Securing Supply Chain Network with Blockchain

By

Amina Zahid

(2018_UMDB_004840)

Malaika Shabbir

(2018_UMDB_004860)

Shuja Abrar

(2018_UMDB_004896)

Supervised By

Dr. Rabia Riaz

Assistant Professor

DEPARTMENT OF COMPUTER SCIENCES &

INFORMATION TECHNOLOGY

FACULTY OF SCIENCES

UNIVERSITY AZAD JAMMU & KASHMIR

MUZAFFARABAD

Session: 2018-2022

UNDERTAKING

I certify that the research work titled Securing Supply Chain Network with Blockchain is my work. The work has not been presented elsewhere for assessment. Where material has been used from other sources it has been properly acknowledged/referred to.

Sign of Student_____

Amina Zahid

(2018_UMDB_004840)

Sign of Student_____

Malaika Shabbir

(2018_UMDB_004860)

Sign of Student_____

Shuja Abrar

(2018_UMDB_004896)

DECLARATION

We hereby, declare that “No portion of the work referred to in this project has been submitted in support of an application for another degree or qualification of this or any other university/institute or other institution of learning”. It is further declared that this undergraduate project, neither as a whole nor as a part there of has been copied out from any sources, wherever references have been provided.

ABSTRACT

The key challenge explored in this thesis is how blockchain technology may be utilized to improve asset traceability in today's food supply chains. In a business network, blockchain is a shared, distributed ledger that employs encryption to validate and record transactions and monitor assets. The objective is to develop a blockchain model that can be used across all food supply chains and to demonstrate the benefits and drawbacks of doing so. One of the most difficult difficulties that FSC firms confront today is food provenance. End-to-end food tracking is unavoidable in the food sector due to a worldwide supply chain network with diverse operational processes and uneven food standards between nations. This study also includes a thorough evaluation of the present issues in food tracking, food safety standards, and food supply chain design. An analysis of the foodborne disease outbreak dataset in the United States from 2007 to 2021 is also given, with an emphasis on two major findings. First, the blockchain model should be focused on food goods with the most food sickness outbreak occurrences, and second, IoT scanning points should be increased at particular phases of the supply chain that have recorded more than 50% of total contamination episodes. Finally, using Smart Contract, a blockchain model is developed, and its advantages over traditional information technology systems and global food monitoring techniques are discussed. Future studies might look into leveraging blockchain-enabled models to replace ERP systems, minimize food waste, and enhance supply management between stages of the food supply chain.

Keywords: supply-chain, blockchain, privacy, information security, transparency, smart contract

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

I would like to acknowledge and give my warmest thanks to my supervisor Dr. Rabia Riaz, Ex Chair-person of the Department of computer Science and Information Technology who made this work possible. Her guidance and advice carried me through all the stages of writing my project. I would also like to thank my committee members for letting my defense be an enjoyable moment, and for your brilliant comments and suggestions, thanks to you.

I would also like to give special thanks to my Parents and my family as a whole for their continuous support and understanding when undertaking my research and writing my project. Your prayer for me was what sustained me this far.

Finally, I would like to thank God, for letting me through all the difficulties. I have experienced your guidance day by day. You are the one who let me finish my degree. I will keep on trusting you for my future.

CHAPTER 01

INTRODUCTION

Blockchain has increased confidence between dispersed parts of a system by introducing new currencies (such as Bitcoin, Ether, etc.), automatically executed digital contracts (such as smart contracts), and intelligent assets that can be tracked and controlled online. [1] The vast majority of existing blockchain research is focused on creating effective applications for various industries. The field of supply-chain management is one where blockchains are used to store and process records and chain-transaction data, intending to boost trust, transparency, and efficacy while lowering total supply-chain costs.[1]

Because Data integrity is ensured through blockchain and defends information from tampering attempts by linking information in the secure method, several supply-chain systems have been constructed using blockchain technology and associated technology. The majority of academic study has concentrated on utilizing blockchain technology to uphold the security of information traded between various company allies and clients. These academic systems enabled us to effectively detect a variety of assaults including data loss and modification. However, while the present blockchain implementation tries to secure communication among dispersed business components, it fails to identify any danger posed by any additional architectural layer, such as an application. The majority of research in the sector has mainly focused on designing and creating tools that use blockchain to facilitate open sharing by autonomously verifying transactions between participants. These techniques, however, once again fail to detect any attack aimed at company-level agreements or other local dangers.

Several literature articles are available that describe current research on this topic. Studies and assessments of blockchain-based supply-chain solutions currently available, on the other hand, have targeted supply-chains that are data-driven and stakeholder protection, To highlight a few, cross-chain and cross-border supply-chain connectivity and supply-chain compliances with various legislation. This survey will assist the interested reader in sorting through the vast amount of material on the protection of blockchain-based supply-chains that are already available. This

survey conducts a systematic analysis of numerous security vulnerabilities by classifying various assaults in the context of blockchain and after several academic and commercial approaches to solve them are examined. End-to-end supply-chain security is something that interests us significantly, this covers the safety of all parties concerned, their business procedures, and related assets. To secure the process's end-to-end security, we've identified the gaps that the many stakeholders and institutions involved in it must fill.

Overall, the study has been paying attention to using blockchain in supply-chain networks to address numerous challenges and enable trustworthy communication among supply-chain interacting parties and stakeholders. Although the supply-chain is now using blockchain technology, new assaults and risks have evolved that must be considered, particularly those involving actual business and assets.

1.1 Aim and Objectives

Any blockchain system's main issue is the security of communication and computing activities. To identify research gaps and opportunities using a systematic framework, research queries are accordingly generated by utilizing current techniques for computational and communication attacks on supply-chain systems. In this study, the following research issues are primarily examined. How is blockchain technology applied in a supply-chain system? What is it? For a blockchain-based supply-chain system, how may smart contracts be used?

- **Analyze** the main computing and communication threats that jeopardize the security of a blockchain-based supply-chain system, and the protection needs for a supply-chain system.
- **Survey** the technologies that have been developed by the industrial sector to address current computing and communication threats by implementing a safe blockchain system.
- **Analysis** of academic research has been done to address computation and interaction assaults on a supply-chain system based on blockchain.
- **Provide Solutions** to existing research gaps and prospective future areas of interest for academics and business.

- **Design and Implementation** of the key elements—and recommendations—to be taken into account when creating a blockchain-based supply-chain system.

1.2 Thesis Organization

In Section 2, this research outlined our survey approach to this purpose. Before going into the details of our research, Section 3 this research provided a summary of pertinent systems such as supply-chain management, blockchain-based supply-chain systems, blockchain, block mining, and smart contracts. In the second part of segment 3, this research first described the basic model for Blockchain - Supply-Chain Management systems in terms of protection requirements, potential attacks, and attack sources. This threat model allowed us to categorize attacks on Blockchain - Supply-Chain Management systems into the computational and communication categories, and in Section 4, this research examined existing academic and commercial initiatives to tackle them. Last but not least, in the fifth segment, This research discovered research gaps that might aid in the development of blockchain-supply-chain solutions that provide more stringent protections for the supply-chain and its related partners and stakeholders. Finally, this paper brings our research to a close.

CHAPTER 02

LITERATURE REVIEW

Blockchain is a method of storing data that makes it difficult or impossible to alter, hack, or cheat it.

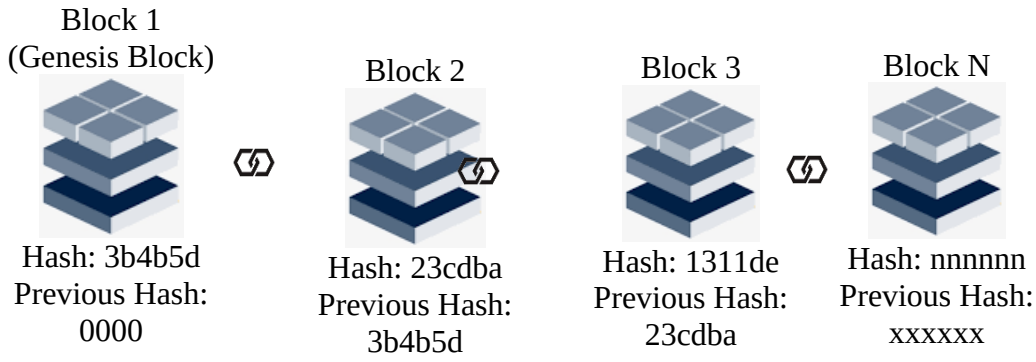

Fig: 2.1 Block Diagram of Blockchain

A blockchain is essentially a **digital ledger** of transactions that is replicated and distributed across the blockchain's whole network of computer systems. Each block in the chain contains a set of transactions, and whenever a new transaction occurs on the blockchain, a record of that transaction is recorded in every participant's ledger. The decentralized database managed by multiple participants is known as **Distributed Ledger Technology (DLT).**

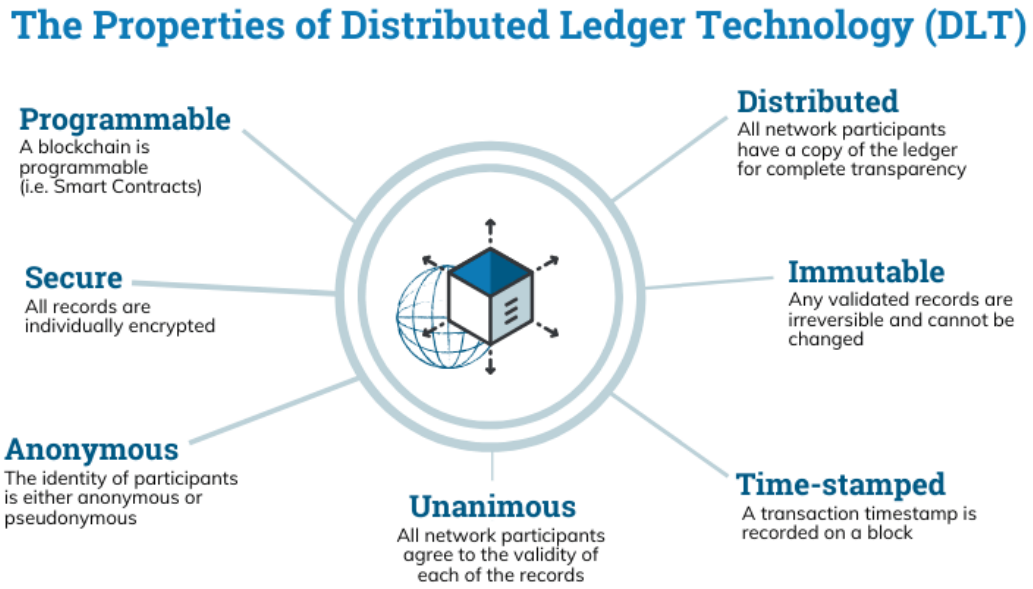

Fig: 2.2 Properties of Distributed Ledger Technology (DLT)

Blockchain is a kind of **distributed ledger technology (DLT)** in which transactions are recorded using an immutable cryptographic signature known as a **hash.**
This means that if one block in one chain was modified, it would be obvious that it had been tampered with. To break a blockchain system, attackers would have to modify every block in the chain, across all distributed versions of the chain. This appears to be impossible because public blockchains are distributed all over the world, and it is impossible to hack or change all the next blocks from all over the world in a very short period.
Blockchains such as **Bitcoin** and **Ethereum** are growing exponentially as blocks are added to the chain, considerably increasing the security of the ledger.

2.1 Web 3.0

When exploring **Web 3.0** technology, there are a few details to consider. For example. First and foremost, the idea is not new. Back in 2006, **Jeffrey Zeldman,** one of the early developers of **Web 1.0** and **Web 2.0** applications, wrote a blog post expressing his support for Web 3.0. However, discussions on this subject began as early as **2001.**

2.1.1 Evolution of the Web 3.0 Technologies

Web 3.0 will emerge from the natural evolution of older-generation web tools combined with cutting-edge technologies such as **AI** and **blockchain**, as well as user interconnection and increased internet usage. **Internet 3.0** appears to be an improvement over its predecessors, **Web 1.0** and **Web 2.0**.

2.1.2 Web 1.0 (1989-2005)

Web 1.0, also known as the **Static Web**, was the first and most functional internet in the **1990s**, while only providing access to limited content with little to no user interaction. Creating user pages or simply commenting on articles were not common practices back then.
Because there were no **algorithms** to sift through internet pages in **Web 1.0**, it was extremely difficult for users to find relevant information. It was like a one-way

street with a narrow footpath where the content was created by a select few and information was mostly sourced from directories.

2.1.3 Web 2.0 (2005-present)

The **Social Web**, or **Web 2.0**, made the internet much more interactive due to advancements in web technologies such as **HTML5**, **CSS** and **javascript,** and others, which enabled startups to build interactive web platforms such as **YouTube**, **Facebook**, **Wikipedia**, and plenty of others.
This paved the way for the growth of social networks and user-generated content production, as data can now be distributed and shared across multiple platforms and applications.
The set of tools in this internet era was pioneered by several web innovators like the aforementioned **Jeffrey Zeldman**.

