------------------------	--	---	---	------

Table 12: Test case for login module

5.2.3 Admin Module

Module Name: Admin dashboard (Overall)

Test Case Description: To verify the functionality of the admin dashboard.

Test Case ID	Description	Test Steps	Test Input	Expected Results	Status
1	Non-admin wallet access	1. Connect non-admin wallet. 2. Navigate to admin dashboard.	-	Redirect to login page.	Pass
2	Admin wallet connection	1. Connect admin wallet. 2. Load all the data to dashboard	Admin Address	Dashboard loads all the data from Ethereum and displays it.	Pass
3	Search by name	1. Type "John" in search 2. Press Enter	Search term: "John"	Filters suppliers where name contains "John" (case-insensitive) and display it.	Pass

4	Filter by status	1. Select "Approved" from dropdown	Filter: Approved	Only shows approved suppliers	Pass
5	Empty search reset	1. Clear search field 2. Press Enter	Search term: ""	Resets to full supplier list	Pass

Table 13 : Test case for admin module overall

Module Name: Admin dashboard (User Management)

Test Case Description: To verify the functionality of the admin dashboard.

Test Case ID	Description	Test Steps	Test Input	Expected Results	Status
1	Approve supplier	1. Click Approve 2. Sign MetaMask tx	Supplier ID: 3 (Pending)	Supplier status changed from pending to approved.	Pass
2	Reject supplier	1. Click Reject 2. Sign MetaMask tx	Supplier ID: 4 (Pending)	Supplier status changed from pending to rejected.	Pass
3	View user details modal	1. Click "View Details"	Supplier ID: 2	Modal shows all supplier details	Pass

4	Edit supplier	1. Click Edit 2. Change user detail 3. Save	New name: "John Smith"	Details update after MetaMask confirmation	Pass
5	Supplier activity summary	-	-	Pending/Approved/Rejected counts update automatically and correctly	Pass

Table 14 : Test case for registration module(User Management)

Module Name: Admin dashboard (Livestock Management)

Test Case Description: To verify the functionality of the admin dashboard.

Test Case ID	Description	Test Steps	Test Input	Expected Results	Status
1	Type filtering	1. Select "Cattle" from dropdown	Filter: Cattle	Only livestock with the type of cattle are shown	Pass
2	Approve arrival (table)	1. Click "Approve Arrival" 2. Confirm 3. Sign MetaMask tx	Livestock ID: 3	Livestock status changes to "Approved", removed from in-transit list	Pass
3	Status counters update	1. Approve arrival 2. Wait 5 second	-	In-transit count decreases, approved-today increases	Pass

Table 15 : Test case for registration module(Livestock Management)

5.2.4 Supplier Module

Module Name: Supplier dashboard (Add Livestock)

Test Case Description: To verify the functionality of the supplier dashboard.

Test Case ID	Description	Test Steps	Test Input	Expected Results	Status
1	Non-approved supplier access	1. Connect non-approved wallet 2. Login to the page.	Non-approved wallet	Redirect to login page with alert	Pass
2	Missing field when adding the new livestock	1. Leave required fields empty 2. Submit form	Incomplete form	The form shows validation errors, no transaction occurs.	Pass
3	Livestock table and statistic display	1.Login as approved supplier 2. The page loaded with correct data	-	Livestock table and statistic data display correctly	Pass
4	View ownership history	1. Click "View History" on any livestock	Livestock ID: 5	Modal shows complete ownership history	Pass

Table 16 : Test case for supplier dashboard module(Add Livestock)

Module Name: Supplier dashboard (Livestock History)

Test Case Description: To verify the functionality of the supplier dashboard.

Test Case ID	Description	Test Steps	Test Input	Expected Results	Status
1	Ownership timeline	1. View details 2. Check timeline	-	Shows complete ownership history in chronological order	Pass
2	Export functionality	1. Click "Export" button	-	Downloads CSV with current filtered data	Pass
3	Hash search	1. Enter partial hash in search 2. Click Search	Search: "a1b2"	Shows records with matching hash	Pass

Table 17 : Test case for supplier dashboard module(Livestock History)

Module Name: Supplier dashboard (Transfer Ownership)

Test Case Description: To verify the functionality of the supplier dashboard.

