

Secure Ration Provision: Ethereum Blockchain Powered Supply Chain Management for India's Public Distribution System

Submitted in partial fulfillment of the requirements for the degree of

Bachelor of Technology

in

Electronics and Communication Engineering

by

B Kishorkumar – 20BEC0084

Zubin James Krishnendu Palit – 20BEC0088

Sabyasachi Mohanty – 20BEC0401

Under the guidance of

Dr. Sujatha R

School of Electronics Engineering

VIT, Vellore.



VIT[®]
Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)

April, 2024

DECLARATION

I hereby declare that the thesis entitled **Secure Ration Provision: Ethereum Blockchain Powered Supply Chain Management for India's Public Distribution System** submitted by me, for the award of the degree of *Bachelor of Technology in Electronics and Communication Engineering* to VIT is a record of bonafide work carried out by me under the supervision of **Dr. Sujatha R.**

I further declare that the work reported in this thesis has not been submitted and will not be submitted, either in part or in full, for the award of any other degree or diploma in this institute or any other institute or university.

Place : Vellore

Date : 30/04/2024



Signature of the Candidate

CERTIFICATE

This is to certify that the thesis entitled **Secure Ration Provision: Ethereum Blockchain Powered Supply Chain Management for India's Public Distribution System** submitted by **B Kishorkumar 20BEC0084, SENSE, Zubin James Krishnendu Palit 20BEC0088, SENSE, Sabyasachi Mohanty 20BEC0401, SENSE, VIT**, for the award of the degree of *Bachelor of Technology in Electronics and Communication Engineering*, is a record of bonafide work carried out by him / her under my supervision during the period, 03.01.2024 to 30.04.2024, as per the VIT code of academic and research ethics.

The contents of this report have not been submitted and will not be submitted either in part or in full, for the award of any other degree or diploma in this institute or any other institute or university. The thesis fulfills the requirements and regulations of the University and in my opinion meets the necessary standards for submission.

Place : Vellore

Date : 30/04/2024

Signature of the Guide

Internal Examiner

External Examiner

Dr. NOOR MOHAMMED V
Department of Communication Engineering
School of Electronics Engineering, VIT

ACKNOWLEDGEMENTS

We extend our deepest gratitude to our Honourable Chancellor, Dr. G. Vishwanathan, , and the entire VIT management for their unwavering support and provision of essential resources, pivotal to the successful culmination of our project.

Our heartfelt appreciation goes to Dr. Sujatha Rajkumar, our guide, whose consistent guidance and encouragement steered us through every phase of our project journey. Her expertise and insights were instrumental in shaping our work. We are also grateful to the School of Electronics Engineering (SENSE) at VIT for fostering an environment conducive to curiosity and learning, significantly contributing to our project's development.

We extend our sincere gratitude to our families and friends for their unyielding belief in us and continuous encouragement, serving as a constant source of inspiration.

Lastly, we express our appreciation to each member of our team, whose dedication, collaboration, and relentless efforts were indispensable in achieving our project's objectives. It is through our collective commitment and synergy that we have reached this milestone of success.

B Kishorkumar (20BEC0084)

Zubin James Krishnendu Palit (20BEC0088)

Sabyasachi Mohanty (20BEC0401)

EXECUTIVE SUMMARY

Exploring the Application of Blockchain Technology in India's Public Distribution System (PDS), this project addresses inefficiencies within the supply chain and enhances its transparency. The focus is on developing a Decentralized Application (dApp) for the PDS, with the primary aim to improve transparency and efficiency and ensure food security among millions of vulnerable individuals. The proposed system provides a transparent and immutable record of transactions throughout the supply chain, enhancing overall efficiency and effectiveness while addressing issues of accountability and data integrity. The methodology involves developing a comprehensive system architecture for integrating blockchain into the existing PDS infrastructure and creating a permissioned blockchain network to ensure only authorized participants engage. This has implications not only for PDS management but also for similar systems globally.

CONTENTS	PAGE NO.
ACKNOWLEDGEMENTS	i
EXECUTIVE SUMMARY	ii
List of Figures	iv
List of Tables	v
List of Abbreviations	vi
Symbols and Notations	vii
1. INTRODUCTION	1
1.1 OBJECTIVE	1
1.2 MOTIVATION	2
1.3 BACKGROUND	2
2. PROJECT DESCRIPTION AND GOALS	4
3. TECHNICAL SPECIFICATION	6
4. DESIGN APPROACH AND DETAILS	12
4.1 DESIGN APPROACH / MATERIALS & METHODS	12
4.2 CODES & STANDARDS	20
4.3 CONSTRAINTS, ALTERNATIVES AND TRADEOFFS	21
5. SCHEDULE, TASKS AND MILESTONES	23
6. PROJECT DEMONSTRATION	26
7. COST ANALYSIS / RESULT & DISCUSSION	33
8. SOCIAL IMPACT	35
9. REFERENCES	36
10. APPENDIX	37

List of Figures

Fig. No.	Title	Page No.
Figure 3.1:	Merkle Tree in Ethereum	6
Figure 3.2:	Blocks in Ethereum	8
Figure 3.3:	Ganache Interface	10
Figure 3.4:	Smart Contract in Ganache	11
Figure 4.1:	Entity Diagram	13
Figure 4.2:	System Architecture	14
Figure 4.3:	Shipment Flow between administrators	15
Figure 4.4:	Customer Purchase Flow	16
Figure 4.5:	Complete Workflow	17
Figure 5.1:	Gantt Chart of Project Timeline	23
Figure 6.1:	Home Page	26
Figure 6.2:	Login Portal	26
Figure 6.3:	Marketyard Home	27
Figure 6.4:	Warehouse Home	28
Figure 6.5:	Confirmation Pop-up	28
Figure 6.6:	Shipment Status Pop-up	28
Figure 6.7:	Customer Home	29
Figure 6.8:	Payment through Metamask	30
Figure 6.9:	Shipment Summary	30
Figure 6.10:	Blockchain Transaction	31
Figure 6.11:	A Block in the Blockchain	31
Figure 6.12:	Consecutive Block in the Blockchain	32
Figure 7.1:	Block Confirmation time in Sepolia	34

List of Tables

Table. No.	Title	Page No.
Table 3.1 :	Software Details	12
Table 4.1 :	Transaction Functions	18
Table 7.1 :	Transaction Costs	33
Table 7.2 :	Block Confirmation time in Sepolia	34

List of Abbreviations

PDS	Public Distribution System
dApp	Decentralized Application
PoS	Proof of Stake
PoW	Proof of Work
ETH	Ethereum
ABI	Application Binary Interface

Symbols and Notations

Notation	Explanation
addrF	Address of farmer
addrM	Address of market yard
addrW	Address of warehouse
addrR	Address of ration shop
addrC	Address of customer
card	Ration card type
ad	Number of adults in the family
ch	Number of children in the family
alloc	Ration allocation for customer
w	Weight of wheat in shipment
r	Weight of rice in shipment
s	Weight of sugar in shipment
□	

1. INTRODUCTION

1.1 OBJECTIVE

The objective of this work is to explore the potential of blockchain technology in revolutionizing the management of the PDS in India, primarily by developing a specialized decentralized application (dApp). This enhances transparency, efficiency, and accountability while progressing food security for the most marginalized communities in India. By taking advantage of blockchain's transparent and decentralized ledger, the project seeks to address the need for a transformative change within the PDS, mitigating longstanding challenges and fostering a more equitable and efficient distribution of essential food commodities.

The primary objectives of the project encompass a comprehensive overhaul of the existing Public Distribution System (PDS) through the integration of blockchain technology. Firstly, the project aims to develop a robust system architecture tailored to the unique requirements and constraints of the ration distribution process. Secondly, the project focuses on the implementation of a permissioned blockchain network, restricting access to authorized participants such as government agencies, ration shop owners, and verified beneficiaries. By leveraging the decentralized ledger, the system will maintain a transparent and immutable record of all transactions within the ration distribution system, enhancing transparency and accountability. By integrating secure identity verification protocols with blockchain technology, the system will safeguard against identity fraud and unauthorized access.

Another key objective of the project is the development of an e-Ration store, a decentralized application designed to digitize end-users' interactions with ration stores. This digital platform will streamline the process of accessing subsidized provisions, enhancing convenience and accessibility for beneficiaries.

Lastly, the project emphasizes stakeholder collaboration, aiming to facilitate seamless interaction and cooperation among government agencies, suppliers, distributors, and beneficiaries. Through effective communication channels and interoperable systems, the project seeks to foster a cohesive ecosystem that maximizes the benefits of blockchain technology in the PDS.

1.2 MOTIVATION

Ensuring food security is paramount for the well-being of the nation's populace, particularly for those living below the poverty line. The PDS plays a crucial role in providing subsidized food grains and essential commodities to economically disadvantaged households, thereby safeguarding their nutritional needs and mitigating hunger and malnutrition.

However, India's PDS has long been plagued by issues such as leakages, corruption, inefficiencies, and lack of transparency. Reports of diversion of subsidized food grains meant for the poor, along with instances of ration card fraud, underscore the urgent need for reform in the distribution system. These challenges not only undermine the effectiveness of welfare programs but also perpetuate food insecurity among marginalized communities.

By leveraging the transparency, immutability, and decentralized nature of blockchain, it becomes possible to create a tamper-proof system that ensures end-to-end traceability and accountability in the distribution of essential commodities. By minimizing leakages and ensuring the targeted delivery of subsidized food grains to genuine beneficiaries, the system can alleviate the plight of the poor, enhance their access to nutritious food, and contribute to poverty alleviation efforts.

1.3 BACKGROUND

India's Public Distribution System was initially introduced in the 1940s to distribute food grains to combat food shortages. Over the years, the PDS has evolved into a critical social welfare program aimed at providing subsidized commodities including rice, wheat and sugar to the destitute population of the countries. PDS plays a crucial role in India's efforts to ensure food security and alleviate poverty by making essential commodities accessible and affordable to millions of people.

Despite its noble objectives, the PDS faces numerous challenges that hamper its efficacy. The most common challenge faced by the PDS is leakages in the supply chain where the subsidized food grains meant for the poor are diverted to the open market or siphoned off through corrupt practices. This not only results in the inefficiency of the PDS but also leads to significant

losses of resources and undermines the intended benefits of the PDS for the deserving beneficiaries.

