

INVENTORY SYSTEM FOR RETAIL USING BLOCKCHAIN TECHNOLOGY

BY

YONG WEI BANG

A REPORT

SUBMITTED TO

Universiti Tunku Abdul Rahman

in partial fulfillment of the requirements

for the degree of

BACHELOR OF INFORMATION TECHNOLOGY (HONOURS)

COMMUNICATIONS AND NETWORKING

Faculty of Information and Communication Technology

(Kampar Campus)

FEBRUARY 2025

COPYRIGHT STATEMENT

© 2025 Yong Wei Bang. All rights reserved.

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

ACKNOWLEDGEMENTS

I would like to express thanks and appreciation to my supervisor, Dr. Zurida Binti Ishak and my moderator, Dr. Adeb Ali Mohammed Ahmed Al-Samet who have given me a golden opportunity to involve in the Blockchain field study. Besides that, they have given me a lot of guidance in order to complete this project. When I was facing problems in this project, the advice from them always assists me in overcoming the problems. Again, a million thanks to my supervisor and moderator.

ABSTRACT

This project is to create an inventory system that would be reliable, secure and operate in a private blockchain setting. Not all conventional methods for inventory management face issues, but they nevertheless have shortcomings that include unclear data presentation, confusing users and inefficient processes, applicable to some methods for inventory control. This work presented blockchain solution as a technological implement to solve the challenges outlined. The project involves a web application development strategy. The system is developed with a mindset that embraces security, scalability, interoperability, performance and user friendliness. Key functionalities include Product Registration, Inventory Tracking and Data Visualization. The system interacts with the chosen blockchain network through smart contracts and stores the data on the blockchain. The project successfully showed the creation of a fully working blockchain-based inventory management system prototype. The system endorses improved security via unalterable data storage and safeguarded transactions on the blockchain. High transparency level makes it possible to monitor the entire supply chain in real time. Moreover, the system achieves efficiency by automating tasks and making inventory management processes easier. The research suggests that in future inventory management procedures, blockchain technology can be a key element of revolution. The developed platform supplies many advantages to businesses like higher trust, operational efficiency and low costs. Along with the investigation of the possibility of implementing advanced functionalities such as disruptive analyzes, decentralized marketplaces.

Area of Study (Minimum 1 and Maximum 2): **Blockchain**

Keywords (Minimum 5 and Maximum 10): **Blockchain Technology, Inventory Management System, Smart Contracts, Decentralized Application (DApp), Automated Reordering, Web3 Application**

TABLE OF CONTENTS

TITLE PAGE	i
COPYRIGHT STATEMENT	ii
ACKNOWLEDGEMENTS	iii
ABSTRACT	iv
TABLE OF CONTENTS	v
LIST OF FIGURES	viii
LIST OF TABLES	xi
CHAPTER 1 INTRODUCTION	1
1.1 Background of the Study	1
1.2 Problem Statement and Motivation	2
1.3 Project Objectives	4
1.3.1 Enhance Data Integrity	5
1.3.2 Improve Traceability	5
1.3.3 Automate Reorder	6
1.4 Project Scope	7
1.5 Contributions	8
1.6 Report Organization	9
CHAPTER 2 LITERATURE REVIEW	9
2.1 Zoho Inventory	9
2.1.1 Strengths and Weakness of Zoho Inventory System	12
2.2 Odoo Inventory System	13
2.2.1 Strengths and Weakness of Odoo Inventory System	17
2.3 Compare with Proposed Solution	19
CHAPTER 3 SYSTEM METHODOLOGY	21
3.1 System Design Diagram	22
3.1.1 System Architecture Diagram	22
3.1.2 Use Case Diagram and Description	23

3.1.3 Activity Diagram	26
CHAPTER 4 SYSTEM DESIGN	38
4.1 System Block Diagram	38
4.2 System Components Specifications	39
4.3 Components Design	39
4.4 System Components Interaction Operations	40
4.5 Flowchart	41
CHAPTER 5 SYSTEM IMPLEMENTATION (FOR DEVELOPMENT BASED PROJECT)	43
5.1 Hardware Setup	43
5.2 Software Setup	43
5.3 Setting and Configuration	44
5.4 System Operation	46
5.5 Implementation Issues and Challenges	59
5.6 Concluding Remark	60
CHAPTER 6 SYSTEM EVALUATION AND DISCUSSION	61
6.1 System Testing and Performance Metrics	61
6.2 Testing Setup and Result	62
6.3 System Usability Survey	63
6.4 Objectives Evaluation	71
6.4.1 Enhance Data Integrity	71
6.4.2 Improve Traceability	71
6.4.3 Automate Reorder	71
6.5 Concluding Remark	72

CHAPTER 7 CONCLUSION AND RECOMMENDATION	73
7.1 Conclusion	73
7.2 Recommendation	74
REFERENCES	75
APPENDIX	A-1
POSTER	A-6

LIST OF FIGURES

Figure Number	Title	Page
Figure 2.1	Zoho Inventory: Home page/ Dashboard	10
Figure 2.2	Zoho Inventory: Item panel	10
Figure 2.3	Zoho Inventory: Active Vendors panel	11
Figure 2.4	Zoho Inventory: Purchase Order panel	11
Figure 2.5	Zoho Inventory: Customer panel.	11
Figure 2.6	Zoho Inventory: Sales Orders panel	12
Figure 2.7	Odoo Inventory: home page	14
Figure 2.8	Odoo Inventory: Dashboard	15
Figure 2.9	Odoo Inventory: inventory overview	15
Figure 2.10	Odoo Inventory: Receipts page	16
Figure 2.11	Odoo Inventory: Delivery orders	16
Figure 2.12	Odoo Inventory: app page	16
Figure 2.13	Odoo Inventory: home page after install app	16
Figure 2.14	Odoo Inventory: sales page	17
Figure 3.1	System Architecture Diagram	22
Figure 3.2	Use Case Diagram	23
Figure 3.3	Activity Diagram (Add Item)	26
Figure 3.4	Activity Diagram (Add Supplier)	27
Figure 3.5	Activity Diagram (Create Purchases)	29
Figure 3.6	Activity Diagram (Create Sales)	30
Figure 3.7	Activity Diagram (Update Item)	32
Figure 3.8	Activity Diagram (Update Purchases Status)	33
Figure 3.9	Activity Diagram (Update Sales Status)	35
Figure 3.10	Activity Diagram (Update Supplier)	36
Figure 4.1	System Block Diagram	38
Figure 4.2	Flowchart	41
Figure 5.1	Login Page	46
Figure 5.2	Home Page	46
Figure 5.3	Logout	47

Figure 5.4	Edit Company Name	48
Figure 5.5	Transaction Confirmation	48
Figure 5.6	Company name updated notification	48
Figure 5.7	Inventory Page	49
Figure 5.8	Search Item	49
Figure 5.9	Add Item	50
Figure 5.10	Add Item confirmation	50
Figure 5.11	Add Item Successful	50
Figure 5.12	Show Added Item	51
Figure 5.13	Edit Item	52
Figure 5.14	Deactivate Item	52
Figure 5.15	Sales Orders Page	53
Figure 5.16	Create Sales Order	53
Figure 5.17	Confirmation Sales Order	53
Figure 5.18	Notification Sales Order Created	54
Figure 5.19	Sales Order Created	54
Figure 5.20	Sales Order Completed	54
Figure 5.21	Purchases Order	55
Figure 5.22	Create Purchase Order	55
Figure 5.23	Notification Purchase Order Created	56
Figure 5.24	Edit Purchase Order	56
Figure 5.25	Purchase Order Completed	56
Figure 5.26	Add Supplier	57
Figure 5.27	Supplier Added	58
Figure 5.28	Search Supplier	58
Figure 6.1	Survey Q1	63
Figure 6.2	Survey Q2	63
Figure 6.3	Survey Q3	64
Figure 6.4	Survey Q4	64
Figure 6.5	Survey Q5	65
Figure 6.6	Survey Q6	65
Figure 6.7	Survey Q7	66
Figure 6.8	Survey Q8	66

Figure 6.9	Survey Q9	67
Figure 6.10	Survey Q10	67
Figure 6.11	Survey Q11	68
Figure 6.12	Survey Q12	68
Figure 6.13	Survey Q13	69
Figure 6.14	Survey Q14	69
Figure 6.15	Survey Q15	70

LIST OF TABLES

Table Number	Title	Page
Table 4.1	System Components Specifications	39
Table 5.1	Specifications of laptop	43
Table 6.1	Test Result	62

Chapter 1

Introduction

In this chapter, we present the background and motivation of our research, our contributions to the field, and the outline of the thesis. The retail sector is a dynamic phase where products move beyond producers to clients and thus to the shophouse of various items and the satisfaction of the buyers. Inventory management is the backbone of the sector, may be considered the cradle in which the retail sector thrives to maintain optimum stock levels, reduce costs, and guarantee customer satisfaction. However, conventional inventory management methodologies have often had to buckle under the changing patterns of the contemporary retail sector. Daily routine manual data entry, one centralized database, and low supply chain visibility not only the issues of inefficiency, inaccuracy and vulnerability but also constitute the weakness of traditional inventory management.

1.1 Background of the Study

Inventory management is a key element of supply chain management as it entails the movement, storage and control of goods as they flow in the system from the original place through many other stages before they reach end-users or customers. The traditional systems for tracking inventories, or stock control, that many businesses use employ central databases and paperwork have some inherent problems, for example inconsistency of data, opaqueness, vulnerability to fraud and slow real-time stock monitoring. These challenges can cause major problems like stockouts, overstocking and wrong records of stock which all affect the functioning and profitability of an organization.

Blockchain technology is a decentralized, immutable and transparent, offering a promising solution to these challenges. First used for the concept of cryptocurrencies such as bitcoins, blockchain technology has slowly found footing in many industries, supply chain management inclusive. In the supply chain management practices, the blockchain can create a secure and transparent record of every inventory movement and transaction that takes place.

In this case, using an inventory management system based on blockchain technology, an organisation will be able to obtain more secure data because every occurrence is encoded and cannot be altered once documented. This helps in maintaining records of inventories in an accurate and credible manner. Furthermore, with blockchain one can have real-time information on the inventory and its flow across all the supply chain members: suppliers, manufacturers or customers.

Other advanced features that the implementation of blockchain brings includes smart contracts which add on the automation of inventory controls. These self-executing contracts can include actions to be taken such as ordering the stocks once they have depleted a certain level, which can greatly reduce the likelihood of human errors as compared to manual reordering.

In conclusion, it can be stated that the incorporation of the blockchain technology into inventory management systems reflects a radical improvement in combating the challenges attributed to the conventional inventory systems. It can therefore be seen that the introduction of blockchain-based inventory management systems offers the potential to dramatically improve supply chain operations, in terms of the accuracy of the system, the potential for cost reduction, and overall business performance. This work will therefore seek to research and explain how blockchain could be harnessed to develop a system that is capable of satisfying the needs of present-day supply chain logistics.

1.2 Problem Statement and Motivation

Traditional inventory systems are challenged in several relevant fields. Traditional supply chain systems are plagued with data inconsistency problems that arise from poor data integration and manual processes. This is where the fraudulent or old details tend to multiply, resulting in a chain reaction of bad impacts. Stockouts that come with incorrect information on inventory levels leave empty shelves which irritate customers, discouraging them from buying. Also, misleading data could wrongly point to lower stock levels when enough products are available, resulting in overstocking. Such stocks consume warehouse spaces, and the products can become unwanted and outdated. In addition, one of the factors that makes proper demand forecasting

challenging is the lack of reliable data. Without reliable forecasts, it is not possible to predict future needs and keep stock levels at the optimal level.

Moreover, the traditional system offers poor visibility and therefore it becomes hard to make an informed decision across all layers of the supply chain. The lack of live data on items or product locations and tracking does not help address the challenges of inventory levels, reordering, and supply chain management. While the data capture without real-time tracking of inventory causes retailers to stock out or have excess stock, this also can drive them to lack optimal inventory levels. The lack of a clear view along the chain of logistics makes the process of optimization more difficult to control, thus resulting in lower efficiency and increased costs. In the absence of obtaining data on product travels in real-time it is more difficult to allocate resources such as delivery schedules, warehousing and route transportation.

These problems prove the point that the traditional inventory management systems should be revised and that they should be developed in a way that would help to overcome problems, like data inconsistency, lack of transparency and limited visibility. The inefficiencies of these channels result in a chain reaction out there that affects the whole logistics ecosystem. A retailer faces the consequence of stockouts as well as overstocking and inaccurate demand forecasting. This also affects distributors who find it difficult to calculate the needs of retailers and manufacturers who are challenged with product supply based on demand expressed by retailers. On the contrary, these inefficiencies manifest themselves in the rise of costs faced by all the participants, which might end up being passed to the consumer.

The motivation for this project is the fact that the current inventory management system has a lot of drawbacks that contribute to negative effects on the retail sector and the entire supply chain. Our efforts will be directed at creating a more advanced and effective system; thereby we will be able to realize several objectives. A high level of precision in inventory accuracy translates to the presence of the product in stock which implies that there are fewer customer complaints resulting in a positive experience. Customers will remain happy if the products they need are readily available to them, thus contemplating the possibility of being held up in the checkout queue by empty shelves becomes a non-issue. Quick order fulfilment with the right pace of supply chain management will also contribute to better customer enjoyment by keeping the inventory on time.

High transparency of data enables better management of stock level, reordering, and logistics as well. This immediately boosts savings on different layers of the supply chain. From data intelligence that is always up-to-date, retailers can resolve as soon as the problem of overstocking and the subsequent storage and product damage costs. With accurate records on the product supply and sales, it is much easier for a retailer to make the right decision of replenishing stock at the right time to avoid stock out and sales loss. Moreover, it prevents the level of surplus stock production which is very costly in terms of storage and transportation. This real-time supervision involves identifying where the products arrive and where they will move throughout the supply chain and this analysis is done to locate loopholes that need to be optimized. Thus, logistics branding efforts are to establish the proper schedules, the useful warehouses and the fastest routes meaning that it lowers the cost of logistics.

A transparent and secure platform on the blockchain can form improved cooperation among retailers, distributors, and manufacturers, from supply to the end-user. All authorized stakeholders have exposure to the real-time unified data on product movement and inventory levels which bring in transparency and excludes communication gaps. The supply chain coordination process becomes more effective as retailers, distributors, and manufacturers have shared knowledge of data insights thus allowing them to anticipate demand shifts and plan production, distribution, and inventory efficiently. The immutable and transparent record storing of transactions on the blockchain prevents the rise of conflicts concerning the origin, movement and ownership of goods within the supply chain's ecosystem in an uncompromised way. A blockchain-powered inventory management system can revolutionize the retail sector by ensuring greater security, transparency, and efficiency. This program can transform the way stock is managed throughout the supply chain and this is beneficial to everyone- the retailers and distributors, producers, as well as the final consumer.

1.3 Project Objectives

The main objective of the project is to enhance the inventory system in the retail sector by implementing a blockchain-based system. This objective can be divided into sub-objectives.

1.3.1 Enhance Data Integrity

The main purpose of blockchain implementation in inventory management systems focuses on achieving strong data integrity. Traditional inventory systems remain exposed to data tampering and unauthorized changes together with unintended deletions that result in various financial irregularities, stock misplacement and fraudulent events. The decentralized system of blockchain enables secure recording of stock entries and sales orders and purchase orders as well as other transactions through an immutable timestamped database.

The blockchain uses cryptographic security to establish a chain of transactions which gets linked automatically to preceding ones while requiring network-wide agreement to prevent later tampering. The system maintains data integrity by making it impossible to modify or delete inventory records after they are recorded. The transparency of blockchain ensures that stakeholders from all departments can rely on correct inventory data while accessing it with external partners.

Through blockchain-based data integrity enhancement the solution minimizes user-generated mistakes and safeguards against unauthorized changes while establishing better responsible actions for all participants. The combination of trust and clarity becomes vital for system ecosystems which require many different stakeholders such as suppliers along with warehouse managers and store operators to work with jointly accessible inventory information.

1.3.2 Improve Traceability

Blockchain integration serves as an important element for inventory management because it allows complete tracking during supply and sales operations. The tracking of product origin along with movement and status becomes complex due to fragmented data that exists in multiple departmental or database systems in traditional inventory systems. Blockchain-based system provides complete inventory visibility because every inventory transaction includes stock addition and reordering recorded on a shared permanent ledger.

Every transaction receives permanent recording that includes an individual timestamp and specific information like product ID alongside the party responsible (e.g. supplier or store manager) and transaction type (e.g. sale or restock). Every authorized individual can access a full item history starting from supplier addition through

customer purchase. The decentralized data storage system ensures reliable truth which authorized users can validate at any time.

Through improved traceability businesses can find issues regarding stock discrepancies as well as delayed deliveries and fraudulent transactions with enhanced speed and accuracy. All participant actions become more accountable through blockchain as these actions create permanent records that allow for easy tracking of their activities.

1.3.3 Automate Reorder

The primary goal of blockchain integration with inventory systems involves the automatic restocking procedure whenever stock levels drop below predetermined thresholds. Traditional methods of inventory tasks such as reordering activities often result in extensive manual processes where human mistakes and long periods of human involvement occur. Personnel frequently overlook placing purchase orders or reorder stock too late or duplicate their orders which creates inventory shortages together with excess stock and process inefficiencies. A combination of blockchain automation with smart contracts provides a dependable approach to resolve this issue.

Smart contracts are computer programs which reside on blockchain platforms and perform automatic operations after conditions in a contract become satisfied. Smart contracts can automatically track inventory levels throughout your system. A purchase order to an enrolled supplier automatically starts when product quantities fall beneath a predetermined limit. The system eliminates human involvement for product inventory commands, so it optimizes both quick stock deliveries and proper inventory amounts.

The blockchain incorporates the reordering logic directly within its platform which results in automatic tamper-proof processes that all participants can trust transparently. All triggered automated processes maintain permanent on-chain records which both trigger events such as low stock and resulting purchase orders to suppliers, so every action stands accountable and visible. The system allows suppliers and store managers to receive instant alerts about recorders because of which restocking operations become quicker.

The system automation of reordering procedures decreases delays and lessens human oversight while boosting your demand response and operationally speeds up your business processes.

1.4 Project Scope

The project scope includes smart contracts creation represents the basis for the supply management on the blockchain. Smart contracts, which will be completed with all sorts of intricacy and care, will represent different elements of inventory control such as inventory items, transactions, and business logic. To this end, smart contracts will be used to automate tasks such as inventory counting, order processing, and supplier management which will make the supply chain reliable, trustworthy, and free of errors.

Implementing the security of the blockchain where the sensitive data is stored should be put first. Secure solutions will be put in place to prevent unauthorized access to the data, the systems being hacked and the data being tampered with. This incorporates encrypting data, utilization of access control mechanisms and security audits in which security loopholes are identified to be either repaired or removed. The key function of the system will be to uphold security, thus raising the trust and confidence of the retailers and guaranteeing the confidentiality and authenticity of their stock data.

The user interface (UI) of the inventory management system will have usability and ease of use as the key principles. The UI will be reachable through web and mobile phone and allow retailers to interact with it without compromising the system consistency through any device used. The design will centrally focus on the convenience of navigation, representation of the inventory data, and instinctive controls, meaning that the users can supervise their inventory correctly and make smart decisions based on the objective data.

Lastly, the system validation should cover the necessary aspects of its functionality, reliability and performance to ensure its effectiveness. Scenario coverage will include among others, unit testing, integration testing and user acceptance testing. It ensures that any bugs identified are addressed before launch. The system will be tested and later deployed on the network of the blockchain that will be chosen, considering the factors of availability, scalability, and capability to withstand high business demands.

1.5 Contributions

The proposed system focuses on adding value to retail inventory management by constructing and verifying the technical viability of blockchain technology applied to inventory systems. The system addresses genuine business requirements through secure transparent and efficient operations which overcome traditional inventory system disadvantages. This system defines blockchain adoption in retail by harnessing its unalterable nature together with decentralized network capabilities and trackable data availability which establishes blockchain as a fundamental technology for retail development.

This project showcases how blockchain technology functions properly for inventory management systems while establishing foundational work for academic advancement in this field. The project makes its source code publication optional alongside project findings distribution to inspire developer and researcher communities for advancing blockchain solutions in inventory control. The implementation of this step builds trust around the technology which will help speed up its industrial adoption.

Completed projects of this nature will lead to secure retail inventory practices that improve customer satisfaction while optimizing business operations and enhancing supply chain protection. The automation of reordering can occur through real-time data combined with smart contracts which reduce stockout frequency and guarantee precise order fulfillment thus delivering quick deliveries to satisfied customers. Product traceability improvements fight counterfeits which builds strong relationships between brands and their customers.

The analysis of current data allows organizations to create supply chain decisions based on facts which helps them save expenses while reducing product waste during distribution processes. The supply chain productivity strengthens considerably when coordination between players improves. Blockchain delivers verifiable data that fights fraudulent activities thus supporting a secure along with transparent supply chain operation. The establishment of reliable relationships and cooperative bonds exists between all parties involved.

1.6 Report Organization

This report is organized into 7 chapters: Chapter 1 Introduction, Chapter 2 Literature Review, Chapter 3 System Methodology, Chapter 4 System Design, Chapter 5 System Implementation, Chapter 6 System Evaluation And Discussion, Chapter 7 Conclusion and Recommendations. The first chapter is the introduction of this project which includes problem statement, project background and motivation, project scope, project objectives, project contribution, highlights of project achievements, and report organization. The second chapter is the literature review carried out on several existing traditional inventory systems in the market to evaluate the strengths and weaknesses of each product. The third chapter discusses the overall system design of this project. The fourth chapter is regarding the details on how to implement the design of the system. Furthermore, the fifth chapter is system implementation. The sixth chapter discusses system evaluation. Finally, the last chapter is the conclusion and recommendations.

Chapter 2

Literature Review

Overview

This literature Review will review two existing online traditional inventory management system to determine and identify the strengths and limitations of the existing inventory management system. It will help generate innovative ideas and improve the system that I will develop in my final year project.

2.1 Zoho Inventory

[1]Zoho Inventory is a popular cloud-based inventory management system that helps small and medium-sized enterprises (SMEs) to manage their inventory effectively. It offers real-time stock tracking, multi-channel selling, order management, and detailed reporting. The system integrates seamlessly with other Zoho products, such as Zoho Books and Zoho CRM, as well as external platforms like Amazon, eBay, Shopify, and various shipping carriers. This extensive integration allows businesses to manage their entire sales and inventory processes from a single platform, streamlining operations and improving efficiency.

Zoho Inventory is a widely recognized inventory management system that caters to small and medium-sized enterprises (SMEs). As part of the Zoho suite, it integrates seamlessly with other applications like Zoho Books and Zoho CRM, providing a holistic approach to managing business operations. Key features of Zoho Inventory include:

Stock Management: Zoho Inventory allows businesses to track stock levels in real time, manage warehouses, and set reorder points to avoid stockouts.

Order Management: The system streamlines order processing by automating order creation, tracking shipments, and syncing orders across various sales channels like Amazon, eBay, and Shopify.

Multi-channel Selling: Zoho Inventory integrates with multiple e-commerce platforms, enabling businesses to manage sales from a single interface.

Reporting and Analytics: The system offers detailed reports and analytics, helping businesses make data-driven decisions.

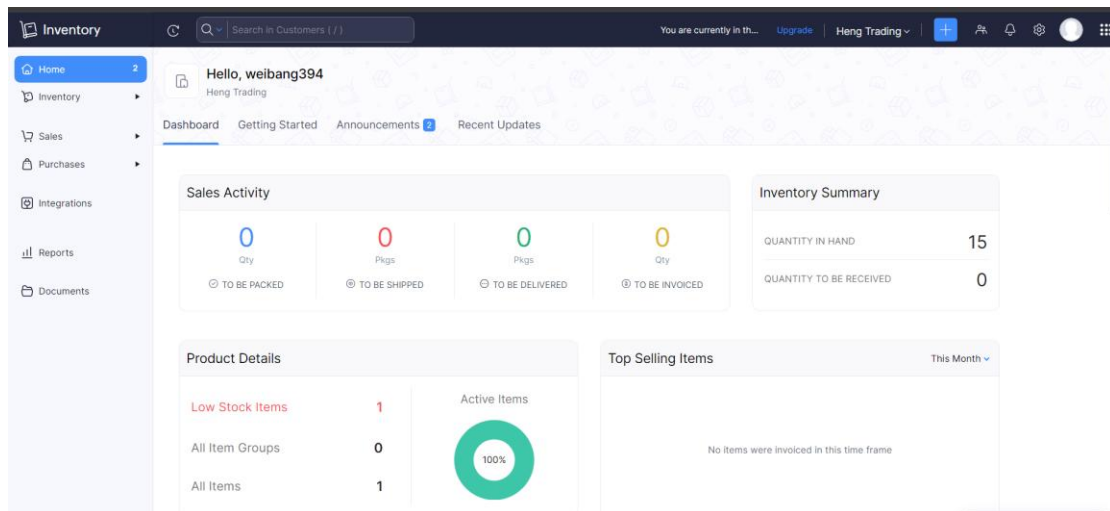


Figure 2.1 Zoho Inventory: Home page/ Dashboard

In Zoho Inventory it has multiple functions such as inventory, sales, purchases, integrations, report and the current page is the home page of the dashboard. Dashboard is a web page to display all the information about the inventory such as sales activity, inventory summary, product detail and top selling items.

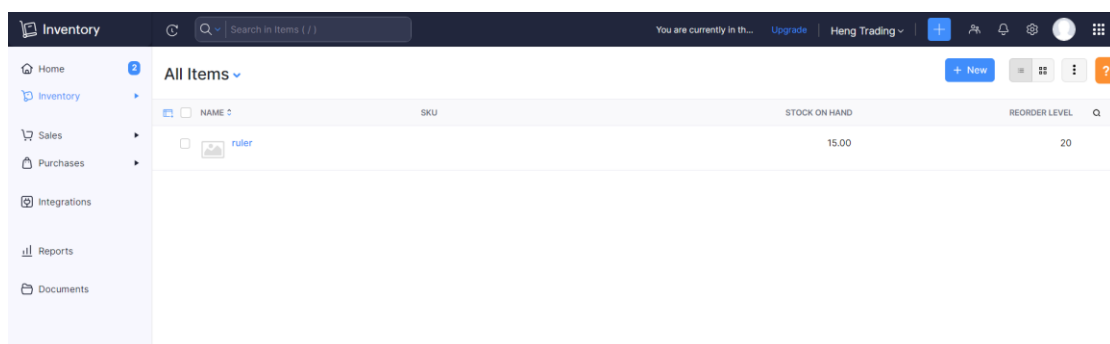
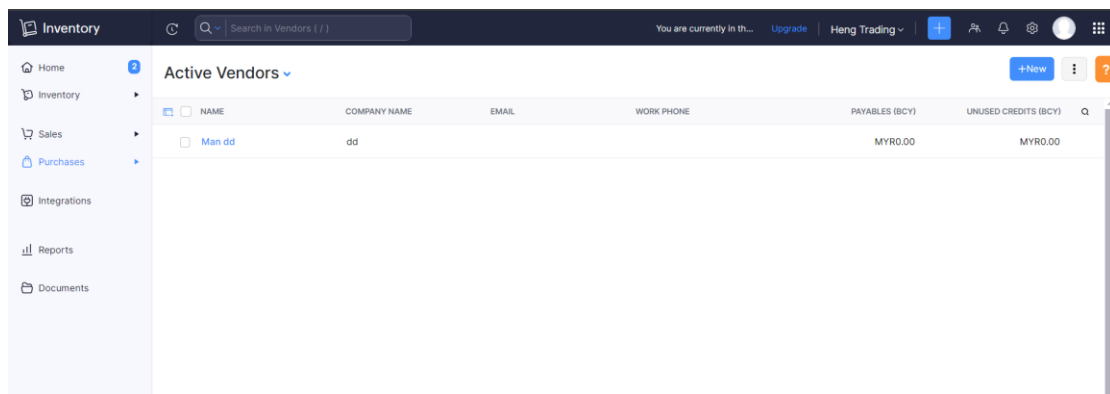


Figure 2.2 Zoho Inventory: Item panel

Other than that, the item panel admin can add new items and also the item information such as price, image of items, reorder level and item name. After adding it will show all the items in this page. In this page admin can easily manage the inventory because it clearly shows the stock level of each item.