Test Case ID	Description	Test Steps	Test Input	Expected Results	Status
1	Livestock selection	1. Select livestock 2. Click "Proceed"	Livestock ID: 5	Moves to recipient selection step	Pass
2	Recipient search	1. Type "Farm" in search	Search: "Farm"	Shows matching suppliers with the name of Farm.	Pass
3	Recipient selection	1. Click on supplier from the list to select the supplier.	Supplier ID: 8	Shows selected supplier card	Pass
4	Transfer confirmation	1. Add notes 2. Click "Proceed"	Notes: "Urgent transfer"	Shows confirmation page with all details	Pass
5	Mandatory confirmation	1. Skip checkbox 2. Try to transfer	Unchecked box	Transfer button disabled	Pass

Table 18: Test case for supplier dashboard module(Transfer Ownership)

5.2.5 Public Module

Module Name: Public dashboard (Overall)

Test Case Description: To verify the functionality of the public dashboard.

Test Case ID	Description	Test Steps	Test Input	Expected Results	Status
1	Only unregistered suppliers and no admin user will be logged in.	1. Login using the meta mask	Unregistered address	Only Unregistered address will access this page	Pass
2	Breed filter	1. Select "Cattle" from dropdown	Breed: Cattle	Only livestock with the type of cattle are shown	Pass
3	View history modal	1. Click "View History" on any row	Livestock ID: 5	Modal shows correct livestock details	Pass
4	Livestock history table and recent activity display	1. Login as unregistered user and no admin user		The livestock history table and recent activity display correct data	Pass

Table 19 : Test case for supplier dashboard module public dashboard

5.3 Non-Funtionality Testing

Non-functional testing is the testing of the proposed system whether it provides the user's a good experience and whether it met the user's expectations. In this proposed system the usability testing, compatibility testing, and reliability testing are implemented.

5.3.1 Usability Testing

Test Case ID	Description	Test Steps	Test Input	Expected Results	Status
1	Registration usability	Observe 5 new users registering and navigating supplier dashboard.	-	Users complete registration and find dashboard features easily.	Pass
2	Supplier livestock management UX	Ask 5 suppliers to add, view, and transfer livestock; check if process is intuitive.	-	Users can complete actions without confusion.	Pass
3	Admin approval UX	Observe admin approving/rejecting suppliers and livestock arrivals.	-	Admin can manage data without extra explanation.	Pass

4	Consistency of UI labels	Review dashboard UI labels and button names across all pages.	UI elements	All labels are consistent in style and terminology.	Pass
---	--------------------------	---	-------------	---	------

Table 20 : Usability testing table

5.3.2 Compatibility Testing

Test Case ID	Description	Test Steps	Test Input	Expected Results	Status
1	Cross-browser compatibility	Test in Chrome, Firefox, Edge, and Safari by logging in and viewing livestock history.	Browsers: Chrome, brave, Edge	All features work and display correctly.	Pass
2	MetaMask version compatibility	Test login and registration using different MetaMask extension versions (old and latest).	Different MetaMask versions	All versions of MetaMask integrate without error.	Pass

Table 21 : Compatibility testing table

5.3.3 Reliability Testing

Test Case ID	Description	Test Steps	Test Input	Expected Results	Status
1	MetaMask unavailability	Disable MetaMask extension, attempt to log in.	MetaMask disabled	Error message: MetaMask is not installed.	Pass
2	Blockchain network interruption	Simulate Ethereum network downtime during livestock approval.	Disconnect blockchain connection	System handles error gracefully, showing retry or error.	Pass
3	Admin approval UX	Observe admin approving/rejecting suppliers and livestock arrivals.	-	Admin can manage data without extra explanation.	Pass
4	Session timeout recovery	Leave supplier dashboard idle for 15 minutes, then try to submit a livestock transaction.	Session inactivity	System prompts to re-login or refresh MetaMask session.	Pass

5	Browser crash recovery	Close browser unexpectedly while adding livestock, re-login and check if data is saved properly.	Browser crash	No partial or corrupt data; system requires full re-entry.	Pass
---	------------------------	--	---------------	--	------

Table 22 : Reliability testing table

5.4 Summary

In this chapter, the testing phase has been thoroughly completed, with results confirming that all the objectives of the project have been achieved. System testing focused on multiple key aspects, including functionality, usability, reliability, and compatibility. The results demonstrate that the blockchain based supply chain product management system meets the functional requirements and provides a secure, user-friendly, and reliable experience across different platforms. This comprehensive testing phase ensures that the system is ready for deployment and aligns with the intended goals of enhancing traceability and transparency in the supply chain.

CHAPTER 6 : CONCLUSION AND FUTURE WORK

6.1 Introduction

This chapter describes the conclusion of the proposed system and the future work that can be implemented in this project. The achievements and limitations of this proposed system will be briefly discussed in this section. Lastly, the possible enhancements for future work will be suggested based on the test results.