Blockchain technology was introduced as the underlying technology for Bitcoin cryptocurrency. Blockchain at its core, is a decentralized and immutable ledger that records transactions across a network of computers, providing transparency, security, and trust among participants. Blockchain has gained widespread attention for its potential to revolutionize various industries, including supply chain management. Blockchain holds the promise to address many of the problems faced by the traditional supply chain management systems such as improving transparency, traceability and accountability throughout the supply chain process. Ethereum is a blockchain platform that extends the capabilities of blockchain beyond simple transactions to support the execution of smart contracts, which are self-executing contracts with predefined rules encoded into them. Ethereum's programmable nature enables developers to build decentralized applications (dApps) that can automate and streamline various processes, including those related to supply chain management.

Several research studies and pilot projects have explored the application of blockchain technology in supply chain management and related fields. Some initiatives have focused on using blockchain to enhance transparency and traceability in food supply chains, ensuring food safety and quality. However very few advancements have been made in the context of Indian Public Distribution System.

In recent years, researchers have proposed innovative solutions for enhancing the efficiency and transparency of India's Public Distribution System (PDS) through the integration of blockchain technology. Himani Mishra et al. [1] and A. N. Shwetha et al. [2] have explored novel approaches to address the challenges faced by the PDS. Mishra et al. [1] focus on managing food grain supply, aiming to reduce corruption and enhance transparency by establishing end-to-end accountability. Their solution involves linking warehouses, Fair Price Shops (FPSs), and consumers through blockchain technology to ensure transparency at every stage of the supply chain. Shwetha et al. [2] advocate for a decentralized blockchain ledger coupled with IoT sensors for real-time monitoring of commodity flows. They address scalability concerns by proposing a hierarchical blockchain approach, which can effectively manage the vast amounts of data generated in the PDS.

T. S. Iddum et al. [3] contribute to the discourse by developing a client-side application for supply chain management using blockchain technology. Their platform emphasizes inventory tracking, order placement, and product attribute information logging, ensuring secure and authentic information delivery throughout the supply chain. Building upon this foundation, Madumidha et al. [4] explore the integration of blockchain and IoT to further improve transparency and efficiency in supply chain logistics. Their research highlights the potential of blockchain technology to address the limitations of traditional logistics systems by providing real-time visibility and accountability.

In addition to PDS-related applications, researchers have investigated blockchain's potential in other domains of supply chain management. Luona Song et al. [5] and Feng Tian [6] have explored blockchain's application in agricultural supply chains and food traceability systems. Song et al. [5] emphasize blockchain's ability to foster collaboration among stakeholders, while Tian [6] address food safety concerns through blockchain-based traceability solutions. Furthermore, Meyliana et al. [7] and Guipeng Zhang et al. [8] contribute to the discourse on authenticity and security within supply chains. Meyliana et al. [7] propose a blockchain-enabled business model to combat counterfeit drugs, while Zhang et al. [8] advocate for blockchain technology to enhance security and information sharing within supply chains.

Expanding the scope of blockchain applications, Benyelles Mohammed El Amine et al. [9] explore its transformative potential in revolutionizing dairy supply chains. Their research sheds light on the opportunities and challenges associated with implementing blockchain technology in dairy supply chains, paving the way for further exploration in this area.

2. PROJECT DESCRIPTION AND GOALS

The project aims to revolutionize the Indian Public Distribution System (PDS) by creating a supply chain management system to tackle prevalent issues like leakages, inefficiencies, and corruption. Key goals include enhancing transparency, efficiency, and accountability within the PDS, reducing leakages, improving beneficiary satisfaction, promoting financial inclusion for local suppliers and farmers, and strengthening governance structures. Through these efforts, the

project seeks to optimize resource allocation, streamline processes, and empower communities reliant on the PDS, ultimately fostering improved livelihoods and well-being.

The project will encompass the design, development, and deployment of a comprehensive blockchain-powered supply chain management system tailored for the Public Distribution System (PDS) in India. This comprehensive system will integrate various key components and functionalities including procurement and sourcing of commodities from suppliers and farmers, distribution and allocation of commodities to ration shops and designated beneficiaries. Additionally, the system will incorporate advanced inventory management capabilities, enabling real-time tracking and monitoring of commodity movements at every stage of the distribution process. Authentication of transactions will be ensured through the utilization of smart contracts, thereby enhancing security and reducing the risk of fraudulent activities. Moreover, the system will provide transparent audit trails, offering stakeholders visibility into the entire supply chain process for improved accountability and oversight. Additionally, user-friendly interfaces will be tailored to meet the diverse needs of stakeholders, including government officials, ration shop owners, and beneficiaries, ensuring seamless interaction and usability for all parties involved.

The implementation of the blockchain-powered supply chain management system is anticipated to yield significant benefits for the Public Distribution System (PDS) in India. It is expected to markedly reduce leakages and losses in the distribution process, ensuring that subsidized food grains and essential commodities reach their intended recipients without diversion or pilferage. Furthermore, the system will enhance transparency, efficiency, and accountability throughout the PDS, fostering greater trust among stakeholders and streamlining operational processes. Improved beneficiary satisfaction and access to subsidized provisions will result from the system's streamlined distribution mechanisms, enabling more timely and reliable provision of essential goods to those in need. Additionally, by promoting financial inclusion and empowering local suppliers and farmers, the system will stimulate economic growth and resilience within communities served by the PDS. Lastly, the implementation of the system will strengthen governance structures and institutional capacities within the PDS, laying the foundation for sustained progress and adaptability in addressing future challenges.

To ensure the successful adoption and utilization of the new supply chain management system, capacity-building initiatives and training programs will be implemented for PDS staff, ration shop owners, and other stakeholders. These initiatives will focus on imparting the

necessary skills and knowledge to effectively operate and manage the system, as well as raising awareness about the benefits of blockchain technology and the importance of data security and privacy.

3. TECHNICAL SPECIFICATION

A. Ethereum Blockchain

Ethereum is a decentralized platform that allows for the creation and execution of smart contracts. Smart contracts are self-executing agreements that have the terms of the contract directly written into code. This allows any program to be written directly to the blockchain. It operates on blockchain technology. This is similar to Bitcoin which is the most popular blockchain technology, but Ethereum has additional features. While Bitcoin primarily serves as a digital currency, Ethereum's primary purpose is to provide a platform for dApps to be built and run on the blockchain.

Fig 2 shows the state trie in Ethereum. It is of a data type called a Merkle tree. These trees are used to efficiently verify authenticity of big data structures without verifying the entire data. In case of Ethereum, this data represents the state of the blockchain, including account balances and smart contract storage data. Each leaf node corresponds to specific data and this data is hashed to produce cypher text. These cypher texts are then recursively combined to form higher-level nodes. This ultimately leads to a root hash, known as the Merkle root. This Merkle root is what is included in the block headers of each block. This means that the block's authenticity can be checked by verifying just the root hash instead of the full content.

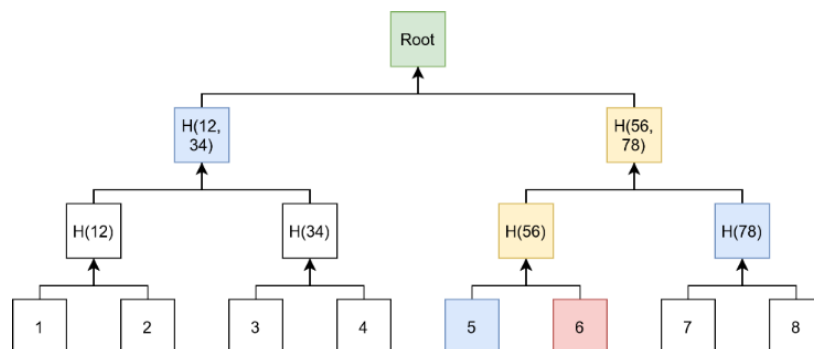


Figure 3.1: Merkle Tree in Ethereum

Miners in a blockchain are entities that are responsible for verifying the transactions and

adding it to the blockchain. They are rewarded for their work through Ether (ETH), which is the native cryptocurrency of Ethereum.

$$1 \text{ gwei} = 10^{-9} \text{ ETH (ether)}$$
$$\text{gas fee} = \text{gas limit} * \text{gas price (in gwei)}$$

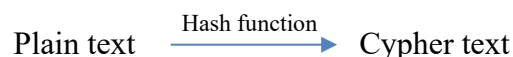
Gas price is denoted in gwei. Gas limit is the maximum amount of gas that the person calling the transaction is willing to spend. Complex transactions require a higher gas limit since there is more work in executing that transaction.

B. Smart Contracts

Smart contracts are self-executing contracts. They are similar to physical contracts but the terms of the agreement are written in code to the blockchain. They automatically execute actions when predefined conditions are met, without the need for intermediaries. This is what makes blockchain “trustless”. Smart contracts have a lot of applications beyond financial transactions. They can be useful in supply chain management, legal agreement, etc. The advantages of smart contracts include automation, transparency, security, and efficiency. They eliminate the need for trusted third parties, reduce transaction costs, and minimize the risk of fraud or manipulation.

C. Encryption in Blockchain

Encryption ensures the confidentiality and security of data stored in blocks. Each block contains a collection of transactions. These transactions are present not as plain text but rather as cypher text generated by using cryptographic encryption algorithms. Encryption algorithms such as Keccak-256 or ECC are some commonly used encryption algorithms.



The Keccak256 function is a one way encrypted algorithm. It is used in a lot of areas within Blockchain. It is used in transaction hashing, block hashing, and address generation. It takes an input message of any length and produces a fixed-size output hash value of 256 bits (32 bytes). This hash value is unique to the input message. This property signifies that even one small change

in the input data will produce a completely different hash output.

$$\text{Keccak256}(m) = H(m)$$

D. Integrity in Blockchain

In blockchain, the transaction information are contained in blocks. Each block contains a timestamp, a reference to the previous block's hash, a Merkle tree root of transaction hashes, and other metadata. Fig 3 shows the blocks in Ethereum. Each block serves as a record of transactions and is linked to the block before it. This forms a chain of blocks, hence the term blockchain.



Figure 3.2: Blocks in Ethereum

This linking of blocks ensures blockchain's immutability and security features. Any alteration to the data in a block would change the block hash. As all blocks contain the hash of the previous block, this would make changes to all the blocks that come after it. This is computationally not possible as the malicious entity would have to keep up with the new transactions happening in the network and keep mining all the blocks. This ensures that blocks in a blockchain can't be tampered.