2.1.4 Web 3.0 (yet to come)

Web 3.0 is the next stage of web evolution that would make the internet more intelligent or process information with near-human-like intelligence through the power of **AI systems** that might run clever algorithms to aid users.
According to **Tim Berners-Lee**, the **Semantic Web** is intended to "**automatically**" interface with systems, people, and home devices. As a result, both humans and machines will be involved in the content creation and decision-making processes. This would allow for the intelligent creation and distribution of highly tailored content directly to each internet user.

2.2 Cryptography

It is a method of developing techniques and protocols to prevent a third party from accessing and acquiring knowledge data from private messages during a communication process.
The term cryptography is derived from two ancient Greek terms: **Kryptos**, which means "**hidden**," and **graphein**, which means "**to write**."

2.2.1 Terminologies

- **Encryption**: It is the transmission of plaintext (normal text) to ciphertext (random sequence of bits).
- **Key:** A small amount of information is required to produce the cryptographic algorithm's output.
- **Decryption:** The inverse process of encryption, converting ciphertext to plaintext.
- **Cipher:** The mathematical function, also known as a cryptographic algorithm, that is used to convert plaintext to ciphertext.

2.2.2 Types of Cryptography

Fundamentally, there are three distinct ways to carry out **cryptographic algorithms.**

2.2.3 Symmetric-Key Cryptography

- We use a **single key** in this encryption method.
- This common key is used for both encryption and decryption.
- Using a common single key raises the issue of securely transferring the key between the sender and the receiver.
- It is also called Secret-Key Cryptography.

2.2.4 Asymmetric-Key Cryptography

- This encryption method uses a **pair of keys**, an **encryption key**, and a **decryption key**, named public key and private key respectively.
- The key pair generated by this algorithm consists of a private key and a unique public key that is generated using the same algorithm.
- It is also called Public-Key Cryptography.

2.2.5 Hash Functions

- It makes no use of keys.

- It is using a cypher to generate a fixed-length cryptographic hash from the plaintext.
- The recovery of plain text contents from ciphertext is nearly impossible.

2.3 Role of Cryptography in Blockchain

Blockchains mainly make use of two types of cryptographic algorithms.

2.3.1 Hash functions

Hash functions are used to provide each participant with the functionality of a single view of the blockchain. Blockchains, in general, use the SHA-256 and RIPEMD-160 hashing algorithms as their hash function.

2.3.2 Why Use Cryptographic Hash Functions

Well, cryptographic hash functions provide the following benefits to the blockchain,

- **Avalanche effect** — i.e., A minor change in the data can result in a noticeably different output.
- **Uniqueness** — i.e., Every input has a different output.
- **Deterministic** — i.e., If the same input is passed through the same hash function, the output will always be the same.
- **Quickness** — The output can be generated in a very short period.
- **Reverse engineering is not possible**, i.e. We can't generate the input if we don't have the output and the hash function.

Furthermore, hash functions play an important role in connecting the blocks and ensuring the integrity of the data stored within each block. Any change to the block data can cause inconsistency and break the blockchain, rendering it **INVALID**. This requirement is met by a property of hash functions known as the **'avalanche effect.'** This feature is what makes blockchain data reliable and secure. Any changes to the block data will result in this difference in the hash value, rendering the blockchain invalid and **immutable**.

2.3.3 Asymmetric-Key Cryptography

It is where a random number algorithm is used to generate the **private key**, and an irreversible algorithm is used to calculate the **public key.**
The asymmetric encryption algorithm has the advantage of having distinct public and private keys that can be transferred over **unsecured networks**.
It is also likely to have several drawbacks, including a slow processing speed and inadequate encryption strength. It is also critical to ensure the security of the asymmetric encryption algorithm during data transmission on the blockchain.

2.3.4 Byzantine Fault Tolerance

The Byzantine Fault Tolerance is the characteristic that defines a system that tolerates the failure class associated with the **Byzantine Generals' Problem**. The most difficult class of failure modes is Byzantine Failure. It implies no constraints and makes no assumptions about the type of behavior that a node can exhibit (e.g. a node can generate any kind of arbitrary data while posing as an honest actor).
Byzantine Faults are the most severe and difficult to resolve. Byzantine Fault Tolerance has been required in airplane engine systems, nuclear power plants, and virtually any system whose actions are dependent on the results of a large number of sensors. Even **SpaceX** was thinking about it as a possible requirement for their systems.
As long as the number of traitors does not exceed one-third of the generals, the algorithm mentioned in the previous section is Byzantine Fault Tolerant. Other solutions exist to make the problem easier to solve, such as using digital signatures or imposing communication restrictions between **network peers**.

2.4 Genesis Block

A genesis block is the first block in a blockchain and is usually hardcoded into its application's software. A blockchain is made up of multiple "blocks" (containing validated transactions and recorded activity data) that are linked together by a **metaphorical chain.**
Each "block" of a crypto asset contains referential data for the previous one and derives its value/legitimacy from it. Thus, the genesis block refers to the first block

(Block 0 or Block 1) of a new blockchain, to which all subsequent blocks are attached.

A genesis block is distinct in that it is the only block in a blockchain that does not refer to a preceding block, and the first mining rewards it unlocks are almost always unspendable.

Genesis blocks are particularly significant because they form the very foundation of a blockchain and frequently contain interesting stories or hidden meanings. For example, Bitcoin's genesis block contains the now-famous message **"The Times 03/Jan/2009 Chancellor on brink of second bailout for banks"** — a reference to the deteriorating financial conditions of the time and the impetus for the creation of cryptocurrencies such as Bitcoin and Ethereum. In 2009, the genesis block of Bitcoin contained 50 BTC.

The Bitcoin genesis block is intriguing not only because of the included message but also because the next block was timestamped nearly six days later (the average time is 10 minutes). The current theory is that the enticing Times headline enticed **Satoshi Nakamoto** to publicly release Bitcoin, but that he actually created the genesis block earlier and changed the timestamp accordingly. Satoshi may have deleted all of the test blocks after testing his software on January 3, 2009, and used the genesis block for his mainnet launch.

2.5 Difference between blockchain and Bitcoin?

Many people wrongly conflate the two.

Although blockchain is the technology that underpins the cryptocurrency Bitcoin, it is not the only version of a blockchain distributed ledger system on the market. Other cryptocurrencies have their blockchain and distributed ledger architectures.

Meanwhile, the decentralization of the technology has resulted in several schisms or forks within the Bitcoin network, resulting in offshoots of the ledger in which some miners use a blockchain with one set of rules and others use a blockchain with another set of rules.

Bitcoin Cash, Bitcoin Gold, and Bitcoin SV, in addition to the original Bitcoin, exist as their cryptocurrency. These cryptocurrency blockchains are more vulnerable to hacking attacks because of their smaller networks, as Bitcoin Gold experienced in 2018.

2.6 Consensus Algorithms in Blockchain

Blockchain, as we know it, is a distributed decentralized network that provides immutability, privacy, security, and transparency. Although there is no central authority to validate and verify transactions, every transaction in the Blockchain is considered to be completely secure and verified. This is only possible due to the presence of the consensus protocol, which is a critical component of any Blockchain network.

A consensus algorithm is a procedure that allows all peers in the Blockchain network to agree on the current state of the distributed ledger. Consensus algorithms achieve reliability in the Blockchain network and build trust between unknown peers in a distributed computing environment in this manner. Essentially, the consensus protocol ensures that every new block added to the Blockchain is the only version of the truth that all nodes in the Blockchain agree on.

The Blockchain consensus protocol has specific goals, such as reaching an agreement, collaboration, cooperation, equal rights for all nodes, and mandatory participation of all nodes in the consensus process. As a result, a consensus algorithm seeks to find a common agreement that benefits the entire network.

Finally, based on their economic stake in the network, a validator is chosen to generate a new block. As a result, PoS encourages validators to reach an agreement through an incentive mechanism.

There are various consensus algorithms out there here are some most popular algorithms and how they work.

2.6.1 Proof of Work (PoW):

This consensus algorithm is used to choose a miner for the next generation of blocks. This PoW consensus algorithm is used by Bitcoin. The main idea behind this algorithm is to solve a complex mathematical puzzle and provide a simple solution. This mathematical puzzle necessitates a significant amount of computational power, so the node that solves the puzzle as soon as possible gets to mine the next block.

2.6.2 Proof of Stake (PoS):

This is the most widely used alternative to PoW. Ethereum's consensus has shifted from PoW to PoS. Instead of investing in expensive hardware to solve a complex puzzle, validators invest in the system's coins by locking up some of their coins as a stake in this type of consensus algorithm. Following that, all validators will begin validating the blocks. If a validator discovers a block that they believe can be added to the chain, they will validate it by placing a bet on it. All validators receive a reward proportionate to their bets based on the actual blocks added to the Blockchain, and their stake increases accordingly.

2.7 Ethereum

Ethereum is a decentralized, open-source blockchain that supports smart contracts. The platform's native cryptocurrency is Ether (ETH or Ξ). In terms of market capitalization, Ether is second only to Bitcoin among cryptocurrencies.

Vitalik Buterin, a programmer, created Ethereum in 2013. Gavin Wood, Charles Hoskinson, Anthony Di Iorio, and Joseph Lubin were also Ethereum founders. The network went live on July 30, 2015, after development work began in 2014 and was crowdfunded. Anyone can use the platform to deploy permanent and immutable decentralized applications with which users can interact. Decentralized finance (Defi) applications offer a wide range of financial services without the need for traditional financial intermediaries such as brokerages, exchanges, or banks, such as allowing cryptocurrency users to borrow against or lend out their holdings for interest. Ethereum also supports the creation and exchange of non-transferable tokens (NFTs), which are non-transferable tokens linked to digital works of art or other real-world items and sold as unique digital property. Furthermore, many other cryptocurrencies run as ERC-20 tokens on top of the Ethereum blockchain and have used the platform for initial coin offerings.

Ethereum has begun implementing a series of upgrades known as Ethereum 2.0, which include a transition to proof of stake and the use of sharding to increase transaction throughput.

2.8 Applications of Ethereum

Ethereum can power several applications offering a wide range of functions:

2.8.1 Currency

If digital currency is accepted as payment, you can use a cryptocurrency wallet to send and receive ether as well as pay for goods and services. Some platforms, such as Coinbase, even allow you to store your coins in a digital wallet, making them less vulnerable to hackers in theory.

2.8.2 Smart contracts

Smart contracts are a type of permission-less app that executes automatically when the contract's conditions are met.

2.8.3 Digital apps, or dapps

Ethereum powers digital apps that allow users to play games, invest, send money, track an investment portfolio, follow social media, and do a variety of other things.

2.8.4 Non-fungible tokens

These tokens, which can be powered by Ethereum, allow artists and others to sell art or other items directly to buyers via smart contracts.

2.8.5 Decentralized finance

Some people may be able to avoid centralized (government) control over the movement of money or other assets by using Ethereum.
Again, it may be more accurate to consider Ethereum as a token that powers various apps rather than a cryptocurrency that allows users to send money to one another.

2.9 Ethereum Virtual Machine (VM)

The physical instantiation of the EVM cannot be described in the same way that a cloud or an ocean wave can, but it does exist as a single entity maintained by thousands of connected computers running an Ethereum client.
The Ethereum protocol exists purely to ensure the continuous, uninterrupted, and immutable operation of this unique state machine; it is the environment in which all Ethereum accounts and smart contracts reside. Ethereum has one and only one 'canonical' state at any given block in the chain, and the EVM is what defines the rules for computing a new valid state from block to block.

2.9.1 Prerequisites

Understanding the EVM requires a basic understanding of computer science terminologies such as bytes, memory, and stack. It would also be advantageous to be familiar with cryptography/blockchain concepts such as hash functions, proof-of-work, and the Merkle tree.

2.9.2 From Ledger to State Machine

The term "distributed ledger" is frequently used to describe blockchains like Bitcoin, which enable a decentralized currency by utilizing fundamental cryptographic tools. Because of the rules that govern what can and cannot be done to modify the ledger, a cryptocurrency behaves like a 'normal' currency. A Bitcoin address, for example, cannot spend more Bitcoin than it has previously received. All transactions on Bitcoin and many other blockchains are governed by these rules. While Ethereum has its native cryptocurrency (Ether), it also enables a much more powerful function: smart contracts. A more sophisticated analogy is required for this more complex feature. Ethereum is a distributed state machine rather than a distributed ledger. The state of Ethereum is a large data structure that contains not only all accounts and balances but also a machine state that can change from block to block according to a predefined set of rules and can execute arbitrary machine code. The EVM defines the specific rules for changing state from block to block.