CHAPTER 2

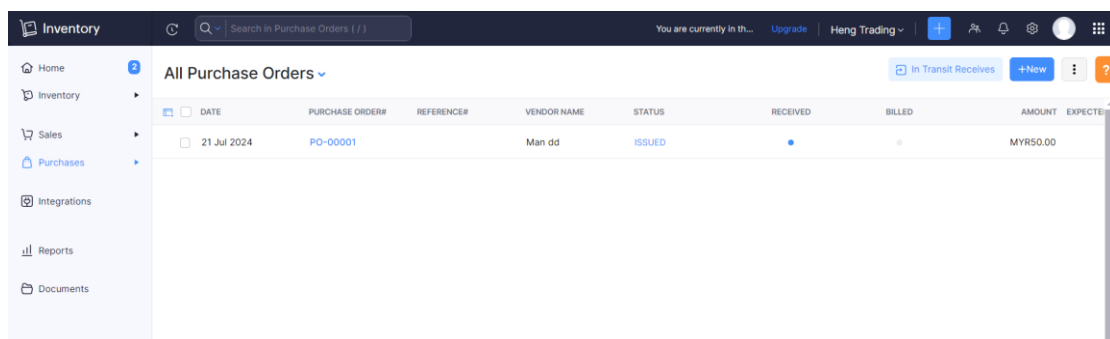


The screenshot shows the 'Active Vendors' panel in Zoho Inventory. The left sidebar contains navigation links: Home, Inventory, Sales, Purchases, Integrations, Reports, and Documents. The main header includes a search bar for vendors, a user profile dropdown, and a 'Heng Trading' label. The table below lists vendors with columns for NAME, COMPANY NAME, EMAIL, WORK PHONE, PAYABLES (BCY), and UNUSED CREDITS (BCY). A single vendor named 'Man dd' is listed with a payable of MYR0.00 and unused credits of MYR0.00.

NAME	COMPANY NAME	EMAIL	WORK PHONE	PAYABLES (BCY)	UNUSED CREDITS (BCY)
Man dd	dd			MYR0.00	MYR0.00

Figure 2.3 Zoho Inventory: Active Vendors panel

Active vendors panel is showing all the vendors. Admin can manage all vendors, add the vendors information such as vendor name and company, payable, email, phone and unused credits.

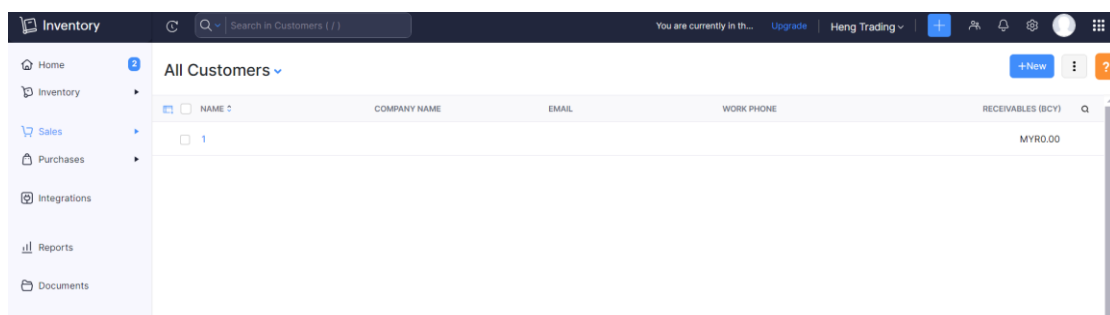


The screenshot shows the 'All Purchase Orders' panel in Zoho Inventory. The left sidebar is the same as in Figure 2.3. The main header includes a search bar for purchase orders, a user profile dropdown, and a 'Heng Trading' label. The table below lists purchase orders with columns for DATE, PURCHASE ORDER, REFERENCE, VENDOR NAME, STATUS, RECEIVED, BILLED, AMOUNT, and EXPECTED. A single purchase order is listed with date 21 Jul 2024, reference PO-00001, vendor Man dd, status ISSUED, and amount MYR50.00.

DATE	PURCHASE ORDER	REFERENCE	VENDOR NAME	STATUS	RECEIVED	BILLED	AMOUNT	EXPECTED
21 Jul 2024	PO-00001		Man dd	ISSUED			MYR50.00	

Figure 2.4 Zoho Inventory: Purchase Order panel

Purchase order panel is a page for creating purchase orders through the vendors that are set in the vendor panel. It displays the date, vendor name, status, received, billed, amount and also expected delivery date.



The screenshot shows the 'All Customers' panel in Zoho Inventory. The left sidebar contains navigation links: Home, Inventory, Sales, Purchases, Integrations, Reports, and Documents. The main header includes a search bar for customers, a user profile dropdown, and a 'Heng Trading' label. The table below lists customers with columns for NAME, COMPANY NAME, EMAIL, WORK PHONE, and RECEIVABLES (BCY). A single customer is listed with a receivable of MYR0.00.

NAME	COMPANY NAME	EMAIL	WORK PHONE	RECEIVABLES (BCY)
1				MYR0.00

Figure 2.5 Zoho Inventory: Customer panel

Moreover, the customer panel displays all the customer information. Admin can add and delete the customer. In this page it displays the customer information such as name,

CHAPTER 2

company name, email, phone and receivables. Admin can easily manage the customer through this page.

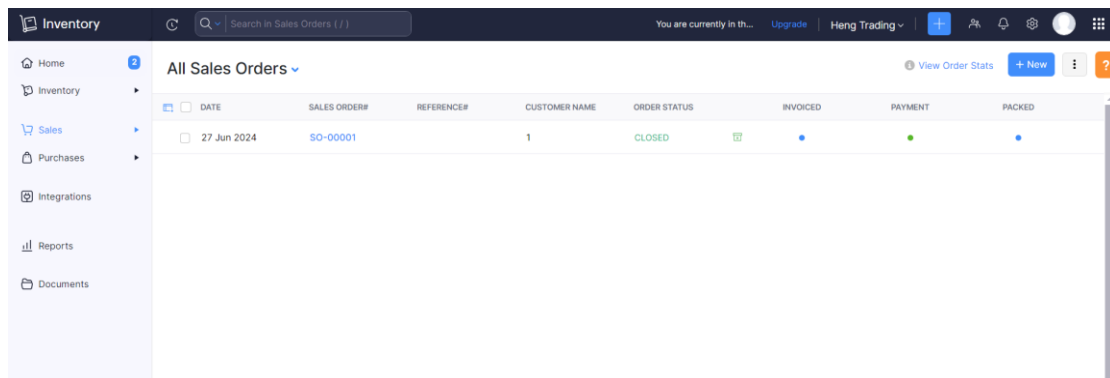


Figure 2.6 Zoho Inventory: Sales Orders panel

Sales order panel is let admin to manage the sales orders. It shows all the sales orders information such as date, sales id, reference, customer name, order status, invoiced, payment status and also packed.

However, there is a main drawback of using the centralized system offered by Zoho Inventory. The various data is consolidated in the Centralized system and can therefore be at risk from hackers, loss of data and system breakdown. These are some of the vulnerabilities that exist in an organization and they may cause inconvenience and may lead to monetary losses and tarnishing of the image of an organization. Also, as discussed above, Zoho Inventory offers many tools and functionalities, but it can still lack certain customizability to fit every company's requirement, especially if those are highly specialized and unusual for inventory management.

2.1.1 Strengths and Weakness of Zoho Inventory System

Zoho Inventory System is a cloud-based inventory management system that provides several advantages for users.

Ease of Use: These systems are equipped by interfaces which can be controlled even by an inexperienced user. These make its adoption and incorporation into business operations to be simpler hence can be easily adopted.

Comprehensive Feature Sets: Mentioned systems, and Zoho Inventory as one of them, provide users with effective instruments for multi-channel selling, orders tracking, and detailed reports. It helps businesses to be more effective in managing their stock and in

the selling of their products.

Integration Capabilities: This feature of expansion of integration with other Zoho products and other platforms like e-Commerce sites and shipping carriers makes it all encompassing to manage business through Zoho Inventory.

Despite their strengths, traditional inventory systems have notable weaknesses:

Centralized Vulnerabilities: These systems are centralized and therefore prone to cyberattacks and system failures as has been observed of late. As much as it is pertinent to note that having one access point weakens business since in the event of a disaster, business operations and data will be affected.

Limited Customization: The limitation with conventional systems is that although many features are delivered, there is little ability to fine tune the system. This can help stop a business from making as many adaptations to the software as it needs to meet its requirements.

Dependency on Internet Connectivity: These are cloud-based systems and as such, they rely on internet connection to run. Establishments depending on the internet connection in delivering their services may experience difficulties in their operations for long periods.

2.2 Odoo Inventory System

Proper inventory management[2] is critical to the operations of various businesses and organizations especially producers, retailers and distributors. A good inventory management system is when a business can make the right decisions regarding the stock, reduce the expenses and satisfy the clients. Odoo Inventory is one of varieties in the Odoo ERP line that provides rather efficient instruments for inventory management. It is more beneficial to lay focus on the accomplishments and disadvantages of Odoo Inventory as it will provide the audience with understanding of the current state of inventory management systems and the fields that need to be developed.

Odoo Inventory is a free and open-source solution that is part of the Odoo Enterprise Resource Planning system. This will assist business entities in their ability to handle stocks effectively with options like stock monitoring, order restocking and a facility to manage several canters. It is part of the Odoo framework, Odoo Inventory is tightly connected with other Odoo modules like Sales, Purchase, and Accounting that create a single platform for business management.

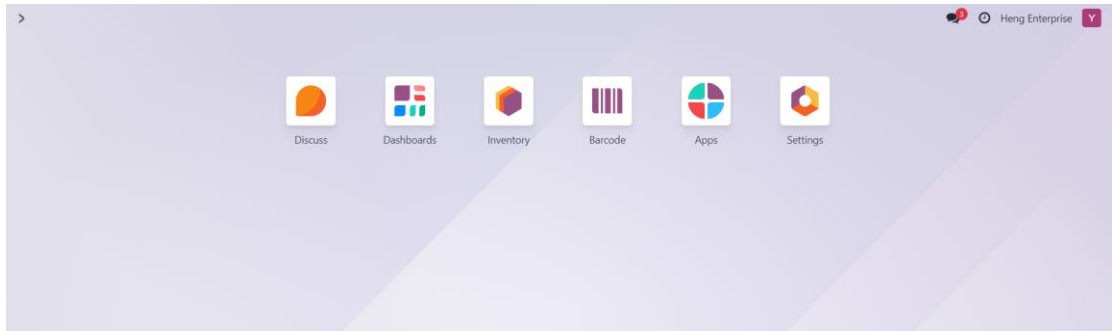


Figure 2.7 Odoo Inventory: home page

The home page of the Odoo Inventory had several function such as discuss, dashboards, inventory, barcode, app and setting.

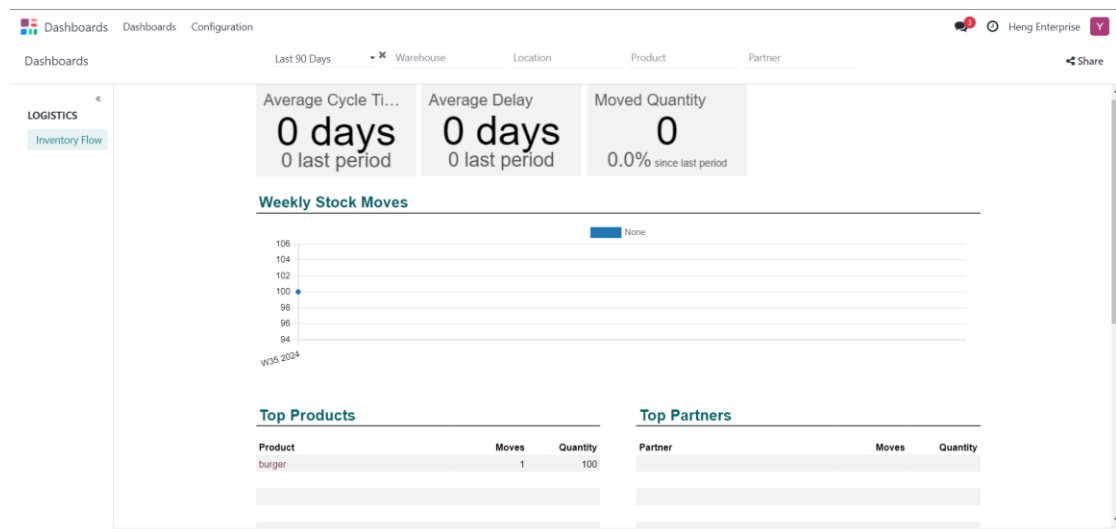


Figure 2.8 Odoo Inventory: Dashboard

This page is a dashboard that shows the overall inventory flow, such as average cycle time, average delay, move quantity, weekly stock moves graph, top product, top partner, top warehouse and top location of the stock. Users also can set the filter to show the period that they need and also can filter the warehouse, location, product and partner.

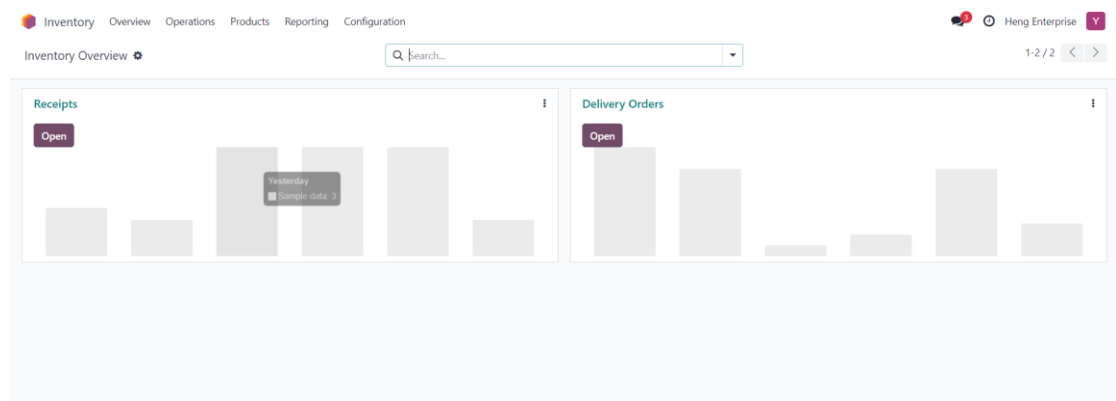


Figure 2.9 Odoo Inventory: inventory overview

CHAPTER 2

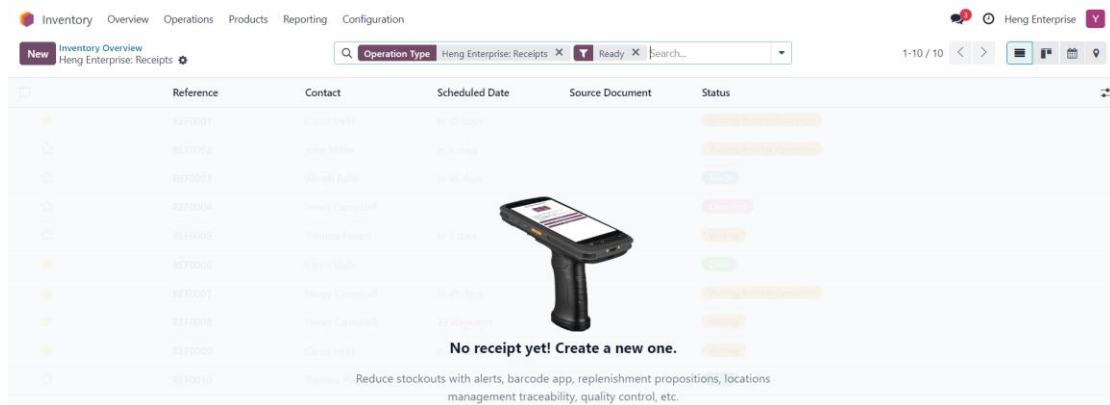


Figure 2.10 Odoo Inventory: Receipts page

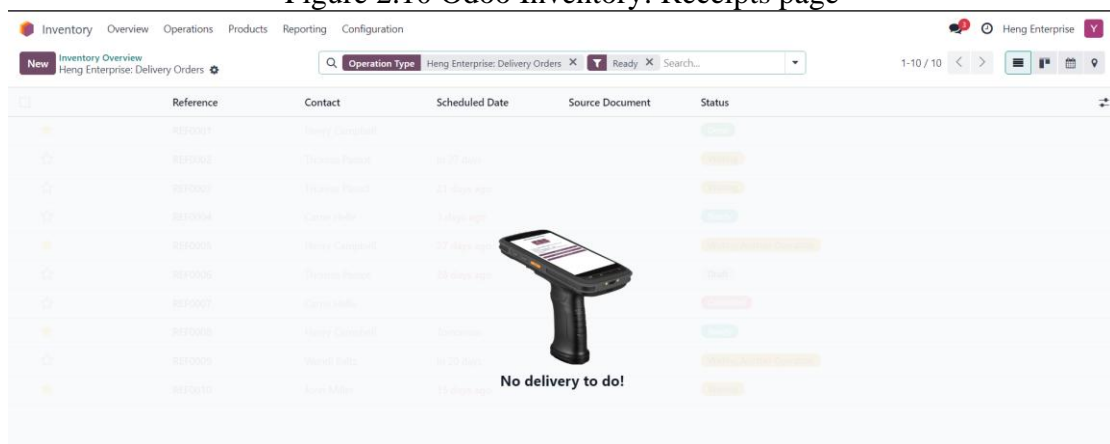


Figure 2.11 Odoo Inventory: Delivery orders

In the inventory overview page, it displays the bar chart about receipt and delivery order. In the receipts page and delivery page the user can manage the receipt. This page displays information such as references, contact, scheduled date, source document and status of the order. Through this page users can easily manage the receipt and delivery orders.

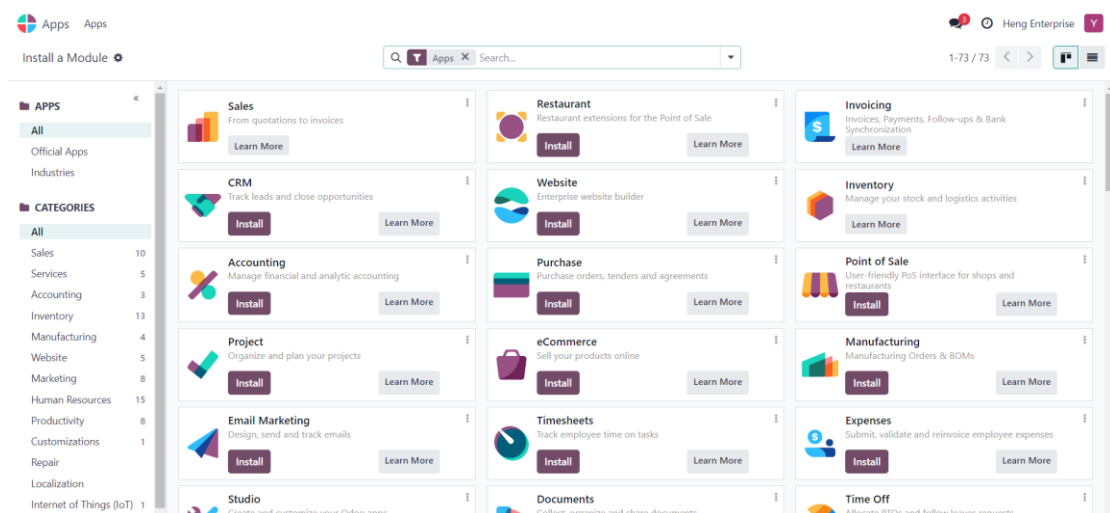


Figure 2.12 Odoo Inventory: app page

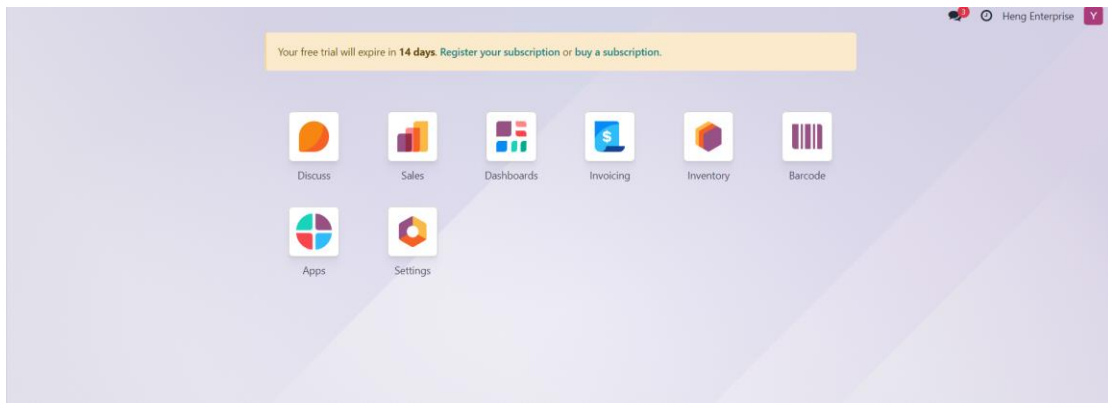


Figure 2.13 Odoo Inventory: home page after install app

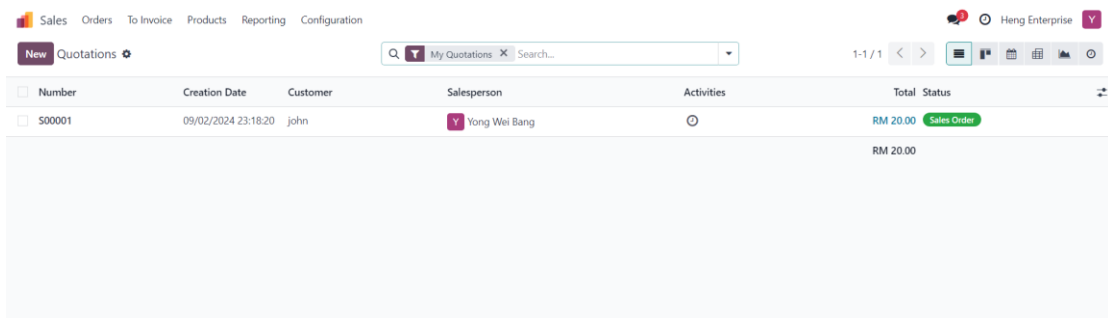


Figure 2.14 Odoo Inventory: sales page

In the app page, users can install many types of the module. After installing the sales module, it will add it to the home page such as sales and invoicing. Then in the sales page it can create sales and on this page it show the information about sales such as id, date, customer, salesperson, activities, total amount and status. It can let users easily manage the sales order.

2.2.1 Strengths and Weakness of Odoo Inventory System

Odoo Inventory is a free and open-source solution that is part of the Odoo Enterprise Resource Planning system that has several strengths.

Comprehensive Feature Set

Real-Time Inventory Tracking: Odoo Inventory offers accurate stock level tracking which then enables businesses to have a clue of their current stock.

Automated Replenishment: The system helps in auto-reorder of the inventory depending on the low levels set to actually help avoid having no stock at all or having an excess of the stock.

Multi-Warehouse Management: The Odoo Inventory module makes it easier for businesses to control their stock inventory from various warehouses and locations, thereby giving a central view.

Integration with Odoo ERP Suite

Unified Platform: It belongs to the wide range of modules that is the Odoo ERP system, and it can work with other Odoo applications such as Sale, Purchase, and Account. This integration makes it possible for businesses to have an integrated system of operation making them more efficient and minimizing on duplication of effort.

Flexibility and Customizability

Open-Source Model: Odoo ERP includes a module known as the Inventory application; the platform is open-source and allows for changes to be made to the software to suit a business organization's requirements. It makes it possible to develop quite flexible solutions which can be easily adjusted according to the necessities of a particular industry.

Scalability: The system has the capability to grow with the expansion of the company by being able to handle increased number of inventories and more warehouses.

Advanced Features

Barcode Scanning: Odoo Inventory has the feature of Barcode, and it is very helpful in Bar-code scanning and other tasks like stock-taking and labelling of products.

Batch and Serial Number Tracking: Some of the elaborate tracking options include batch tracking and serial number tracking which is very essential for organizations that have strict tracking needs.

Despite their strengths, Odoo inventory systems weaknesses

Complexity and Learning Curve:

System Complexity: Although the Odoo Inventory offers numerous features and settings, finding and enabling them could be a challenge since it is an open-source system – this may be challenging for small and medium-scale businesses as well as firms with less IT knowledge.

Learning Curve: Sometimes the users, during the initial phase, take some time in familiarising themselves with various options offered by the system.

Customization and Technical Expertise

Need for Technical Expertise: Most of the times, tuning the Odoo Inventory to a customer's needs may be complex, and one has to seek the help of the developers. This is a challenge for businesses which do not have an internal IT department or a team to sort it out.

Implementation Challenges: As and perhaps the initial installation process of the system may involve several steps and configurations may take time, professional help may be called in which may add to the cost of the implementation.

Cost Considerations

Additional Costs: This is correct as Odoo has their Odoo community edition which is free but the enterprise edition that has more features and support costs licenses. In particular, customization and implementation services can also bring about an increase in the cost.

Integration with Non-Odoo Systems

Limited Integration Options: Despite the fact that Odoo Inventory works well with the other Odoo modules, businesses using other systems may face difficulties on the integration of external software/systems to the Odoo solution.

2.3 Compare with Proposed Solution

Comparing Zoho Inventory and Odoo Inventory with a blockchain-based inventory management system, several key differences emerge that highlight the advantages of the blockchain approach.

One of the main challenges observed in inventory management system is security. Unfortunately, Zoho Inventory has a central architecture which makes this application insecure to cyberattacks and data breaches. The central server is quite vulnerable since a compromise in it would lead to vulnerability of the whole system. Regarding the Odoo Inventory module, data is stored in the classical Database which means that despite the fact the data is safe, some users can manipulate it, which leads to the trust issue. On the other hand, a blockchain-based system provides a decentralized structure for the data, and its copies are stored in many nodes, and hackers cannot target one centralized location to breach in and alter the records easily. Moreover, it is to be noted that due to the typical characteristics of the blockchain, once data is written it

cannot be changed, which in turn adds more layer of security and reliability to the application.