E. Consensus Algorithm

In Ethereum, the consensus algorithm plays a critical role in ensuring agreement among network participants regarding the state of the blockchain digital ledger, which is distributed in nature. Ethereum currently utilizes a Proof of Stake (PoS) consensus mechanism. Unlike Proof of Work (PoW), where miners compete to solve complex puzzles, PoS relies on validators who are chosen to create new blocks and validate transactions based on the amount of cryptocurrency they hold and are willing to "stake" as collateral. The more cryptocurrency a validator holds and is willing to lock up, the higher the chance they have of being selected to create the next block.

and earn rewards. This design incentivizes validators to act honestly and secure the network, as they have a financial stake in the system. Additionally, PoS is generally more energy-efficient compared to PoW because it doesn't require vast amounts of computational power to solve complex puzzles.

F. Solidity

Solidity, a high-level, statically typed programming language specifically designed for Ethereum smart contract development, plays a pivotal role in the architecture of this blockchain-powered supply chain management system. Its versatility and robust features make it an ideal choice for defining the logic and behavior of smart contracts that automate various aspects of the supply chain operations. One of the key advantages of Solidity is its compatibility with the Ethereum Virtual Machine (EVM), which allows smart contracts written in Solidity to be deployed and executed on the Ethereum blockchain. This compatibility ensures seamless integration with the Ethereum network, enabling the supply chain management system to leverage Ethereum's decentralized infrastructure and native cryptocurrency, Ether (ETH), for secure and transparent transactions.

Moreover, Solidity incorporates advanced security features to mitigate common vulnerabilities in smart contracts, such as integer overflow/underflow and reentrancy attacks. These built-in security measures help to enhance the resilience and trustworthiness of the supply chain management system, ensuring that critical transactions and data remain secure and tamper-proof on the blockchain.

In this project, Solidity is utilized to create a variety of smart contracts that govern different aspects of the supply chain, including product tracking, inventory management, and transaction settlement. These smart contracts define the rules and conditions under which supply chain transactions are executed, providing transparency and accountability throughout the entire supply chain process. However, the development of smart contracts in Solidity also presents certain challenges, such as the need for thorough testing to ensure the security and reliability of the deployed contracts, as well as optimization techniques to minimize gas costs associated with executing transactions on the Ethereum network.

G. Ganache Private Blockchain

Ganache is a personal Blockchain for Ethereum development that offers a built-in Ethereum client that supports the latest Ethereum standards. It also provides a user-friendly interface to view and debug transactions, events, and Blockchain states. It provides testing accounts that come with ETH that otherwise have to be bought as they are required for making any transactions on the blockchain.

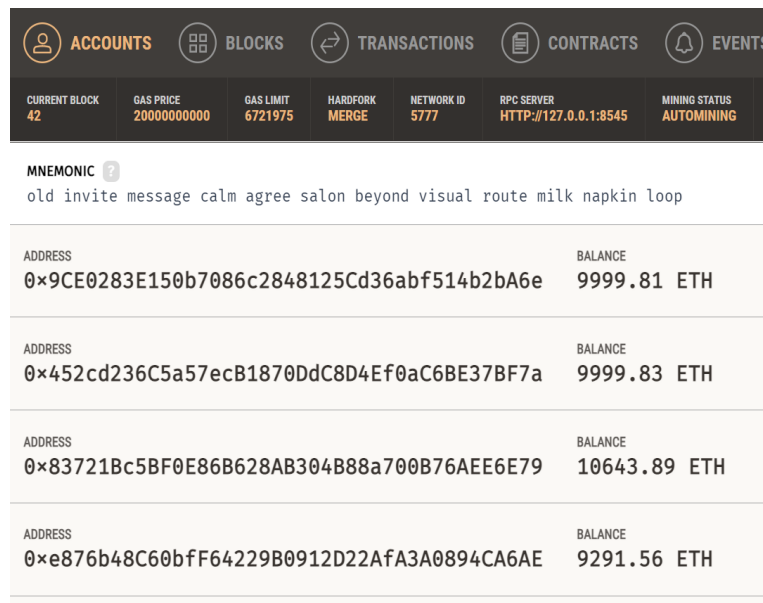


Figure 3.3: Ganache Interface

The smart contract is deployed on the Ganache blockchain. A smart contract has its own address, balance and state variables which are a key mechanism for storing and managing data on the blockchain. These variables are declared within the contract and hold data that represents the contract's state. Fig 3.4 displays a smart contract named 'pds' deployed on the Ganache blockchain network.

When the web dApp is created, it provides a user interface to interact with the smart contract using its contract address and ABI.

ACCOUNTS BLOCKS TRANSACTIONS **CONTRACTS** EVENTS LOGS SEARCH

CURRENT BLOCK 73 GAS PRICE 2000000000 GAS LIMIT 6721975 HARDFORK MERGE NETWORK ID 5777 RPC SERVER HTTP://127.0.0.1:8545 MINING STATUS AUTOMINING WORKSPACE DRAGONFRUIT

← BACK **pds**

ADDRESS
0x1Ec15Aa4852601A00E11182a7b3DD367e523de5a

BALANCE
92.00 ETH

CREATION TX
0x033BB2a6df09A4A7E0Ba9150598aC34B78bEf39021DcB71aaEc
F3cC4A2a9b7dC

STORAGE

```
{
  13 items
  owner : address "0x9CE0283E150b7086c2..."
  marketyardAdmin : address "0x9CE0283E150b7086c2..."
  warehouseAdmin : address "0x452cd236C5a57ecB18..."
  rationshopAdmin : address "0x83721Bc5BF0E86B628..."
  ▶ marketyard : { ... } struct 3 items
  ▶ warehouse : { ... } struct 3 items
  ▶ rationshop : { ... } struct 3 items
  ▶ arrWHapproval : [ ... ] 5 items
  ▶ arrRSapproval : [ ... ] 4 items
  ▶ REJbyWH : [ ... ] 5 items
  ▶ REJbyRS : [ ... ] 2 items
  ▶ Summary : [ ... ] 11 items
  shipmentID : uint 14
}
```

TRANSACTIONS

TX HASH
0x9c0feb9ce1b9d6f669cf36e6bb6f982e188357c77b6e160af6fdea737a412ded

FROM ADDRESS	TO CONTRACT ADDRESS	GAS USED	VALUE
0x9CE0283E150b7086c2848125Cd36abf514b2bA6e	pds	40130	0

Figure 3.4: Smart Contract in Ganache

H. Metamask

In this blockchain-powered supply chain management system, the Ethereum network has been leveraged for its robustness, security, and smart contract capabilities. As part of the technical specifications, MetaMask has been incorporated as the primary payment wallet solution for users interacting with the platform.

MetaMask serves as a browser extension or mobile application that enables users to manage their Ethereum accounts, interact with decentralized applications (DApps), and securely store and transfer Ether (ETH) and other ERC-20 tokens. It provides a user-friendly interface for accessing blockchain-based services directly from a web browser, enhancing accessibility and usability for both experienced and novice users.

One of the key features of MetaMask is its seamless integration with web-based

applications, allowing users to interact with the supply chain management platform directly through their preferred web browsers. This integration streamlines the user experience, eliminating the need for additional software installations or complex setup processes.

By integrating MetaMask as the payment wallet solution in the blockchain-powered supply chain management system, it is possible to enhance user accessibility, security, and convenience, empowering users to seamlessly participate in the decentralized economy while leveraging the benefits of Ethereum's blockchain infrastructure.

I. Software Details

Table 3.1: Software Details

Backend (Blockchain)	Frontend (Website)
Solidity	HTML
Truffle	CSS
Ganache	React JS
Next JS	Semantic UI
Node JS	
MetaMask	
Web3JS	

4. DESIGN APPROACH AND DETAILS

4.1 DESIGN APPROACH / MATERIALS & METHODS

A. Entity Diagram

After extensive research on India’s PDS, an entity diagram has been devised to map out the supply chain dynamics. This system revolves around five key entities: farmers, market yards, warehouses, ration stores, and customers—the end users. Fig 4.1 shows the various entities and their interactions with each other. Farmers are the first part of the supply chain who produce the various commodities . The market yard administrator is in charge of the procurement of

agricultural produce from farmers at designated wholesale markets or mandis. Warehouse administrators manage the storage and distribution of food grains procured from farmers and wholesale markets. They are in charge of the inventory management, maintenance, and security of warehouses where commodities are stored before distribution to ration shops. Ration shop administrators operate the ration stores where subsidized food grains are distributed to eligible beneficiaries under the PDS. They maintain records, manage inventory, and distribute essential commodities to consumers at government-regulated prices. Customers in the PDS system are the beneficiaries who purchase subsidized food grains from ration shops for their consumption.

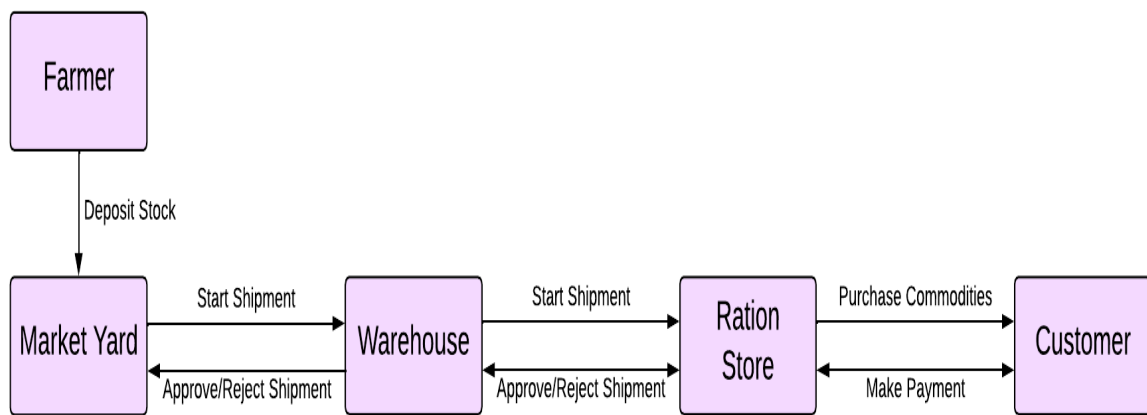


Figure 4.1: Entity Diagram

The system will encompass all entities within the supply chain and manage their interactions through the utilization of smart contracts. The process will commence with the farmer depositing their stock at the market yards, initiating a sequence of actions by different administrators to facilitate the movement of stock throughout the supply chain until it reaches the final consumer.

B. System Architecture

In the system architecture, a website serves as the primary interface with a blockchain as its foundational backend infrastructure. A smart contract is deployed on the blockchain network. The smart contract governs the transactions initiated by users via the website and provides the transaction status updates. Each transaction leads to the generation and addition of a new block to the blockchain.