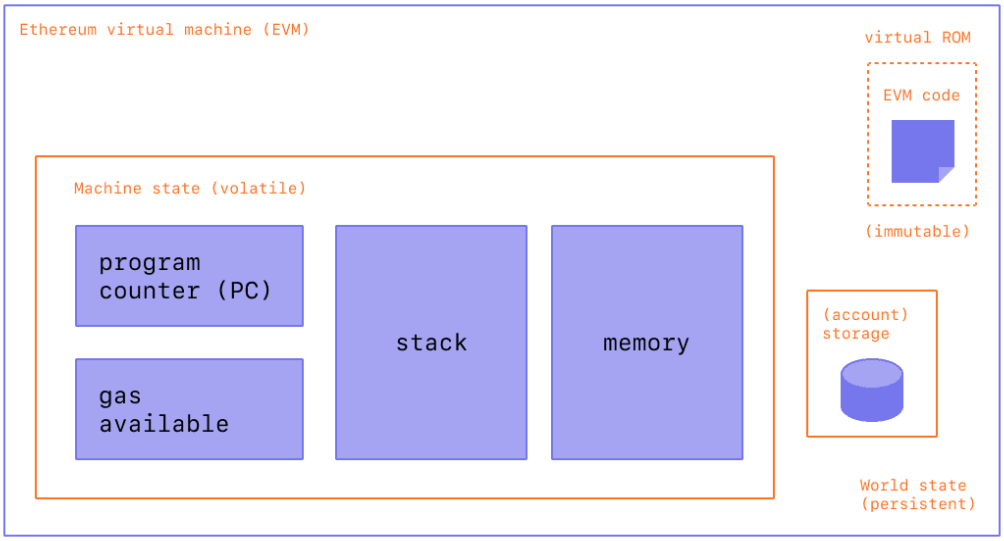

Fig: 2.3 Ethereum (EVM) State Machine

2.9.3 EVM Instructions

The EVM runs as a stack machine with a 1024-item depth. Each item is a 256-bit word that was chosen for its compatibility with 256-bit cryptography (such as Keccak-256 hashes or secp256k1 signatures).

The EVM maintains transient memory (as a word-addressed byte array) during execution, which does not persist between transactions.

Contracts, on the other hand, contain a Merkle Patricia storage trie (as a word-addressable word array), which is associated with the account in question and is part of the global state.

Compiled smart contract bytecode executes as a series of EVM opcodes that perform standard stack operations such as XOR, AND, ADD, SUB, etc. The EVM also implements several blockchain-specific stack operations like ADDRESS, BALANCE, BLOCK HASH, and so on.

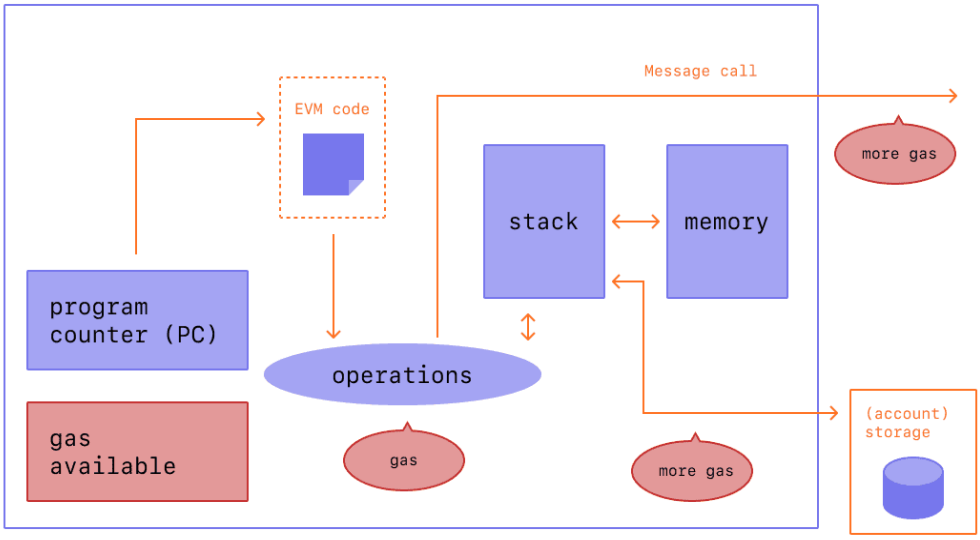

Fig: 2.4 EVM Instructions

2.9.4 EVM Implementations

All EVM implementations must adhere to the specifications outlined in the Ethereum Yellow paper.

The EVM has undergone several revisions over Ethereum's 5-year history, and there are several implementations of the EVM in various programming languages.

An EVM implementation is included in all Ethereum clients. There are also several stand-alone implementations, such as:

- Py-EVM – Python
- evmone - C++
- **ethereumjs**-VM – JavaScript
- **eEVM** - C++
- Hyperledger Burrow – Go
- **hevm –** Haskell

2.10 Smart Contracts

A smart contract is a computer program or a transaction protocol that is designed to execute, control, or document legally relevant events and actions by the terms of a contract or agreement. The goals of smart contracts are to reduce the need for trusted intermediaries, arbitration and enforcement costs, fraud losses, and malicious and

accidental exceptions. Vending machines are mentioned as the oldest piece of technology that can be used to implement smart contracts. The Ethereum white paper from 2014 describes the Bitcoin protocol as a weak version of the smart contract concept as defined by computer scientist, lawyer, and cryptographer Nick Szabo. Scripting languages have been supported by various cryptocurrencies since Bitcoin, allowing for more advanced smart contracts between untrusted parties. Smart contracts should not be confused with smart legal contracts. The latter is a traditional natural language legally binding agreement in which specific terms are expressed and implemented in machine-readable code.

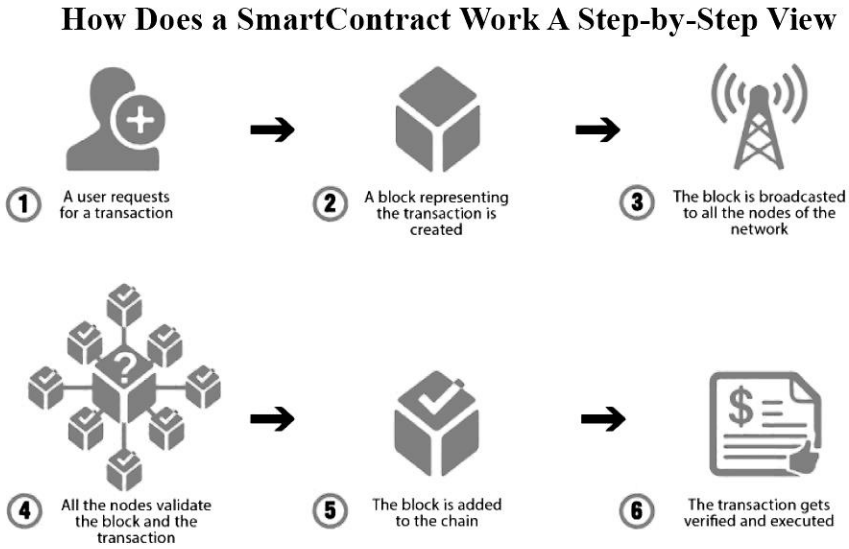

Fig:2.5 Working of SmartContract

2.10.1 Decentralized Applications (DApps)

Decentralized applications (dApps) are digital applications or programs that exist on a blockchain or peer-to-peer (P2P) network of computers rather than a single computer. DApps (also spelled "apps") exist outside of the jurisdiction and control of a single authority. DApps, which are frequently built on the Ethereum platform, can be used for a variety of purposes such as gaming, finance, and social media.

2.10.2 Understanding Decentralized Applications (dApps)

A standard web app, such as Uber or Twitter, runs on a computer system owned and operated by an organization, granting it complete control over the app and its

operations. On one side, there may be multiple users, but the backend is controlled by a single organization.

DApps can run on either a peer-to-peer or a blockchain network. BitTorrent, Tor, and Popcorn Time, for example, are applications that run on computers that are part of a P2P network, in which multiple participants are consuming content, feeding or seeding content, or performing both functions at the same time.

In the context of cryptocurrencies, dApps operate on a blockchain network in a public, open-source, decentralized environment, free of control and interference from any single authority. A developer, for example, can create a Twitter-like dApp and place it on a blockchain where any user can publish messages. No one, not even the app's creators, can delete the messages once they've been posted.

2.10.3 Advantages and Disadvantages of dApps

- **Advantages**

Many of the benefits of dApps revolve around the program's ability to protect user privacy. Users of decentralized apps do not need to submit personal information to use the app's functions. Smart contracts are used by DApps to complete transactions between two anonymous parties without the need for a central authority.

Proponents of free speech point out that dApps can be developed as alternative social media platforms. Because no single participant on the blockchain can delete or block messages from being posted, a decentralized social media platform would be immune to censorship.

Ethereum is a versatile platform for developing new dApps, providing the infrastructure required for developers to concentrate their efforts on discovering novel uses for digital applications. This could allow for the rapid deployment of dApps in a wide range of industries, including banking and finance, gaming, social media, and online shopping.

- **Disadvantages**

Because the use of dApps is still in its early stages, it is experimental and subject to certain problems and unknowns. There are concerns about the applications' ability to scale effectively, particularly if an app requires significant computations and overloads a network, causing network congestion.

Another concern is the ability to create a user-friendly interface. Most users of traditional centralized institutions' apps have an expectation of ease-of-use that encourages them to use and interact with the app. To persuade people to use dApps, developers must create an end-user experience and level of performance that rivals popular and well-established programs.

Another limitation of dApps is the difficulty in modifying code. Once deployed, a dApp will almost certainly require ongoing changes to make enhancements or to correct bugs or security risks. According to Ethereum, developers may find it difficult to make necessary updates to dApps because the data and code published to the blockchain are difficult to modify.

2.11 Solidity Compiler

One of the languages used to create smart contracts is Solidity. The next sections will go into great detail about smart contracts. A Solidity compiler is used to compile Solidity-written code, producing the byte code and other artifacts required for smart contract deployment.

2.11.1 ABI

The default method of interacting with contracts in the Ethereum ecosystem, both from outside the blockchain and for contract-to-contract communication, is the Contract Application Binary Interface (ABI).

2.11.2 Byte Code

The data that our Solidity code is "converted" into is bytecode. It includes binary instructions for the computer. Numeric codes, constants, and other types of information are typically stored in bytecode.

2.12 Distributed peer-to-peer network

A peer-to-peer (P2P) service is a decentralized platform in which two individuals interact directly with each other without the use of a third-party intermediary. Instead, the buyer and seller transact directly with each other via the P2P service.

The P2P platform may offer services such as search, screening, rating, payment processing, and escrow.

2.12.1 Understanding Peer-to-Peer (P2P) Services

The modern peer-to-peer concept was popularised by file-sharing systems, such as the music-sharing application Napster, which debuted in 1999. The peer-to-peer movement enabled millions of internet users to connect directly, form groups, and collaborate to function as user-created search engines, virtual supercomputers, and file systems. This network arrangement differs from the client-server model, in which communication is typical to and from a central server.

2.12.2 Examples of Peer-to-Peer (P2P) Services

- **Open-source Software**

Anyone can view and/or modify the software's code. Open-source software attempts to eliminate the need for a central publisher/editor of software by crowdsourcing software coding, editing, and quality control among writers and users.

- **Filesharing**

Filesharing is the exchange of media and software files between uploaders and downloaders. Filesharing services, in addition to peer-to-peer networking, can provide scanning and security for shared files. They may also provide users with the ability to bypass intellectual property rights anonymously, or they may provide intellectual property enforcement.

- **Online Marketplaces**

Online marketplaces are networks that allow private sellers of goods to find interested buyers. Online marketplaces can provide sellers with promotion services, buyer and seller ratings based on history, payment processing, and escrow services.

- **Cryptocurrency and Blockchain**

The blockchain is a component of cryptocurrency technology. It is a network in which users can make, process, and verify payments without the need for a central currency issuer or clearinghouse. Blockchain technology enables people to conduct business using cryptocurrencies, as well as create and enforce smart contracts.

- **Home sharing**

Home sharing enables property owners to rent out all or a portion of their property to short-term renters. Payment processing, quality assurance, or rating and qualification of owners and renters are common services provided by home-sharing services.

- **Ridesharing**

Ridesharing is a platform that allows car owners to provide chauffeur services to people looking for a taxi ride. Ridesharing platforms provide similar services to home-sharing platforms.

2.13 Gas Price

Gas is the fee, or pricing value, required to complete a transaction or execute a contract on the Ethereum blockchain platform. The gas, which is priced in small fractions of the cryptocurrency ether (ETH), is used to allocate resources of the Ethereum virtual machine (EVM) so that decentralized applications such as smart contracts can self-execute in a secure but decentralized fashion.

The exact price of the gas is determined by supply and demand between network miners, who can refuse to process a transaction if the gas price does not meet their threshold, and network users looking for processing power.

2.14 DAO Attack

The DAO (Decentralized Autonomous Organization) - a program built on the Ethereum Blockchain platform - was breached earlier this year, resulting in the theft of $50 million in Ether.