6.2 Achivement

The blockchain-based supply chain management system has successfully achieved its primary goal of enhancing transparency and trust within livestock tracking. By leveraging blockchain technology and MetaMask integration, the system ensures secure and tamper-proof recording of supplier and livestock information. Comprehensive testing across functional and non-functional aspects has confirmed that the system is reliable, user-friendly, and compatible across various devices and platforms.

6.3 Limitations

Despite its strengths, the system has some limitations that need to be addressed. Its reliance on the MetaMask extension can be a barrier for less technically inclined users, potentially hindering adoption. Additionally, blockchain transaction times can fluctuate based on network conditions, occasionally leading to delays in updating livestock records. Another important limitation is the potential high gas fees associated with transactions on

the Ethereum network, which could make certain operations costly and discourage frequent use. The system's scalability is also a concern if it is expected to handle significantly larger datasets and more complex interactions in the future.

6.4 Future Work

To overcome these limitations and expand the system's impact, several areas for future work have been identified. Integrating support for alternative blockchain networks, such as layer 2 solutions or less congested blockchains, can help reduce transaction costs and gas fees while improving scalability. Additionally, exploring the development of a dedicated blockchain network specifically for the supply chain system could provide greater control, lower fees, and tailored features to better meet the industry's needs. Developing a dedicated mobile application with offline capabilities would also enhance accessibility for users in areas with poor internet connectivity. Streamlining the onboarding process will be important for easing adoption among new users and reducing reliance on MetaMask for certain interactions.

REFERENCE

- Androulaki, E., Barger, A., Bortnikov, V., et al. (2018) Hyperledger Fabric: A Distributed Operating System For Permissioned Blockchains. Proceedings of the Thirteenth EuroSys Conference, Porto, April 2018, 1-15.
<https://doi.org/10.1145/3190508.3190538>
- Buterin, V. (2017). A Next-Generation Smart Contract and Decentralized Application Platform. Ethereum.
- Chandran, R., Shukla, A., & Sharma, S. (2014). Challenges and Opportunities in Traditional Supply Chain Systems. International Journal of Modern Computer Science, 2(1), 15-20. Retrieved from
<https://airccse.org/journal/mvsc/papers/5114ijmvsc03.pdf>
- Ge, Z., Loghin, D., Ooi, B. C., Ruan, P., & Wang, T. (2022). Hybrid blockchain database systems: Design and performance. *Proceedings of the VLDB Endowment*, 15(5), 1092–1104.
- He, J., & Xu, J. (2019). "Blockchain-based supply chain management: A survey." *Computers, Materials & Continua*, 59(3), 847-864.
- Kshetri, N. (2018). 1 Blockchain's roles in meeting key supply chain management objectives. *International Journal of Information Management*, 39, 80-89.

<https://doi.org/10.1016/j.ijinfomgt.2017.12.005>

Kumar, S., & Gupta, A. (2017). Inefficiencies in Traditional Supply Chain Management: A Case Study. *International Journal of Advanced Research in Science, Engineering, and Technology*, 4(3), 55-60. Retrieved from <https://iarjset.com/upload/2017/si/NCDMETE-2017/IARJSET-NCDMETE%2011.pdf>

Liu, J., Yan, L., & Wang, D. (2022). A hybrid blockchain model for trusted data of supply chain finance. *Wireless Personal Communications*, 127(2), 919–943.

Luu, L., et al. (2019). "Blockchain-based Supply Chain Management with Hyperledger Fabric." *International Journal of Computer Applications*, 175(10), 21-28.

Mougayar, W. (2016). *The Business Blockchain: Promise, Practice, and the Next Big Thing*.

Narayanan, A., Bonneau, J., Felten, E., Miller, A., & Shacham, H. (2016). *Bitcoin and Cryptocurrency Technologies*. Princeton University Press.

Ranjan, J., & Thomas, P. (2019). "Hyperledger Fabric in Supply Chain Management: A Survey of Applications and Challenges." *Proceedings of the 3rd International Conference on Blockchain Technology*.

Rodeck, D., & Curry, B. (2022). *What is blockchain? Forbes Advisor*.
<https://communications.pasenategop.com/wp-content/uploads/sites/15/2022/06/What->

Supply chain transparency: The key to food safety excellence. Blockchain for Food Safety, Traceability and Supplychain Transparency. (2023).
<https://tracextech.com/supply-chain-transparency-food-safety/>

Tapscott, D., & Tapscott, A. (2016). *Blockchain Revolution: How the Technology Behind Bitcoin and Other Cryptocurrencies Is Changing the World.*

Vukolic, M. (2016). "The Interoperability of Blockchains." Springer.

Vyas, N., Beije, A., & Krishnamachari, B. (2019). *Blockchain and the supply chain: concepts, strategies and practical applications.* Kogan Page Publishers, New York.