A database is used to store the authentication information of the users. Users input their login credentials which are cross-referenced with the data stored in the database. Upon successful verification, users are directed to their designated webpages. There are 2 classes of users - admins and customers. Administrators are further categorized into three distinct roles—marketyard, warehouse, and ration store. Each admin oversees crucial operations such as shipment initiation, approval, rejection, and movement of commodities along the supply chain until the ration store. The pinnacle or end stage of the supply chain are the customers who purchase commodities from the ration store. As customers proceed with their purchases, payments are simultaneously executed from their cryptocurrency wallets to the smart contract balance and thereafter routed to the designated wallet of the ration store admin.

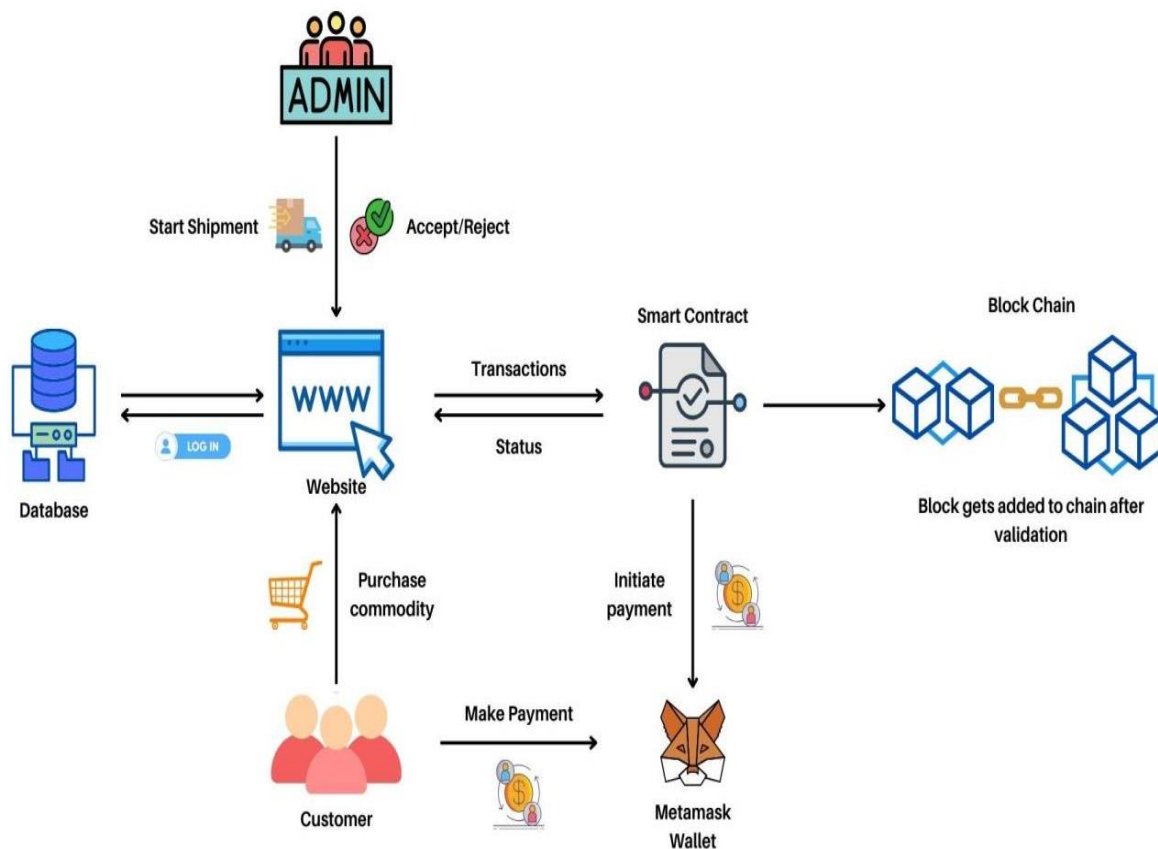


Figure 4.2: System Architecture

C. System Flow Diagram

The entire system can be divided into two major components - Admins initiating and approving/rejecting the shipments & Customer purchasing commodities

1) Administrators Initiate and Approve/Reject Shipments

Fig 4.3 illustrates the shipment flow in the supply chain. The current stage administrator verifies and processes received shipments, entering commodity weights into the system to ensure they match expected quantities. This validation occurs twice: between the market yard and warehouse, and between the warehouse and ration shop. If entered weights match expected quantities, the shipment is approved for further processing and distribution. However, if weights do not match, it raises a discrepancy and prompts further investigation. Such instances result in the shipment being flagged as rejected, indicating it does not meet required standards for approval.

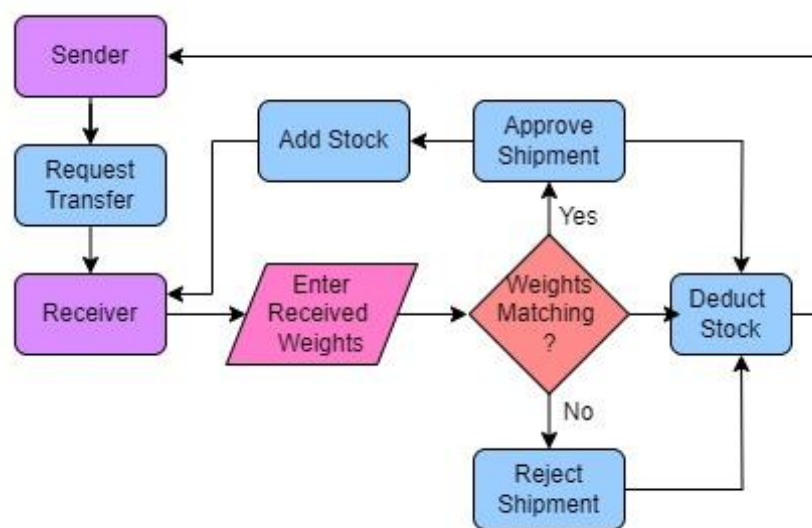


Figure 4.3: Shipment Flow between administrators

2) Customers Purchase Commodities

Fig 4.4 shows the purchase flow. Upon accessing the system, the customer is presented with the option to select their card type, specifying their eligibility for certain commodities. Additionally, they input the number of adults and children in their family, which determines the quantity of commodities they are entitled to receive. Based on the entered information, the system

designates the appropriate commodities for the customer, ensuring compliance with their entitlements. Subsequently, the customer proceeds to make the payment using their cryptocurrency wallet. The payment transaction is executed, simultaneously debiting the required amount from the customer's wallet and crediting it to the ration store's cryptocurrency wallet. This seamless payment process ensures efficient and secure transactions between the customer and the ration store, facilitating the exchange of commodities for cryptocurrency.

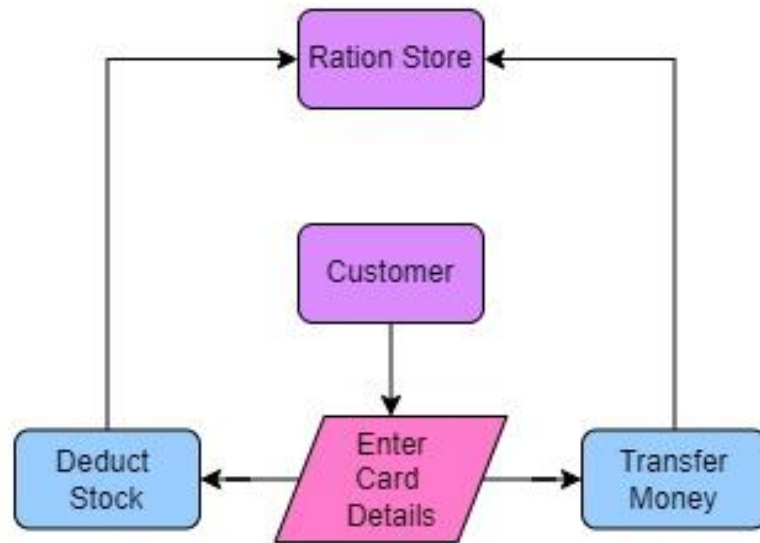


Figure 4.4: Customer Purchase Flow

3) Complete Workflow

Fig 4.5 shows the complete workflow comprising of all entities in the supply chain. The supply chain begins with the farmer depositing stock at the Market yard, initiating the distribution process. Administrators oversee and authorize shipments through various stages until they reach the Ration store. Upon receipt, administrators verify weights, and approved shipments progress. Discrepancies may prompt further investigation or rejection. Customers select card type and family details to determine ration entitlements. The system calculates available commodities, and customers complete transactions for acquisition from the Ration store. This process ensures transparency, efficiency, and accountability.