Only a few weeks after one of the largest crowd funding projects in history, the DAO appeared to be a promising application that contributed to the hype surrounding the Blockchain space. One hacker discovered a flaw in the DAO's code and managed to drain 3.6 million Ether into a personal account, sending the Ethereum community into panic mode, causing the price to plummet and creating a reluctance among the community to invest. The price of Ether has since recovered somewhat, and trust has been restored to a degree, but the attack demonstrated that Blockchain technology is not without flaws.

The attack and subsequent events have altered perceptions of the Ethereum network's security while also shining a light on the "grey area" surrounding cryptocurrency and Blockchain technology in general.

View the EMEA Grid Blockchain Hub's article: How the Ethereum Community Reacted to the DAO Attack to learn how the Ethereum community reacted to the DAO attack. Whether to fork or not to fork.

2.15 Initial Coin Offering (ICO)

The cryptocurrency industry's equivalent of an initial public offering (IPO) is an initial coin offering (ICO) (IPO). An ICO is launched by a company looking to raise funds to create a new coin, app, or service.

Interested investors can participate in the offering in exchange for a new cryptocurrency token issued by the company. This token may have some utility in using the company's product or service, or it may simply represent a stake in the company or project.

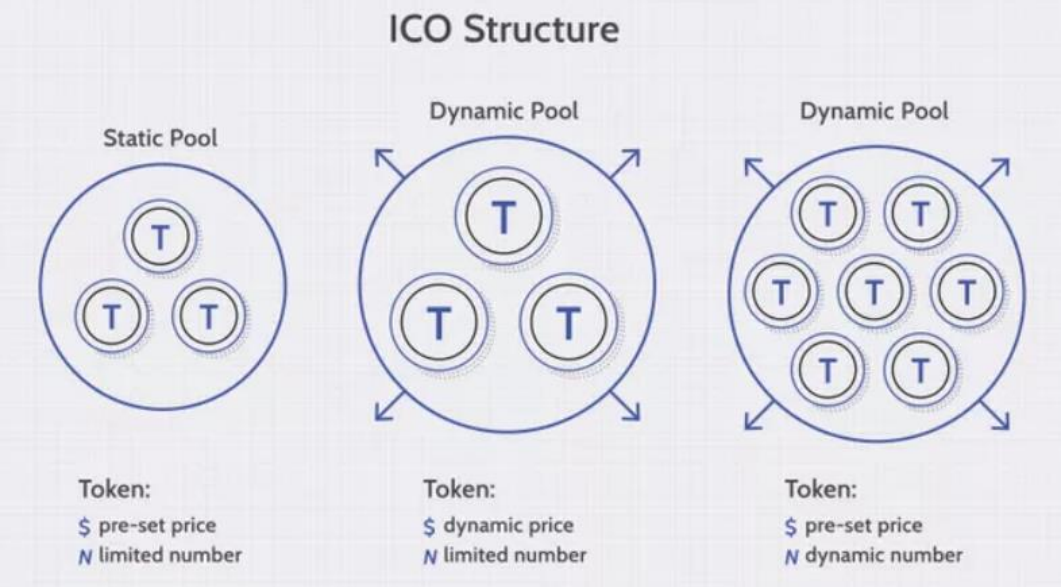

Fig: 2.6 ICO Structure

2.16 MainNet and TestNet

The testnet serves as a testing ground for blockchain initiatives before they are implemented on the mainnet, with the mainnet serving as the actual world.

It doesn't matter whether the mainnet is superior to the testnet or vice versa; it all depends on the application you need.

Testnet is preferable at the developmental stage, while mainnet use begins once the project development is in its completion stage.

2.16.1 Rinkeby Faucet

Before releasing their DApps on the Ethereum mainnet, ETH developers may enhance them with the aid of Rinkeby Faucet.

2.17 Metamask

A software cryptocurrency wallet called MetaMask is used to communicate with the Ethereum network. Users can utilize a browser extension or mobile app to access their Ethereum wallet, which can then be used to connect with decentralized applications.

2.18 Online IDE

A cloud IDE is a web-based integrated development platform (IDE).
An integrated development environment (IDE) is a programming environment that has been packaged as an application and typically includes a code editor, a compiler, a debugger, and a graphical user interface (GUI) builder. Cloud IDEs are frequently not only cloud-based but also designed for the development of cloud apps. Some cloud IDEs, on the other hand, are designed specifically for the development of native apps for smartphones, tablets, and other mobile devices.
The advantages of cloud IDEs include access from anywhere in the world and on any compatible device; minimal-to-no download and installation; and ease of collaboration among geographically dispersed developers.
Because HTML 5 supports browser-based development, it is frequently cited as a key enabler of cloud IDEs. Other important factors include the growing popularity of mobility, cloud computing, and open-source software.
Cloud IDEs include Remix, Cloud9, OxE, Neutron, Orion, and shiftEdit.

2.18.1 Remix

Remix IDE is a free and open-source desktop and web application. It promotes a quick development cycle and includes a large number of plugins with user-friendly

interfaces. Remix is used throughout the contract development process, as well as a playground for learning and teaching Ethereum.

Remix IDE is a component of the Remix Project, which is a platform for plugin-based development tools. It includes sub-projects such as the Remix Plugin Engine, Remix Libs, and, of course, the Remix-IDE.

Remix IDE is a powerful open-source tool for writing Solidity contracts directly from the browser.

It is written in JavaScript and can be used in the browser, but run locally, or on the desktop.

Remix IDE includes modules for testing, debugging, and deploying smart contracts, among other things.

Remix-IDE is available at remix.ethereum.org, and more information can be found in these docs. Our IDE tool is available on GitHub.

here is the interface of the online REMIX IDE.

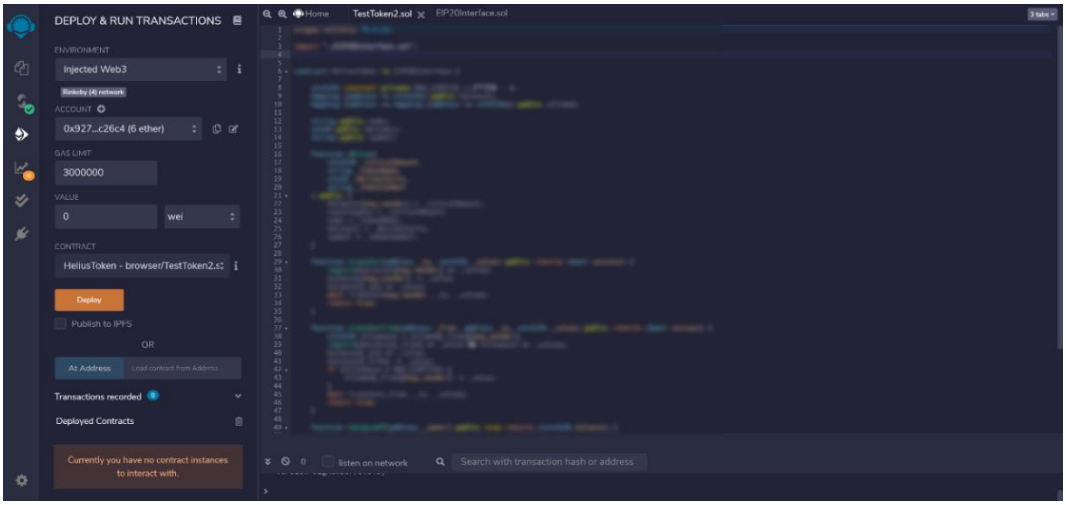

Fig: 2.7 Remix IDE

2.19 Truffle

A world-class development environment, testing framework, and asset pipeline for blockchains based on the Ethereum Virtual Machine (EVM), to make life easier for developers. You get the following with Truffle:

- Built-in smart contract compilation, linking, deployment, and binary management.
- Automated contract testing for rapid development.
- Scriptable, extensible deployment & migrations framework.

- Network management for deploying to any number of public & private networks.
- Package management with EthPM & NPM, using the ERC190 standard.
- Interactive console for direct contract communication.
- Configurable build pipeline with support for tight integration.
- External script runner that executes scripts within a Truffle environment.

2.20 Ganache

Ganache is a personal blockchain that allows for the rapid development of Ethereum and Corda distributed applications. Ganache can be used throughout the development cycle, allowing you to develop, deploy, and test your dApps in a secure and deterministic environment.

Ganache is available in two flavors: UI and CLI. Ganache UI is a desktop application that works with both Ethereum and Corda. For Ethereum development, the command-line tool ganache-cli (formerly known as the TestRPC) is now available. Do you prefer command-line interfaces? This documentation will only cover the Ganache UI flavor. For command-line documentation, please see the Ganache CLI Readme.

All versions of Ganache are available for Windows, Mac, and Linux.

Here is a screenshot of the GUI Version of Ganache

When Ganache starts, the Ganache screen will appear as shown below −

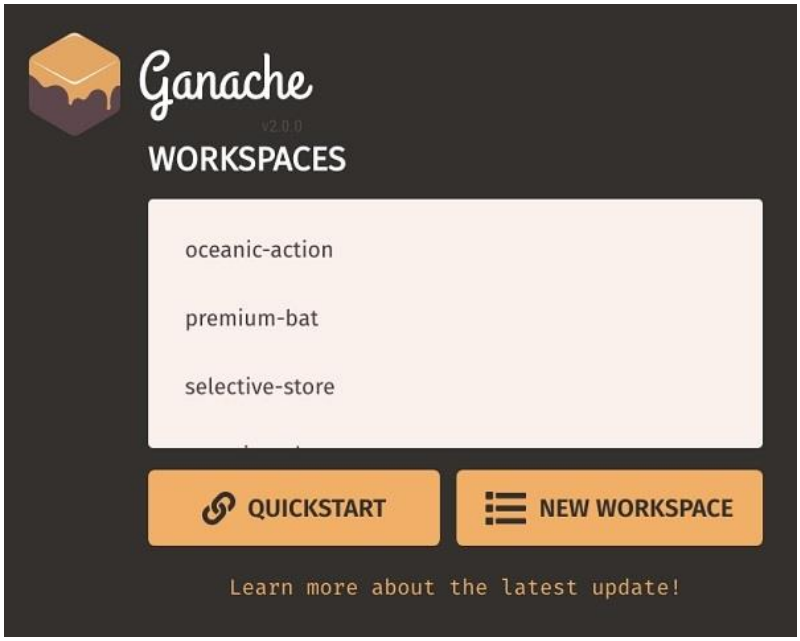

Fig: 2.8 Ganache Interface (GUI)

Click **QUICKSTART** to start Ganache. You will see the Ganache console as shown below –

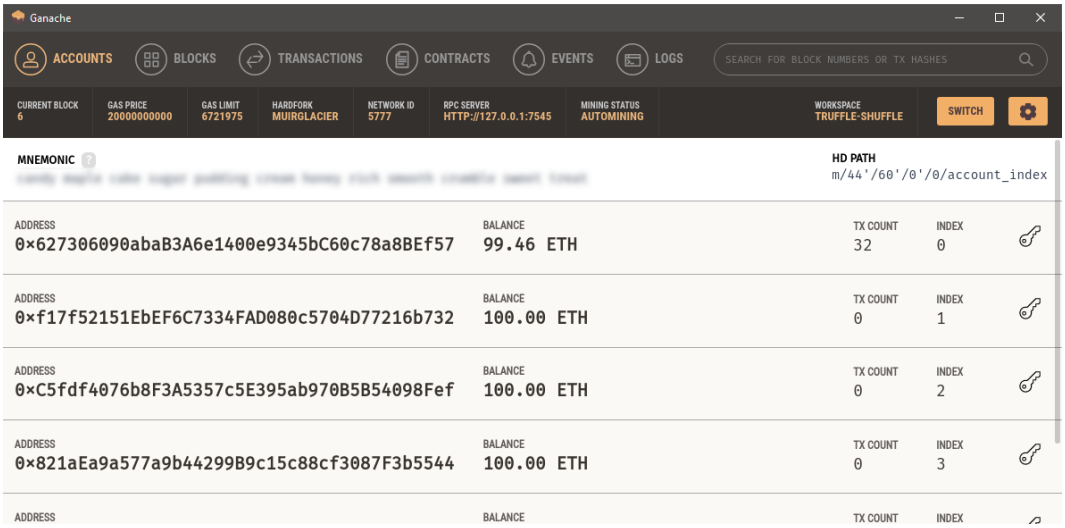

Fig: 2.9 Ganache Interface (GUI)

2.21 Hyperledger

Hyperledger is a global enterprise blockchain project that provides the framework, standards, guidelines, and tools needed to create open-source blockchains and related applications for use in a variety of industries. Among Hyperledger's projects are several enterprise-ready permissioned blockchain platforms in which network participants are known to one another and thus have an inherent interest in participating in the consensus-making process.

Using the Hyperledger components, a company can implement various modular blockchain solutions and services to significantly improve the performance of their operations and the efficiency of their business processes.

2.22 Key Takeaways

Hyperledger is an open-source community dedicated to the creation of a stable framework, tools, and libraries for permissioned, enterprise-grade blockchain deployments.

