

**RAJIV GANDHI INSTITUTE OF TECHNOLOGY
GOVERNMENT ENGINEERING COLLEGE**

KOTTAYAM-686 501



DEPARTMENT OF COMPUTER APPLICATIONS

20MCA246 - MAIN PROJECT REPORT

**VACCINE SUPPLY CHAIN MANAGEMENT THROUGH
BLOCKCHAIN TECHNOLOGY**

Submitted By

GOPIKA KRISHNAN S

(KTE22MCA-2025)

Supervisor Name

Dr. Sangeetha Jose



APJ ABDUL KALAM TECHNOLOGICAL UNIVERSITY

THIRUVANANTHAPURAM

APRIL 2024

DECLARATION

I, hereby declare that the project report entitled “**Vaccine Supply Chain Management Through Blockchain Technology**”, submitted in partial fulfillment of the requirements for the award of the degree of Master of Computer Applications of the APJ Abdul Kalam Technological University, Kerala, is a bonafide work done by me under the supervision of **Dr. Sangeetha Jose**. This submission represents my ideas in my own words and where ideas or words of others have been included, I have adequately and accurately cited and referenced the original sources. I also declare that I have adhered to the ethics of academic honesty and integrity and have not misrepresented or fabricated any data or idea or fact or source in my submission. I understand that any violation of the above will be a cause for disciplinary action by the institute and/or the University and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been obtained. This report has not been previously submitted as the basis for the award of any degree, diploma, or similar title of any other University.

Place: Kottayam

Date: 22/04/2024

GOPIKA KRISHNAN S

ACKNOWLEDGEMENT

I want to express my gratitude to everyone who has supported me throughout the endeavour. First and foremost, I give thanks to God Almighty for His mercy and blessings, for without His unexpected direction, this would still be only a dream.

I sincerely thank **Dr. Prince A**, Principal, Rajiv Gandhi Institute of Technology, Kottayam, for providing the environment in which this project could be completed.

I owe a huge debt of gratitude to **Dr. Reena Murali**, Head of the Department of Computer Applications, for granting permission and making available all of the facilities needed to complete the project properly.

I am grateful to my project guide **Dr. Sangeetha Jose**, for her helpful criticism of my thesis.

I am grateful to the project coordinators **Dr. John C John** and **Prof. Sreejith V P**, for providing constructive suggestions and inspiration throughout the project.

Finally, I would like to take this chance to express my gratitude to the faculty and technical staff of the Department of Computer Applications.

GOPIKA KRISHNAN S

ABSTRACT

Vaccine supply chains are vulnerable to various threats, including tampering and counterfeit vaccines. To address these challenges, a decentralized system built on the Hyperledger Fabric blockchain is proposed. This integration enhances data immutability, security, and transparency throughout the vaccine distribution process. Methodologically, the project involved API design, data modeling, middleware development, and integration with Hyperledger Fabric, alongside robust authentication implementation. The key findings reveal that the blockchain implementation successfully achieves several objectives, including enhanced security, transparency, immutability, traceability, and operational efficiency. Notably, the system achieves these benefits without incurring gas fees typically associated with public blockchains. By leveraging cryptographic hashing, consensus algorithms, and a shared ledger, stakeholders are empowered with real-time visibility into vaccine movements, streamlined operations, and reduced errors. The decentralized nature of the system also enhances resilience against disruptions or attacks. In essence, this project represents a successful implementation of a permissioned blockchain system tailored for a secure, transparent, and efficient vaccine supply chain. Through precise design and integration, the system addresses critical vulnerabilities present in traditional supply chains. This initiative not only mitigates risks associated with vaccine distribution but also contributes to broader efforts in enhancing the integrity and efficiency of supply chain management practices.

Keywords: Hyperledger Fabric, APIs, Blockchain technology, Vaccine, Middleware

Contents

DECLARATION	i
ACKNOWLEDGEMENT	ii
ABSTRACT	iii
LIST OF FIGURES	vi
LIST OF TABLES	vii
LIST OF ABBREVIATIONS	viii
1 INTRODUCTION	1
1.1 Need for the Project	1
1.2 Objective	1
1.3 Scope of the Project	2
1.4 Organisation of Thesis	2
2 LITERATURE REVIEW	4
2.1 Existing System	4
2.2 System Study	4
2.3 Gap Identification	7
3 PROPOSED METHODOLOGY	10
3.1 Features of Proposed System	10