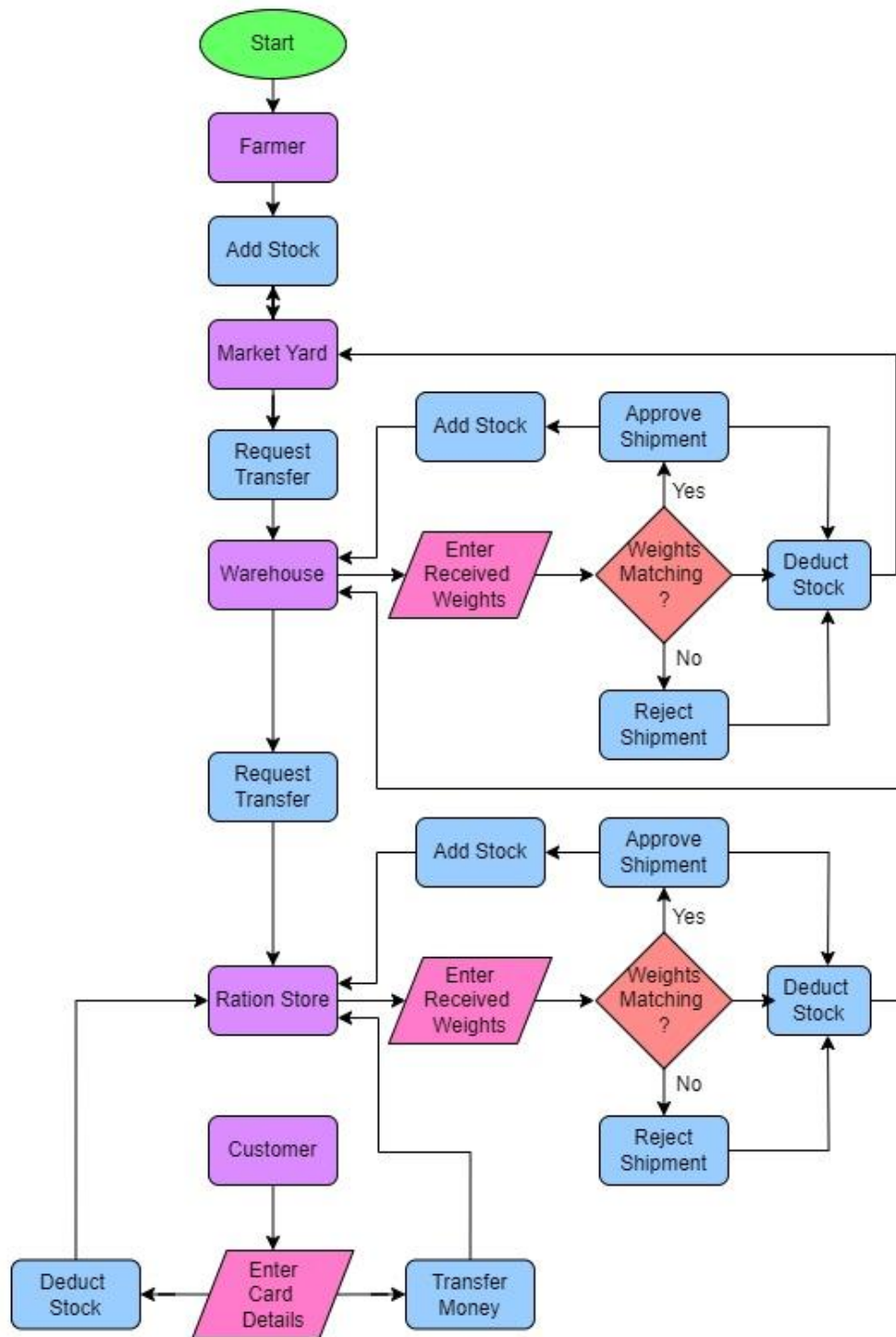


Figure 4.5: Complete Workflow

D. Transaction Functions

Table 4.1 delineates the core transactional functionalities embedded within the proposed blockchain-based supply chain management system along with a brief explanation of their functionality.

Table 4.1: Transaction Functions

<i>Functions</i>	<i>Description</i>
<i>myDeposit</i>	Farmer deposits commodities to Market Yard
<i>requestShipmentToWarehouse</i>	Market Yard admin initiates shipment to Warehouse
<i>approveShipmentToWarehouse</i>	Warehouse admin approves shipment
<i>rejectShipmentToWarehouse</i>	Warehouse admin rejects shipment
<i>requestShipmentToRationShop</i>	Warehouse admin initiates shipment to Ration Shop
<i>approveShipmentToRationShop</i>	Ration shop admin approves shipment
<i>rejectShipmentToRationShop</i>	Ration shop admin rejects shipment
<i>payMoney</i>	Customer pays money to Ration Shop to purchase commodities
<i>reduceStock</i>	Ration Shop stock gets updated on purchase of commodities by customer

E. Algorithms/Pseudo-codes

Algorithm 1, myDeposit function, enables farmers to deposit wheat, rice, and sugar to the market yard. It verifies the caller's identity, increments the transaction ID for a unique identifier, and updates the stock.

```
function: myDeposit (addrF, addrM, w, r, s)
//check identification of farmer
require(msg.sender == addrF)
//generate transaction ID
transactionID++
//update market yard stock
//add to existing stock
updateMStock(w, r, s);
```

Algorithm 2, requestShipmentToRationShop function, allows the warehouse administrator

to request a shipment of wheat, rice, and sugar to the ration shop. It validates the caller's identity, increments the transaction ID for a unique identifier, and updates the stock. The same algorithm is also applicable for market yard to request shipment to warehouse.

```
function: requestShipmentToRationShop (addrW, addrR, w, r, s)
    //check identification of sender
    require(msg.sender == addrW)
    //generate transaction ID
    transactionID++
    //update warehouse stock
    //subtract from existing stock
    updateWStock(w, r, s);
    //update ration shop stock
    //add to existing stock
    updateRStock(w, r, s);
```

Algorithm 3, receiveShipmentRationShop function, allows the ration shop administrator to acknowledge receipt of a shipment containing wheat, rice, and sugar. It verifies the caller's identity, validates received commodity weights, and updates warehouse and ration shop stocks. This process is also applicable for warehouse receipt of shipments from the market yard.

```
function: receiveShipmentRationShop (addrW, addrR, w, r, s)
    //check identification of sender
    require(msg.sender==addrR)
    //get transaction ID
    //check received weights
    if (w, r, s, == received w, r, s)
        approveShipmentToRationShop
    //update warehouse stock
    //subtract from existing stock
    updateWStock(w, r, s);
    //update ration shop stock
    //add to existing stock
    updateRStock(w, r, s);
    if (w, r, s != received w, r, s)
        rejectShipmentToRationShop();
```

Algorithm 4, purchaseCommodities function, manages the purchasing process for customers. It verifies customer identification, calculates commodity allocation based on provided details, and checks ration shop stock availability. The payMoney function initiates payment by transferring funds from the customer's account to the smart contract, including a 5% fee for gas. Funds are then transferred from the contract account to the ration shop to complete payment. After successful payment, stock levels in the ration shop are updated to reflect purchased commodities.

```
function: purchaseCommodities (addrC, addrR, card, ad, ch)
    //check identification of customer
    require(msg.sender == addrC)
    //get card type
    //get family details
```

```

//calculate allocation based on details
alloc= {w, r, s}
//check stock availability in ration shop
require(stock >= alloc)
//initiate payment
//5% extra from customer
payMoney(addrR);
//update stock
reduceStock(w, r, s);

```

4.2 CODES & STANDARDS

A) Regulatory Requirements:

India's Public Distribution System (PDS) is subject to various regulatory requirements aimed at ensuring the fair and transparent distribution of subsidized food grains and essential commodities to eligible beneficiaries. The following regulatory requirements are relevant to the design and implementation of the blockchain-powered supply chain management system:

1) National Food Security Act (NFSA): The NFSA, enacted in 2013, mandates the provision of subsidized food grains to identified priority households through the Targeted Public Distribution System (TPDS). Compliance with the NFSA is essential to ensure that the blockchain system accurately identifies and serves eligible beneficiaries under the TPDS.

2) Food Safety and Standards Act (FSSA): The FSSA sets standards for food safety and quality assurance throughout the food supply chain. The blockchain system must comply with FSSA regulations to ensure the integrity and safety of the distributed commodities, including proper storage, handling, and distribution practices.

3) Data Protection and Privacy Laws: Compliance with data protection and privacy laws, such as the 'Digital Personal Data Protection Act, 2023' and the Information Technology (Reasonable Security Practices and Procedures and Sensitive Personal Data or Information) Rules, is essential to safeguard the personal information of beneficiaries stored and processed within the blockchain system.

B) Blockchain Standards:

The design and implementation of the blockchain-powered supply chain management system adhere to blockchain-specific standards and protocols to ensure compatibility, security, and efficiency. The following blockchain standards and protocols are considered in system development:

1) Ethereum Improvement Proposals (EIPs): Ethereum Improvement Proposals are technical specifications proposed by the Ethereum community to enhance the Ethereum blockchain's functionality and performance. The blockchain system follows relevant EIPs to leverage new features and improvements introduced to the Ethereum network.

2) Ethereum Request for Comment (ERC) Standards: ERC standards define common interfaces and protocols for Ethereum tokens, smart contracts, and decentralized applications (DApps). The blockchain system complies with ERC standards, such as ERC-20 for fungible tokens and ERC-721 for non-fungible tokens, to ensure compatibility and interoperability with Ethereum-based applications and wallets.

3) Decentralized Identity Standards: Standards such as the Decentralized Identifiers (DIDs) and Verifiable Credentials (VCs) specifications are followed to implement decentralized identity solutions within the blockchain system. These standards enable secure and privacy-preserving identity management for stakeholders interacting with the system.

4.3 CONSTRAINTS, ALTERNATIVES AND TRADEOFFS

A) Alternatives Considered:

Alternate Blockchain Platform:

During the planning phase of the project, Various blockchain platforms were explored as potential alternatives to Ethereum for implementing the supply chain management system. Each platform offered distinct features, consensus mechanisms, and architectural designs that were evaluated based on their suitability for the project's requirements. The following alternative blockchain platforms were considered:

1) Hyperledger Fabric:

Hyperledger Fabric is a permissioned blockchain platform developed under the Linux Foundation's Hyperledger project. It offers a modular and customizable architecture, allowing for greater control over consensus mechanisms, membership access, and privacy features.

One of the key advantages of Hyperledger Fabric is its support for permissioned networks, which are well-suited for enterprise applications requiring strict access controls and privacy measures.

Additionally, Hyperledger Fabric provides support for pluggable consensus protocols, enabling flexibility in choosing the consensus mechanism that best aligns with the project's performance and security requirements.

2) Quorum:

Quorum is an Ethereum-based blockchain platform developed by J.P. Morgan, tailored for enterprise use cases such as financial services and supply chain management. It combines the transparency and security of the Ethereum network with enhancements for privacy and scalability.

One of the distinguishing features of Quorum is its support for private transactions using privacy-preserving techniques such as zero-knowledge proofs and private smart contracts. This enables sensitive business transactions to be conducted securely while still benefiting from the transparency of the underlying blockchain.

Quorum also offers integration with enterprise-grade tools and protocols, making it easier for organizations to deploy and manage blockchain solutions within their existing infrastructure.

Evaluation Criteria:

Each alternative blockchain platform was evaluated based on several criteria, including:

- 1) Scalability: Ability to handle a high volume of transactions efficiently and scale to meet future demand.
- 2) Privacy: Support for privacy features such as confidential transactions and data sharing

- controls.
- 3) Developer Ecosystem: Availability of developer tools, documentation, and community support for building and deploying applications.
 - 4) Interoperability: Compatibility with existing systems and standards, enabling seamless integration with external data sources and networks.
 - 5) Regulatory Compliance: Adherence to regulatory requirements and standards relevant to the project's jurisdiction and industry sector.

After a thorough evaluation of these criteria, Ethereum was ultimately chosen as the preferred blockchain platform for the project due to its mature ecosystem, robustness, and broad developer adoption. However, the exploration of alternative platforms provided valuable insights into different architectural approaches and considerations for future projects or enhancements to the system.