It is a global collaboration hosted by The Linux Foundation that includes member organizations from the finance, banking, Internet of Things, supply chains, manufacturing, and technology industries.

Several sub-projects exist, including Hyperledger Fabric, Sawtooth, Composer, and Cello.

2.23 Understanding Hyperledger

The Linux Foundation, based in San Francisco, California, founded the Hyperledger project in December 2015. It began with 30 member firms and has since grown to more than 120 member firms.

Hyperledger was founded to accelerate industry-wide collaboration in the development of high-performance and dependable blockchain and distributed ledger-based technology frameworks that could be used across multiple industry sectors to improve the efficiency, performance, and transactions of various business processes.

Hyperledger is a global collaboration of leading companies in finance, banking, the Internet of Things (IoT), supply chain management, manufacturing and production, and technology. They include well-known companies such as Bosch, Daimler, IBM, Samsung, Microsoft, Hitachi, American Express, JP Morgan, and Visa, as well as a slew of blockchain-based startups such as Blockforce and ConsenSys.

2.24 Hyperledger's Organizational Structure

In essence, Hyperledger is not a company, a cryptocurrency network, or a blockchain system. It does not support cryptocurrencies such as bitcoin, but it functions by providing the infrastructure and standards required for the development of various blockchain-based systems and applications for industrial use. Consider Hyperledger to be a hub for various individual blockchain-based projects and tools that adhere to its defined design philosophy.

Various projects include the following:

Hyperledger Fabric is a platform for building various blockchain-based products, solutions, and applications for business use. A now-defunct layer called Hyperledger Composer has since been merged with Fabric as well.

Hyperledger Cello allows blockchain to be used through an on-demand “as-a-service” deployment model (Blockchain-as-a-Service).

Hyperledger Explorer is a dashboard utility that allows for the monitoring, searching and maintenance of blockchain developments and related data.

Hyperledger Burrow is a permissioned Ethereum smart-contract blockchain node that handles transactions and executes smart contract code on the Ethereum Virtual Machine (EVM).

Hyperledger Sawtooth is an enterprise-level, permissioned, modular blockchain platform that uses an innovative Proof of Elapsed Time consensus algorithm.

Hyperledger Caliper is a blockchain benchmark tool that is used to evaluate the performance of a specific blockchain implementation.

All such projects under the Hyperledger umbrella follow the design methodology that supports a modular and extensible approach, interoperability, and security features. The projects remain agnostic to a particular token or cryptocurrency, though a user can create one as required.

2.25 Hyperledger Technology Layers

In terms of architecture, Hyperledger uses the following key business components:

The consensus layer takes care of creating an agreement on the order and confirming the correctness of the set of transactions that constitute a block.

The smart contract layer is responsible for processing transaction requests and authorizing only valid transactions.

The communication layer takes care of peer-to-peer message transport.

Identity management services are a necessary function for maintaining and validating the identities of users and systems and establishing trust in the blockchain.

The API, or application programming interface, enables external applications and clients to interface with the blockchain.

2.26 React

React (also known as **React.js** or **ReactJS**) is a free and open-source Front-end JavaScript library for designing UI components-based user interfaces. Meta (formerly Facebook) and a community of individual developers and corporations manage it.

2.26.1 Uses of React

React can be used to build single-page or mobile apps as a foundation. React, on the other hand, is only concerned with state management and rendering that information to the DOM (Document Object Model), hence constructing react applications usually necessitates the usage of extra libraries for routing and client-side functionality.

2.26.2 What is Babel?

Babel is a JavaScript compiler capable of converting markup and programming languages into JavaScript.
With Babel, you can use the newest features of JavaScript (ES6 - ECMAScript 2015).
Babel is available for different conversions. React uses Babel to convert JSX into JavaScript.
Please note that <script type="text/babel"> is needed for using Babel.

2.26.3 JSX?

JSX stands for **J**ava**S**cript **X**ML.
JSX is an XML/HTML like extension to JavaScript.

2.26.4 JSX Expressions

In JSX, you can use expressions by wrapping them in curly brackets.

2.26.5 React DOM Render

To render (display) HTML elements, use the ReactDom.render() method.

2.26.6 React Elements

The majority of React applications are built around a single HTML element.
This is typically referred to as the root node (root element) by React developers:

React elements look like this:

const element = <h1>Hello Blockchain World!</h1>

Elements are rendered (displayed) with the ReactDOM.render() method:

ReactDOM.render(element, document.getElementById('root'));

React elements can't be changed. They are unchangeable. A React element can only be changed by rendering a new one each time.

2.26.7 ReactJS Components

Components are the basic building blocks of React code. The React DOM library allows you to render components to a specific element in the DOM. When rendering a component, values known as "props" can be passed in:

ReactDOM.render(<Greeter greeting="Hello Blockchain!" />, document.getElementById('myDApp'));

Function components and class-based components are the two most common ways to declare components in React.

2.26.8 Types of Components:

In React, we mainly have **two** types of components:

- **Functional Components:**

JavaScript functions are used to create functional components. A JavaScript function can be used to construct a functional component in React. Data may or may not be passed as parameters to these functions. In React, the following example displays a valid functional component:

```
const DemoComponent=()=>
{
return <h1>Welcome Message!</h1>;}
```

- **Class Components:**

The functional components are a little more complicated than the class components. The functional components of your application are unaware of the other components in your application, whereas the class components can collaborate. We can send information from one class component to another. In React, we can use JavaScript ES6 classes to make class-based components. In React, the following example displays a valid class-based component:

```
class DemoComponent extends React.Component
{
render(){
return <h1>Welcome Message!</h1>;}
}
```

2.26.9 Create React Application

Facebook has built a Create React Application tool that includes everything you'll need to get started with React.

It is a development server that uses Webpack to compile React, JSX, and ES6, auto-prefix CSS files.

The Create React App uses ESLint to test and warn about mistakes in the code.

To create a Create React App run the following code on your terminal:

Example

```
npx create-react-app react-tutorial
```

2.26.10 React hooks:

Hooks are functions that allow developers to use function components to "hook into" React state and lifecycle features. Hooks don't work within classes; instead, they allow you to use React without them.

There are a few built-in hooks in React.

like useState, useContext, useReducer , useMemo, and useEffect.

Others are documented in the Hooks API Reference. useState, useReducer, and useEffect, which are the most used, are for controlling state and side effects respectively.

- **Rules of hooks:**

Hooks have rules that explain the distinctive code pattern that hooks rely on. It's the most up-to-date approach to working with the state in React.

- Hooks should only be used at the very top of the hierarchy (not inside loops or if statements).
- Hooks should only be called from React function components, not from other types of functions or classes.

Although these rules can't be enforced at runtime, code analysis tools such as linters can be configured to detect many mistakes during development. The rules apply to both usage of hooks and the implementation of custom hooks, which may call other hooks.

2.26.11 What are props?

Props are used to communicate data between React components and are short for properties. The data flow between components in React is one-way (from parent to child only).

- **How do you pass data with props?**

Here is an example of how data can be passed by using props:

```
class ParentComponent extends Component {
   render() {
      return (
         <ChildComponent name="First Child" />
      );
   }
}

const ChildComponent = (props) => {
   return <p>{props.name}</p>;
};
```

To begin, we must define/get some data from the parent component and apply it to the "prop" attribute of a child component.

```
<ChildComponent name="First Child" />
```

Here, "Name" is a defined prop that holds text data. Then, just like passing an argument to a method, we can pass data with props:

```
const ChildComponent = (props) => {
  // statements
};
```

And finally, we use dot notation to access the prop data and render it:

```
return <p>{props.name}</p>;
```

2.26.12 What is State?

Another built-in object in React is called state, which allows components to build and maintain their data. Components cannot send data with state, unlike props, but they can create and maintain it themselves.

Here is an example showing how to use state:

```
class Test extends React.Component {
   constructor() {
      this.state = {
         id: 1,
         name: "test"
      };
   }

   render() {
      return (
         <div>
           <p>{this.state.id}</p>
           <p>{this.state.name}</p>
         </div>
      );
   }
}
```

- **How do you update a component's state?**

The state should not be modified directly, but it can be modified with a special method called setState( ).

```
this.state.id = "2020"; // wrong

this.setState({        // correct
   id: "2020"
});
```

- **What happens when the state changes?**

A state change occurs as a result of user input, triggering an event, and so on. React components (with the state) are also rendered using the data stored in the state. The initial data is stored in the state.

When the state of a component changes, React is notified and re-renders the DOM — but just the component with the new state. One of the reasons React is so quick is because of this.

And how does React get notified? With setState( ), you guessed it. The re-rendering procedure for the changed portions is triggered by the setState() method. React receives the information, determines which parts of the DOM to alter, and makes the changes rapidly without re-rendering the entire DOM.

In conclusion, there are two essential components to remember when using state:

- The state shouldn't be modified directly – the setState( ) should be used
- Because the state has an impact on the performance of your application, it should not be used needlessly.
- Can I use state in every Component?

In the early days, the state could only be used in **class components**, not in functional components.

As a result, stateless components were also known as functional components. State, on the other hand, can now be used in both class and functional components thanks to the introduction of React Hooks.

If your project is not using React Hooks, then you can only use state in class components.

2.26.13 What are the differences between props and state?

The main differences between props and state:

- Props allow components to receive data from the outside world, whereas the state allows them to produce and maintain their data.
- Props are used to pass data around, whereas the state is used to keep track of it.
- Data from props is read-only, and cannot be modified by a component that is receiving it from outside

- State data is editable by its component, but it is kept secret (cannot be accessed from outside)
- Props can only be passed from parent to child component (unidirectional flow)
- Modifying state should happen with the setState ( ) method

Every front-end developer should be familiar with React.js, one of the most frequently used JavaScript libraries today.

CHAPTER 03

METHODOLOGY

This research methodically conducted a study, selecting the most important research findings in the field of BC-SCM[2]. According to Kitchenham's standards, this paper retrieved material and summarized the available literature. Our method is based on a systematic literature review, which begins with the download of relevant research papers, reports, theses, and business tools that use keyword-based query-searching as shown in **Table 3.1**. At the outset of the research, this paper discovered that businesses helped define the significance of blockchain in supply-chain management and developing blockchain-based products. As a result, this research divided our survey into two sections: industry activities and academic endeavors. The development of blockchain-based supply-chain management systems is a particular focus of industrial activities. In contrast, academic efforts have led to the publication of research literature focusing on blockchain technology research for supply-chain operations. [3].

Table: 3.1 Search Keywords

Blockchain AND	banking OR supply-chain finance
	Agriculture, food, health care, or medical
	Regulation or Compliance?
	(Computation or communication), risks
	(Security or privacy), risks
	(Computation OR communication) AND risks
	"Automotive industry" or "manufacturing industry
	("smart contracts," "cryptocurrency," or "transaction"), risks
	business control or "process control,"
	risk factors for security analysis or confirmation

To acquire a comprehensive view of the present literature, this paper examined research that was published in five interdisciplinary digital bibliographic databases, including Springer, Elsevier, ACM, IEEE, and Science Direct. Since blockchain technology has only just gained popularity[4], it has only recently begun to be

utilized for the supply chain management. As a result, this research looked at papers published between 2010 and 2020. Although this research discovered over 30,000 blockchain-related papers, this research chose the top 150 in Table 2.1 to satisfy our search criteria. The study questions and associated abstracts were used to critically assess the downloaded publications, and half of the papers were eliminated during this stage. Twenty additional publications are removed through Snowball sampling and full-text screening. On the references of the remaining papers, the backward and forward selection is undertaken, resulting in the identification of 7 new suitable papers. This study includes 29 online reports and industrial tools in addition to research papers and theses. From 2010 through 2020, **Fig 3.1** depicts the total number of research examined each year.

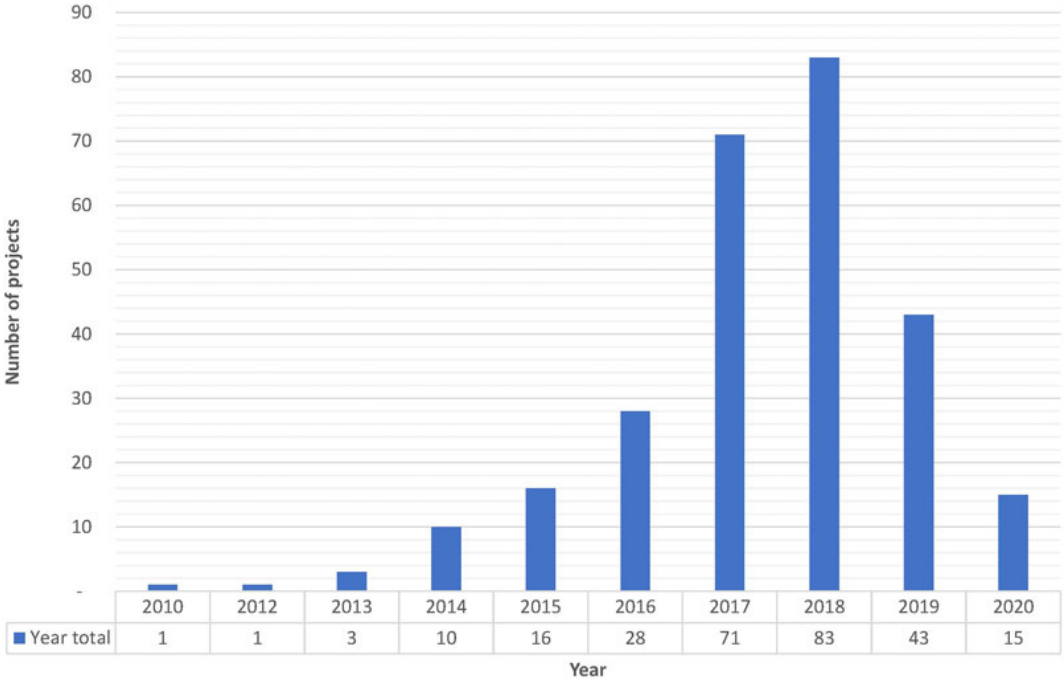

Fig: 3.1 Statistical data for the works cited [5].