Another difference is the level of transparency and the trust which is required for successful operation. Zoho and Odoo Inventory systems use conventional databases in which the information storing remains valid only if the authority oversees the system. This means that there could be cases of differences and less trust between users especially when the data has to be audited or checked. A blockchain based system however, has the provision to maintain the records of all the transactions which are secured and cannot be modified. It also guarantees full disclosure and will be very helpful in places where there are strict regulatory compliance and business workflow documentation since no one has the ability to alter the records.

Regarding customization and automation, Zoho Inventory provides a rich list of tools, nevertheless, its customization possibilities are not very vast, so it is not very suitable for niche-oriented organizations. One advantage of free Odoo Inventory is that this software is more flexible compared to others since it is open-source, but one disadvantage is that companies which decided to use open-source software have to solve many problems themselves and it can be very difficult if they have no IT specialists in their team. Smart contracts in a blockchain-based system provide a great deal of flexibility in terms of customization and opportunities to automate reordering, payments and stock management when it is set by certain conditions that are already programmed to be engaged without human intervention. This increases efficiency and reduces human error, making the system more adaptable and efficient than traditional inventory management systems.

Scalability and integration are other good examples of features that must not be overlooked when choosing between the two models. Zoho inventory is best suitable for small and medium enterprises but sometimes they may face performance problems with increasing number of transactions. Odoo Inventory is also an enterprise-level app that has problems connecting with other non-Odoo systems especially for organizations that use a variety of systems for operations. The blockchain based system is designed to connect a large number of participants in a distributed environment and maintain or even improve the speed of operation with increasing data and transaction rates. Additionally, with a blockchain based system, one can easily link to other blockchain based systems and while with traditional systems they can always link to traditional as

CHAPTER 2

well as blockchain based systems using APIs, there seems to be a better cohesiveness to a business environment.

In conclusion it can be stated that the implementation of an inventory management system with application of blockchain is free from many of the shortcomings that are found in Zoho and Odoo Inventory. Thus, the blockchain approach has pros of security and transparency, possibility of customization, automation, scalability, and integration as compared to the traditional methods of inventories' management and is preferable for industries needing advanced and highly reliable systems.

Chapter 3 System Methodology

Proposed Method/Approach

The Inventory Management System depends on blockchain technology to generate complete secure transactions along with full transparency that tracks inventory changes without alteration. The system operates through decentralized application (dApp) software connections to public blockchain systems while providing user access through an internet web interface.

The system foundation comprises smart contracts that enable the automation of essential business operations starting from add supplier then moving to product storage all the way to sales and purchase order administration along with inventory reordering automation. The system adopts smart contracts for transactions which provide unalterable verification, thus improving data reliability between all stakeholders.

The system implements Agile principles with modifications for single developer operations. The multiple development sprints maintain short durations for managing specific tasks that include smart contract programming and frontend user interface design and backend integration work. A product review takes place after each sprint to execute necessary updates that rely on test outcomes and usability objectives.

Software development as a single-person project requires proper planning together with self-assessment and iterative methodology for preserving both system code quality and user-friendly features. Continuous testing checks for potential bugs and conflicts which may come from new updates. The system stability alongside reliability continues as an active factor throughout its entire developmental phase.

A solo project benefits from Agile implementation because it maintains adaptability to new obstacles that arise in the development process. Leaders in academia and industry now possess an expandable and protected blockchain inventory system to address actual retail constraints while creating opportunities for research growth or business development.

3.1 System Design Diagram

3.1.1 System Architecture Diagram

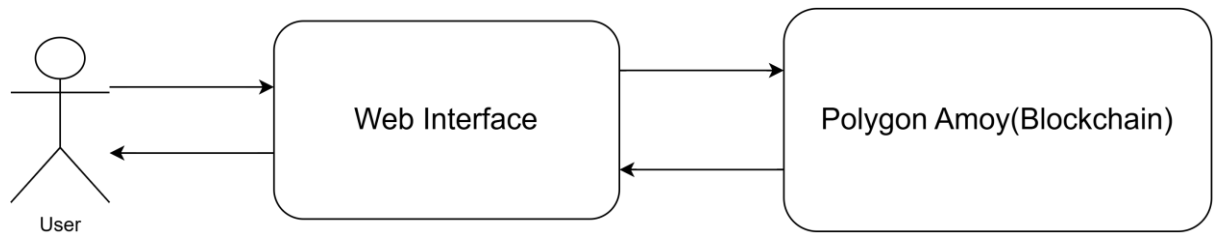


Figure 3.1 System Architecture Diagram

The system consists of three main components:

User, Web Interface and Polygon Amoy Blockchain

- **User:** This represents the stock manager interacting with the system. The user accesses the system through a web-based interface to perform various actions like manage stock, create order and manage suppliers.
- **Web Interface:** This acts as the front-end of the decentralized application (dApp). It is built using modern web technologies such as React and connects to the blockchain network via libraries like Ethers.js. The interface captures user inputs, communicates with smart contracts, and displays data retrieved from the blockchain.
- **Polygon Amoy (Blockchain):** This is the testnet version of the Polygon network used during development and testing. It hosts the deployed smart contracts that manage core functionalities such as product registration, stock control, and transaction recording. All business logic is executed within these smart contracts to ensure transparency, immutability and trustless operations.

The arrows in the diagram show the bidirectional flow of data between components. Users interact with the web interface, which in turn sends transactions to the blockchain. Once a transaction is confirmed, the updated information is sent back and displayed on the user interface.

This architecture ensures a secure, decentralized, and user-friendly environment for managing inventory data, reducing the risk of tampering, data loss or central point of failure.

3.1.2 Use Case Diagram and Description

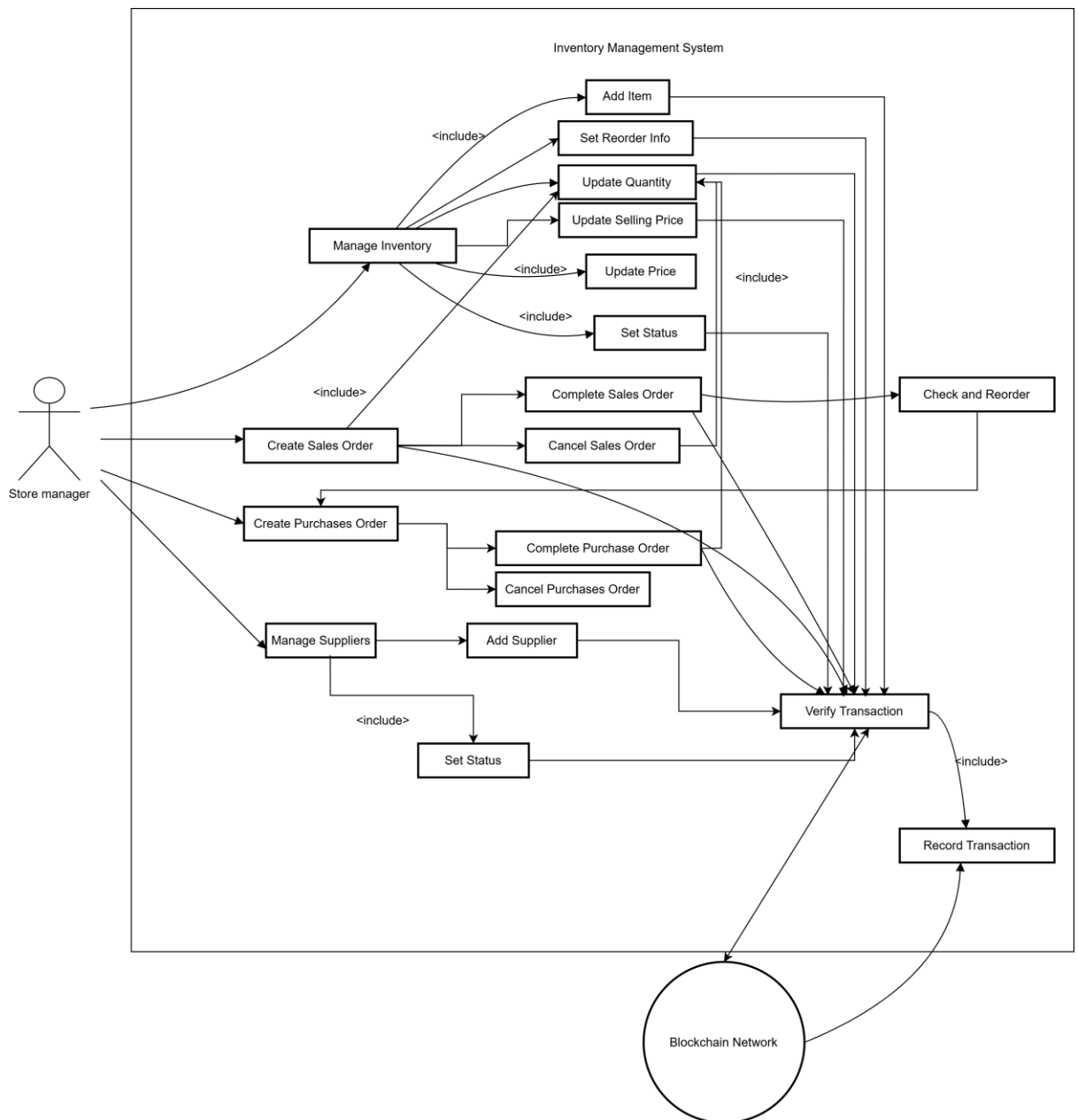


Figure 3.2 Use Case Diagram

The diagram shows the core functionalities and interactions within the Inventory Management System and how to relate the Blockchain Network.

Main Actor:

- User: The user interacts with the system to perform inventory-related tasks such as managing stock, creating orders, and verifying transactions.

CHAPTER 3

Use Case Groups and Functions:

1. Manage Inventory

This includes all inventory-related operations:

- Add Item: Add new products to the inventory.
- Set Reorder Info: Define reorder thresholds and restock levels.
- Update Quantity: Adjust product quantities based on new purchases or sales.
- Update Selling Price: Change the selling price of a product.
- Set Status: Change item status (e.g., Active, Disable).
- These actions all include interaction with the Verify Transaction process to ensure data integrity.

2. Create and Manage Sales Orders

- Create Sales Order: Create a new sales record.
- Complete Sales Order: Confirm the delivery or completion of a sale.
- Cancel Sales Order: Void an existing sales order.
- Each order-related action connects to:
 - Check and Reorder: Automatically checks stock levels to trigger restock if needed.
 - Verify Transaction: Ensure changes are valid before saving to blockchain.

3. Create and Manage Purchase Orders

- Create Purchases Order: Create a new purchase order to restock inventory.
- Complete Purchase Order: Confirm receipt of goods from supplier.
- Cancel Purchases Order: Cancel any purchase order.
- All operations go through Verify Transaction for validation.

4. Manage Suppliers

- Add Supplier: Register new suppliers.
- Set Status: Update supplier statuses (e.g. Active, disable).
- Involves blockchain verification for legitimacy and accuracy.

Blockchain Integration

- Verify Transaction: Every action that changes data (inventory, orders, or suppliers) is first verified.
- Record Transaction: Once verified, transactions are immutably saved on the blockchain network (Polygon Amoy).

System Goals Reflected in the Diagram:

- Transparency: Every operation is verified and recorded on the blockchain.
- Automation: Reordering and transaction validation are built in.
- Security: Use of smart contracts ensures trust and eliminates tampering.
- Traceability: Every step in the system is traceable via blockchain logging.

3.1.3 Activity Diagram

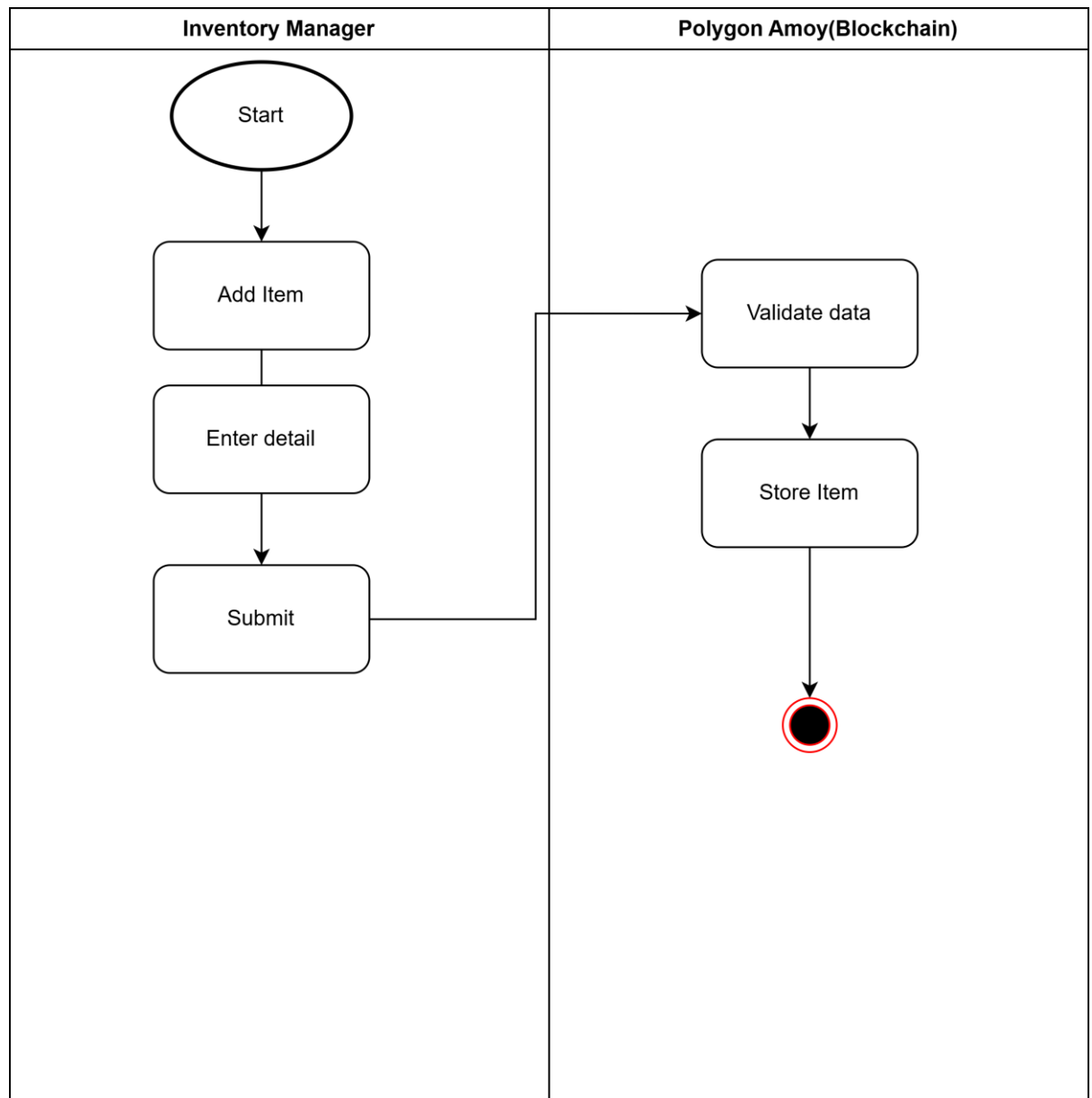


Figure3.3 Activity Diagram (Add Item)

The activity diagram shows that adding a new inventory item into the system. The diagram is divided into two swim lanes representing the **Inventory Manager** and the **Polygon Amoy (Blockchain)** components, showing the interaction between the user and the blockchain network.

Description of Activities:

- **Start:** The process begins when the Inventory Manager initiates the "Add Item" function.
- **Add Item:** The manager selects the option to add a new inventory item.

- **Enter Detail:** The user inputs relevant item information such as item name, quantity, supplier, etc.
- **Submit:** Once all details are entered, the manager submits the item information.
- **Validate Data (Blockchain):** The submitted data is sent to the blockchain (Polygon Amoy testnet) where it is validated to ensure correctness and authenticity.
- **Store Item (Blockchain):** If validation passes, the item data is recorded immutably on the blockchain.
- **End:** The activity concludes once the item is successfully stored.

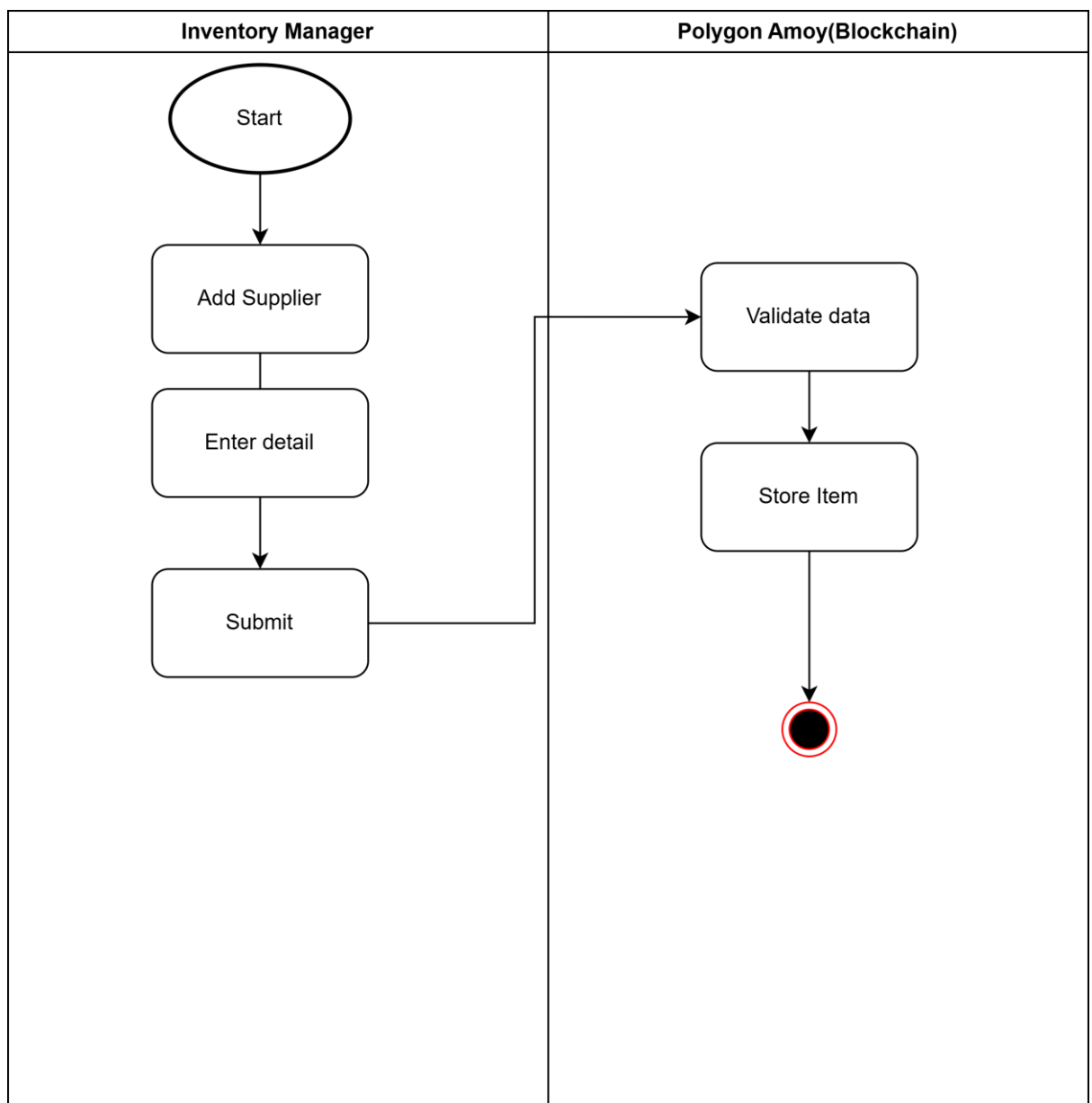


Figure3.4 Activity Diagram (Add Supplier)

The activity diagram shows that adding a new supplier into the system. The diagram is divided into two swim lanes representing the **Inventory Manager** and the **Polygon Amoy (Blockchain)** components, showing the interaction between the user and the blockchain network.

Description of Activities:

- **Start:** The process begins when the Inventory Manager initiates the "Add Supplier" function.
- **Add Supplier:** The manager selects the option to add a new supplier.
- **Enter Detail:** The user inputs relevant item information such as name and contact information.
- **Submit:** Once all details are entered, the manager submits the information.
- **Validate Data (Blockchain):** The submitted data is sent to the blockchain (Polygon Amoy testnet) where it is validated to ensure correctness and authenticity.
- **Store Supplier (Blockchain):** If validation passes, the data is recorded immutably on the blockchain.
- **End:** The activity concludes once the supplier is successfully stored.

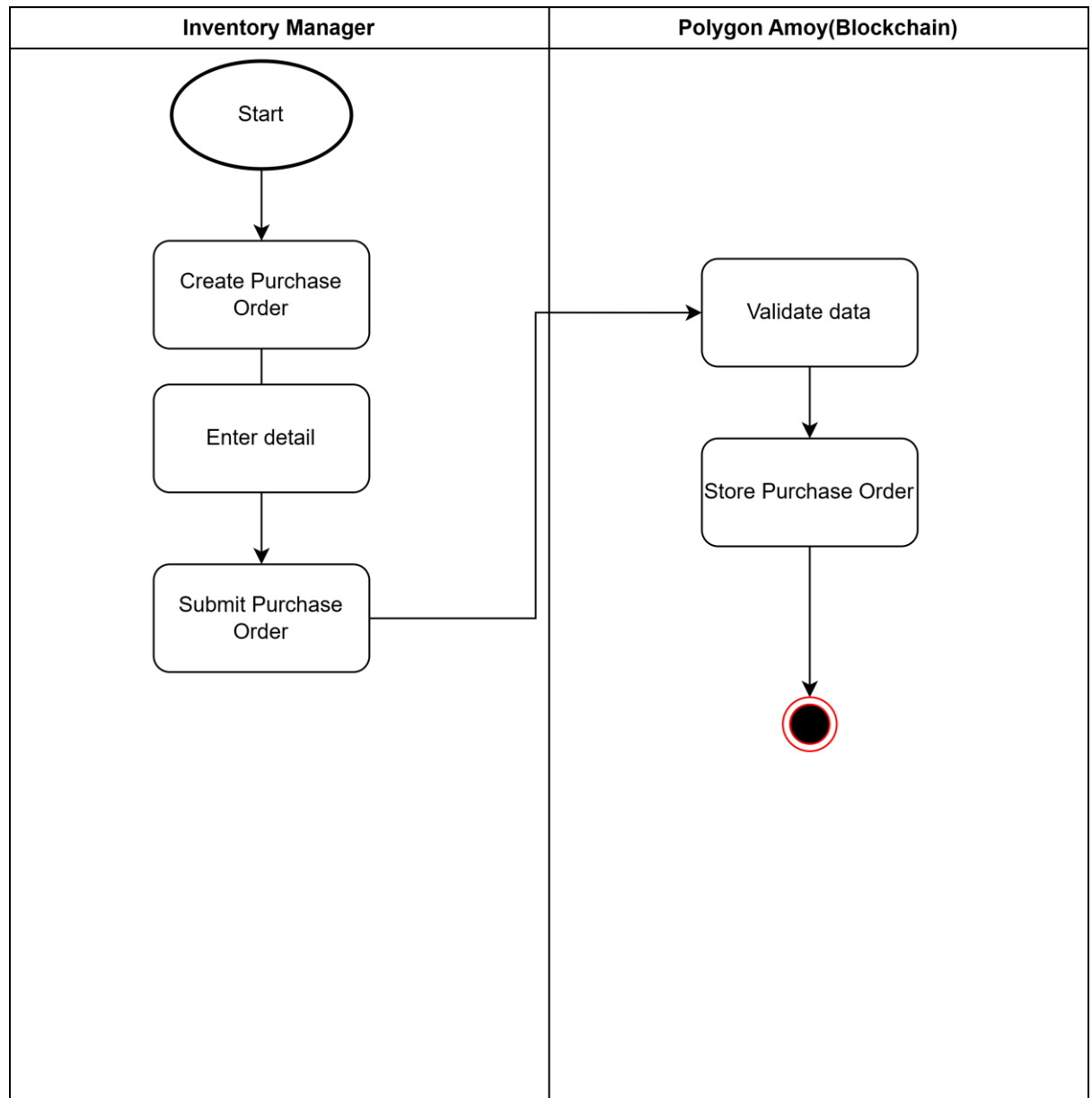


Figure 3.5 Activity Diagram (Create Purchases)

The activity diagram shows that adding a new purchase order into the system. The diagram is divided into two swim lanes representing the **Inventory Manager** and the **Polygon Amoy (Blockchain)** components, showing the interaction between the user and the blockchain network.

Description of Activities:

- **Start:** The process begins when the Inventory Manager initiates the "Create Purchase" function.
- **Create Purchase Order:** The manager selects the option to create a purchase order.

- **Enter Detail:** The user inputs relevant item information such as name, price, supplier, etc.
- **Submit:** Once all details are entered, the manager submits the information.
- **Validate Data (Blockchain):** The submitted data is sent to the blockchain (Polygon Amoy testnet) where it is validated to ensure correctness and authenticity.
- **Store Purchases Order (Blockchain):** If validation passes, the data is recorded immutably on the blockchain.
- **End:** The activity concludes once the supplier is successfully stored.