5. SCHEDULE, TASKS AND MILESTONES

A) Schedule:

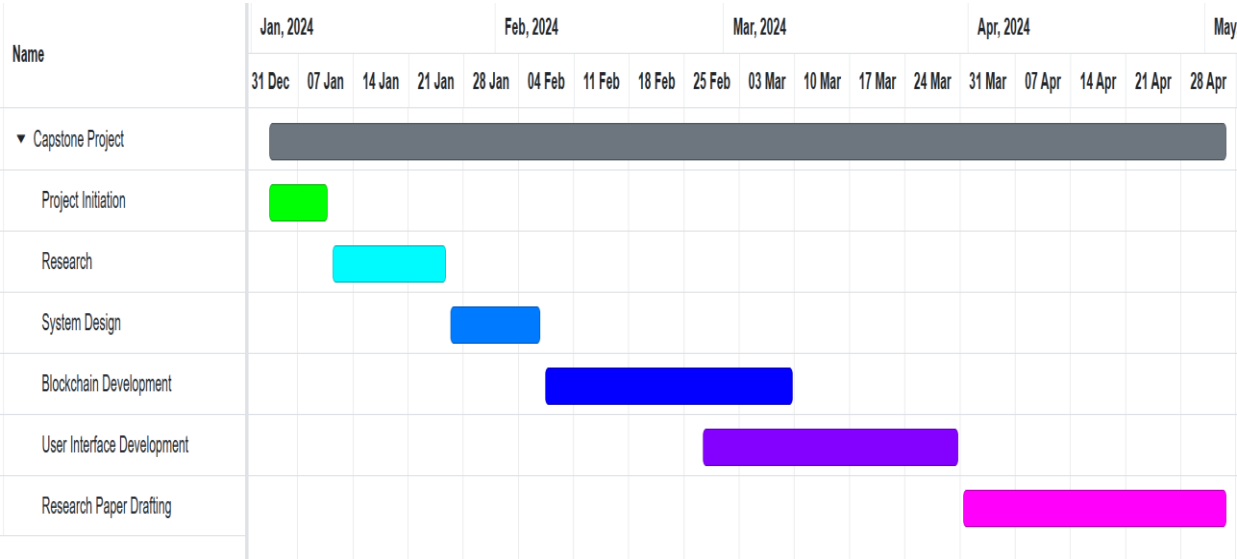


Figure 5.1: Gantt Chart of Project Timeline

January:

- Finalize project topic and objectives
- Research suitable softwares
- Determine required tasks and implementation strategy
- Set initial timeline for project phases

February:

- Smart Contract Development:
 - ◆ Initial contract development on Remix IDE
 - ◆ Contract testing on Ganache personal blockchain
 - ◆ Deploying contract on Sepolia Testnet

March:

- Web dApp Development:
 - ◆ Develop app functionality according to planned requirements
- Research Paper Drafting:
 - ◆ Draft research paper outlining project methodology and findings
 - ◆ Explore future research areas such as iot integration

April:

- App development continuation:
 - ◆ Develop app functionality according to planned requirements
- Work on poster:
 - ◆ Prepare poster summarizing project
- Finalize report:
 - ◆ Compile and finalize project report
 - ◆ Review and edit research paper
 - ◆ Ensure completion of app development and testing

May

- Submission of thesis:
 - ◆ Submit finalized thesis report

B) Individual Tasks/Contribution of Team Members:

Kishorkumar

- Working on the backend and frontend of the project.
- In the backend, the focus is on developing and testing the smart contract to ensure the proper functioning of the various transactions.
- In the frontend, the focus is on invoking the transactions through various elements in the web page.

Zubin James

- Working on the backend and frontend of the project.
- In the backend, the focus is on deploying the smart contract to the Ganache blockchain.
- In the frontend, the focus is on building the user interface and working on the routing within various pages in the project.

Sabyasachi

- Working on the frontend of the project, focusing on the visuals and aesthetics of the web pages.
- Handling research and documentation for the project.

6. PROJECT DEMONSTRATION

Fig. 6.1 shows the home page of the web dApp. The home page serves as the primary landing page for the users. It displays some basic information about the project and team members and also allows users to login.

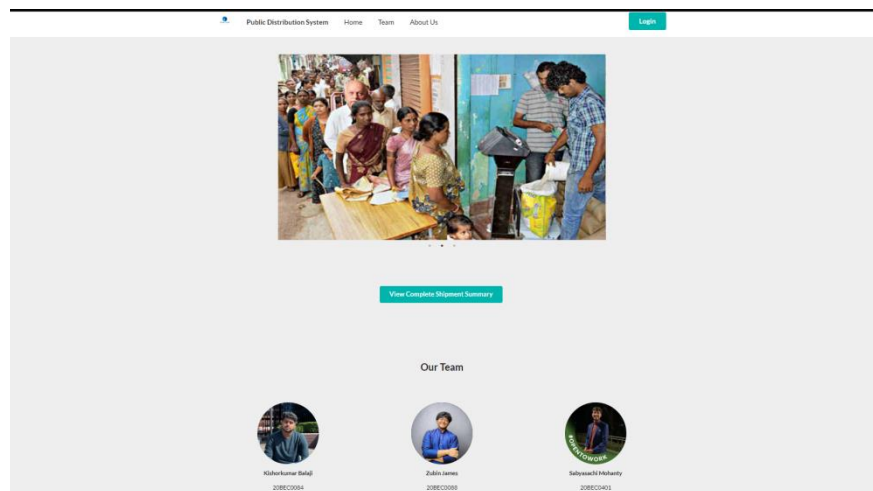


Figure 6.1: Home Page

Fig 6.2 shows the login portal for the users where they can access their respective pages by signing in with their username and passwords. Each user has their own webpage upon login, that contains user specific functions and options.

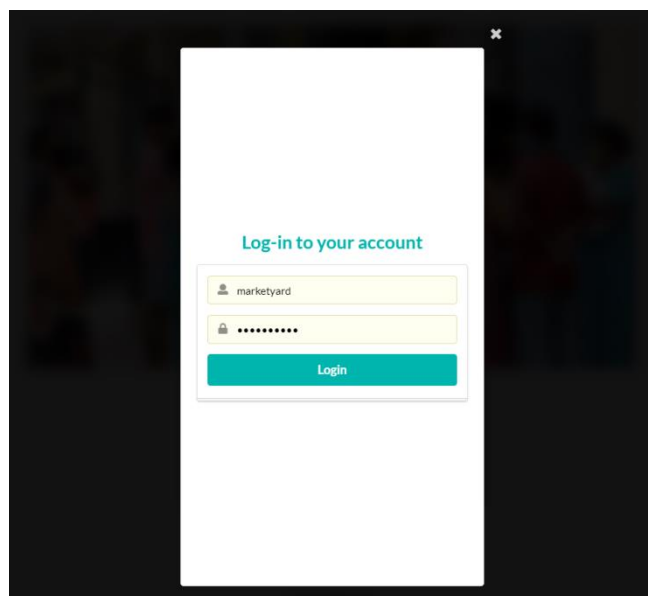


Figure 6.2: Login Portal

Fig. 6.3 shows the homepage for the market yard admin. Admins can view the stock available to them currently. They can enter the weights of the goods to start a shipment and move the commodities along the supply chain. Market yard admin can start a shipment to the warehouse. And the warehouse admin can start a shipment to the ration shop.

The screenshot displays the 'Market Yard Home' interface. At the top, there is a home icon, the title 'Market Yard Home', and a 'Logout' button. The main content area is divided into two primary sections. The left section, titled 'Market Yard Stock', shows current stock levels: WHEAT: 1765, RICE: 1865, and SUGAR: 1645. The right section, titled 'Add Stock to Market Yard', contains input fields for Wheat, Rice, and Sugar, followed by a 'Submit' button. Below these, a 'Transfer to Warehouse' section features input fields for Wheat, Rice, and Sugar, each with the value '50' entered, and a 'Submit' button. At the bottom center, there is a button labeled 'View Complete Shipment Summary'.

Figure 6.3: Marketyard Home

Fig. 6.4 shows the homepage for the warehouse admin. Admins can view all the shipments that are pending approval. They enter the received weights of the shipment they want to approve/reject. If the weights are matching the shipped weights, the shipment is approved otherwise its rejected. Fig. 6.5 shows the confirmation popup. When the received shipment weights are entered, a popup appears to confirm that the values are entered correctly. Fig. 6.6 shows the status popup. When the entered details are confirmed, the shipment status is shown as approved or rejected.

Warehouse Stock

WHEAT: 400

RICE: 400

SUGAR: 400

Pending Approvals (5)

ID: 12

<Wheat: 150>

Rice: 150

Sugar: 150

Enter Received Weights

Wheat

50

Rice

50

Sugar

30

Submit

Figure 6.4: Warehouse Home

Are you sure you want to submit the weights? Wheat: 50, Rice: 50, Sugar: 30

Cancel

OK

Figure 6.5: Confirmation Pop-up

Shipment Status

The shipment has been rejected.


Close

Figure 6.6: Shipment Status Pop-up


Fig. 6.7 shows the customer interface. Customer can view the stocks available in the ration shops. They have to enter the details of the number of adults and children and select their ration card type. Their ration allocation along with prices is displayed to them. They can submit to automatically make the payment from their wallet to the ration shop wallet.

Customer Interface


Stock



Wheat
Stock: 475



Rice
Stock: 465



Sugar
Stock: 468

Card Type
NPHH-S ▾

Number of Adults
2

Number of Children
2

Submit

Selected Card Information

Card Type: NPHH-S

Card Description: Non-Priority House Hold - Sugar Only

Wheat: 5 Kgs per card @ Rs.7.50 per Kg

Rice: Not Applicable

Sugar: 500 gms per head @ Rs.25 per Kg plus 3 Kgs of Sugar

Figure 6.7: Customer Home

Fig 6.8 shows the payment through MetaMask directly from the customer to the ration shop. The payment happens automatically on submission of card type and details by the customer. The money is deducted from the customer's cryptocurrency wallet and is added to the ration shop admin's cryptocurrency wallet. The customer is also charged a nominal extra fee to cover the gas fee required by the blockchain network.