3.1 MRO Supply Chain Methodology

The purpose in any supply chain is to ensure that right product is produced at the right time and is at proper flow in the chain. It is highly dependent on the external factors like mode of transportation and internal factors like availability of inventories to produce and meet the required demands. These chains consist of exorbitant amount of flow information, product and money.

- Components of supply chain management are complex but based on:

- Planning of Inventory i.e. from which part of manufacturing hub, it will be easy to procure the required product, etc.
- Development in making relation with supplier of raw material.
- Make i.e. manufacturing and measuring the quality of product.
- Delivering final product to required party.
- Run i.e. finally testing or running the product as required.

There are instances when an aircraft is required service, part replacement, or sometime overhauling where it as a whole may be used for performance testing. Aircrafts move from one place to another for getting serviced as that specific service could be present at some particular place hub. This ensures the quality and performance marking and if there is any error of fault, then it is either repaired or replaced.

CHAPTER 04

DESIGN AND IMPLEMENTATION

Pakistan has been severely impacted by the coronavirus epidemic. The issue is not improved by the failing healthcare system or the lack of vaccines. Although vaccinations are the sole effective defense against this dangerous illness, people have been having difficulty getting their shots due to downed servers, a lack of supplies, and black-market trade. A blockchain-based vaccination system called Health Chain aims to reach people at all points along the vaccine chain, from the manufacturer to the general population.

To meet the needs and requirements of everyone, Health Chain addresses all points along the vaccine supply chain, from consumers to vaccine producers. From a user's standpoint, the platform has the following characteristics:

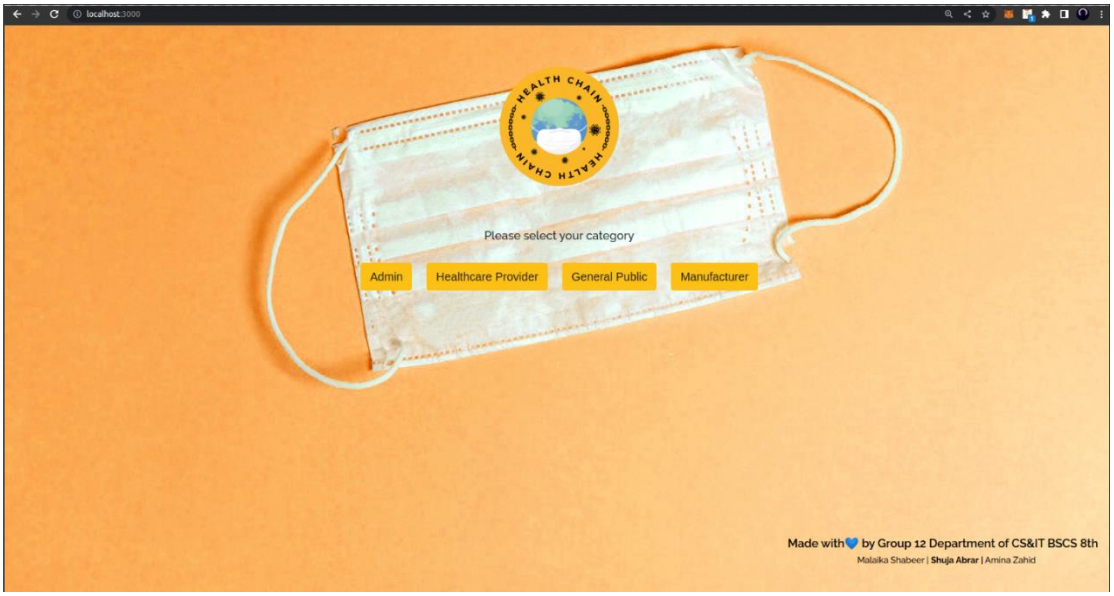

Fig: 4.1 Home page

Citizens

- View all available healthcare providers' immunization costs and NABH registration.
- The quantity and state of the vaccination supply

- Using CELO, make an appointment and pay the hospital directly at a reasonable fee.
- Observe dosages and schedule follow-up sessions

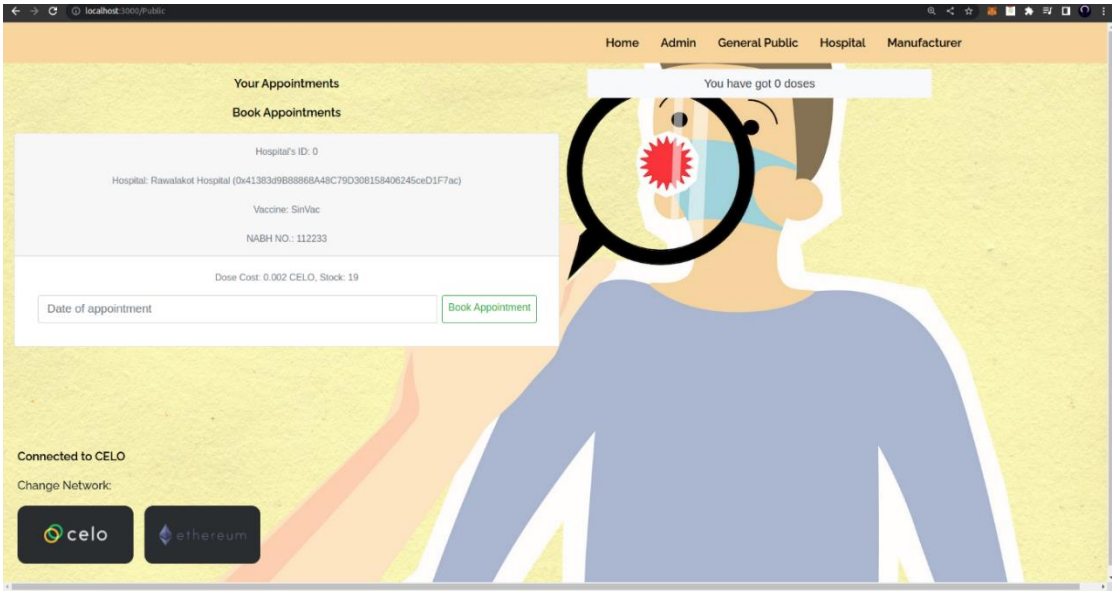

Fig: 4.2 General Public

Admin

- a brief review of the population's vaccination rate
- Before vaccinating the people, check the certifications of the healthcare providers.

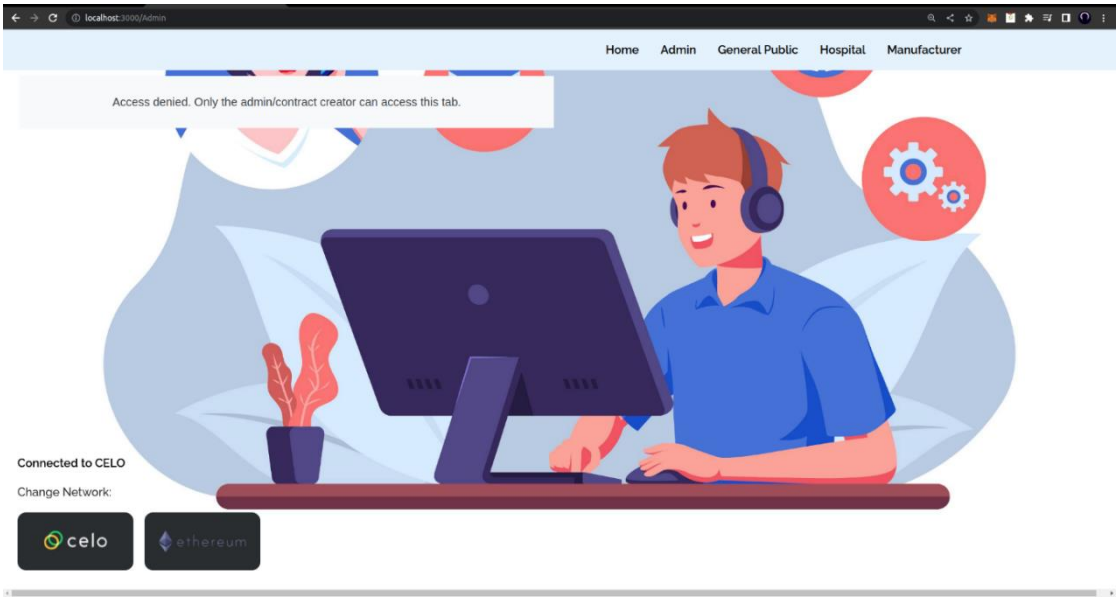

Fig: 4.3 Admin

*only admin can access this tab by there unique Celo or Ethereum account access otherwise access is denied.

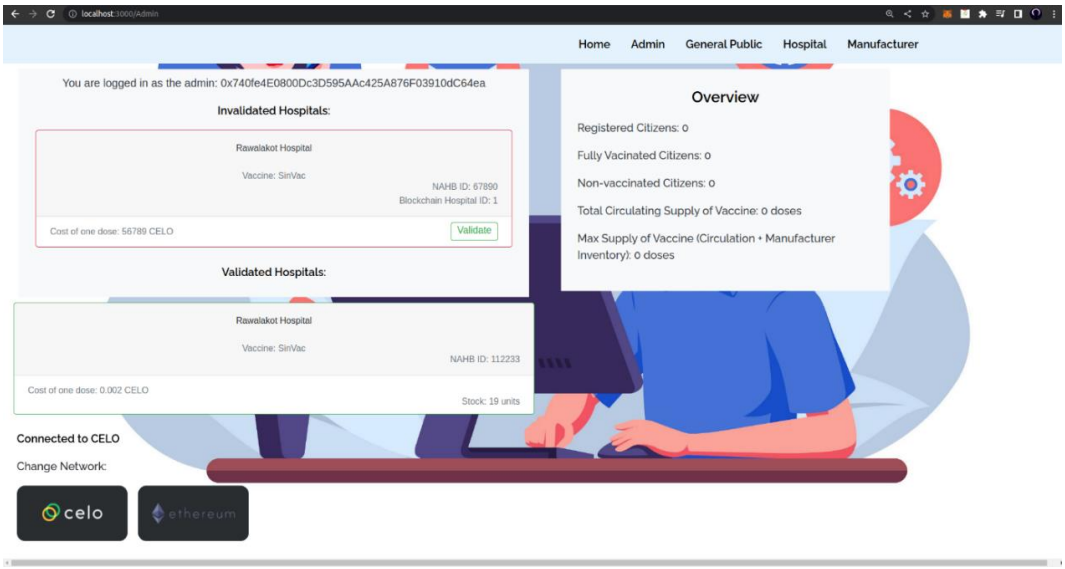

Fig: 4.4 Admin 2

Healthcare Providers (Hospitals or Clinics)

- Obtain vaccines at a fair price from a variety of manufacturers.
- Accept cryptocurrency payments as an asset.
- Using this system, record those who have been immunized, automatically updating the inventory and dose counts.