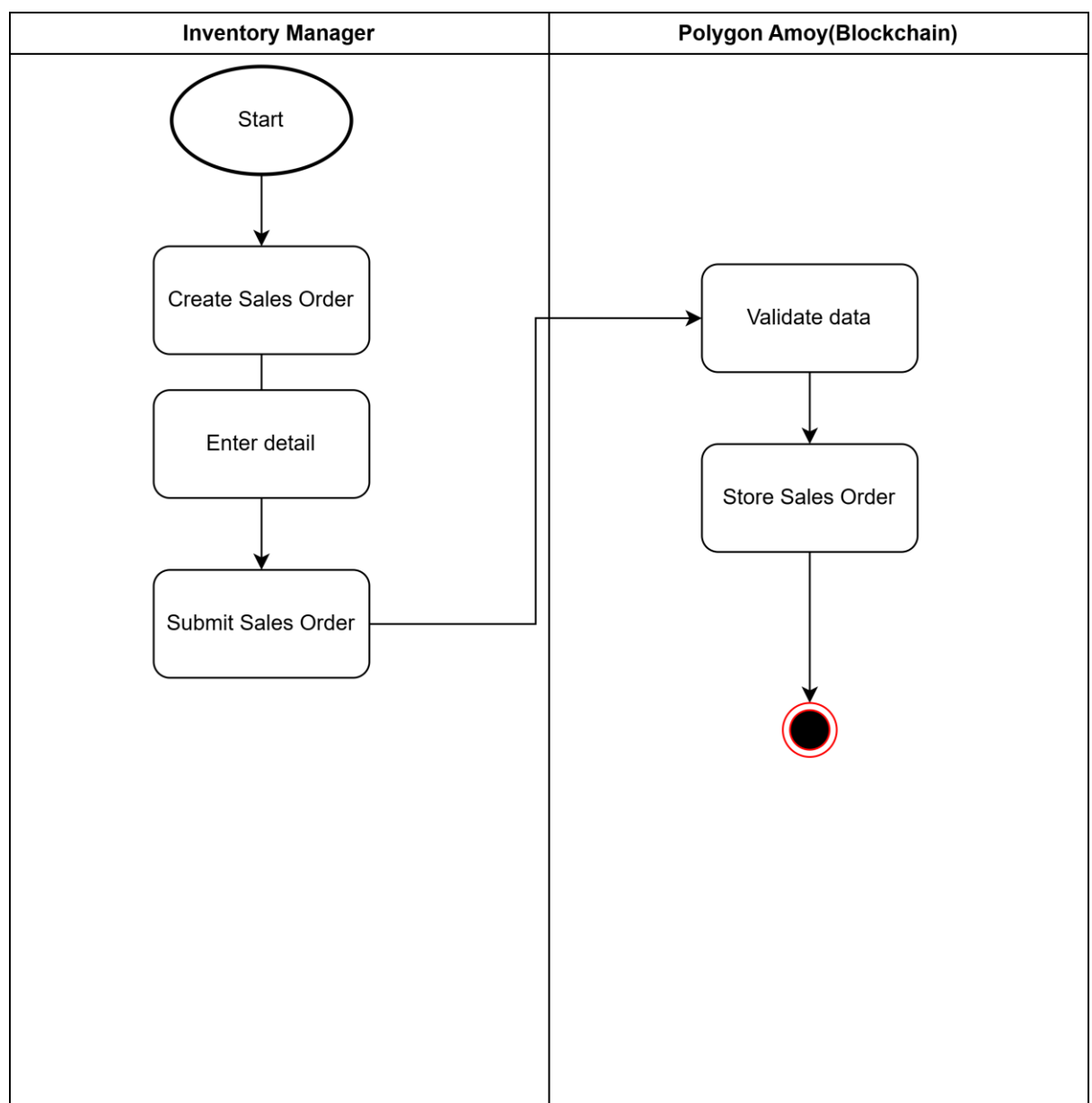


Figure 3.6 Activity Diagram (Create Sales)

The activity diagram shows that adding a new sales order into the system. The diagram is divided into two swim lanes representing the **Inventory Manager** and the **Polygon Amoy (Blockchain)** components, showing the interaction between the user and the blockchain network.

Description of Activities:

- **Start:** The process begins when the Inventory Manager initiates the "Create Sales" function.
- **Create Sales Order:** The manager selects the option to a create sales order.
- **Enter Detail:** The user inputs relevant item information such as name, price, customer name, etc.
- **Submit:** Once all details are entered, the manager submits the information.
- **Validate Data (Blockchain):** The submitted data is sent to the blockchain (Polygon Amoy testnet) where it is validated to ensure correctness and authenticity.
- **Store Sales Order (Blockchain):** If validation passes, the data is recorded immutably on the blockchain.
- **End:** The activity concludes once the supplier is successfully stored.

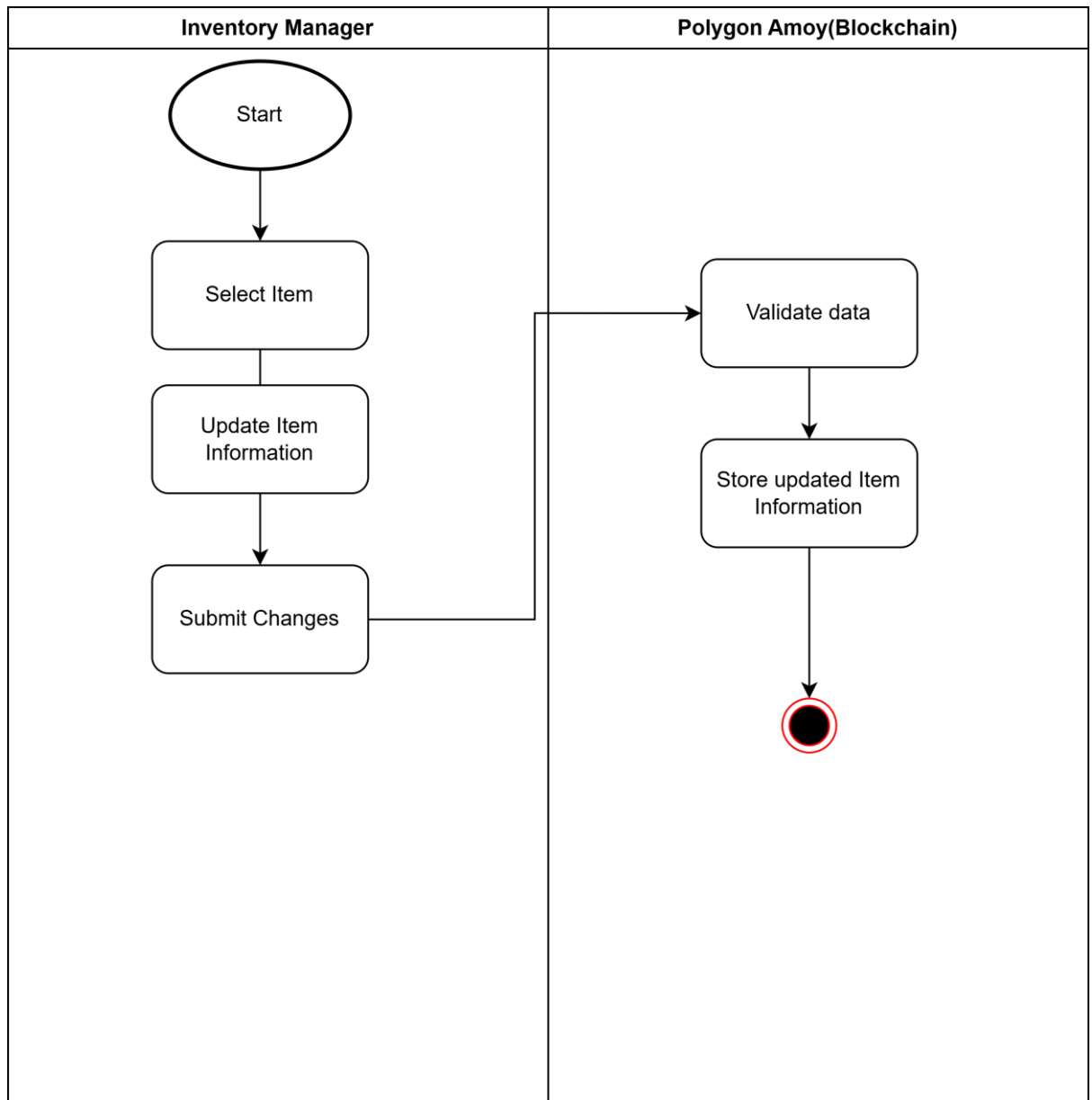


Figure 3.7 Activity Diagram (Update Item)

The activity diagram shows that updating items into the system. The diagram is divided into two swim lanes representing the **Inventory Manager** and the **Polygon Amoy (Blockchain)** components, showing the interaction between the user and the blockchain network.

Description of Activities:

- **Start:** The process begins when the Inventory Manager initiates the "update item" function.
- **Select Item:** The manager selects the item to update the item info.

- **Update Item Information:** The user updates relevant item information such as name, price, supplier, etc.
- **Submit:** Once all details are entered, the manager submits the changes.
- **Validate Data (Blockchain):** The submitted data is sent to the blockchain (Polygon Amoy testnet) where it is validated to ensure correctness and authenticity.
- **Store Updated Info (Blockchain):** If validation passes, the data is recorded immutably on the blockchain.
- **End:** The activity concludes once the supplier is successfully stored.

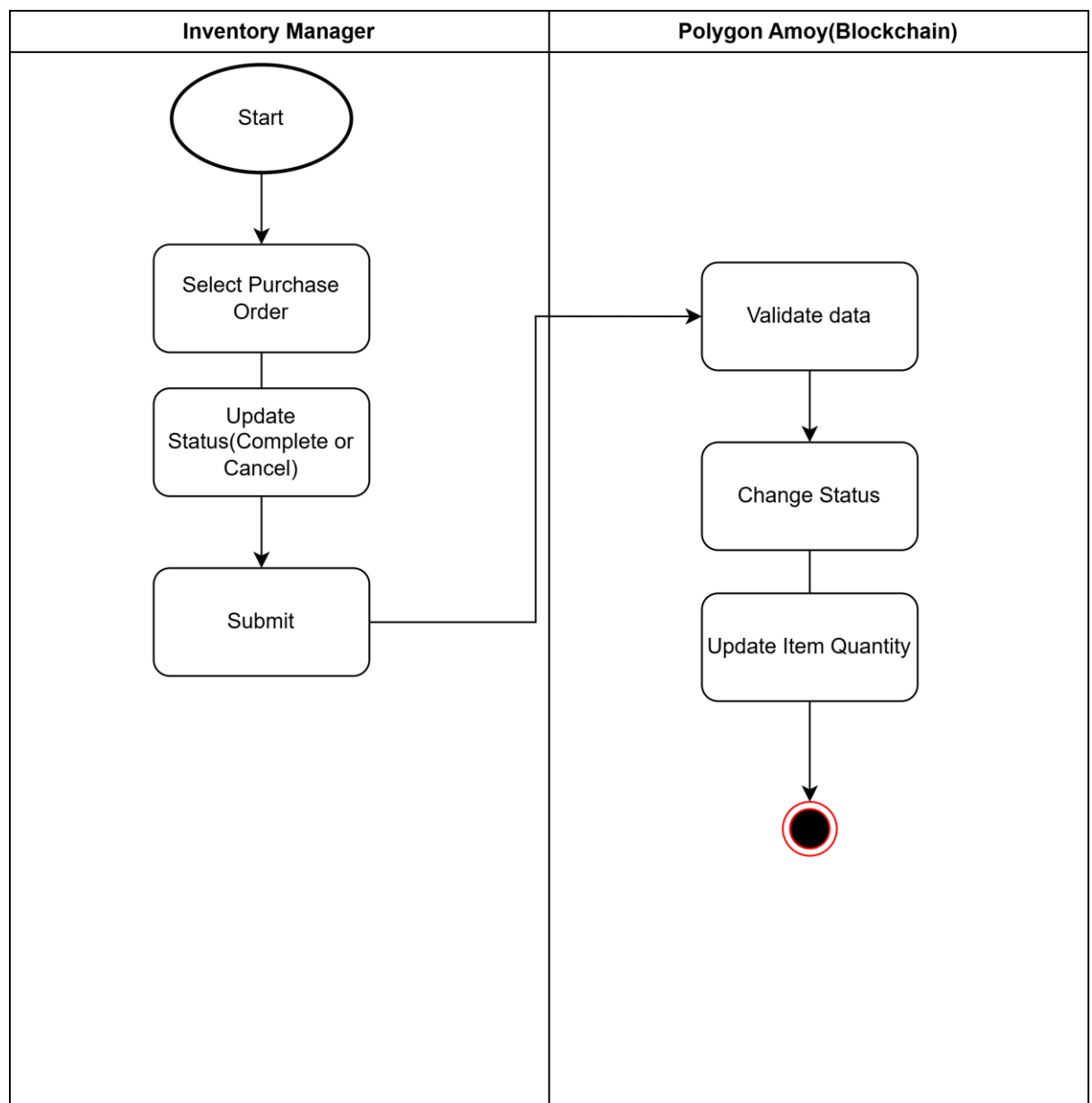


Figure 3.8 Activity Diagram (Update Purchases Status)

The activity diagram shows that updating purchases status into the system. The diagram is divided into two swim lanes representing the **Inventory Manager** and the **Polygon Amoy (Blockchain)** components, showing the interaction between the user and the blockchain network.

Description of Activities:

- **Start:** The process begins when the Inventory Manager initiates the "update purchases status" function.
- **Select Purchase Order:** The manager selects the purchases order to update the status.
- **Update Purchases Status:** The user update the status (Complete or Cancel)
- **Submit:** Once all details are entered, the manager submits the changes.
- **Validate Data (Blockchain):** The submitted data is sent to the blockchain (Polygon Amoy testnet) where it is validated to ensure correctness and authenticity.
- **Change Status (Blockchain):** If validation passes, the data is recorded immutably on the blockchain.
- **Update Item Quantity:** After the status change the item quantity will be updated.
- **End:** The activity concludes once the supplier is successfully stored.

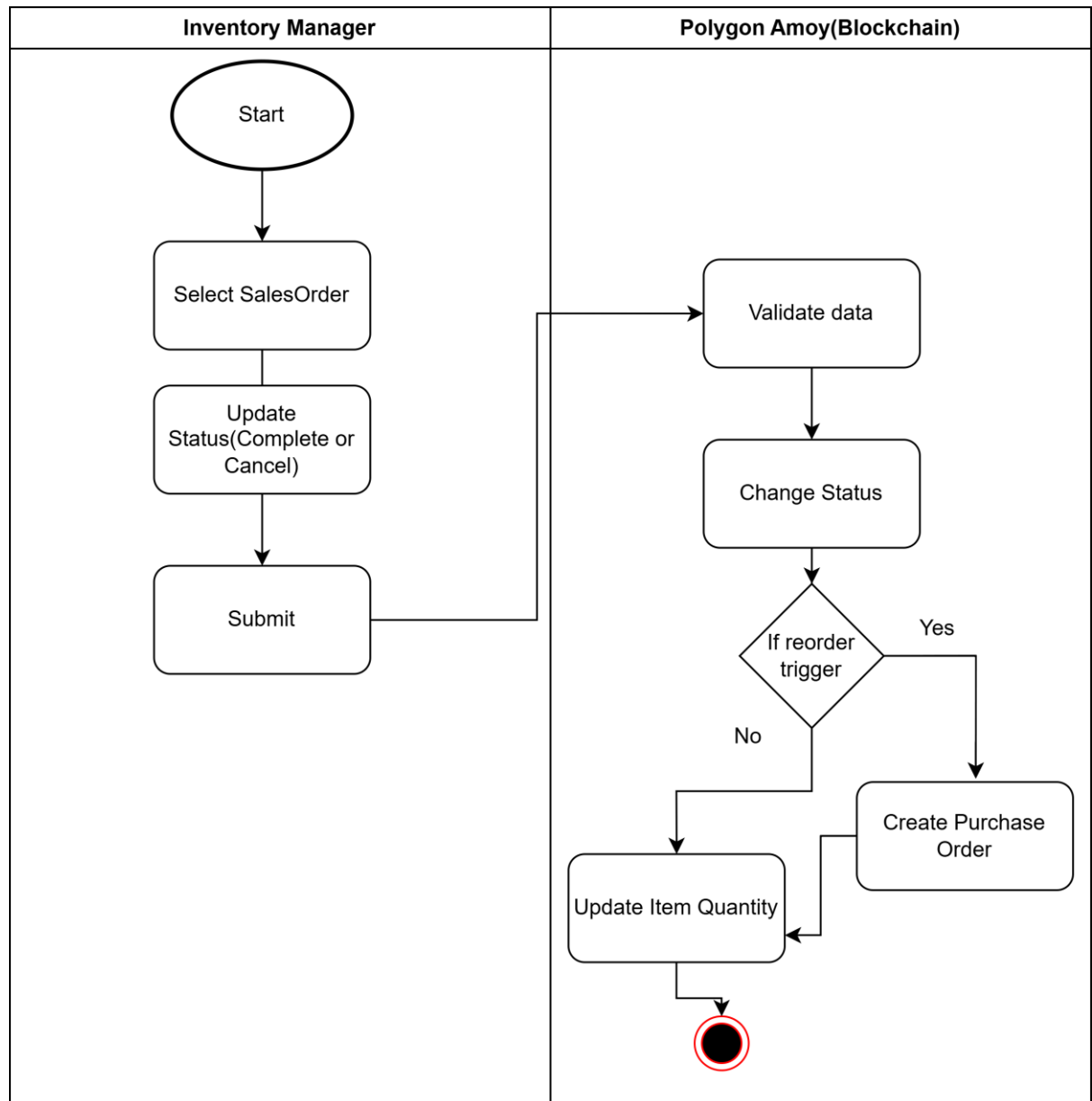


Figure 3.9 Activity Diagram (Update Sales Status)

The activity diagram shows that updating sales status into the system. The diagram is divided into two swim lanes representing the **Inventory Manager** and the **Polygon Amoy (Blockchain)** components, showing the interaction between the user and the blockchain network.

Description of Activities:

- **Start:** The process begins when the Inventory Manager initiates the "update purchases status" function.
- **Select Sales Order:** The manager selects the sales order to update the status.
- **Update Sales Status:** The user updates the status (Complete or Cancel)
- **Submit:** Once all details are entered, the manager submits the changes.

- **Validate Data (Blockchain):** The submitted data is sent to the blockchain (Polygon Amoy testnet) where it is validated to ensure correctness and authenticity.
- **Change Status (Blockchain):** If validation passes, the data is recorded immutably on the blockchain.
- **If reorder trigger:** If the stock lower then the reorder point that set then it will create purchases order, if not it will directly go through update item quantity.
- **Update Item Quantity:** After the status change the item quantity will be updated.
- **End:** The activity concludes once the supplier is successfully stored.

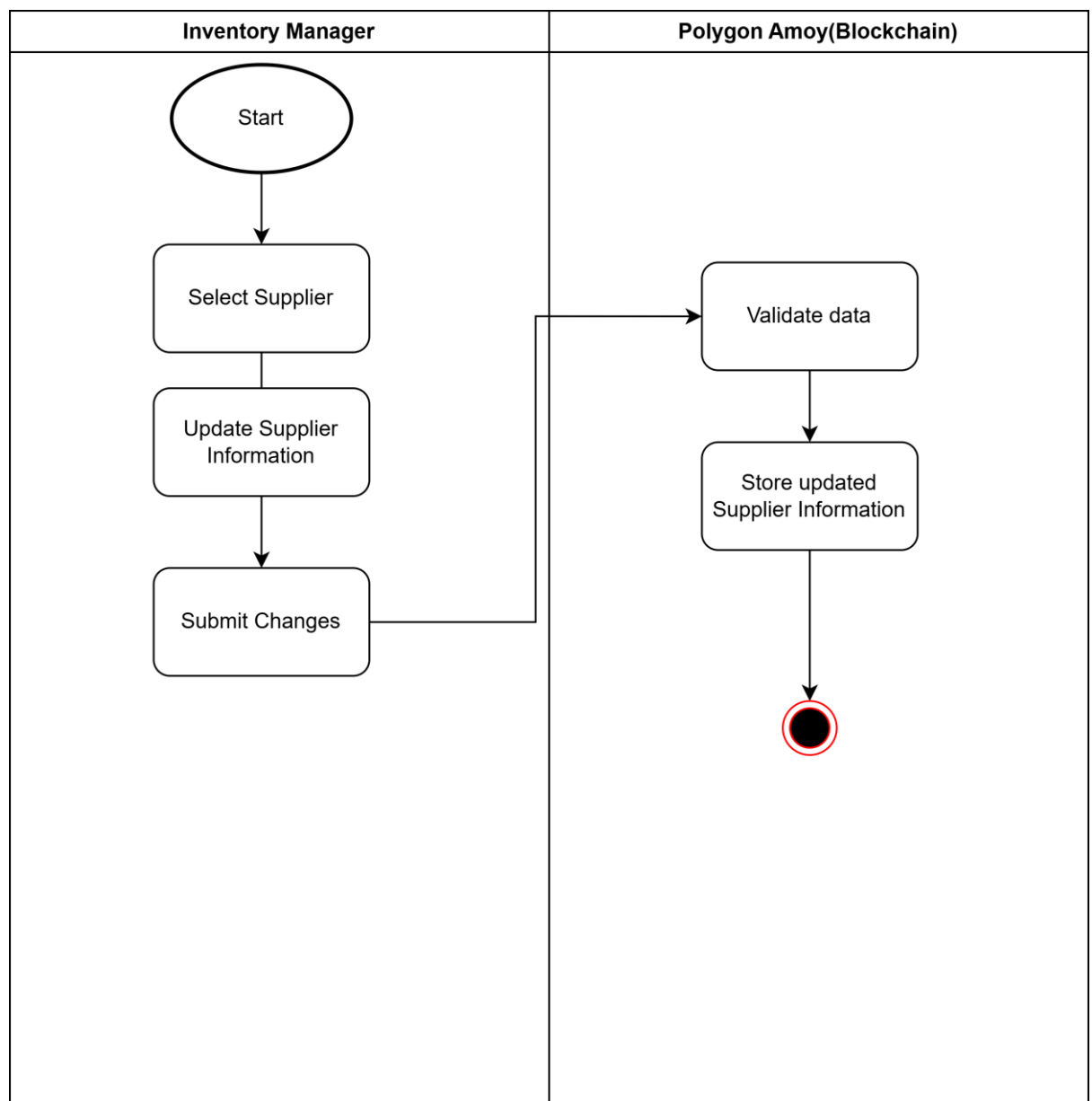


Figure 3.10 Activity Diagram (Update Supplier)

The activity diagram shows that updating suppliers into the system. The diagram is divided into two swim lanes representing the **Inventory Manager** and the **Polygon Amoy (Blockchain)** components, showing the interaction between the user and the blockchain network.

Description of Activities:

- **Start:** The process begins when the Inventory Manager initiates the "update item" function.
- **Select Supplier:** The manager selects the item to update the supplier info.
- **Update Supplier Information:** The user updates relevant supplier information such as name, contact, etc.
- **Submit:** Once all details are entered, the manager submits the changes.
- **Validate Data (Blockchain):** The submitted data is sent to the blockchain (Polygon Amoy testnet) where it is validated to ensure correctness and authenticity.
- **Store Updated Info (Blockchain):** If validation passes, the data is recorded immutably on the blockchain.
- **End:** The activity concludes once the supplier is successfully stored.

Chapter 4 System Design

4.1 System Block Diagram

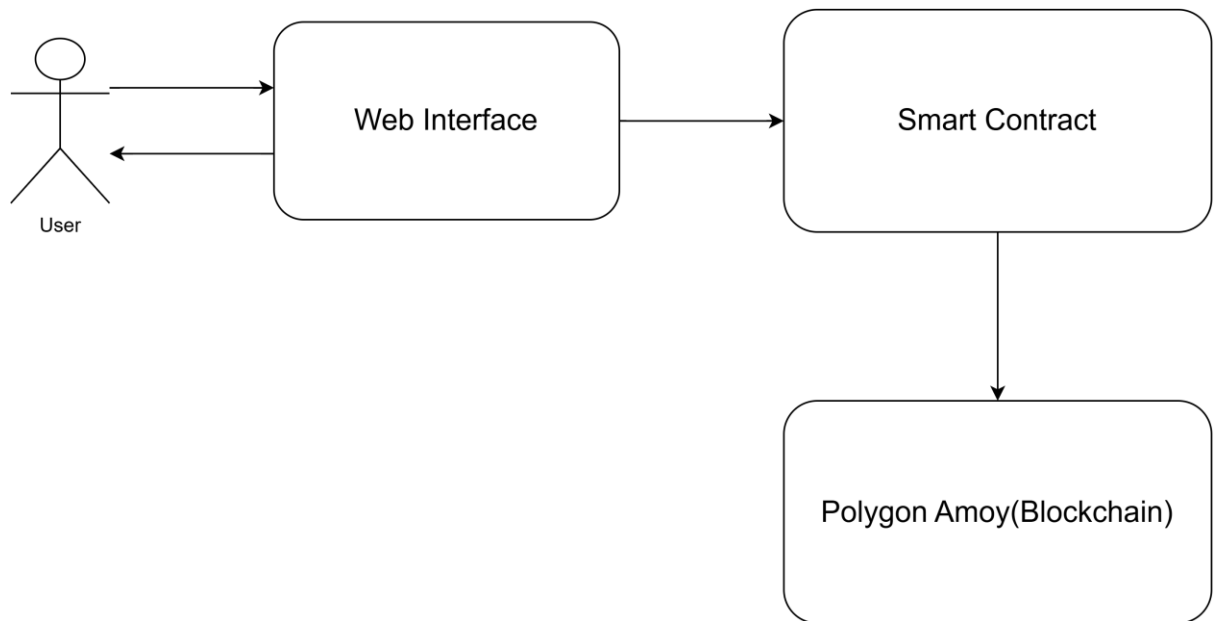


Figure 4.1 System Block Diagram

The system block diagram provides a top-down view of the major components involved in the development of the Inventory Management System using blockchain technology.

The system with three primary blocks:

- **Web Interface:** The frontend developed using React and TypeScript. It also used some React libraries to build up the UI. It allows store managers to manage inventory, create orders, and interact with the blockchain.
- **Smart Contracts Layer:** The smart contracts written in Solidity, with all the core business logic, including inventory tracking, supplier registration and sales/purchase order management.
- **Polygon Amoy Blockchain Network:** A decentralized testnet where all contract-related transactions are deployed, verified, and stored immutably.

These components work together to ensure that all actions performed via the frontend are processed through smart contracts and recorded securely on the blockchain network.

4.2 System Components Specifications

Table 4.1 System Components Specifications

Component	Technology Used	Purpose
Web Interface	ReactJS, TypeScript, Ethers.js	Provides the graphical interface for users to interact with the system
Smart Contracts	Solidity	Manages inventory data and order processing on-chain
Deployment Tool	Hardhat	Compiles and deploys smart contracts to the testnet
Blockchain Network	Polygon Amoy Testnet	Executes and records smart contract transactions
Wallet Integration	MetaMask	Enables secure transaction signing and authentication
Version Control	Git	Tracks source code and development progress

4.3 Components Design

As this is a software-only solution, this section discusses the design of logical components and contract architecture.

- **Smart Contract Design**

- Contract: InventoryManager.sol, Reordering.sol, OrderManagement.sol
- Key Functions:
 - addSupplier
 - createSalesOrder
 - createPurchaseOrder
 - setStatus(orderId, status)
 - checkAndReorder()
 - addItem
 - updateItem
 - updateSupplier

- **Frontend Contract Interaction**

- Uses WAGMI and Ethers.js hooks to read and write data

- MetaMask prompts users to approve transactions
- **Security Measures**
 - Input validations
 - require statements for data integrity

4.4 System Components Interaction Operations

This section outlines the operational flow of how different system components interact:

1. User Interaction

- The user accesses the Web Interface and connects their wallet.

2. Performing Actions

- Users initiate actions such as adding a supplier, creating orders, or updating inventory.

3. Smart Contract Invocation

- Frontend sends transactions via MetaMask to the appropriate smart contract function.

4. Blockchain Confirmation

- Transactions are confirmed on the Polygon Amoy Testnet.

5. UI Update

- The system fetches the updated data from the blockchain and reflects it on the Web Interface.

6. Automatic Logic Execution

- Functions like `checkAndReorder()` are called when stock levels drop below threshold, automatically initiating a reorder.

This interaction cycle ensures data integrity, transparency, and traceability in inventory management activities.