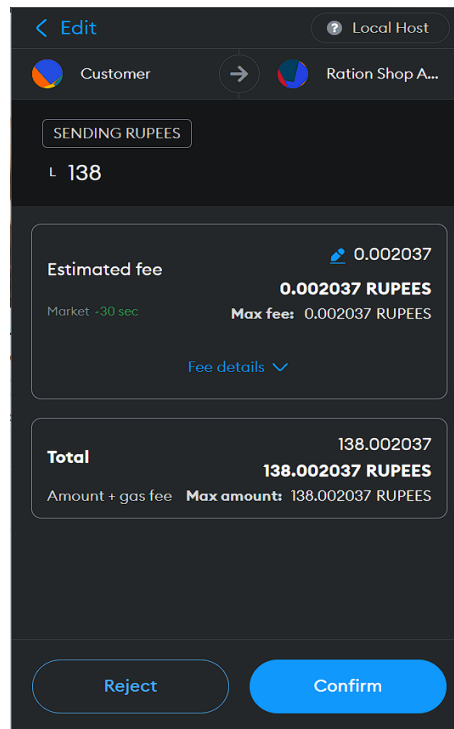


Figure 6.8: Payment through Metamask

Fig. 6.9 shows the shipment summary. This will include details of – Shipment ID, weights of shipped goods, sender and receiver of the shipment, along with the approved/rejected status of the shipment. The approved shipments are highlighted in green and the rejected shipments are highlighted in red to improve readability.

Supply Chain Information

ID	Wheat	Rice	Sugar	From	To	Status
1	10	10	10	Market Yard	Warehouse	Approved
2	10	11	12	Market Yard	Warehouse	Approved
3	4	5	6	Market Yard	Warehouse	Approved
4	17	18	19	Market Yard	Warehouse	Rejected
5	6	7	8	Market Yard	Warehouse	Rejected
6	100	120	110	Market Yard	Warehouse	Approved
7	12	10	11	Warehouse	Ration Shop	Approved
8	3	4	5	Warehouse	Ration Shop	Rejected
9	5	6	7	Warehouse	Ration Shop	Approved
10	11	12	13	Warehouse	Ration Shop	Rejected
11	8	9	10	Warehouse	Ration Shop	Approved
12	4	4	4	Warehouse	Ration Shop	Approved

Figure 6.9: Shipment Summary

Fig. 6.10 shows the content of a transaction made in the web dApp. This specific block shows a transaction of value 138 being called by the dApp. IT has data such as the value, gas used, gas price, gas limit and block number.

CURRENT BLOCK
73

GAS PRICE
2000000000

GAS LIMIT
6721975

HARDFORK
MERGE

NETWORK ID
5777

RPC SERVER
HTTP://127.0.0.1:8545

MINING STATUS
AUTOMINING

WORKSPACE
DRAGONFRUIT

SWITCH

← BACK

TX

0x978aeba2a4610d5fa8743209e5901451f73b6b911a488236fa285ce7c32f6aa1

SENDER ADDRESS

TO ADDRESS

0xe876b48c60bfF64229B0912D22AfA3A0894CA6AE0x83721Bc5BF0E86B628AB304B88a700B76AEE6E79

VALUE TRANSFER

VALUE

GAS USED

GAS PRICE

GAS LIMIT

MINED IN BLOCK

138.00 ETH

21000

96999959800

21000

73

TX DATA

0x

Figure 6.10: Blockchain Transaction

Fig 6.11 shows the contents of a block that gets created for calling the transaction. This block gets mined and is added to the blockchain network. It has the block hash and the transaction hash of the transaction that is mined in that block. It also has other information about the sender and receiver addresses, gas details, etc.

← BACK

BLOCK 73

GAS USED

2100

GAS LIMIT

672197

MINED ON

2024-03-07 20:58:50

BLOCK HASH

0x2c90148129d75c2f55827632f8963acd0744c0b1c04654c9680de28bf32c6e26

TX HASH

0x978aeba2a4610d5fa8743209e5901451f73b6b911a488236fa285ce7c32f6aa1

VALUE TRANSFER

FROM ADDRESS

0xe876b48c60bfF64229B0912D22AfA3A0894CA6AE

TO ADDRESS

0x83721Bc5BF0E86B628AB304B88a700B76AEE6E79

GAS USED

21000

VALUE

13800000000000000000

Figure 6.11: A Block in the Blockchain

Fig. 6.12. shows the new mined blocks being chained to the previous blocks that hold the older transactions.

CURRENT BLOCK	GAS PRICE	GAS LIMIT	HARDFORK	NETWORK ID	RPC SERVER	MINING STATUS	WORKSPACE	SWITCH	⚙
73	20000000000	6721975	MERGE	5777	HTTP://127.0.0.1:8545	AUTOMINING	DRAGONFRUIT		
BLOCK 73	MINED ON 2024-03-07 20:58:53		GAS USED 21000		1 TRANSACTION				
BLOCK 72	MINED ON 2024-03-07 20:57:22		GAS USED 33265		1 TRANSACTION				
BLOCK 71	MINED ON 2024-03-07 20:57:22		GAS USED 38470		1 TRANSACTION				
BLOCK 70	MINED ON 2024-03-07 20:57:08		GAS USED 33265		1 TRANSACTION				
BLOCK 69	MINED ON 2024-03-07 20:57:08		GAS USED 38470		1 TRANSACTION				

Figure 6.12: Consecutive Block in the Blockchain

7. COST ANALYSIS / RESULT & DISCUSSION

A web dApp was developed with different pages and functions for the various entities involved in the supply chain. A smart contract of the proposed system was developed and deployed on the ganache blockchain for testing. After thorough testing, the final smart contract was deployed on the Ethereum Sepolia Testnet for validation and the results obtained are summarized in this section.

A) Transaction Cost

Transaction cost and execution cost are two integral aspects of blockchain transactions, each representing different components of the overall cost incurred during transaction execution. Execution cost specifically pertains to the computational expenses associated with executing the operations specified in the transaction, influenced by factors such as smart contract complexity and computational resources required. On the other hand, transaction cost encompasses the broader spectrum of expenses. The equation below shows the exact composition of transaction cost. Table 7.1 contains the transaction costs for the various functions present in the smart contract.

$$\text{transaction cost} = \text{execution cost} + \text{inherited transaction cost} + \text{call data cost}$$

Table 7.1: Transaction Costs

Transaction	Transaction Cost
myDeposit	40130 gwei
requestShipmentToWarehouse	123467 gwei
approveShipmentToWarehouse	214546 gwei
rejectShipmentToWarehouse	305217 gwei
requestShipmentToRationShop	323494 gwei
approveShipmentToRationShop	219526 gwei
rejectShipmentToRationShop	305217 gwei
payMoney	33265 gwei
reduceStock	38470 gwei

B) Block Confirmation Time

To study the block confirmation times, all transactions in the supply chain as listed in Table 7.1 were called sequentially. Three different gas prices were used – 5 gwei, 3 gwei, and 1 gwei, and the confirmation times for each transaction was measured. Fig 8 shows the findings plotted in a graph.

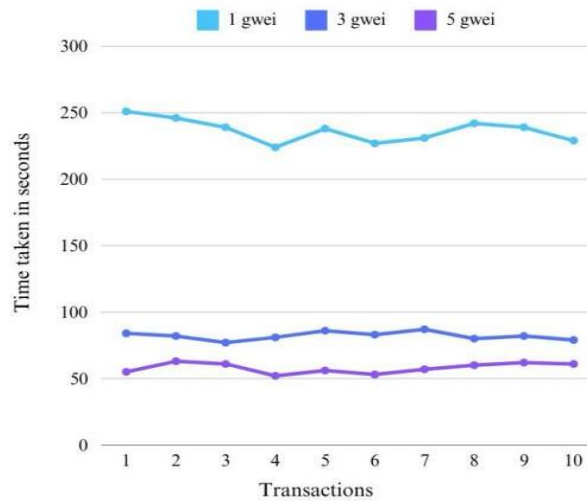


Figure 7.1: Block Confirmation time in Sepolia

Table 7.2 summarizes the findings for the average time taken for block confirmation for different gas prices.

Table 7.2: Block Confirmation time in Sepolia

<i>Gas price</i>	<i>Minimum time taken</i>	<i>Maximum time taken</i>	<i>Mean time taken</i>
5 gwei	52 seconds	63 seconds	58 seconds
3 gwei	77 seconds	87 seconds	82 seconds
1 gwei	224 seconds	251 seconds	236 seconds

This research has explored the transformative potential of blockchain technology in PDS in India. Through practical implementations, the feasibility and effectiveness of integrating blockchain solutions into the PDS has been demonstrated. The proposed system architecture offers a comprehensive framework for leveraging blockchain to streamline supply chain processes. The discussion on transaction costs and block confirmation times sheds light on the practical implications and considerations of implementing blockchain solutions in real-world scenarios. In the future, there are opportunities to explore scalability of the blockchain network and integrate technologies like IoT to automate some of the processes.

8. SOCIAL IMPACT

A) Impact on Beneficiaries

- (i) Improved Access to Subsidized Commodities: By reducing leakages, delays, and instances of corruption, eligible households will have better access to the provisions they are entitled to, thereby improving their food security and overall well-being.
- (ii) Enhanced Accountability and Transparency: Beneficiaries will have increased visibility into the movement of goods from procurement to distribution, empowering them to hold authorities accountable for any discrepancies or irregularities.

B) Socio-Economic Empowerment:

- (i) Empowering Local Suppliers and Farmers: The adoption of blockchain technology facilitates direct procurement from local suppliers and farmers, bypassing middlemen and empowering them economically. This way, small-scale producers receive fair compensation for their produce, thereby enhancing their livelihoods and contributing to rural development.
- (ii) Financial Inclusion and Digital Literacy: The digitization of the distribution system through blockchain technology can promote financial inclusion by encouraging the adoption of digital payment methods among beneficiaries. Efforts to integrate digital literacy programs alongside the implementation of the blockchain system can empower beneficiaries to leverage technology for their financial transactions, thereby enhancing their digital skills and economic resilience.

C) Governance and Institutional Strengthening:

- (i) Streamlined Administration and Oversight: Authorities responsible for monitoring and managing the distribution process will have access to real-time data and analytics, enabling proactive decision-making and resource allocation.
- (ii) Combatting Corruption and Fraud: By leveraging blockchain's tamper-proof ledger, the project can effectively combat corruption and fraudulent activities within the distribution system. The immutability of records will deter unauthorized manipulation of data, thereby ensuring the integrity and trustworthiness of the distribution process.