Fig: 4.5 Hospital

4.4 Vaccine Manufacturers

- Keep an eye on your stock and add additional batches.
- Automated collection of payments in Crypto for vaccination delivery

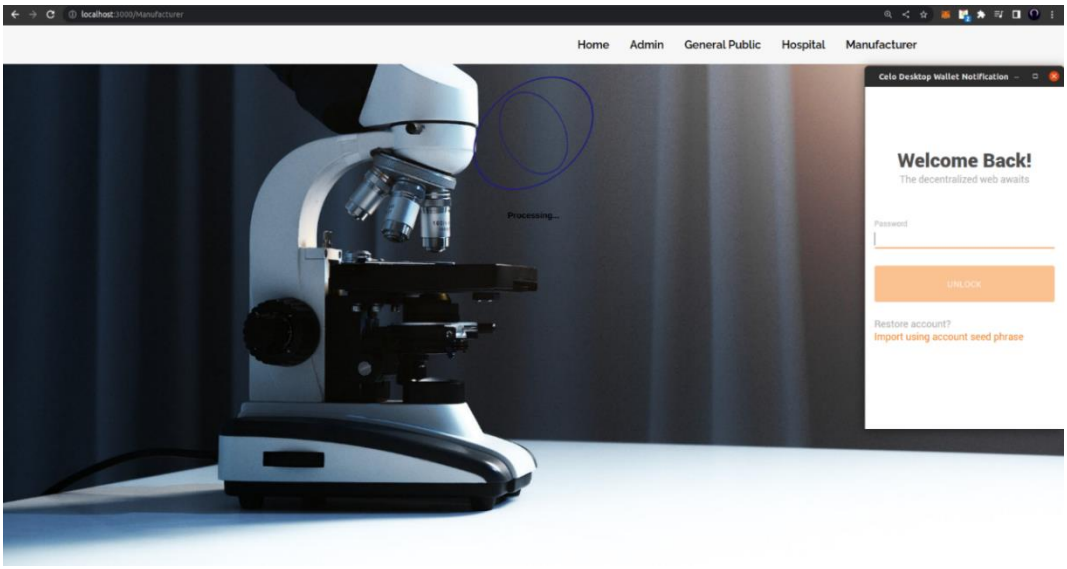

Fig: 4.6 Manufacturer 1

*Pop-up of celo wallet will appear if not **signed** in the wallet. (Remain same for all pages)

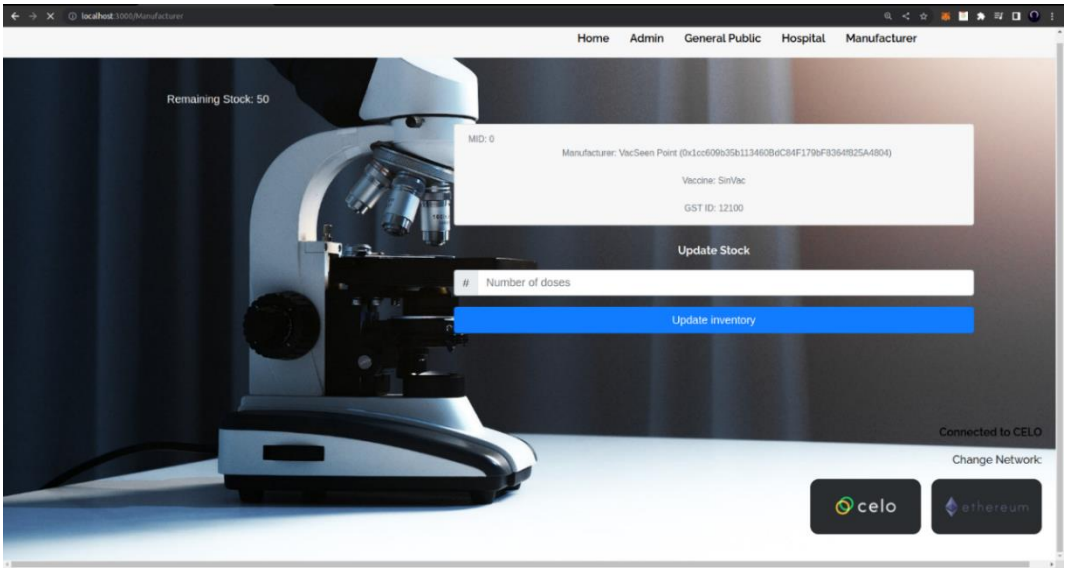

Fig: 4.7 Manufacturer 2

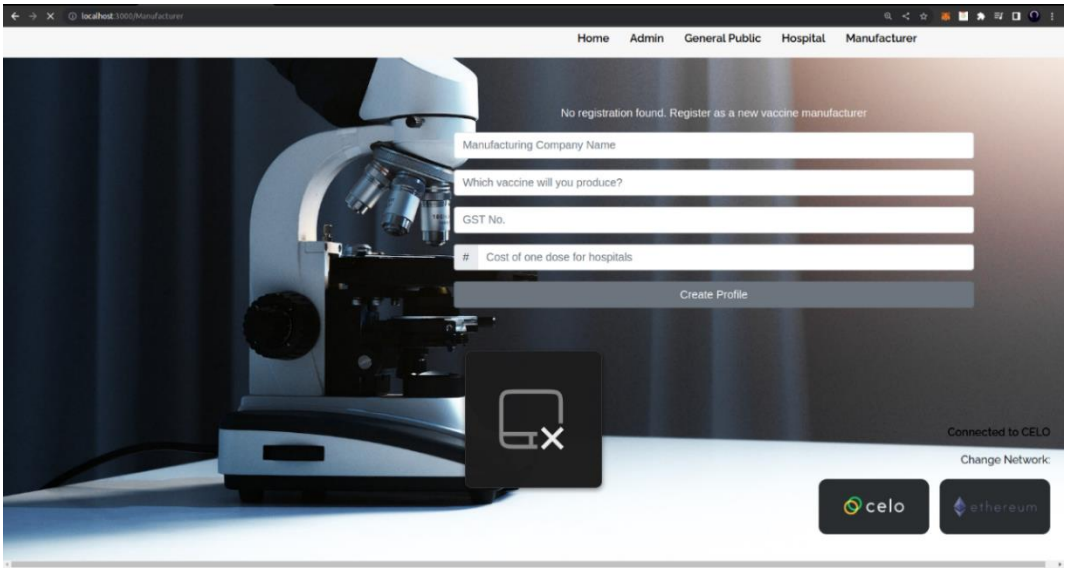

Fig: 3.8 Manufacturer 3

CHAPTER 05

RESULTS AND EVALUATION

There are two different sorts of interactions in a typical supply-chain application. The first is the communication between internal stakeholders or processes, such as finance, inventory, and warehousing. The second is an external procedure or group of parties, including banks, vendors, and suppliers. Because they are affiliated with the same business and are often totally digital and automated, the previous interaction and related procedures may be trusted[23]. However, the latter interaction requires a combination of automated and manual interaction.

A common supply-chain application that needs a system to support the distributed ledger technology used by Blockchain-based Supply-Chain Management (BC-SCM) is an example of such an application. This survey's main objective is to examine current BC-SC M systems for potential assaults on their computational and communication systems and related defenses.

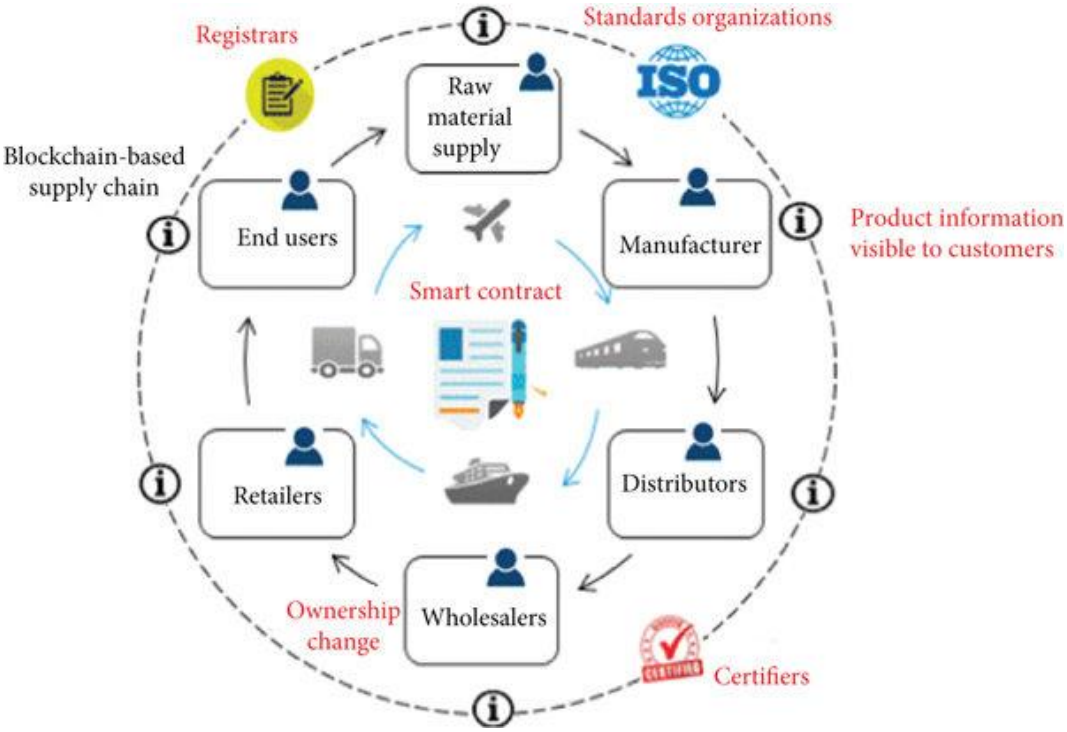

Fig: 5.1 Supply chain management based on blockchain[22]

5.1 Attacks' sources

This research look at the assaults that primarily result from the insecure development of smart contracts and apps, the blockchain execution environment (BEE)[20], and a hostile environment among highly integrated supply-chain services. The primary cause of many assaults is the design of smart contracts and apps (DSC)[21]. Importantly, faith in digital and automated sub-processes is also in jeopardy, mostly as a result of the delivery of context-blind services.

Table 5.1 depicts the relationship between the abovementioned attacks, supply-chain management system security needs, and probable attack reasons.

Table: 5.1 Mapping of Attacks to Requirements.

Type of Attack	Method of Attack	Required Changes	Effects
Computational	Agreement amendment	Tracking and tracing, Privacy	DSC, BEE
	Modification of execution	Tracking and tracing, Privacy	DSC, BEE, TIS
	Exploiting vulnerabilities	Tracking and tracing, Transparency, Privacy	DSC, BEE
Communication	Altered input values	Tracking and tracing, Transparency, Privacy	BEE, TIS
	Data integrity breach	Tracking and tracing, Transparency, Privacy	BEE, TIS
	Injection of false information	Tracking and tracing, Transparency, Privacy	DSC, BEE, TIS

5.2 Systems for Supply-Chains' Security Requirements

Confidentiality, traceability, and traceability are essential components of a secure supply-chain system. When a calculation or communication unexpected happens, such as when a smart contract's execution is affected or a communicative party is affected and shares unwanted data, such solutions are unable to identify any danger in real-time. According to the thread model outlined in Section 4, and the types and sources of assaults, this research examine the current solutions. The next sections each cover the academic and industry endeavors separately[24, 25].

5.3 Academic Efforts in Supply-Chain Management Based on Blockchain:

5.3.1 Modern research

In this section, this research looked into existing blockchain-based solutions developed by academia in the field of supply-chain management, with a focus on communication and computational vulnerabilities[26].

5.3.2 Actions were taken to prevent communication attacks

To manage communication assaults, most research ideas for blockchain-based supply-chain systems centered on four functions.

- Maintaining the privacy of shared information among parties.
- Strengthening governance
- Enabling interoperability across diverse systems and infrastructures.
- Safeguarding the data sharing method is the activity.

The security needs and risk model, as mentioned in Sections 5 & 5.2, are used to assess academic attempts to achieve these activities.

5.3.3 Activities to Prevent Computational Attacks

Computing attacks, in contrast to communication attacks, look for flaws in the system's computational operations. Existing BC-SCM systems provide a defense against computational attacks by

- assuring the compliance and privacy of transactions and activities.
- finding unique methods for smart contract vulnerability identification and creating confidence in their execution environment.

5.4 Tools for Industrial Supply-Chain Management Using Blockchain

Along with academic research, the industry has created a variety of blockchain-based supply-chain technologies to combat the possibility of computational & communication threats. In this part, we'll talk about cutting-edge supply-chain

management tools[27] produced by businesses while keeping the system safe against computation and communications threats.

5.4.1 Actions were taken to prevent communication attacks

To prevent communication attacks, the created technologies (1) traceability and product provenance should be established to reduce fraud. (2) increase financial transaction security while lowering costs, and (3) give protection to information transmitted between cross-border parties and dispute resolution. Table 5.2 summarizes the examination of various industrial instruments, taking into account the security needs and threat model described in Section 3.

Table: 5.2 Attacks on Communication: Industrial Solutions.

Paper	Areas of Application	Operational Targets	Important Components	Required Changes	Possible Attack Techniques
[28-30]	Food business, Airbus	Provenance	Traceability of products, Product provenance, Easy to find traces	Secure communication, Access based on privileges, Case resolution	Traceability, Transparency, Privacy
[28, 29, 31-34]	Cargo, Financial trading, Crypto flow	streamlined financial operations, continuous interoperability, Settlement	adherence to a directive, Transparent communication reliable trading platform, a clear financial flow	Traceability, Transparency, Privacy	Traceability, Transparency, Privacy
[35-37]	Cross-border trade, Trade dispute	Interoperability between international parties	Secure communication, Privileged based access, Resolution of disputes	Traceability, Transparency, Privacy	Data integrity breach, Tampered input data, injection of false information

5.4.2 Actions were taken to prevent Computational Attacks

Computing assaults, once again, take advantage of flaws in the system's computational operations. Industries address these vulnerabilities in the supply-chain sector by developing supply-chain compliance and crypto-currency transactions and detecting weaknesses in smart contracts and their underlying execution environment (e.g., EVM). According to the Blockchain-based supply-chain networks' security needs and threat model, this research analyzed these industry activities in depth in the following subsections. Table 4.3 summarizes the findings of the inquiry.