4.5 Flowchart

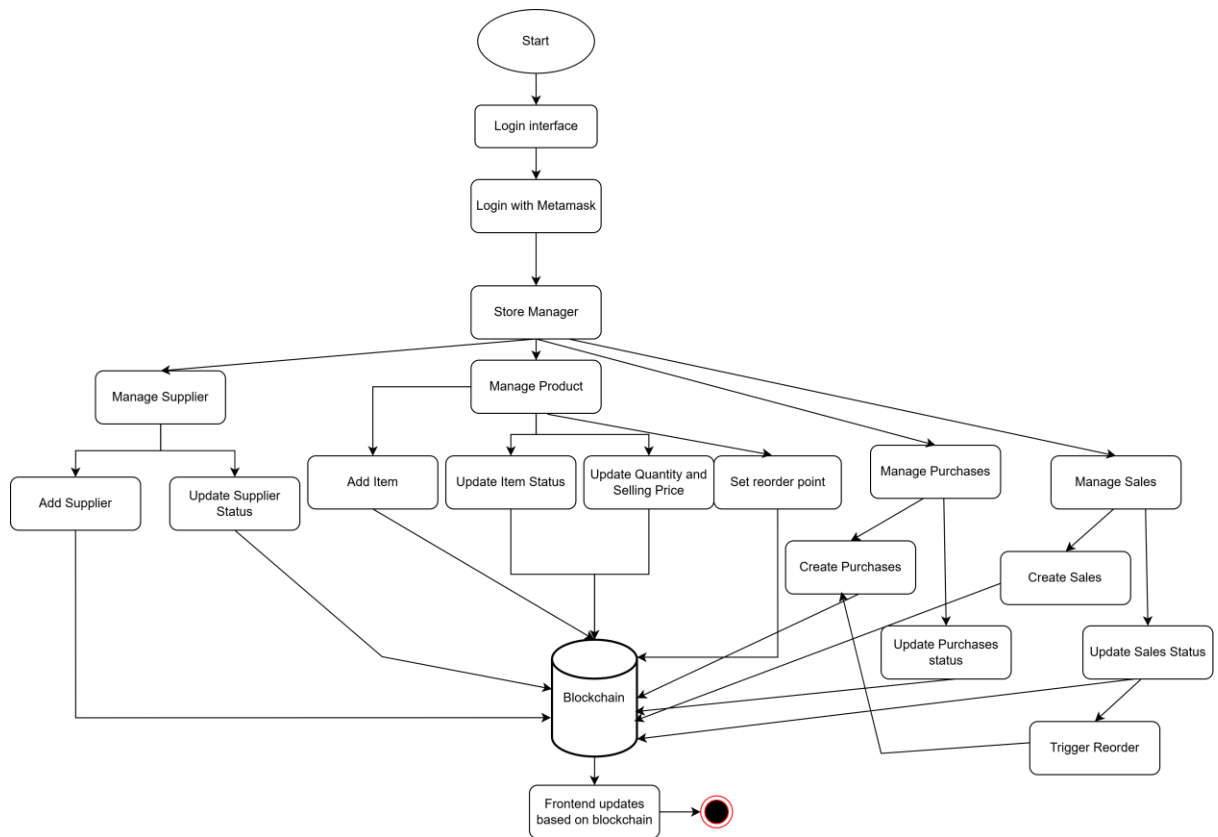


Figure 4.2 Flowchart

The flowchart illustrates the overall process flow of the blockchain-based Inventory Management System, focusing on the actions performed by the Store Manager through a user interface connected to the blockchain backend.

1. Start & Login

The process begins with a login interface, where the store manager logs in using **MetaMask**, a blockchain wallet that ensures secure authentication and enables transaction signing.

2. Store Manager Role

- **Manage Supplier:** Add new suppliers or update existing supplier information. These actions interact with the blockchain to store or modify supplier records.
- **Manage Product:**
 - Adding new items.
 - Updating item status (e.g., availability).

- Updating quantity and selling price.
- Setting reorder points to automate restocking.

All product-related data is stored or updated on the blockchain.

- **Manage Purchases:** Create purchase orders and update their status (e.g., completed or pending), recorded immutably on the blockchain.
- **Manage Sales:** Create sales orders and update their status, if after sales the stock is hit the threshold then it will auto create a purchase order the blockchain.

3. **Blockchain**

Integration

All key operations (add/update supplier, add/update products, create purchases/sales, set reorder points) interact directly with the blockchain, ensuring transparency, immutability, and decentralized storage of data.

4. **Frontend**

Updates

The frontend is dynamically updated based on the blockchain data, it allows real-time visibility of current inventory, supplier and transaction.

So that the system ensures secure and tamper-proof management of inventory activities, enhances transparency and reduces the risk of data manipulation or loss.

Chapter 5 System Implementation

5.1 Hardware Setup

The hardware involved in this project is a laptop with following specifications to create and support a Web 3 development environment and run the implemented codes.

Table 5.1 Specifications of laptop

Description	Specifications
Model	Acer Enduro Urban
Processor	Intel Core i5-1135G7
Operating System	Windows 11
Memory	8GB DDR4 RAM
Storage	512GB SSD

Internet Access: It required to interact with the blockchain network (e.g., Polygon Amoy Testnet)

5.2 Software Setup

The main software and frameworks used:

- IDE: Visual Studio Code, Remix IDE
- Smart Contracts: Solidity (Hardhat framework)
- Frontend: React with TypeScript
- Blockchain Network: Polygon Amoy (Testnet)
- Wallet: MetaMask
- Node.js & npm: v18+
- Libraries:
 - ethers.js, wagmi (for Web3 interactions)
 - tailwindcss (styling)
 - react-toastify (to show toast notifications (React Toast))
 - react-loader-spinner (customize loading indicators)

5.3 Setting and Configuration

Blockchain: Connect wallet to Polygon Amoy network using RPC and Chain ID

Hardhat: Set up hardhat.config.ts with contract details and deploy script

Frontend: Configure wagmi client with correct provider and connectors

Environment File (.env):

AMOY_RPC_URL="https://polygon-

amoy.g.alchemy.com/v2/mO8dMTLzK2bRxvIqBiJqxHVWIgkD0hG-"

PRIVATE_KEY=b76e7e060b27e0af05fa8afb4cdca0e7d80caf090e2d1bb5f69804580
8b876a5

Oklink_API_key="https://oklink.com/account/04bafbcb-d107-4e30-9e93-
af91b7a16559"

Package.json - Declares project dependencies:

```
{
  "name": "webapp",
  "version": "0.1.0",
  "private": true,
  "dependencies": {
    "@tanstack/react-query": "^5.52.2",
    "@testing-library/jest-dom": "^5.17.0",
    "@testing-library/react": "^13.4.0",
    "@testing-library/user-event": "^13.5.0",
    "@types/jest": "^27.5.2",
    "@types/node": "^16.18.106",
    "bootstrap": "^5.3.3",
    "chart.js": "^4.4.4",
    "ethers": "^6.13.2",
    "react": "^17.0.2",
    "react-chartjs-2": "^5.2.0",
    "react-dom": "^17.0.2",
    "react-scripts": "5.0.1",
    "react-select": "^5.2.1",
    "wagmi": "^2.12.7",
    "web-vitals": "^2.1.4"
  },
}
```

CHAPTER 5

```
"resolutions": {
  "react": "^17.0.2",
  "react-dom": "^17.0.2"
},
"scripts": {
  "start": "react-scripts start",
  "build": "react-scripts build",
  "test": "react-scripts test",
  "eject": "react-scripts eject"
},
"eslintConfig": {
  "extends": [
    "react-app",
    "react-app/jest"
  ]
},
"browserslist": {
  "production": [
    ">0.2%",
    "not dead",
    "not op_mini all"
  ],
  "development": [
    "last 1 chrome version",
    "last 1 firefox version",
    "last 1 safari version"
  ]
},
"devDependencies": {
  "@types/react": "^18.3.4",
  "@types/react-dom": "^18.3.0",
  "typescript": "^5.5.4"
}
```

}

5.4 System Operation

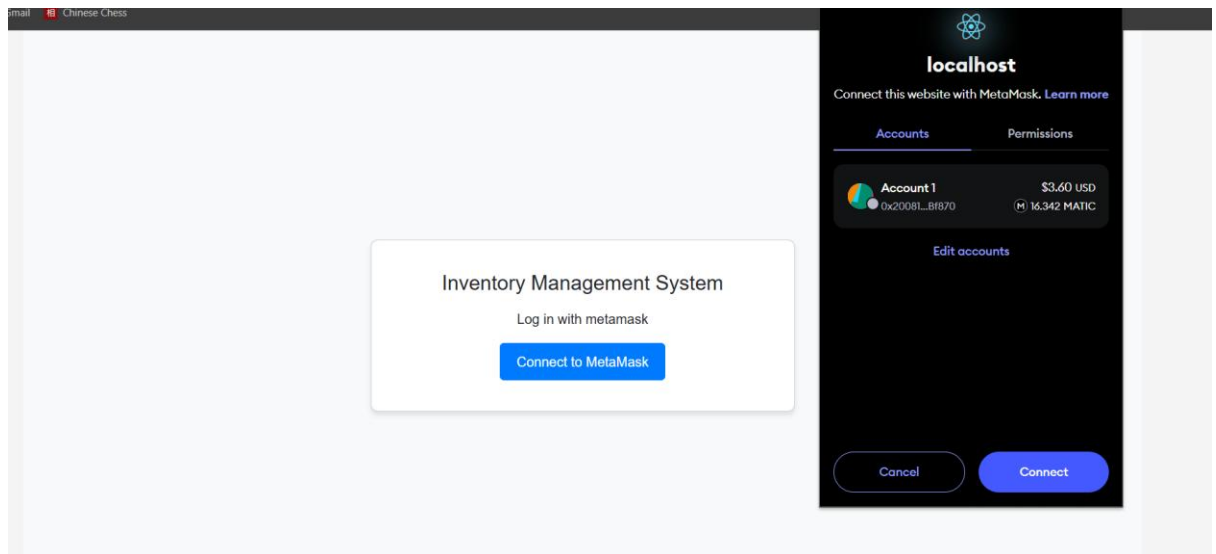


Figure 5.1 Login Page

User (Inventory Manager) logs into the system using MetaMask Wallet. To browse the web application, the user is prompted to connect their wallet, which serves as the authentication mechanism. Once connected, the system verifies the wallet address and grants access to the appropriate inventory management features.

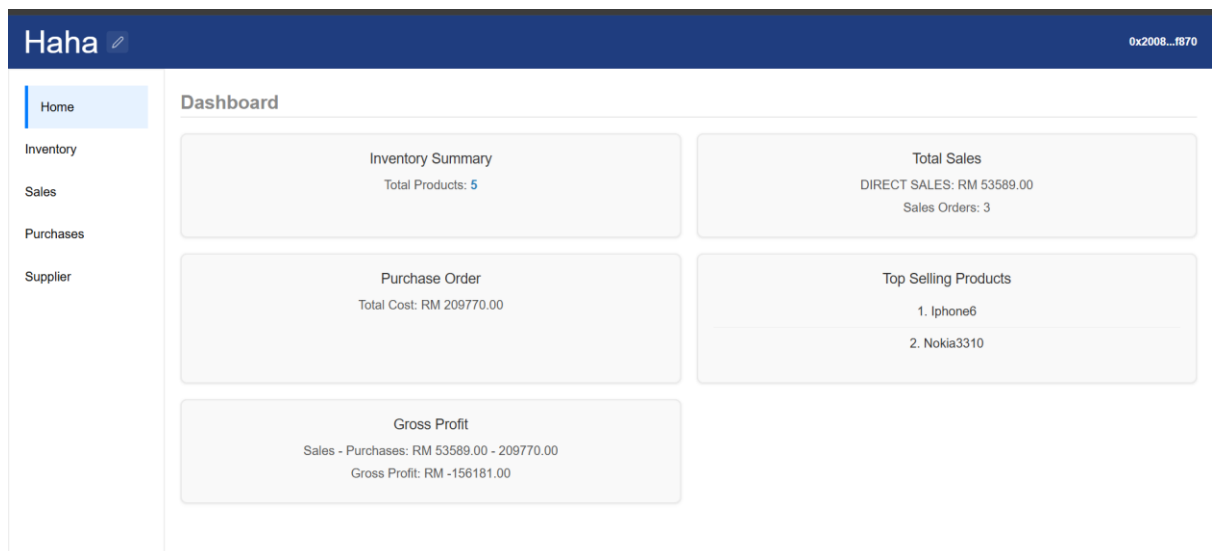


Figure 5.2 Home page

After the user logs in to the system using their MetaMask wallet, they are directed to the **Home Page**. The Home Page serves as the dashboard and provides a quick overview of the current inventory and business performance.

The dashboard includes:

- **Total Products** – The number of unique product types available in inventory.
- **Total Sales** – The total number of sales orders created or completed.
- **Total Purchases** – The total number of purchase orders recorded.
- **Top 3 Selling Products** – A summary of the best-performing products based on quantity sold.
- **Gross Profit** – An estimate of the total profit based on sales and purchase prices.

In addition:

- The **company name** is displayed in the top-left corner of the page.
- The **connected wallet address** is shown in the top-right corner for easy user identification.
- A **navigation bar** is clearly visible, allowing users to access different pages such as Inventory, Sales, Purchases and Supplier.

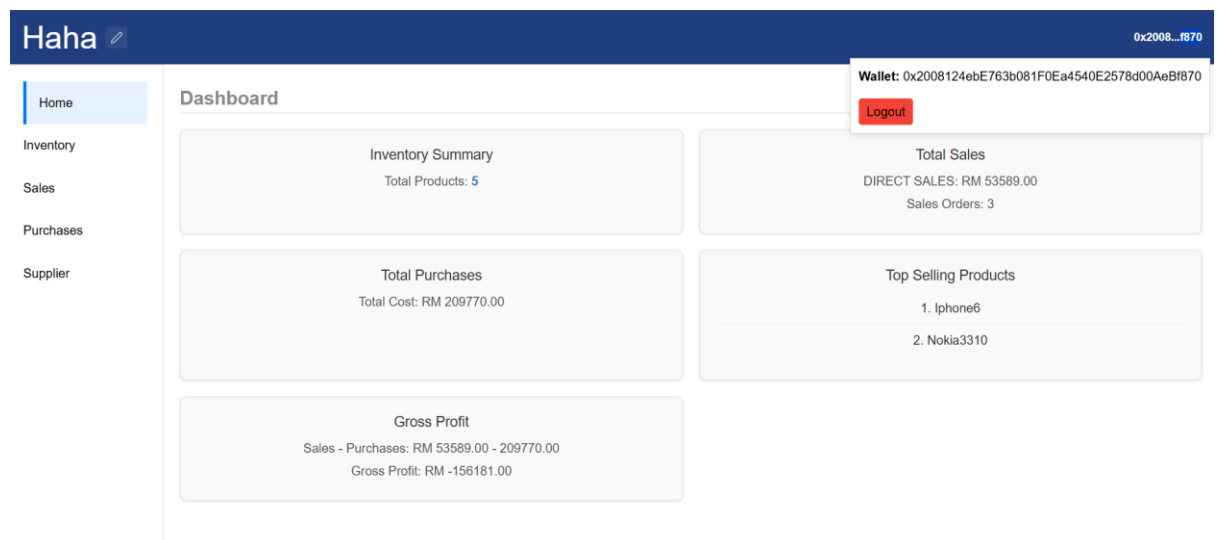


Figure 5.3 Logout

When the user clicks the top right corner profile it will show out the full wallet address and a logout button that allows the user to logout the system.

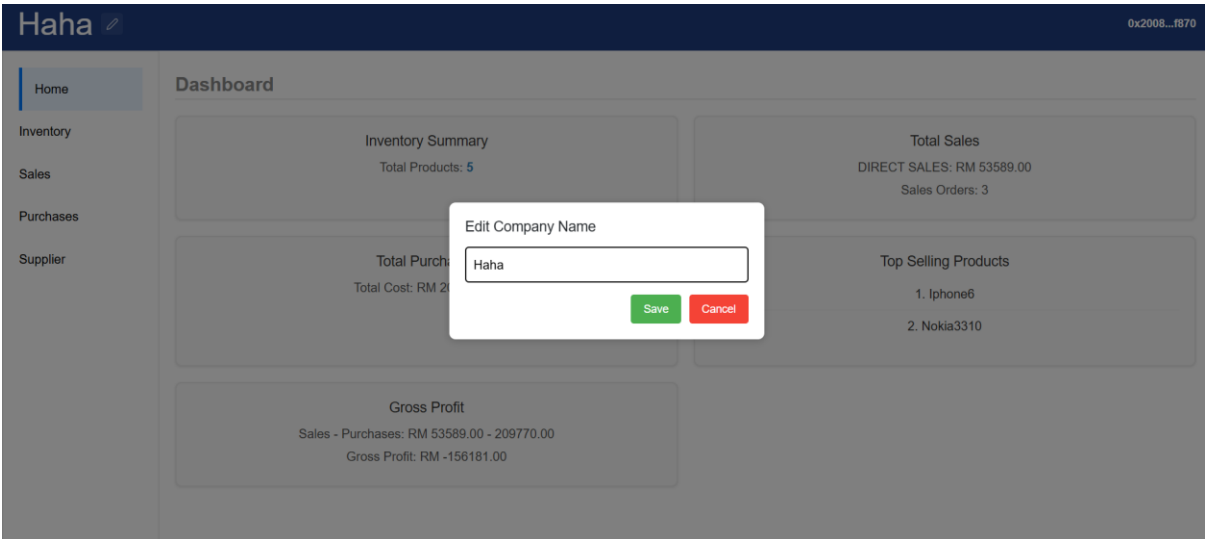


Figure 5.4 Edit Company Name

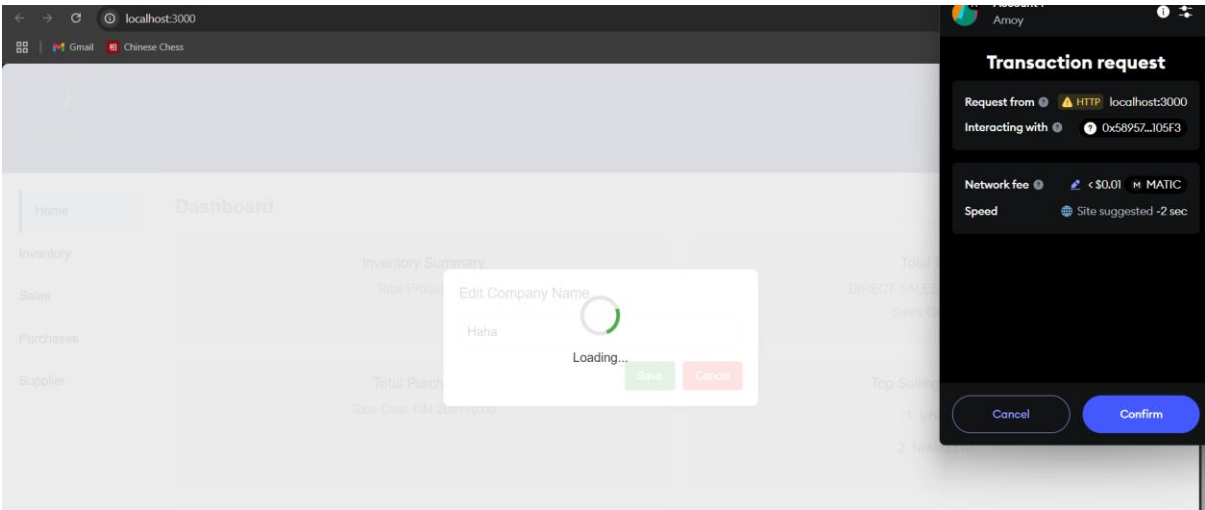


Figure 5.5 Transaction Confirmation

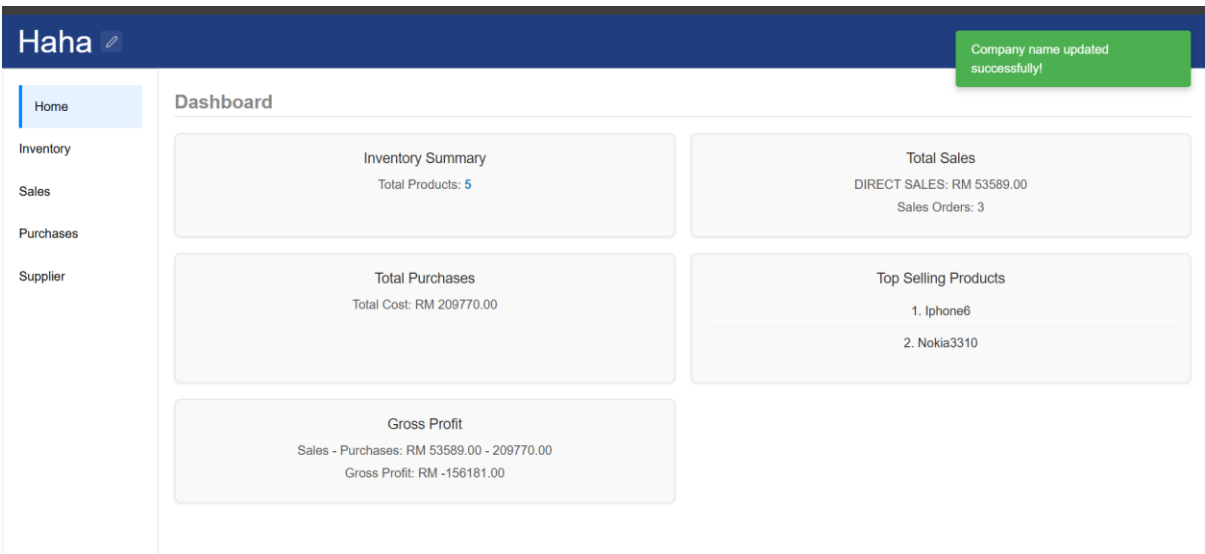


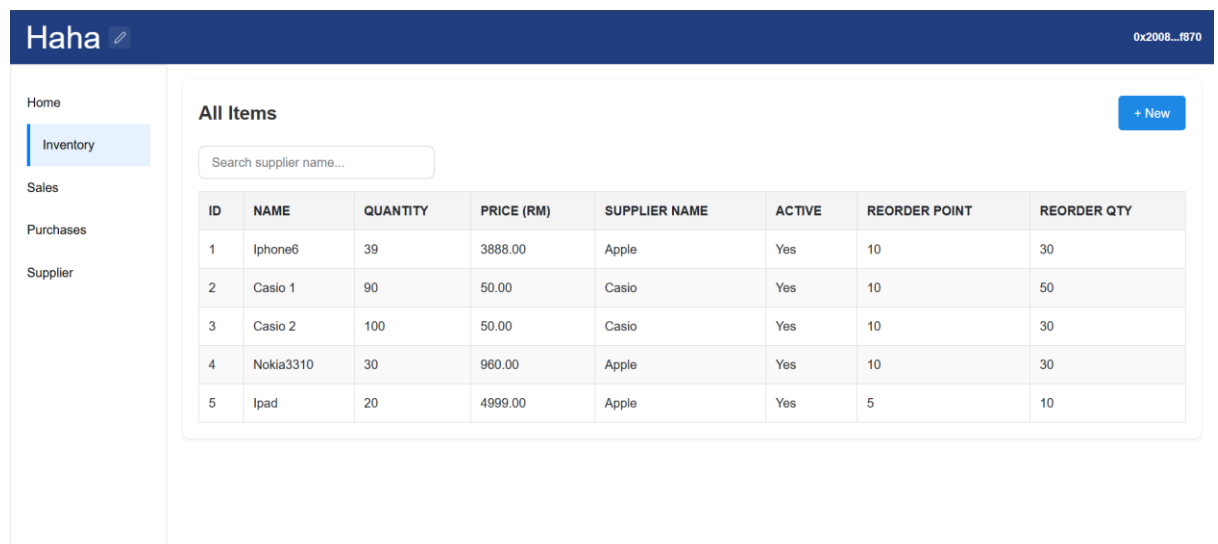
Figure 5.6 Company name updated notification

CHAPTER 5

Users can click the edit button beside the company name to edit the company name. After updating the company name and submitting it, the system initiates a transaction on the blockchain. This transaction records company name securely and immutably through a smart contract.

Then, the MetaMask wallet automatically prompts the user with a pop-up window to review and confirm the transaction, including the estimated gas fee required to process it on the blockchain. The user must approve the transaction in MetaMask for it to be successfully submitted and recorded on the blockchain network.

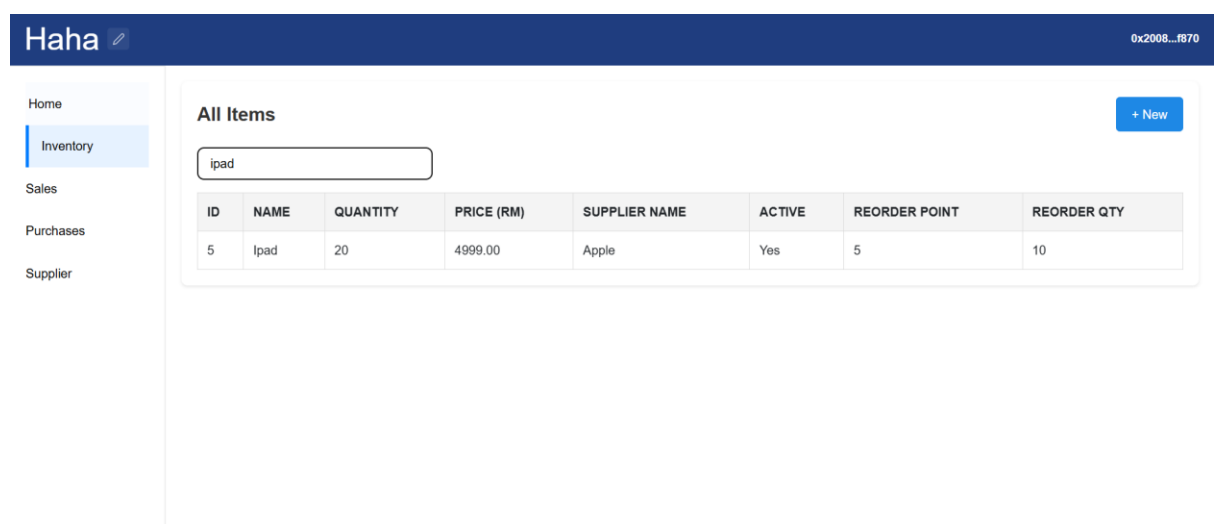
Once confirmed, the system waits for the blockchain to process the transaction. Upon successful completion, a toast message is displayed to the user, indicating that the company name has been created.



The screenshot shows the 'Inventory' page of a web application. The header is dark blue with the text 'Haha' and a small icon. On the right of the header is a user identifier '0x2008...f870'. A sidebar on the left contains links: Home, Inventory (highlighted), Sales, Purchases, and Supplier. The main content area is titled 'All Items' and features a search bar with the placeholder text 'Search supplier name...'. Below the search bar is a table with 8 columns: ID, NAME, QUANTITY, PRICE (RM), SUPPLIER NAME, ACTIVE, REORDER POINT, and REORDER QTY. The table contains 5 rows of data.