9. REFERENCES

- [1] Mishra, H. and Maheshwari, P, "Blockchain in Indian public distribution system: a conceptual framework to prevent leakage of the supplies and its enablers and disablers", *Journal of Global Operations and Strategic Sourcing*, Vol. 14 No. 2, pp. 312-335, 2021.
- [2] A. N. Shwetha and C. P. Prabodh, "Blockchain - Bringing Accountability in the Public Distribution System," 2019 4th International Conference on Recent Trends on Electronics, Information, Communication & Technology (RTEICT), Bangalore, India, 2019, pp. 330-335, 2019
- [3] T. S. Iddum, S. Sahu, R. Sujatha and C. V. R. Kumar, "Ethereum blockchain based logistics application for smart supply chain Management," 2022 IEEE Global Conference on Computing, Power and Communication Technologies (GlobConPT), New Delhi, India, 2022, pp. 1-7, 2022.
- [4] S. Madumidha, P. S. Ranjani, S. S. Varsinee and P. S. Sundari. "Transparency and traceability: In food supply chain system using blockchain technology with internet of things," 2019 3rd International Conference on Trends in Electronics and Informatics (ICOEI), Tirunelveli, India, pp. 983-987, 2019.
- [5] Song L, Luo Y, Chang Z, Jin C, Nicolas M. "Blockchain adoption in agricultural supply chain for better sustainability: A game theory perspective", *Sustainability*. 2022; 14(3):1470.
- [6] Feng Tian. "A supply chain traceability system for food safety based on HACCP, blockchain & internet of things," 2017 International Conference on Service Systems and Service Management, Dalian, pp. 1-6, 2017.
- [7] Meyliana, E. Fernando, Surjandy and Marjuky. "The business process of good manufacturing practice based on blockchain technology in the pharmaceutical industry," 2021 Fifth International Conference on Information Retrieval and Knowledge Management (CAMP), Kuala Lumpur, Malaysia, pp. 91-95, 2021.
- [8] Guipeng Zhang, Zhenguo Yang, Wenyin Liu. "Blockchain-based decentralized supply chain system with secure information sharing", *Computers & Industrial Engineering*, Volume 182, 109392, ISSN 0360-8352, 2023
- [9] B. M. El Amine and S. -T. Lamia. (2023) "Blockchain adoption for a sustainable agricultural supply chain: Opportunities and Challenges for the Dairy Industry," 2023 International Conference on Decision Aid Sciences and Applications (DASA), Annaba, Algeria, pp. 331-336, 202

10. APPENDIX

Smart Contract written in Solidity

```
// SPDX-License-Identifier: GPL-3.0
pragma solidity >=0.7.0 <0.9.0;

contract pds {

    address public owner;
    address public marketyardAdmin =0xAe16151AA4c36430d4C4B1Dc2b1326b67dfCa78F;
    address public warehouseAdmin =0xFE2EeC754D21471B40EE8164c8e8afAc34C73806;
    address public rationshopAdmin =0x47ceb78b198B8637EE0578f3eE62258281CB547c;

    struct entity {
        uint256 wheat;
        uint256 rice;
        uint256 sugar;
    }

    struct shipments {
        uint256 id;
        uint256 wheat;
        uint256 rice;
        uint256 sugar;
    }

    struct summary {
        uint256 id;
        uint256 wheat;
        uint256 rice;
        uint256 sugar;
        string from;
        string to;
        string status;
    }

    entity public marketyard;
    entity public warehouse;
    entity public rationshop;

    shipments[] public arrWHapproval;
    shipments[] public arrRSapproval;
    shipments[] public REJbyWH;
    shipments[] public REJbyRS;
    summary[] public Summary;

    uint256 public shipmentID; // Counter for incrementing transfer request IDs
```

```

modifier onlyOwner() {
    require(msg.sender == owner, "Only contract owner can call this function");
    _;
}

modifier onlyMarketyardAdmin() {
    require(msg.sender == marketyardAdmin, "Only marketyard admin can call this function");
    _;
}

modifier onlyWarehouseAdmin() {
    require(msg.sender == warehouseAdmin, "Only warehouse admin can call this function");
    _;
}

modifier onlyRationshopAdmin() {
    require(msg.sender == rationshopAdmin, "Only rationshop admin can call this function");
    _;
}

constructor() {
    owner = msg.sender;
    marketyard = entity(0, 0, 0);
    warehouse = entity(0, 0, 0);
    rationshop = entity(0, 0, 0);
    shipmentID = 0; // Initialize the counter
}

function MYdeposit(uint256 _wheat, uint256 _rice, uint256 _sugar) public
onlyMarketyardAdmin {
    marketyard.wheat += _wheat;
    marketyard.rice += _rice;
    marketyard.sugar += _sugar;
}

function requestTransferToWarehouse(uint256 _wheat, uint256 _rice, uint256 _sugar) public
onlyMarketyardAdmin {
    // Increment the transfer request counter
    shipmentID++;
    // Add the request with the incremented ID
    arrWHapproval.push(shipments(shipmentID, _wheat, _rice, _sugar));
}

function approveTransferToWarehouse(uint256 id) public onlyWarehouseAdmin {
    for (uint256 i = 0; i < arrWHapproval.length; i++) {
        if (arrWHapproval[i].id == id) {
            shipments storage request = arrWHapproval[i];

            marketyard.wheat -= request.wheat;

```



```

        marketyard.rice -= request.rice;
        marketyard.sugar -= request.sugar;

        warehouse.wheat += request.wheat;
        warehouse.rice += request.rice;
        warehouse.sugar += request.sugar;

        Summary.push(summary(id, request.wheat, request.rice, request.sugar,"Market
Yard","Warehouse","Approved"));
        // Remove the request from the array after approval
        removeWHApprovalRequest(i);
        return;
    }
}
revert("Request with given ID not found");
}

function getWHApprovalRequestsCount() public view returns (uint256) {
    return arrWHApproval.length;
}

function removeWHApprovalRequest(uint256 i) internal {
    for (uint256 j = i; j < arrWHApproval.length - 1; j++) {
        arrWHApproval[j] = arrWHApproval[j + 1];
    }
    arrWHApproval.pop();
    return;
}

function rejectTransferToWarehouse(uint256 id) public onlyWarehouseAdmin {
    for (uint256 i = 0; i < arrWHApproval.length; i++) {
        if (arrWHApproval[i].id == id) {
            shipments storage request = arrWHApproval[i];
            // Deduct the rejected shipment from the source inventory
            marketyard.wheat -= request.wheat;
            marketyard.rice -= request.rice;
            marketyard.sugar -= request.sugar;
            // Add the rejected shipment to the REJbyWH array
            REJbyWH.push(request);
            Summary.push(summary(id, request.wheat, request.rice, request.sugar,"Market
Yard","Warehouse","Rejected"));
            // Remove the request from the array after rejection
            removeWHApprovalRequest(i);
            return;
        }
    }
    revert("Request with given ID not found");
}

```

```

function requestTransferToRationShop(uint256 _wheat, uint256 _rice, uint256 _sugar)
public onlyWarehouseAdmin {
    // Increment the transfer request counter
    shipmentID++;
    // Add the request with the incremented ID
    arrRSapproval.push(shipments(shipmentID, _wheat, _rice, _sugar));
}

function approveTransferToRationShop(uint256 id) public onlyRationshopAdmin {
    for (uint256 i = 0; i < arrRSapproval.length; i++) {
        if (arrRSapproval[i].id == id) {
            shipments storage request = arrRSapproval[i];

            warehouse.wheat -= request.wheat;
            warehouse.rice -= request.rice;
            warehouse.sugar -= request.sugar;

            rationshop.wheat += request.wheat;
            rationshop.rice += request.rice;
            rationshop.sugar += request.sugar;

            Summary.push(summary(id, request.wheat, request.rice,
request.sugar, "Warehouse", "Ration Shop", "Approved"));
            // Remove the request from the array after approval
            removeRSApprovalRequest(i);
            return;
        }
    }
    revert("Request with given ID not found");
}

function getRSApprovalRequestsCount() public view returns (uint256) {
    return arrRSapproval.length;
}

function getSummaryCount() public view returns (uint256) {
    return Summary.length;
}

function removeRSApprovalRequest(uint256 i) internal {
    for (uint256 j = i; j < arrRSapproval.length - 1; j++) {
        arrRSapproval[j] = arrRSapproval[j + 1];
    }
    arrRSapproval.pop();
    return;
}

function rejectTransferToRationShop(uint256 id) public onlyRationshopAdmin {
    for (uint256 i = 0; i < arrRSapproval.length; i++) {
        if (arrRSapproval[i].id == id) {

```

```

        shipments.storage.request = arrRSapproval[i];
        // Deduct the rejected shipment from the source inventory
        warehouse.wheat -= request.wheat;
        warehouse.rice -= request.rice;
        warehouse.sugar -= request.sugar;
        // Add the rejected shipment to the REJbyRS array
        REJbyRS.push(request);
        Summary.push(summary(id, request.wheat, request.rice,
request.sugar, "Warehouse", "Ration Shop", "Rejected"));
        // Remove the request from the array after rejection
        removeRSapprovalRequest(i);
        return;
    }
}
revert("Request with given ID not found");
}

function getarrWHapproval() public view returns( shipments[] memory ){
    return arrWHapproval;
}

function getarrRSapproval() public view returns( shipments[] memory ){
    return arrRSapproval;
}

function reduceStock(string memory cardType, uint256 numAdults, uint256 numChildren)
external {
    require(numAdults > 0 || numChildren > 0, "At least one adult or child required");

    uint256 wheatQty = 5;
    uint256 wheatpriceinpaise = 3750;
    uint256 riceQty;
    uint256 sugarQty;
    uint256 sugarpriceinpaise;

    if (keccak256(bytes(cardType)) == keccak256(bytes("PHH"))){
        riceQty = 4 * numAdults + 2 * numChildren;
        sugarQty = (numAdults + numChildren) ;
        sugarpriceinpaise = sugarQty * 1350 ;
    } else if (keccak256(bytes(cardType)) == keccak256(bytes("PHH-AAY"))){
        riceQty = 35;
        sugarQty = (numAdults + numChildren) ;
        sugarpriceinpaise = sugarQty * 1350;
    } else if (keccak256(bytes(cardType)) == keccak256(bytes("NPHH"))){

```

```

    riceQty = 4 * numAdults + 2 * numChildren;
    sugarQty = (numAdults + numChildren) ;
    sugarpriceinpaise = sugarQty * 2500;

} else if (keccak256(bytes(cardType)) == keccak256(bytes("NPHH-S"))) {

    riceQty = 0;
    sugarQty = 3 +(numAdults + numChildren);
    sugarpriceinpaise = (numAdults + numChildren) * 2500;

} else {
    revert("Invalid card type");
}

require((rationshop.wheat ) >= wheatQty, "Insufficient wheat stock");
require((rationshop.rice )>= riceQty, "Insufficient rice stock");
require((rationshop.sugar)>= sugarQty, "Insufficient sugar stock");

rationshop.wheat -= wheatQty;
rationshop.rice -= riceQty;
rationshop.sugar -= sugarQty;
}

}

```