Wan, C. (2019). *Cordas: Blockchain-inspired design.* Medium. <https://clemens-wan.medium.com/cordas-blockchain-inspired-design-104bfd5509d3>

APPENDIX

Table 23 : Project Schedule

Task	Expected Date
First meeting with FYP sv for FYP title	10/10/2024
Submit brief project description for reviewing	17/10/2024
Submit brief project description to the system	19/10/2024
Third meeting for further requirement gathering	22/10/2024
Submit full proposal to system	14/11/2024
Submission of chapter 1	21/11/2024
Submission of chapter 2	13/12/2024
Submission of chapter 3	5/1/2025
Submission of FYP 1 final report & paper for assessment	17/1/2025
Submission of revised structure of FYP report	1/4/2025
Submission of first draft of chapter 4	15/5/2025
Submission of First Draft for Chapter 5, 6 & Abstract for Paper	31/5/2025
Submission of First Draft for FYP 2 Full Report & Paper	10/6/2025
Submission of Final Report, source code, installation kits, user manual and Paper for Assessment	23/6/2025

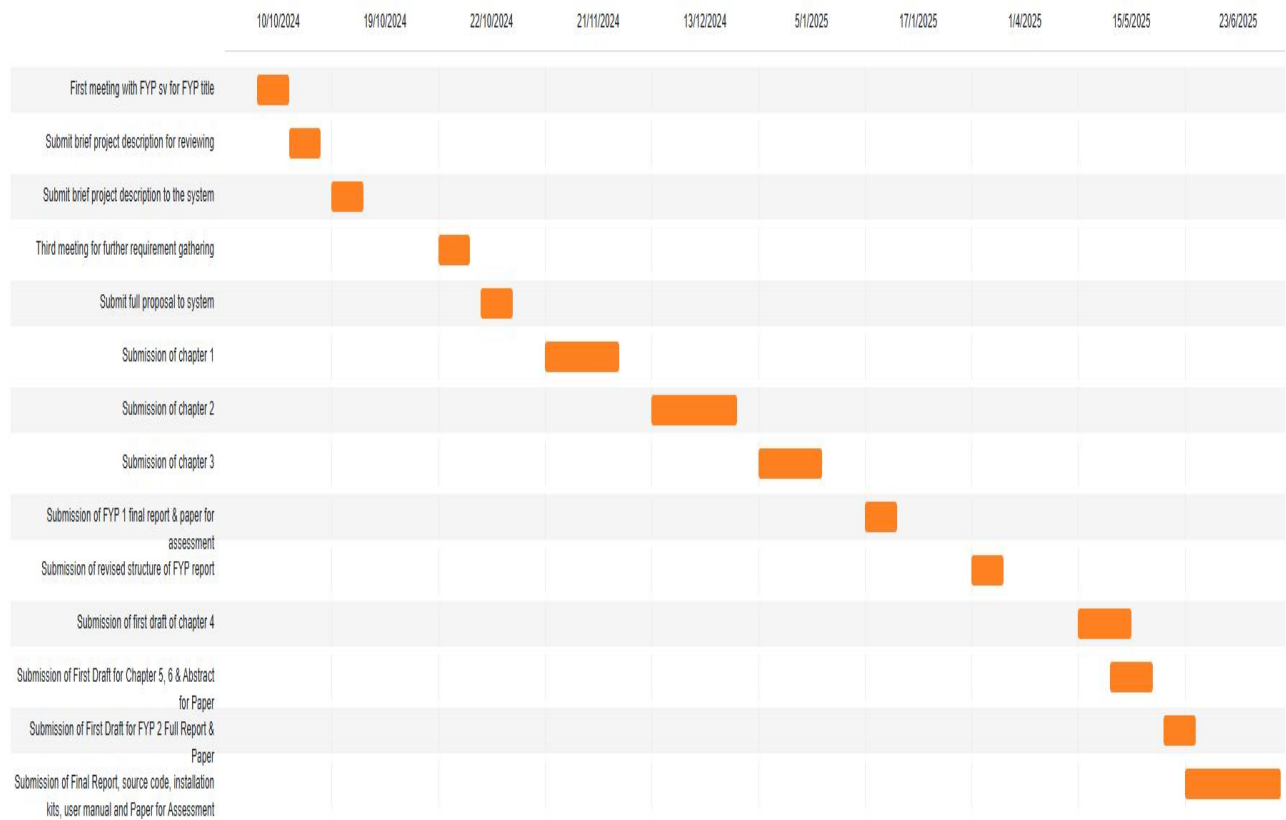


Figure 36 : Project schedule gantt chart