ID	NAME	QUANTITY	PRICE (RM)	SUPPLIER NAME	ACTIVE	REORDER POINT	REORDER QTY
1	Iphone6	39	3888.00	Apple	Yes	10	30
2	Casio 1	90	50.00	Casio	Yes	10	50
3	Casio 2	100	50.00	Casio	Yes	10	30
4	Nokia3310	30	960.00	Apple	Yes	10	30
5	Ipad	20	4999.00	Apple	Yes	5	10

Figure 5.7 Inventory Page



The screenshot shows the 'Search Item' page of the same web application. The header and sidebar are identical to Figure 5.7. The main content area is titled 'All Items' and features a search bar with the text 'ipad' entered. Below the search bar, the table from Figure 5.7 is displayed, but it is filtered to show only the item with 'Ipad' in the NAME column (ID 5).

ID	NAME	QUANTITY	PRICE (RM)	SUPPLIER NAME	ACTIVE	REORDER POINT	REORDER QTY
5	Ipad	20	4999.00	Apple	Yes	5	10

Figure 5.8 Search Item

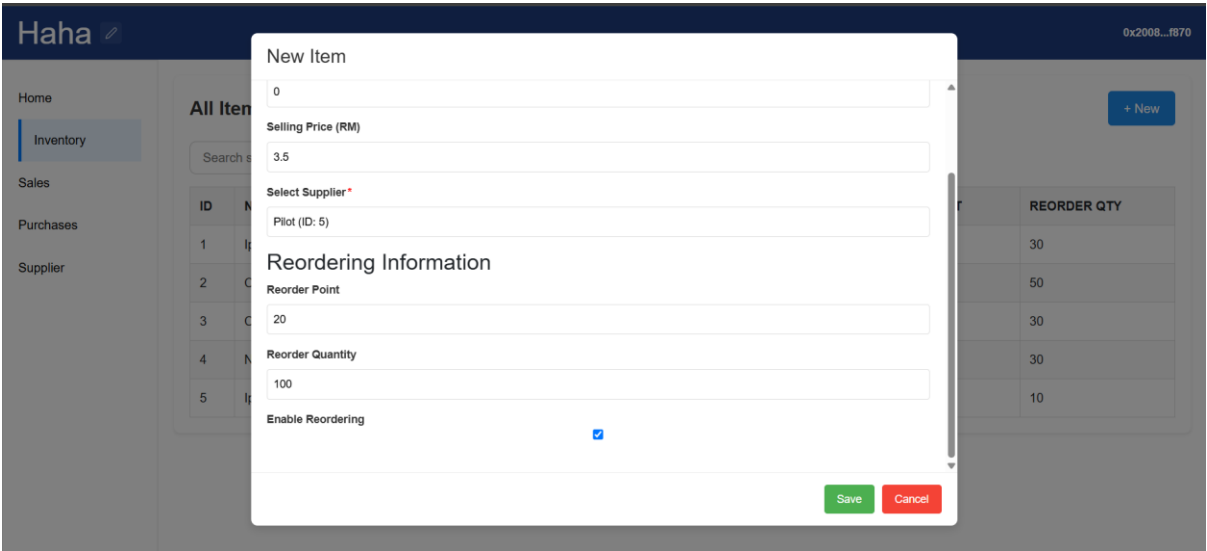


Figure 5.9 Add Item

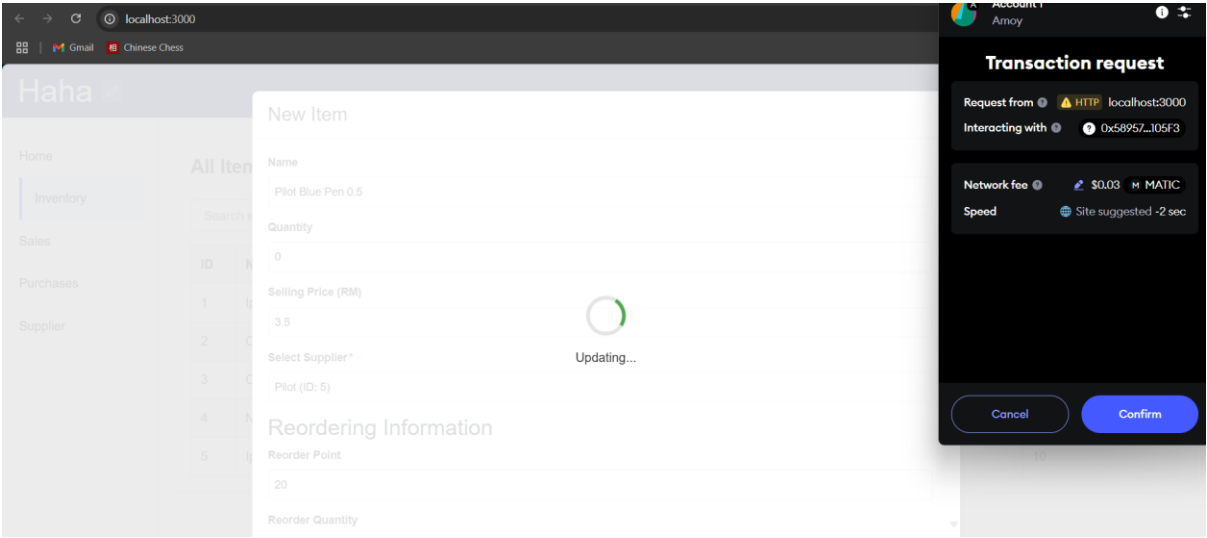


Figure 5.10 Add Item confirmation

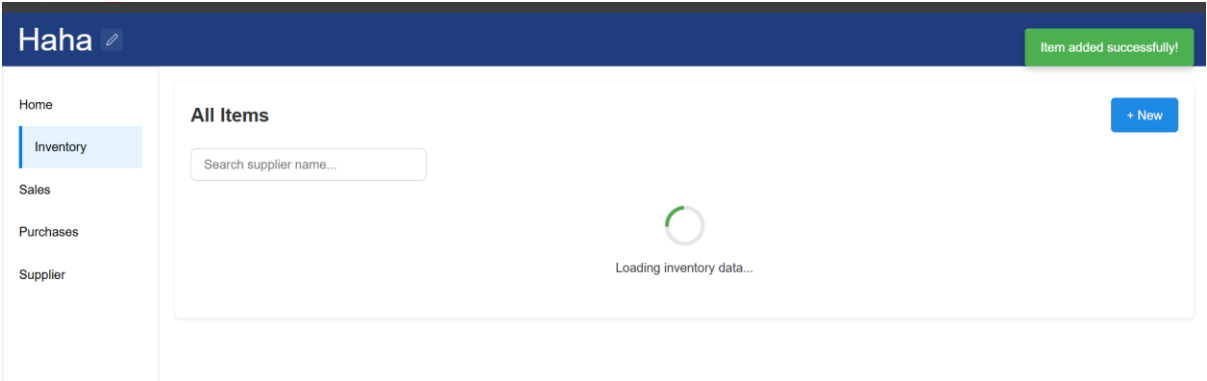


Figure 5.11 Add Item Successful

5	Ipad	20	4999.00	Apple	Yes	5	10
6	Pilot Blue Pen 0.5	0	3.50	Pilot	Yes	20	100

Figure 5.12 Show Added Item

Next, the inventory page shows out all the items that are stored in the blockchain. Then, the user can use the search bar to find the item. Then, the user can click the new button to add a new item, after clicking the button the user can fill in the detail for the item. After filling in the detail and submitting it, the system initiates a transaction on the blockchain. This transaction records company name securely and immutably through a smart contract.

Then, the MetaMask wallet automatically prompts the user with a pop-up window to review and confirm the transaction, including the estimated gas fee required to process it on the blockchain. The user must approve the transaction in MetaMask for it to be successfully submitted and recorded on the blockchain network.

Once confirmed, the system waits for the blockchain to process the transaction. Upon successful completion, a toast message is displayed to the user and the user can see that the added item had shown in the table.

Figure 5.13 Edit Item

All items + New

Search supplier name...

ID	NAME	QUANTITY	SELLING PRICE (RM)	SUPPLIER NAME	ACTIVE	REORDER POINT	REORDER QTY
1	Iphone6	39	3888.00	Apple	No	10	30
2	Casio 1	90	50.00	Casio	Yes	10	50
3	Casio 2	100	50.00	Casio	Yes	10	30
4	Nokia3310	30	960.00	Apple	Yes	10	30
5	Ipad	20	4999.00	Apple	Yes	5	10
6	Pilot Blue Pen 0.5	0	3.50	Pilot	Yes	20	100

Figure 5.14 Deactivate Item

Users can click on the item to edit the item information, and it also can deactivate the item if accidentally adding the wrong item. After deactivating the item, we can see that it will show deactivated information in the item list.

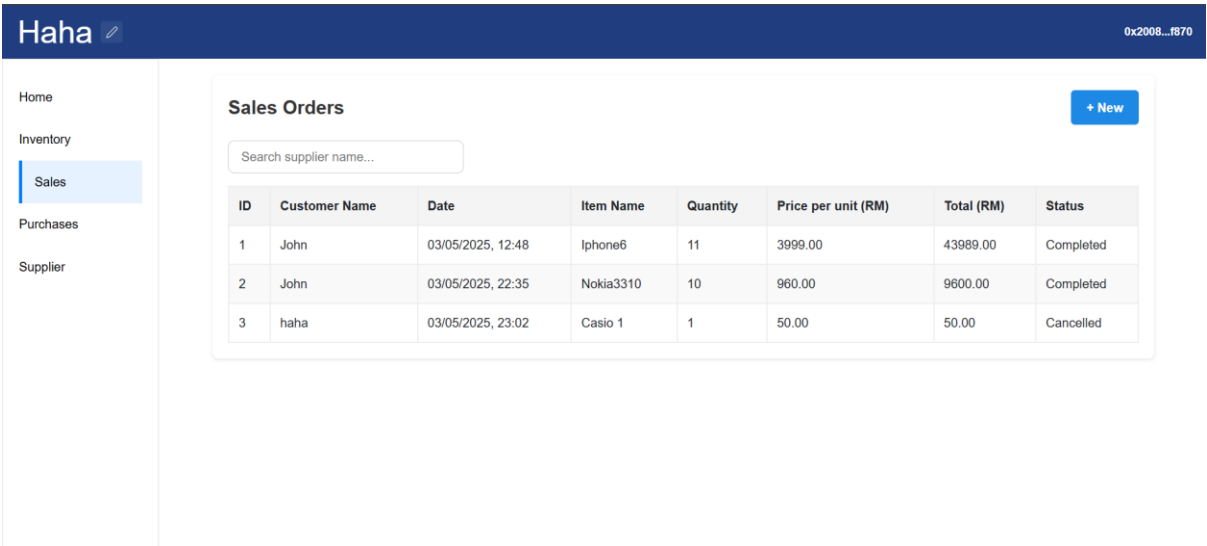


Figure 5.15 Sales Orders Page

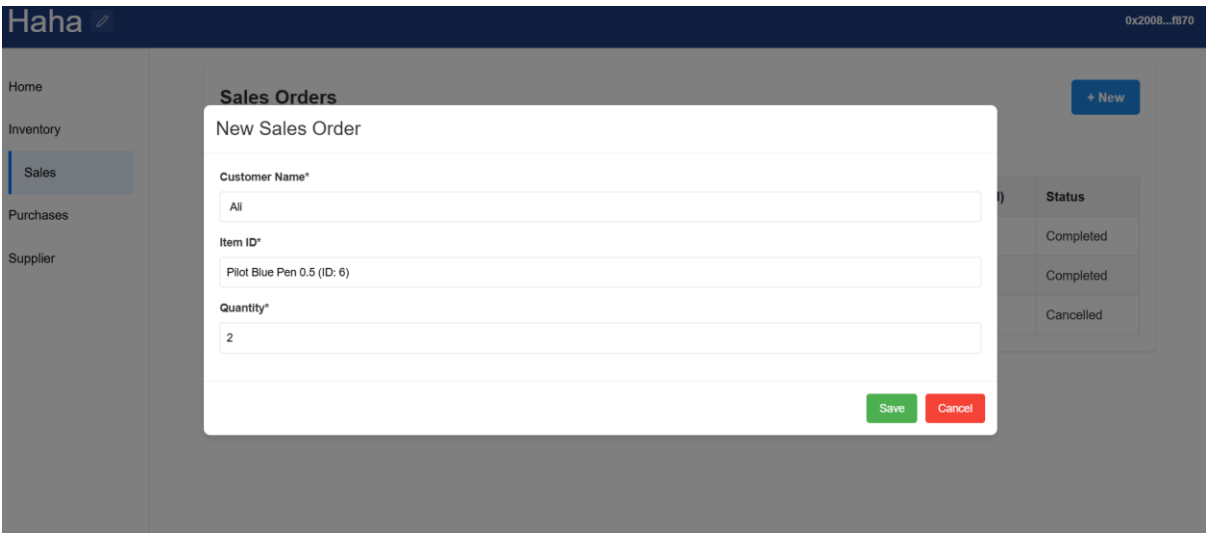


Figure 5.16 Create Sales Order

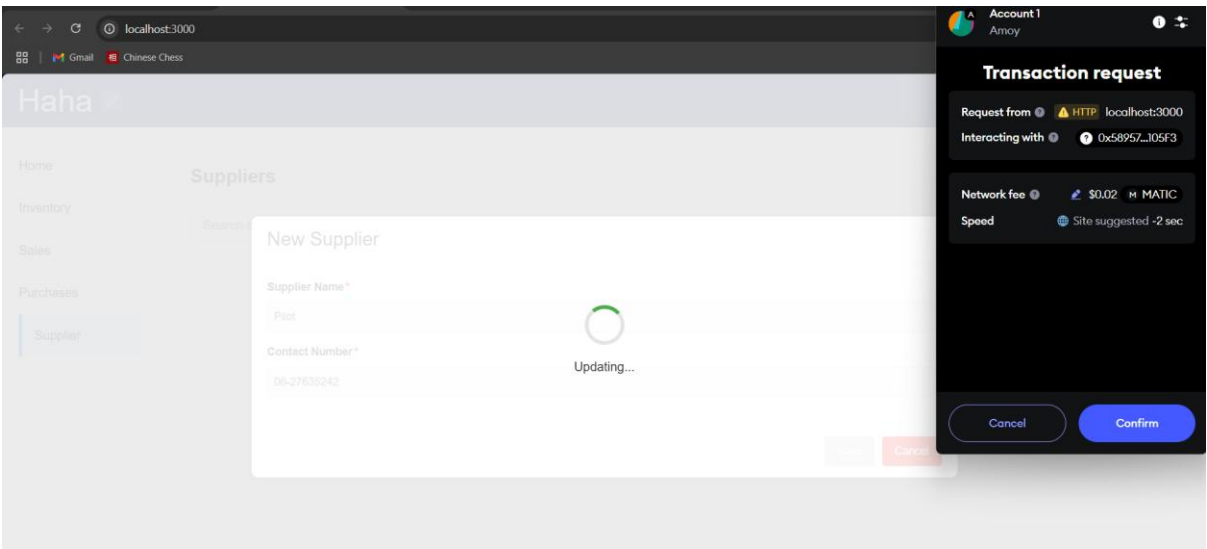


Figure 5.17 Confirmation Sales Order

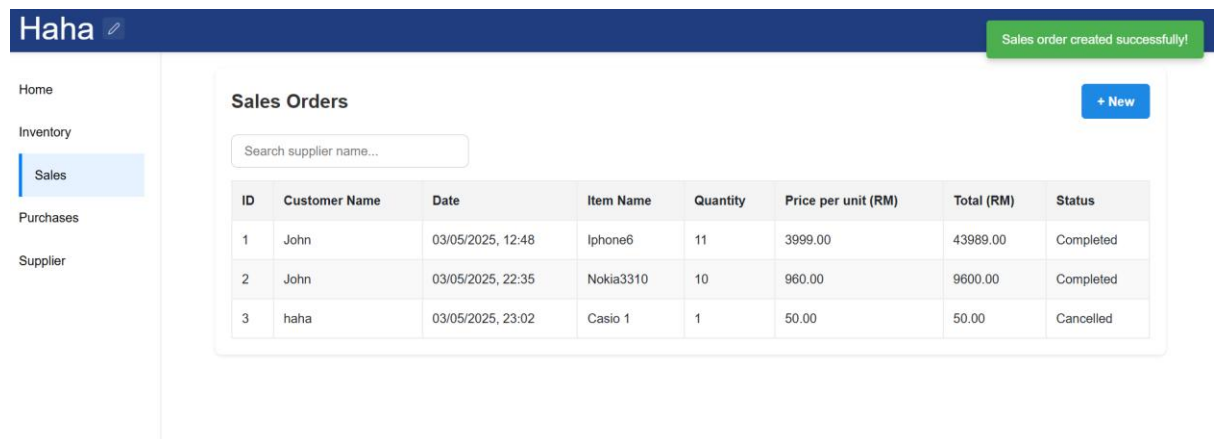


Figure 5.18 Notification Sales Order Created

2	John	03/05/2025, 22:35	Nokia3310	10	960.00	9600.00	Completed
3	haha	03/05/2025, 23:02	Casio 1	1	50.00	50.00	Cancelled
4	Ali	06/05/2025, 17:31	Pilot Blue Pen 0.5	2	3.50	7.00	Pending

Figure 5.19 Sales Order Created

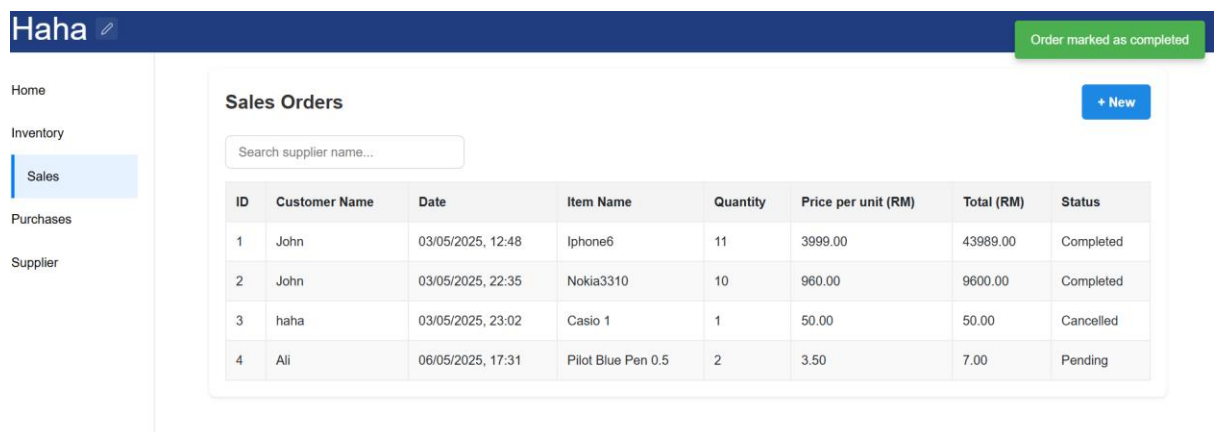


Figure 5.20 Sales Order Completed

In the sales orders page, it can show all sales orders that were created. Then, the user can use the search bar to find the order. Then, the user can click the new button to add a new sales order, after clicking the button the user can fill in the detail for the order. After filling in the details and submitting it, the system initiates a transaction on the blockchain. This transaction records company name securely and immutably through a smart contract.

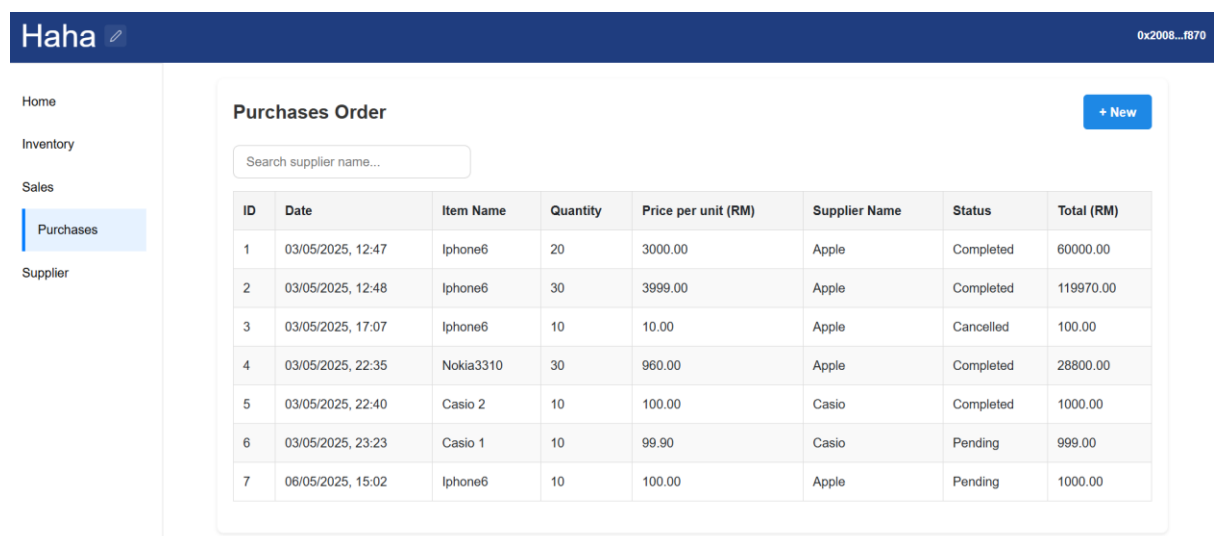
Then, the MetaMask wallet automatically prompts the user with a pop-up window to review and confirm the transaction, including the estimated gas fee required to process

CHAPTER 5

it on the blockchain. The user must approve the transaction in MetaMask for it to be successfully submitted and recorded on the blockchain network.

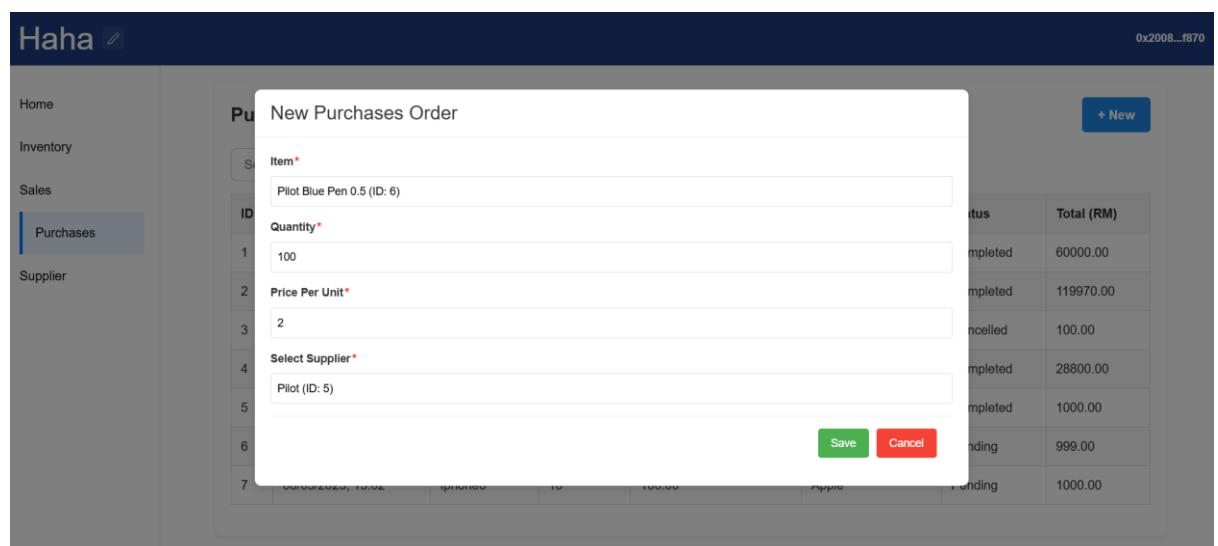
Once confirmed, the system waits for the blockchain to process the transaction. Upon successful completion, a toast message is displayed to the user, and user can see that the added order had shown in the table.

Then, user can click on the item to set the status of the order either complete or cancel. After that, it will show the status on the table and the quantity of the item will update. If the stock meet the threshold, it will auto create a purchase order.