Table: 5.3 Attacks on Computational: Industrial Solutions.

Paper	Areas of Application	Operational Targets	Important Components	Required Changes	Possible Attack Techniques
[38-40]	Authorities, Finance, security companies, Internet of Things	Compliance with transactions	Data conformity, Adherence to regulations, and Information monitoring	Traceability, Privacy	Modification of execution (rewriting attacks)
[41-43]	Business enterprise network	identification of vulnerabilities	Transfer amount versus the required amount, transaction troubleshooting, graph of a control flow, Non-validated arguments, Exception handling, Withdrawal and Overflow	Traceability, Transparency, Privacy	Exploiting vulnerabilities, altering how things are done

CHAPTER 06

CONCLUSION & FURTHER WORK

Considering the supply-chain systems' security needs as illustrated in Table 4.1 and an assessment of educational and commercial attempts to avoid computational & communication attacks, this research recognized the following gap that exists (opportunities) that need to be taken into account to better support the end-to-end security of blockchain-based supply-chain companies against computational & communication attacks.

6.1 Prevention of Communication Attacks

The mentioned changes were made to better defend against communication attacks. Blockchain-based systems place a high priority on maintaining the confidentiality of communicated information, whereas communication attacks aim to find ways to compromise identity and information in various ways[44].

6.2 Prevention of Computational Attacks

To increase the security against computational attacks, deficiencies in self-awareness, Integration of corporate processes, and assistance with business evolution must be resolved. The BC-SCM systems must assure operational compliance with specific regulations and directives, as well as identify and exploit vulnerabilities in various computational portions of the system, such as smart contracts and virtual machines[45].

6.3 Recommendations

On the bases above-mentioned research gaps, it is required to develop current blockchain-based solutions with details about actual commercial procedures (i.e., communication components) and communication infrastructure (i.e., computational components), allowing various stakeholders to interact in a blockchain-based supply-chain environment. We'll look at how business process-level information

from engaging stakeholders can help secure BC-SCM systems' end-to-end operations in the following sections.

To begin with, knowledge of the business processes of the interacting stakeholders reinforces the security of the blockchain-enabled supply-chain management system by ensuring that stakeholder engagement, including smart contracts, complies not only with the regulations encapsulated in smart contracts but also with the associated company operations. This will provide perfect transparency in all exchanges while also adhering to the company's business processes. This also aids in the detection of dangers that have a cascading effect on the businesses they engage with.

Second, by giving the real-world business context of contacts (such as smart contract-based interactions) and shielding data from known and unknown attack variations, such data would support the creation of self-aware and intelligent defense mechanisms for involved parties. Additionally, the information might be used to ensure that smart contract-based transactions between parties follow their respective business procedures. Additionally, it is possible to identify any flaws or inconsistencies in the stakeholder's business procedures that enable end-to-end information protection.

Third, information at the business process level facilitates updating corporate policies/agreements without affecting the real execution of blockchain-based connections. This is vital because corporate processes are always altering in response to changing business requirements or to ensure compliance with new laws and policies, such as GDPR.

Fourth, information about company processes could aid in determining the true cause of a risk or assault. The system will be aware of the end-to-end specifics of the relevant business process as well as the interactions (that are frequently just interfaces for transactions). In reality, existing blockchain-based systems just keep track of transactions (such as data), which can only be used to identify any data breaches but not the compliance or integrity of the data's actual details.

Fifth, because the system can pinpoint the specific asset or component that has been hacked, this information may be sufficient to both restore damaged operations and automatically lessen the effect of known threats and assaults.

As a result, the data will aid in the detection of happenings as well as their consequences.

Last but not least, blockchain-based systems would be more transparent and provide known likely causes for threats or attack occurrence if this information was available. Because present interactions among stakeholders—including transaction rules—are based on smart contracts, and they have insufficient information about the stakeholders' real business processes, they act as backboxes. As a result, by making the business process transparent to partners, this information can help to build confidence among interacting parties.

As a result, by giving end-to-end details of interactions, supply-chain stakeholders' business process information would increase the protection given by BC-SCM systems. (i) This will help in detecting assault situations as well as assessing and minimizing their real effects. (ii) offering a context-aware audit of the occurrences that might spot tainted data and guarantee the veracity of the data's contents, and (iii) due to interactions being transparent from beginning to end, stakeholder confidence is growing. It includes the openness of all interactions as well as the transparency of smart contract transactions. Finally, such information could aid in the management of security flaws in business processes, allowing stakeholders to better meet their business demands and objectives.

6.4 Conclusion

In this research, this research establishes essential security needs and risk models for the supply-chain management system. This research contends that a system that implements the following suggestions can deliver sufficient data for the strict protection of key assets and information against both known and unknown assaults. Such data enables solutions to, on the one side, automatically execute different regulations and instructions &, on the other, to identify any issue or system defect (i.e., a discrepancy between a business process and its underlying implementation).

References

1. Farahani, B., F. Firouzi, and M. Luecking, *The convergence of IoT and distributed ledger technologies (DLT): Opportunities, challenges, and solutions.* Journal of Network and Computer Applications, 2021. **177**: p. 102936.
2. Chabani, Z. and W. Chabani, *The Role of Blockchain Technology in Enhancing Security Management in the Supply Chain*, in *Advances in Blockchain Technology for Cyber Physical Systems*. 2022, Springer. p. 211-231.
3. Wang, Y., J.H. Han, and P. Beynon-Davies, *Understanding blockchain technology for future supply chains: a systematic literature review and research agenda.* Supply Chain Management: An International Journal, 2018.
4. Abioye, S.O., et al., *Artificial intelligence in the construction industry: A review of present status, opportunities, and future challenges.* Journal of Building Engineering, 2021. **44**: p. 103299.
5. Vadgama, N. and P. Tasca, *An analysis of blockchain adoption in supply chains between 2010 and 2020.* Frontiers in Blockchain, 2021. **4**: p. 610476.
6. Chang, S.E., Y.-C. Chen, and M.-F. Lu, *Supply chain re-engineering using blockchain technology: A case of smart contract based tracking process.* Technological Forecasting and Social Change, 2019. **144**: p. 1-11.
7. Stephens, S., *Supply chain operations reference model version 5.0: a new tool to improve supply chain efficiency and achieve best practice.* Information Systems Frontiers, 2001. **3**(4): p. 471-476.
8. https://blockchain.oodles.io/blog/solving-supply-chain-management-challenges-blockchain/.
9. Kannan, G., et al. *Managing the Supply Chain for the Crops Directed from Agricultural Fields using Blockchains*. in *2022 International Conference on Electronics and Renewable Systems (ICEARS)*. 2022. IEEE.
10. Nofer, M., et al., *Blockchain.* Business & Information Systems Engineering, 2017. **59**(3): p. 183-187.
11. Romero Ugarte, J.L., *Distributed ledger technology (DLT): introduction.* Banco de Espana Article, 2018. **19**: p. 18.
12. Chowdhury, M.J.M., et al., *A comparative analysis of distributed ledger technology platforms.* IEEE Access, 2019. **7**: p. 167930-167943.

13. https://www.researchgate.net/figure/The-properties-of-distributed-ledger-technology-39_fig1_355223429.

14. Sobti, R. and G. Geetha, *Cryptographic hash functions: a review*. International Journal of Computer Science Issues (IJCSI), 2012. **9**(2): p. 461.

15. Cash, M., and M. Bassiouni. *Two-tier permission-ed and permission-less blockchain for secure data sharing*. in *2018 IEEE International Conference on Smart Cloud (SmartCloud)*. 2018. IEEE.

16. Usman, M. and U. Qamar, *Secure electronic medical records storage and sharing using blockchain technology*. Procedia Computer Science, 2020. **174**: p. 321-327.

17. Liang, Y. *Identity verification and management of electronic health records with blockchain technology*. in *2019 IEEE international conference on healthcare informatics (ichi)*. 2019. IEEE.

18. Raikwar, M., D. Gligoroski, and K. Kralevska, *SoK of used cryptography in blockchain.* IEEE Access, 2019. **7**: p. 148550-148575.

19. Zheng, Z., et al., *An overview on smart contracts: Challenges, advances, and platforms.* Future Generation Computer Systems, 2020. **105**: p. 475-491.

20. Kolvart, M., M. Poola, and A. Rull, *Smart contracts*, in *The Future of Law and etechnologies*. 2016, Springer. p. 133-147.

21. Luu, L., et al. *Making smart contracts smarter*. in *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*. 2016.

22. Saberi, S., et al., *Blockchain technology and its relationships to sustainable supply chain management.* International Journal of Production Research, 2019. **57**(7): p. 2117-2135.

23. Swaminathan, J.M., S.F. Smith, and N.M. Sadeh, *Modeling supply chain dynamics: A multiagent approach.* Decision sciences, 1998. **29**(3): p. 607-632.

24. Hou, Y., J. Such, and A. Rashid. *Understanding security requirements for industrial control system supply chains*. in *2019 IEEE/ACM 5th International Workshop on Software Engineering for Smart Cyber-Physical Systems (SEsCPS)*. 2019. IEEE.

25. Abdel-Basset, M., G. Manogaran, and M. Mohamed, *Internet of Things (IoT) and its impact on supply chain: A framework for building smart, secure and efficient systems.* Future Generation Computer Systems, 2018. **86**(9): p. 614-628.

26. Kosmarski, A., *Blockchain adoption in academia: Promises and challenges.* Journal of Open Innovation: Technology, Market, and Complexity, 2020. **6**(4): p. 117.

27. Holmström, J., et al., *Evaluating research-in-process: At the cutting edge of solution design and supply chain management theory.* Helsinki University of Technology, Department of Industrial Engineering and Management, Working Paper, 2004(2004/4).

28. Al-Farsi, S., M.M. Rathore, and S. Bakiras, *Security of blockchain-based supply chain management systems: challenges and opportunities.* Applied Sciences, 2021. **11**(12): p. 5585.

29. Krstić, M. and L. Krstić, *Hyperledger frameworks with a special focus on hyperledger fabric.* Vojnotehnički glasnik/Military Technical Courier, 2020. **68**(3): p. 639-663.

30. Tselios, N., *Blockchain: the phare of Innovative Entrepreneurship.* 2019.

31. Casey, M., et al., *The impact of blockchain technology on finance: A catalyst for change.* 2018.

32. Chen, J., et al., *A blockchain-driven supply chain finance application for auto retail industry.* Entropy, 2020. **22**(1): p. 95.

33. Cozzi, F., *Will blockchain technologies strengthen or undermine the effectiveness of global trade control regulations and financial sanctions?* Global Jurist, 2020. **20**(2).

34. Niforos, M., V. Ramachandran, and T. Rehermann, *Block Chain.* 2017.

35. Schmitz, A.J., *There's an App for That: Developing Online Dispute Resolution to Empower Economic Development.* Notre Dame JL Ethics & Pub. Pol'y, 2018. **32**: p. 1.

36. Bebbington, A., B. Fash, and J. Rogan, *Socio-environmental conflict, political settlements, and mining governance: A cross-border comparison, El Salvador and Honduras.* Latin American Perspectives, 2019. **46**(2): p. 84-106.

37. Koh, H.H., *The" Gants Principles" for Online Dispute Resolution: Realizing the Chief Justice's Vision for Courts in the Cloud.* BCL Rev., 2021. **62**: p. 2768.

38. Zhang, L., et al., *The challenges and countermeasures of blockchain in finance and economics.* Systems Research and Behavioral Science, 2020. **37**(4): p. 691-698.

39. Ross, E.S., *Nobody puts blockchain in a corner: The disruptive role of blockchain technology in the financial services industry and current regulatory issues.* Cath. UJL & Tech, 2016. **25**: p. 353.

40. Yussof, S.A. and A.M.H. Al-Harthy, *Cryptocurrency as an alternative currency in Malaysia: issues and challenges.* ICR journal, 2018. **9**(1): p. 48-65.

41. Lucena, P., et al., *A case study for grain quality assurance tracking based on a Blockchain business network.* arXiv preprint arXiv:1803.07877, 2018.

42. Li, D., W.E. Wong, and J. Guo. *A survey on blockchain for enterprise using hyperledger fabric and composer*. in *2019 6th International Conference on Dependable Systems and Their Applications (DSA)*. 2020. IEEE.

43. Gaur, N., *Blockchain challenges in adoption.* Managerial Finance, 2019.

44. Cetinkaya, A., et al., *Randomized transmission protocols for protection against jamming attacks in multi-agent consensus.* Automatica, 2020. **117**: p. 108960.

45. Rivain, M. and J. Wang, *Analysis and improvement of differential computation attacks against internally-encoded white-box implementations.* IACR Transactions on Cryptographic Hardware and Embedded Systems, 2019: p. 225-255.