ID	Date	Item Name	Quantity	Price per unit (RM)	Supplier Name	Status	Total (RM)
1	03/05/2025, 12:47	Iphone6	20	3000.00	Apple	Completed	60000.00
2	03/05/2025, 12:48	Iphone6	30	3999.00	Apple	Completed	119970.00
3	03/05/2025, 17:07	Iphone6	10	10.00	Apple	Cancelled	100.00
4	03/05/2025, 22:35	Nokia3310	30	960.00	Apple	Completed	28800.00
5	03/05/2025, 22:40	Casio 2	10	100.00	Casio	Completed	1000.00
6	03/05/2025, 23:23	Casio 1	10	99.90	Casio	Pending	999.00
7	06/05/2025, 15:02	Iphone6	10	100.00	Apple	Pending	1000.00

Figure 5.21 Purchases Order



New Purchases Order

Item *
Pilot Blue Pen 0.5 (ID: 6)

Quantity *
100

Price Per Unit *
2

Select Supplier *
Pilot (ID: 5)

Save Cancel

Figure 5.22 Create Purchase Order

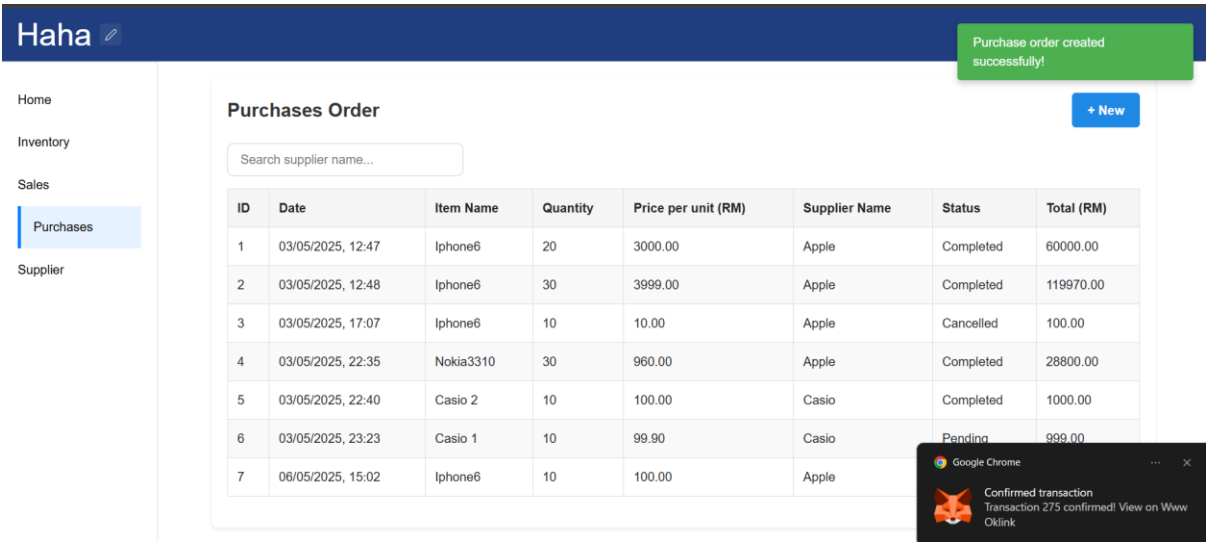


Figure 5.23 Notification Purchase Order Created

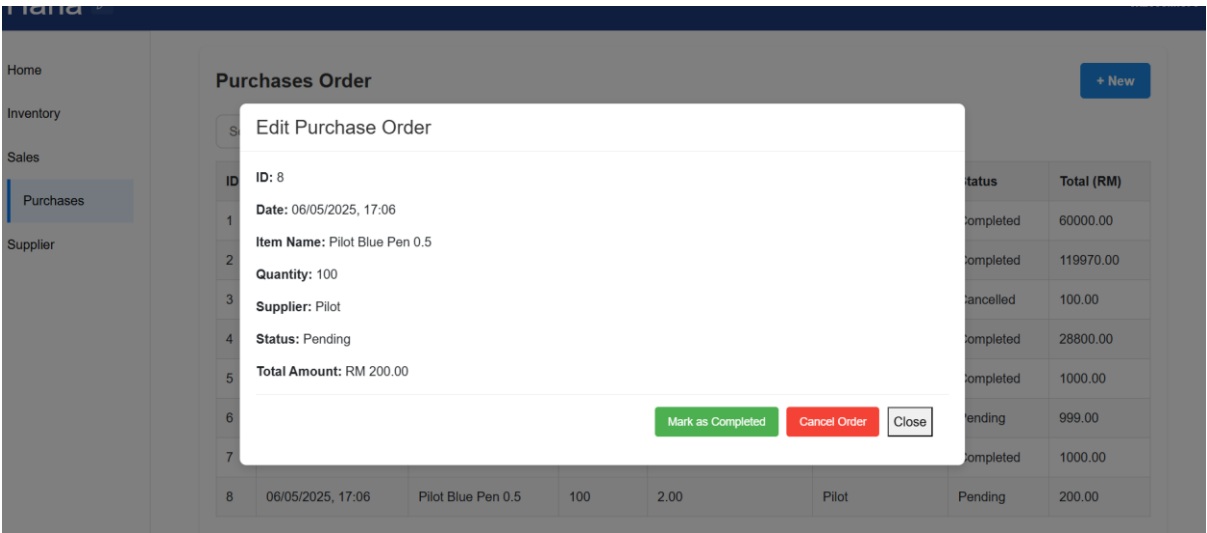


Figure 5.24 Edit Purchase Order

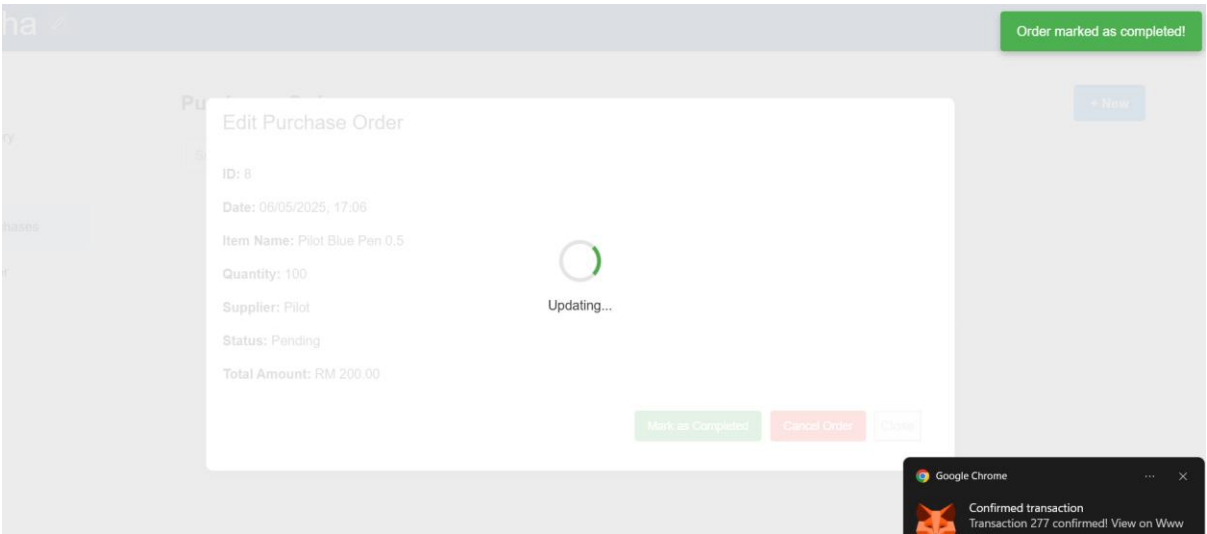


Figure 5.25 Purchase Order Completed

CHAPTER 5

In the purchase orders page, it can show all purchase orders that created. Then, the user can use the search bar to find the order. Then, the user can click the new button to add new a sales order, after clicking the button the user can fill in the detail for the order. After filling in the details and submitting it, the system initiates a transaction on the blockchain. This transaction records company name securely and immutably through a smart contract.

Then, the MetaMask wallet automatically prompts the user with a pop-up window to review and confirm the transaction, including the estimated gas fee required to process it on the blockchain. The user must approve the transaction in MetaMask for it to be successfully submitted and recorded on the blockchain network.

Once confirmed, the system waits for the blockchain to process the transaction. Upon successful completion, a toast message is displayed to the user, and user can see that the added order had shown in the table.

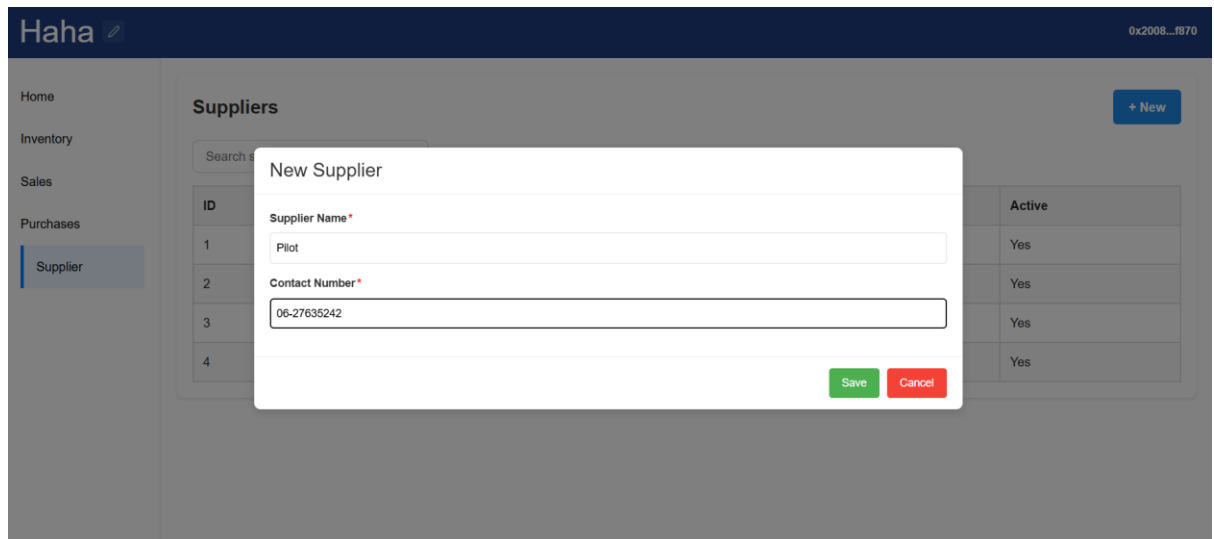


Figure 5.26 Add Supplier

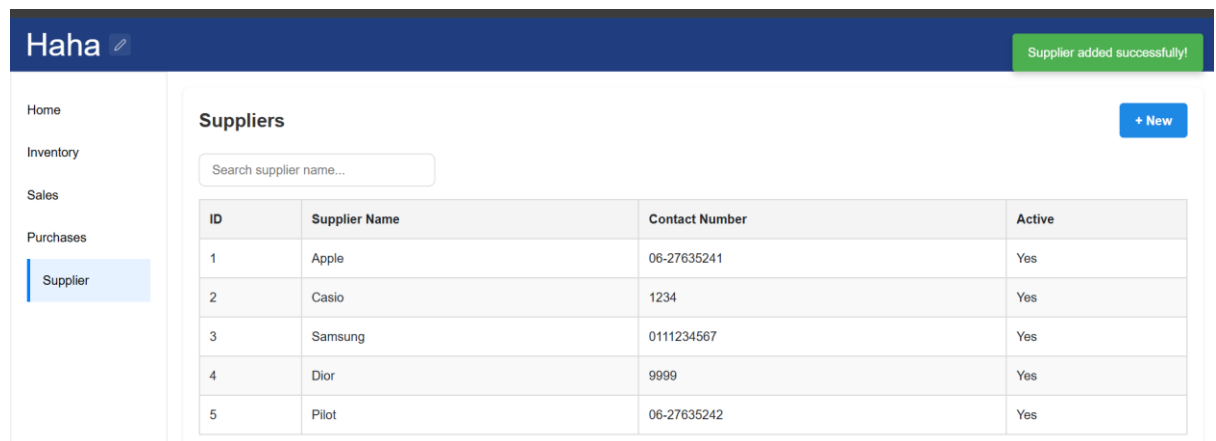


Figure 5.27 Supplier Added

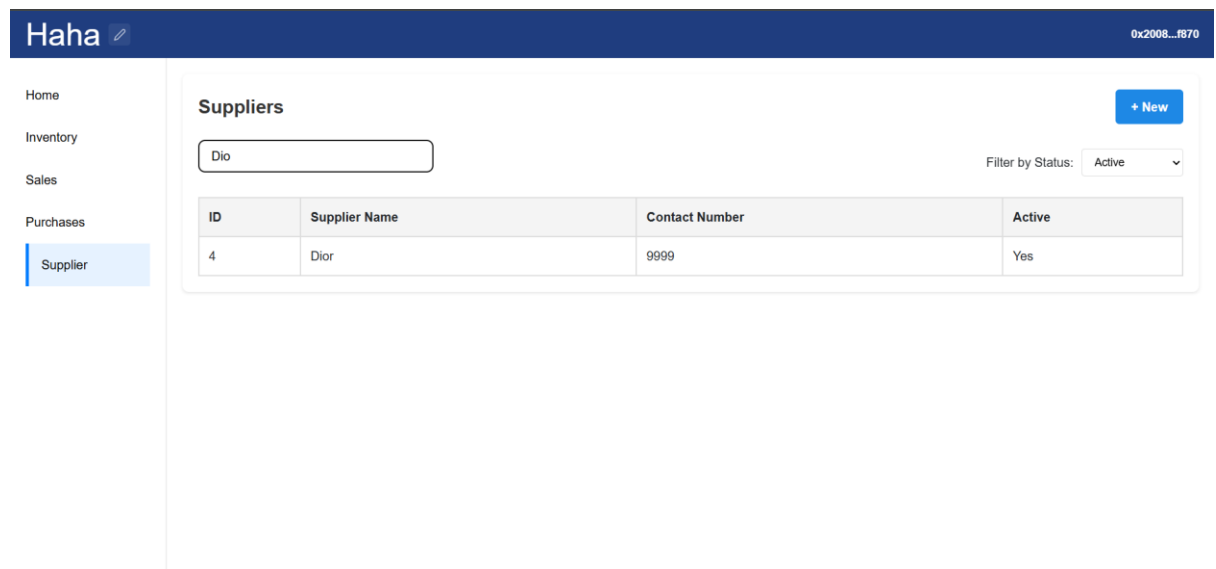


Figure 5.28 Search Supplier

Next, the supplier page it shows out all the supplier that stored in the blockchain. Then, the user can use the search bar to find the supplier. Then, the user can click the new button to add a new supplier, after clicking the button the user can fill in the detail for the supplier. After filling in the details and submitting it, the system initiates a transaction on the blockchain. This transaction records company name securely and immutably through a smart contract.

Then, the MetaMask wallet automatically prompts the user with a pop-up window to review and confirm the transaction, including the estimated gas fee required to process it on the blockchain. The user must approve the transaction in MetaMask for it to be successfully submitted and recorded on the blockchain network.

Once confirmed, the system waits for the blockchain to process the transaction. Upon successful completion, a toast message is displayed to the user, and user can see that the added supplier had shown in the table.

5.5 Implementation Issues and Challenges

During the development and deployment of the blockchain-based inventory management system, several technical challenges were encountered. These issues mainly revolved around smart contract behaviour, blockchain transaction handling and deployment workflows. The most notable challenges included the following:

1. Gas Estimation Problems

While interacting with the blockchain network (Polygon Amoy), the system encountered inaccurate gas estimations for certain transactions. Some transactions failed due to consume more gas than expected, leading to unnecessary costs during testing. To address unpredictable gas estimation and avoid failed transactions during testing on the Polygon Amoy network, the following manual gas configuration was implemented in the frontend when interacting with smart contracts. This configuration helped stabilize transaction success rates and avoid "out of gas" errors. It also allowed better control and predictability when testing contract interactions through MetaMask.

2. Debugging Blockchain Events and Transaction Logs

Debugging blockchain transactions posed a challenge due to the asynchronous nature of block confirmations and the lack of standard debugging tools. When transactions failed, it was necessary to analyse the event logs and transaction receipts in detail using Hardhat and the browser console tools (e.g., MetaMask and Etherscan).

This process required reviewing:

- Emitted event outputs (e.g., SalesOrderCreated, StockUpdated)
- Reverted transactions and their reasons
- Internal transaction traces, using tools like Hardhat console logs

Understanding the root cause of failure often involved replaying the transaction flow and manually tracing smart contract logic.

3. Smart Contract Function Errors and Redeployment

In several cases, certain smart contract functions did not behave as intended due to logic errors.

Some events were not emitted, making it harder to trace order lifecycles.

Automatic stock adjustment logic had flaws when linked to purchase and sales transactions.

To resolve these issues, the smart contracts had to be:

- **Rewritten or refactored**
- **Recompiled and redeployed** to the test network
- **Reconnected** with the frontend using updated ABI and contract address

This iterative process added complexity, especially when existing data was lost or when UI logic had to be re-synced with the contract's new structure.

5.6 Concluding Remark

The inventory management system using blockchain has been successfully completed through a combination of hardware configuration, software development and system integration. The system use MetaMask for secure authentication, smart contracts for managing inventory operations and a user-friendly web interface for seamless interaction. During the development process, several technical issues were encountered, such as gas estimation, debugging event logs and redeploying smart contracts cause by logic errors. These issues were systematically addressed by adjusting transaction parameters, analysing blockchain logs and rewrite contract functions.

Overall, the system demonstrates the feasibility of using blockchain technology to enhance transparency, automation and security in inventory management. The successful implementation is important for future improvements and scalability in real-world retail environments.

Chapter 6 System Evaluation And Discussion

6.1 System Testing and Performance Metrics

A series of functional and performance tests were conducted on the blockchain-based inventory management system to verify its fulfillment of requirements. There was testing to confirm that smart contract functions alongside their user interface met standards regarding reliability and accuracy along with response time.

- **Transaction Latency:** Record the duration from initial transaction request until blockchain verifies and confirms it. The system's average transaction completion took between 15–30 seconds because of changes in network bandwidth.
- **Gas Usage:** Tested the gas consumption metrics for the contract functions which included createSale, createPurchase and addProduct. The optimization of smart contracts minimized transaction expenses.
- **Throughput:** The system's performance was measured regarding its capacity to process transactions while functioning normally.
- **Error Rate:** Testing verified the frequency at which functions stopped working due to validation mistakes or network connectivity problems.
- **User Interface Responsiveness:** Tested frontend performance for loading times, navigation, and responsiveness to user actions.

Smart contract functions were thoroughly tested using Hardhat to ensure correctness and security. Frontend testing included both manual and automated UI testing.

6.2 Testing Setup and Result

The testing environment included the following:

- **Test Network:** Polygon Amoy (Testnet)
- **Wallet:** MetaMask
- **Frameworks:** Hardhat for smart contracts, React TypeScript with WAGMI for frontend
- **Testing Tools:** Hardhat test suite, console logging, transaction logs, and block explorers (OKLink)

Test Results:

Table 6.1 Test Result

Functionality	Expected Result	Actual Result	Status
Add Product	Product added successfully	Works as expected	Pass
Add Supplier	Supplier added successfully	Work as expected	Pass
Update product detail	Update product selling price and other information	Work as expected	Pass
Update product status	Set active or inactive	Work as expected	Pass
Update supplier status	Set active or inactive	Work as expected	Pass
Create Sales Order	Sale created + triggers reorder	Works, triggers reorder correctly	Pass
Metamask Interaction	Prompt gas fee confirmation	Metamask popup shown	Pass
Order Status Update	Completed or cancelled	Work as expected	Pass
UI Navigation	Smooth and reactive	No delay or glitches	Pass
Blockchain Logging	Transaction confirmed	Logs readable via block explorer	Pass
Search Function in Inventory, Purchase, Sales and Supplier page	Can search item probably	Work as expected	Pass

The system behaved as expected during most tests, though minor issues were identified and corrected during debugging.

6.3 System Usability Survey

For the google form survey, it had 15 questions and answer by 21 responses. The rating 1 to 5 is from (Strongly Disagree to Strongly Agree).

1. I found the system interface easy to understand.

21 responses

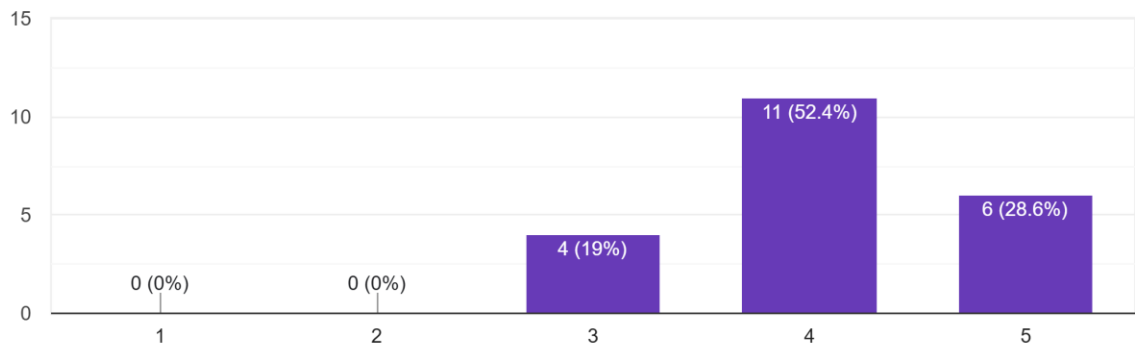


Figure 6.1 Survey Q1

Based on the feedback from 21 respondents, the overall user experience with the system was positive. Most users found the system interface easy to understand, with 52.4% rating it at 4 and 28.6% giving it the highest rating of 5.

2. I was able to navigate the system without difficulty.

21 responses

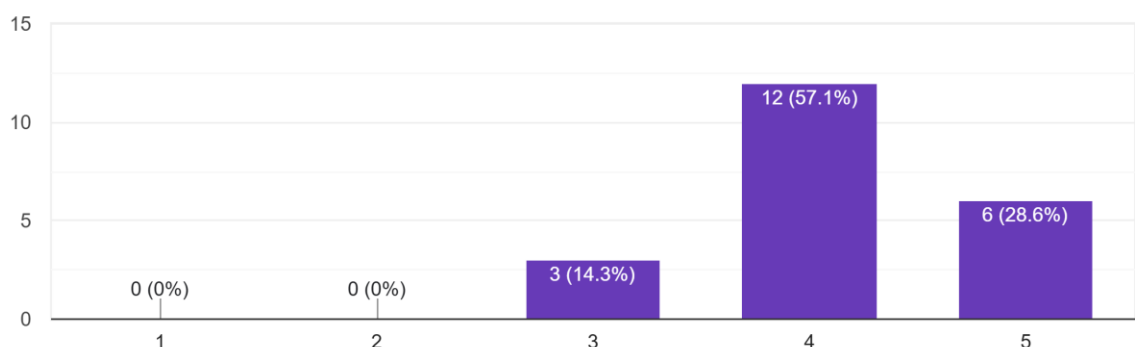


Figure 6.2 Survey Q2

CHAPTER 6

For navigation, 57.1% agreed they were able to use the system without difficulty (rating 4), and 28.6% strongly agreed (rating 5), indicating that the majority found the system intuitive.

3. I found the labels and instructions in the system clear and helpful.

21 responses

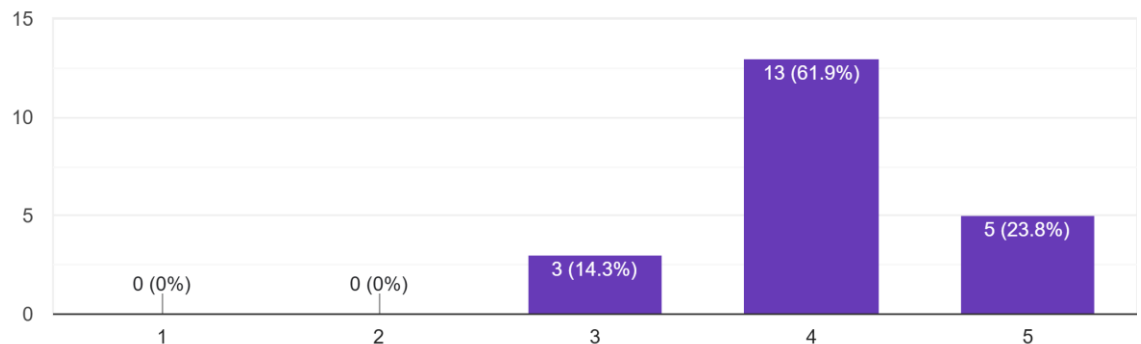


Figure 6.3 Survey Q3

For the clarity of labels and instructions, 61.9% rated it a 4 and 23.8% rated it a 5, showing that the system's guidance and labels were generally clear and helpful. A small percentage of users (14–19%) gave moderate ratings of 3 across the questions, suggesting minor areas where usability could be improved.

4. I experienced fast response times when using the system.

21 responses

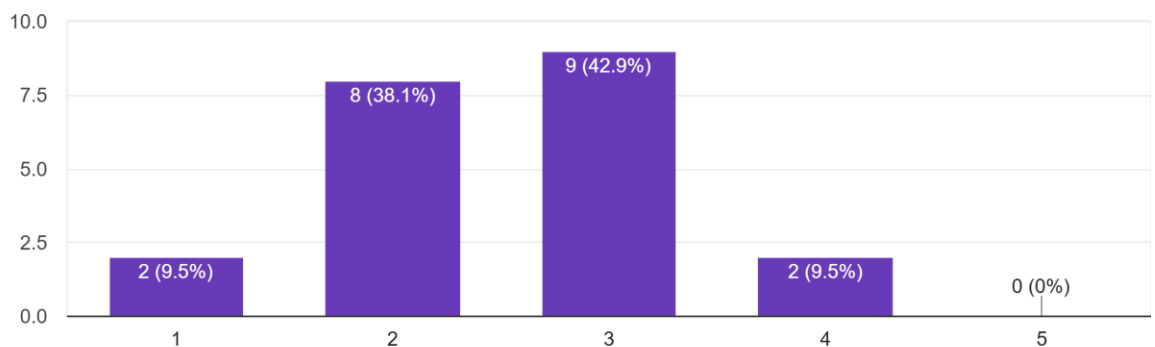


Figure 6.4 Survey Q4

Feedback for the remaining aspects of the system shows mixed responses in certain areas. Regarding system performance, 42.9% of users rated the response time as

CHAPTER 6

average (3), while 38.1% rated it lower (2), and only 9.5% rated it above average (4).

This suggests performance optimization may be needed to improve speed.

5. I found the system visually appealing.

21 responses

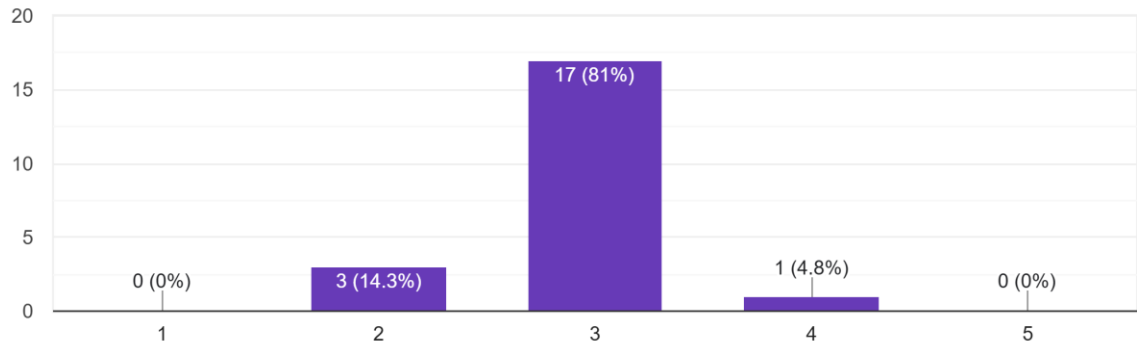


Figure 6.5 Survey Q5

For visual appeal, the majority (81%) rated it a 3, indicating the system design is acceptable but not particularly attractive. Only 4.8% rated it higher (4), and none gave it a 5.

6. I was able to add and manage inventory easily.

21 responses

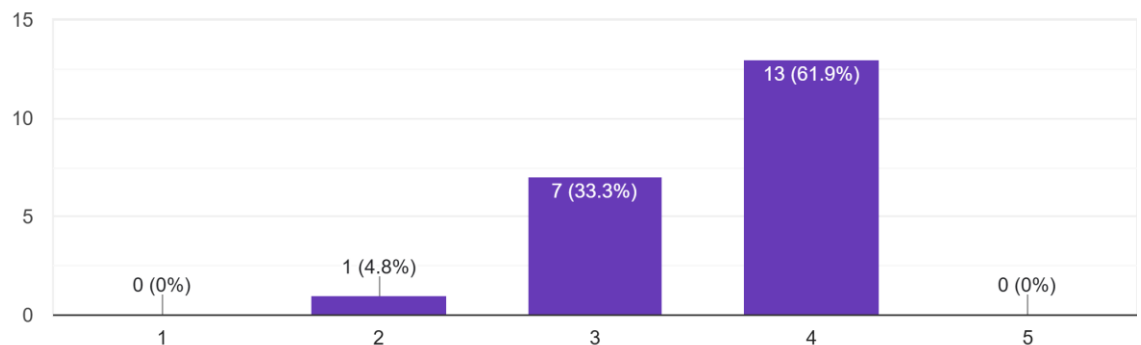


Figure 6.6 Survey Q6

Users responded more positively about inventory management, with 61.9% rating it a 4 and 33.3% giving it a 3, showing that most users found it relatively easy to add and manage inventory within the system.

7. I felt the system performed efficiently and reliably.

21 responses

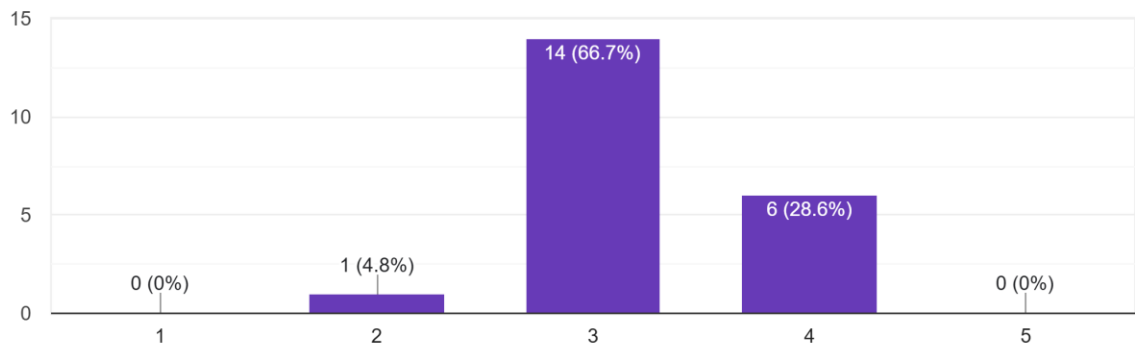


Figure 6.7 Survey Q7

The responses reflect generally positive impressions of the system's reliability and usability features. For system performance, 66.7% of users rated it a 3, indicating it was moderately efficient and reliable, while 28.6% rated it slightly higher at 4.

8. I found it easy to connect my wallet.

21 responses

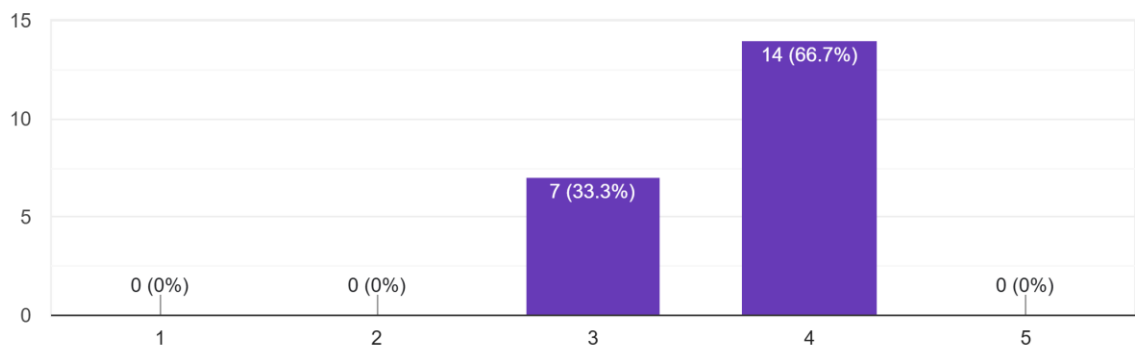


Figure 6.8 Survey Q8

For wallet connectivity, 66.7% of respondents found it easy (rated 4), and 33.3% rated it as average (3), showing no major issues with connecting their wallet.

9. I received clear feedback after performing actions in the system.

21 responses

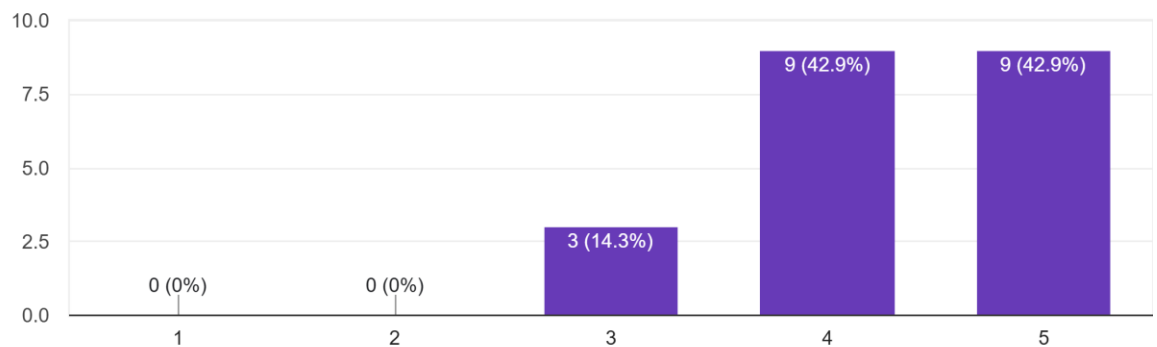


Figure 6.9 Survey Q9

For system feedback, users responded very positively: 42.9% rated the clarity of feedback a 4 and another 42.9% rated it a 5, most users clearly understood the system's responses after taking action.

10. I felt confident using the system after a short period.

21 responses

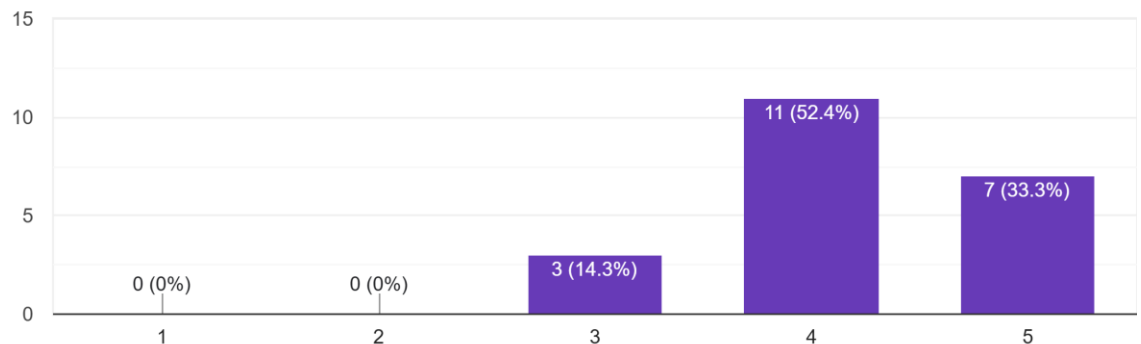


Figure 6.10 Survey Q10

For Question 10, which asked users if they felt confident using the system after a short period, the majority responded favorably, with 52.4% rating it a 4 and 33.3% rating it a 5. This indicates that most users were able to quickly adapt to and feel comfortable using the system, suggesting it is intuitive and user-friendly.

11. I found the help messages, tooltips, or error prompts useful.

21 responses

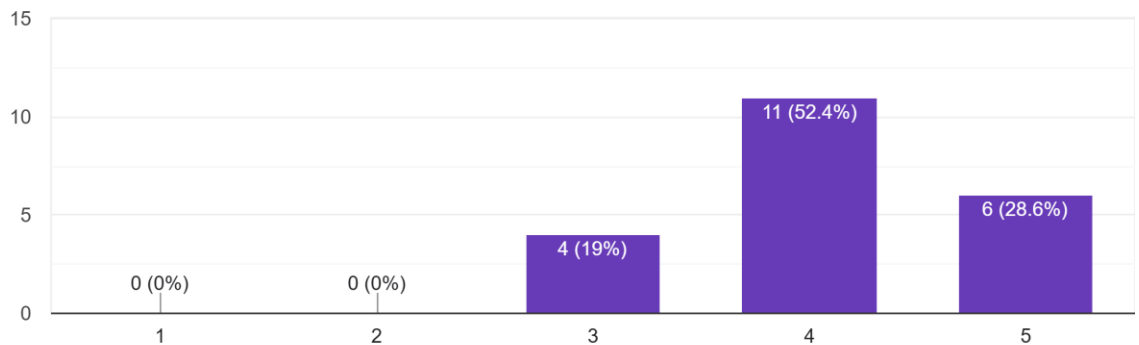


Figure 6.11 Survey Q11

Question 11 focused on the usefulness of help messages, tooltips, or error prompts.

Here, 52.4% of users gave a rating of 4 and 28.6% rated it a 5, showing that the built-in guidance tools are considered effective and supportive.

12. I found the inventory and transaction records to be accurate.

21 responses

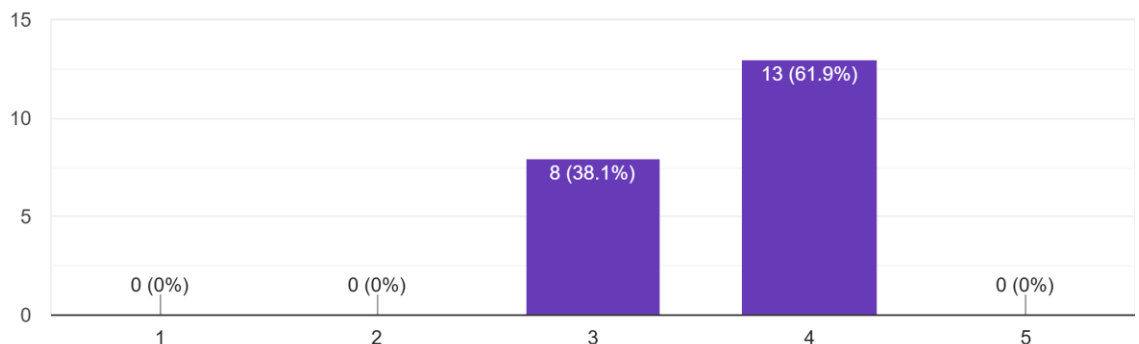


Figure 6.12 Survey Q12

For Question 12, which assessed the accuracy of inventory and transaction records, the responses were slightly more moderate. While 61.9% rated it a 4 and 38.1% gave it a 3, no users rated it a 5. This suggests that while the records are generally reliable, there may be room for improvement to achieve a higher level of trust and accuracy from users.

13. I felt secure using the system.

21 responses

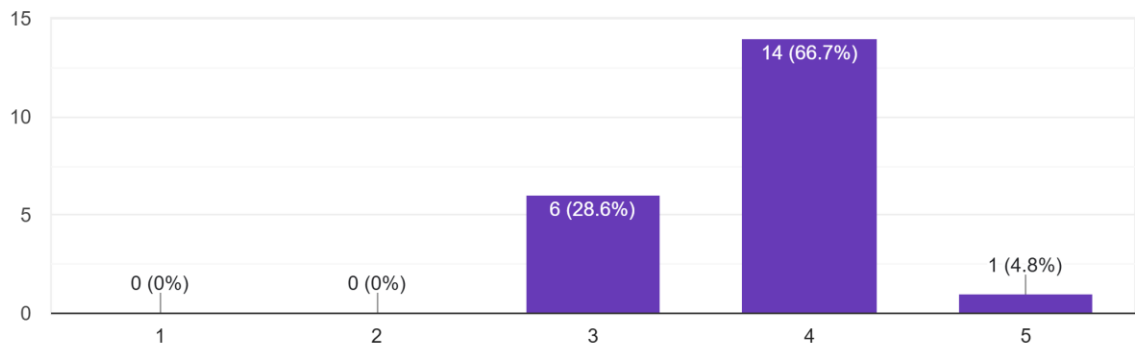


Figure 6.13 Survey Q13

For the question on security, most respondents (66.7%) felt secure using the system, rating it a 4 out of 5, while a smaller group (28.6%) gave it a 3. Only one respondent gave the highest rating and none rated it poorly, suggesting that while users trust the system to a fair extent, there's still a gap before full confidence is achieved.

14. I believe the system met my expectations for a blockchain-based solution.

21 responses

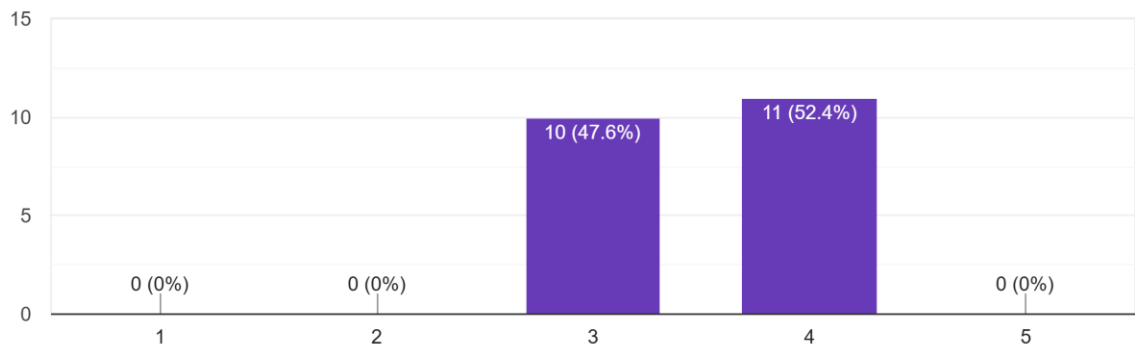


Figure 6.14 Survey Q14

For whether the system met expectations for a blockchain-based solution, the majority (52.4%) rated it a 4, and the rest (47.6%) gave it a 3. This indicates that while users felt the system fulfilled its basic promise, it did not significantly exceed expectations.

15. I would recommend this system to others.

21 responses

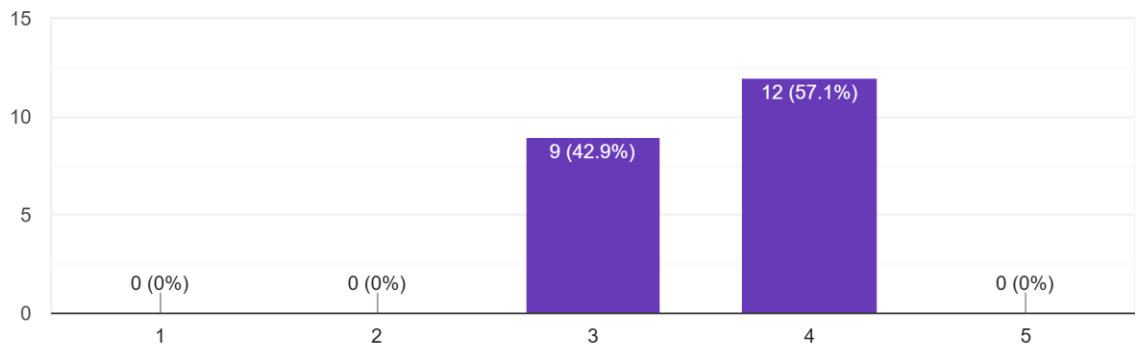


Figure 6.15 Survey Q15

When asked if they would recommend the system to others, 57.1% of users responded with a 4, and 42.9% with a 3, again showing moderate satisfaction but no strong advocacy.

The survey results from 21 respondents reveal positive assessment of system use by showing several benefits alongside some anatomical limitations. Users discovered that the system provided seamless usability and efficient inventory management and navigation because of its convenient functional design. Most users managed to link their wallets successfully and they received prompt understanding whenever they executed actions which supported their smooth use of the system.

The survey participants expressed mixed opinions about the system's speed together with its visual presentation and reliability perceptions. Users overall found that the blockchain system met target functionality standards yet failed to surpass typical blockchain specifications. The level of security matched expectations but users maintained the system needed improvement in its help tools and documentation. Most of the users-maintained self-assurance in their ability to navigate the system efficiently and showed mild enthusiasm about recommending it to others. Focused development work on performance and visual design with system responsiveness should allow the system to reach exceptional user satisfaction rates and wider adoption.

6.4 Objective Evaluation

The main objective of the project is to enhance the inventory system in the retail sector by implementing a blockchain-based system. This section determines the success rate of each established objective through analysis of the system implementation and testing period.

6.4.1 Enhance Data Integrity

The project implemented blockchain-based storage systems to document inventory transactions that tracked product additions and purchase orders and sales orders. All inventory data becomes securely stored in an immutable ledger through the deployment of smart contracts on blockchain. Protected data storage on the blockchain ensures no one can tamper with records beyond their recording and confirmation state.

The testing revealed that permanent storage of transaction data (such as product sales) appears on the blockchain where it becomes impossible to modify. This can enhance trust among stakeholders and ensures that inventory information remains consistent, traceable and secure. The achievement of data integrity enhancement goals has met the specified objectives.

6.4.2 Improve Traceability

The traceability of inventory was achieved through blockchain logging of all transactions including product creation, sales, purchases and reorders. Blockchain records transactions that contain timestamps and actor addresses (supplier or store manager) along with transaction data that includes product ID and quantity. A blockchain search enabled users to view complete transaction records containing product information for every transaction. This makes it easy to identify supply chain issues such as missing items or unusual activity.

6.4.3 Automate Reorder

The system designed automatic reordering through smart contracts that automatically track inventory quantities constantly. When the quantity of a product drops below the threshold that had set, the system automatically generates a new purchase order and records it on the blockchain. Transaction execution between the system and store managers and suppliers causes restocking prompts to happen.

Smart contracts proved they could automate reorder functions after developers solved minor production issues including gas fee adjustments and function rewrites. Through the testing showed that reorders are triggered accurately and recorded transparently, reducing manual workload and human errors. Success in achieving this objective validates the work.

6.5 Concluding Remark

The tests on the blockchain-based inventory management system proved successful and demonstrated its ability to solve fundamental inventory system problems. Through testing and analysis, found that the system successfully enhances data integrity, improves traceability and automates the reorder process using smart contracts.

Different operations did not impact the system performance because blockchain provided immutable records which secured and verified inventory records. Although some technical issues such as gas estimation tuning and smart contract redeployment were encountered during development, these challenges were resolved and led to a deeper understanding of blockchain implementation in real-world applications.

The final system successfully achieved its goals by implementing a robust transparent automated system to track inventory management. Learning captured during development testing phases will create a strong framework for evolving and enlarging the system in the future.

Chapter 7 Conclusion and Recommendation

7.1 Conclusion

The blockchain-based Inventory Management System solves essential retail inventory issues that stem from data unreliability and fraud along with improper reordering.

Through the integration of blockchain technology specifically smart contracts and decentralized storage, the system introduces a new level of transparency, security and automation into inventory workflows.

The current development phase of the system reveals successful operations through initial testing conducted on Polygon Amoy testnet. The backend smart contracts along with their management of customers, products and orders and automatic reordering system adhere to a modular design. An interface exists between the TypeScript-based frontend powered by React while MetaMask handles the secure transaction confirmation path.

The result show that blockchain is immutability and transparency significantly improves trust and accuracy in inventory operations. Smart contracts handle automatic reordering tasks which eliminate human-handled processes and produces improved results than conventional procedures. Furthermore, the system's component-based architecture allows for scalability and future expansion.

The present prototype demonstrates remarkable potential despite needing deployment and validation tests with real-world data. Through its implementation the project acts as an important solution for inventory management problems via decentralized systems. Future tests and system improvements will make the system's stability and operational effectiveness.

7.2 Recommendation

Based on the outcomes of this project, the following recommendations are proposed for future enhancement and deployment of the system:

1. Enhance UI/UX Design

While the current interface is functional, more intuitive visual elements, loading states during MetaMask transactions and mobile responsiveness could improve overall user experience.

2. Integrate QR Code or RFID Scanning

To further streamline inventory input and output, future versions could integrate hardware features like QR code or RFID scanners to automate product identification and tracking.

3. Implement Role-Based Access Control

Currently, permissions are handled simply through the smart contract. Adding a more detailed access control mechanism will support scalability across organizations with different user levels (e.g., admin, customer, supplier).

4. Connect to Off-Chain Databases

For performance and reporting purposes, integrating an off-chain database (e.g., Firebase, PostgreSQL) synchronized with on-chain events will allow for efficient data retrieval, filtering and analytics without compromising blockchain integrity.

5. Improve Smart Contract Efficiency

Review and refactor smart contracts to optimize gas usage further. Although gas fees were manually configured, optimizing logic flow and reducing redundant storage calls will minimize transaction costs.

6. Deploy on a Mainnet

Once the system is fully optimized and tested, deploying on a production blockchain (e.g., Polygon mainnet) would offer real-world functionality and readiness for integration with real businesses.

In conclusion, this project lays a solid foundation for a secure, automated and traceable inventory system using blockchain. With further improvements, it has strong potential for adoption in small and medium retail enterprises seeking reliable stock management solutions.

REFERENCES

- [1] S. Vembu, “Online inventory management software: Zoho inventory,” Inventory management software | Online inventory management for businesses - Zoho Inventory, <https://www.zoho.com/inventory/> (accessed Mar. 30, 2024).
- [2] F. Pinckaers, “The #1 open source inventory management,” Odoo, <https://www.odoo.com/app/inventory> (accessed Mar. 30, 2024).
- [3] SoluLab, “Blockchain in retail industry is beyond inventory management, why?,” Blockchain Technology, Mobility, AI and IoT Development Company USA, Canada, <https://www.solulab.com/blockchain-retail-inventory-management/> (accessed Mar. 30, 2024).
- [4] J. Anglen, “Blockchain powered retail industry 2023 - supply chain and inventory management to ensuring product provenance,” On Time, Every Time: AI & Blockchain Solutions In 90 Days, <https://www.rapidinnovation.io/post/blockchain-powered-retail-industry-2023-supply-chain-and-inventory-management-to-ensuring-product-provenance> (accessed Mar. 30, 2024)
- [5] “A complete guide to inventory management with the use of Blockchain,” BSV Blockchain, <https://www.bsvblockchain.org/news/a-complete-guide-to-inventory-management-with-the-use-of-blockchain> (accessed Feb. 4, 2025).
- [6] B. A. Kurdi, H. M. Alzoubi, I. Akour, and M. T. Alshurideh, “The effect of Blockchain and smart inventory system on supply chain performance: Empirical evidence from retail industry,” *Uncertain Supply Chain Management*, vol. 10, no. 4, pp. 1111–1116, 2022. doi:10.5267/j.uscm.2022.9.001
- [7] P. Helo and Y. Hao, “Blockchains in operations and Supply Chains: A model and reference implementation,” *Computers & Industrial Engineering*, vol. 136, pp. 242–251, Oct. 2019. doi:10.1016/j.cie.2019.07.023

REFERENCES

- [8] A. Rejeb, K. Rejeb, S. Simske, and J. G. Keogh, "Exploring blockchain research in supply chain management: A latent Dirichlet allocation-driven systematic review," *Information*, vol. 14, no. 10, p. 557, Oct. 2023. doi:10.3390/info14100557
- [9] E. P. Mondol, "The impact of blockchain and Smart Inventory System on supply chain performance at retail industry," *International Journal of Computations, Information and Manufacturing (IJCIM)*, vol. 1, no. 1, Dec. 2021. doi:10.54489/ijcim.v1i1.30
- [10] H. Magd, M. S. Ansari, and S. Negi, "Impact of blockchain technology on operations and Supply Chain Management Performance," *Proceedings of the 1st International Conference on Innovation in Information Technology and Business (ICIITB 2022)*, pp. 22–35, 2023. doi:10.2991/978-94-6463-110-4_3
- [11] S. Al-Farsi, M. M. Rathore, and S. Bakiras, "Security of blockchain-based supply chain management systems: Challenges and opportunities," *Applied Sciences*, vol. 11, no. 12, p. 5585, Jun. 2021. doi:10.3390/app11125585
- [12] "React-loader-spinner," npm, <https://www.npmjs.com/package/react-loader-spinner> (accessed Apr 7, 2025).
- [13] "Git-push documentation," Git, <https://git-scm.com/docs/git-push> (accessed Apr 7, 2025).
- [14] J. R. Admin, "Item benchmarks for the system usability scale - jux," *JUX - The Journal of User Experience*, <https://uxpajournal.org/item-benchmarks-system-usability-scale-sus/> (accessed Apr 7, 2025).

APPENDIX

FYP System Usability Survey-Inventory Management System for Retail Using Blockchain

This form is designed to collect user feedback on the usability of our blockchain-based inventory management system. Your responses will help us evaluate the system's performance, ease of use, and areas for improvement.

Please answer all questions honestly based on your experience.

All responses will be kept confidential and used strictly for academic and system development purposes.

* Indicates required question

1. I found the system interface easy to understand. *

	1	2	3	4	5	
Strongly Disagree	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Strongly Agree

2. I was able to navigate the system without difficulty. *

	1	2	3	4	5	
Strongly Disagree	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Strongly Agree

3. I found the labels and instructions in the system clear and helpful. *

	1	2	3	4	5	
Strongly Disagree	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Strongly Agree

4. I experienced fast response times when using the system. *

	1	2	3	4	5	
Strongly Disagree	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Strongly Agree

5. I found the system visually appealing. *

	1	2	3	4	5	
Strongly Disagree	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Strongly Agree

6. I was able to add and manage inventory easily. *

	1	2	3	4	5	
Strongly Disagree	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Strongly Agree

7. I felt the system performed efficiently and reliably. *

	1	2	3	4	5	
Strongly Disagree	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Strongly Agree

8. I found it easy to connect my wallet. *

	1	2	3	4	5	
Strongly Disagree	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Strongly Agree

9. I received clear feedback after performing actions in the system. *

	1	2	3	4	5	
Strongly Disagree	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Strongly Agree

10. I felt confident using the system after a short period. *

	1	2	3	4	5	
Strongly Disagree	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Strongly Agree

11. I found the help messages, tooltips, or error prompts useful. *

	1	2	3	4	5	
Strongly Disagree	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Strongly Agree

12. I found the inventory and transaction records to be accurate. *

	1	2	3	4	5	
Strongly Disagree	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Strongly Agree

13. I felt secure using the system. *

	1	2	3	4	5	
Strongly Disagree	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Strongly Agree


14. I believe the system met my expectations for a blockchain-based solution. *

	1	2	3	4	5	
Strongly Disagree	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Strongly Agree

15. I would recommend this system to others. *

	1	2	3	4	5	
Strongly Disagree	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Strongly Agree

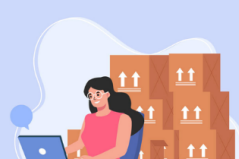
This form was created inside of Universiti Tunku Abdul Rahman.
Does this form look suspicious? [Report](#)



UTAR
UNIVERSITI TUNKU ABDUL RAHMAN

**FACULTY OF INFORMATION COMMUNICATION
AND TECHNOLOGY**

Inventory System fo Retail using Blockchain Technology

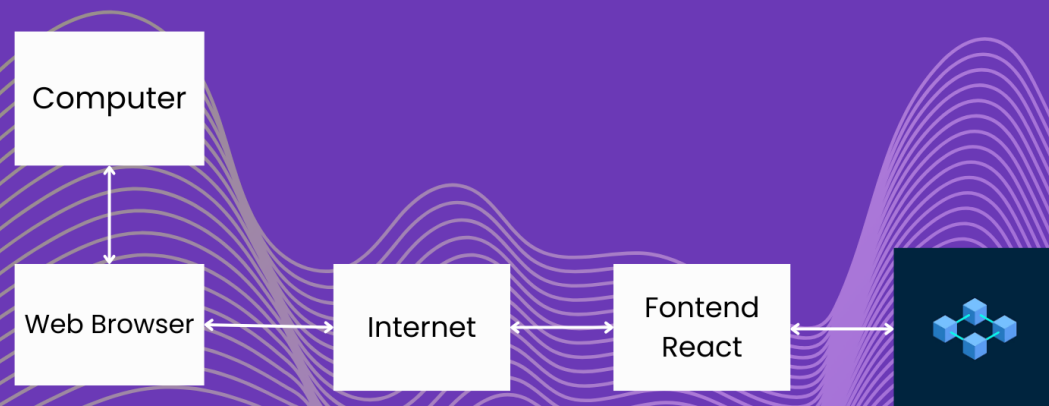


Introduction

Inventory Management System using blockchain is an innovative approach that leverages decentralized ledger technology to enhance the accuracy, transparency and security of inventory tracking and management.

Objective

- Develop a blockchain-based Inventory management system
- Automate Reorder
- Enhance Data Integrity
- Traceability



```

graph LR
    Computer[Computer] <--> WebBrowser[Web Browser]
    WebBrowser <--> Internet[Internet]
    Internet <--> Frontend[Fontend React]
    Frontend <--> Blockchain[Blockchain]
    
```

Project Developer: Yong Wei Bang
Project Supervisor: Dr Zurida Binti Ishak