3.2	Materials and Methods	11
3.2.1	Methodology	11
3.2.2	Algorithm	12
3.2.3	System Architecture	14
3.2.4	Database Schema	15
3.2.5	System Specification	17
3.2.6	Software Tools	18
4	RESULTS AND ANALYSIS	20
4.1	Results	20
4.2	Performance Analysis	21
5	CONCLUSION	24
6	FUTURE SCOPE	25
	REFERENCES	26
	APPENDIX	27

LIST OF FIGURES

3.1	System Architecture	14
4.1	Impact of Transaction Rates and Transaction Types on the Blockchain Through- put and Latency	22
4.2	Impact of the Number of Participants on Throughput and Latency	23
6.1	User Login	39
6.2	Add Vaccine	39
6.3	Data Stored on Blockchain	40
6.4	Bitbucket History	41

LIST OF TABLES

2.1	Literature Review	6
2.2	Detailed Study of Current and Proposed System	8
3.1	User Login Table	15
3.2	Vaccine Table	16
3.3	Manufacturer Table	16
3.4	Vaccinerequest Table	17
4.1	Outcomes Achieved	21

LIST OF ABBREVIATIONS

Abbreviations	Definition
Golang	Go programming language
API	Application Programming Interface
JWT	JSON Web Token
Hyperledger	Hyperledger Fabric
HTTP	Hypertext Transfer Protocol

Chapter 1

INTRODUCTION

This Chapter introduces the critical necessity for transforming vaccine supply chains, tackling challenges such as counterfeit vaccines and transparency gaps. It outlines the goal of creating a decentralized data management system using blockchain, emphasizing authenticity and traceability.

1.1 Need for the Project

The project on "Vaccine Supply Chain Management through Blockchain Technology" [1] is crucial because current vaccine supply chains face significant problems. Vaccines, essential for human health, are at risk due to issues like fake vaccines, tampered expiration dates, and a lack of transparency in tracking their journey. Traditional systems relying on manual records are prone to data tampering, affecting the reliability of vaccine information. The global impact of poor-quality drugs, including counterfeit vaccines, has resulted in numerous deaths, emphasizing the urgent need for a better solution. The complexity of the supply chain, involving various stakeholders, creates challenges in maintaining trust and transparency. Specific issues in the vaccine supply chain, such as the need for a cold chain and vulnerabilities in centralized systems, underline the need for an innovative solution.

Utilizing blockchain technology as a transformative solution to these challenges, the project ensures transparency, traceability, and immutability by storing transaction records in secure and interconnected blocks. The decentralized nature of blockchain mitigates the risk of a single point of failure, prevalent in current centralized systems. Integration of smart contracts automates transactions, eliminating the need for cumbersome paperwork and reducing the likelihood of errors. Blockchain's ability to provide a secure and transparent reputation system addresses concerns related to feedback and user comments, ensuring the reliability of information.

1.2 Objective

Focusing on designing a decentralized data management system using blockchain technology to address vulnerabilities present in centralized approaches, this initiative aims to ensure the authenticity, immutability, and traceability of vaccine-related information. Leveraging

blockchain features for tamper-proof records and transparent transaction history throughout the supply chain is a primary goal. Additionally, evaluating the performance of the proposed framework through various metrics, assessing its effectiveness, efficiency, and overall functionality in enhancing vaccine supply chain management, is crucial. These objectives collectively outline a strategic approach to mitigate existing challenges in vaccine supply chains, providing a secure, transparent, and efficient solution through blockchain integration.

1.3 Scope of the Project

The project's scope is extensive, encompassing crucial aspects aimed at addressing challenges prevalent in contemporary vaccine supply chains. Firstly, it involves the integration of blockchain technology into the supply chain, creating a decentralized framework to capitalize on blockchain's features of transparency, immutability, and traceability. Additionally, the project delves into securing and maintaining the integrity of vaccine-related data throughout the supply chain process. Through the implementation of smart contracts using the Solidity programming language, the reliance on paperwork is minimized, leading to increased efficiency. Furthermore, the project explores Hyperledger tools, tailoring their usage to enhance security measures. Performance metrics will be employed to evaluate the effectiveness, efficiency, and overall functionality of the blockchain-based system. Gas management optimization in public blockchain scenarios is another focal point, striving to reduce costs and enhance transactional efficiency. The versatility of the developed framework for operation on public blockchains is considered, ensuring adaptability across various vaccine supply chain scenarios. Addressing the rise of online pharmacies, security measures are incorporated to safeguard against counterfeit vaccines entering the legitimate supply chain. Cold chain management for temperature-controlled vaccines is a critical component, ensuring proper refrigeration transport within the supply chain entities. Emphasizing global applicability, the project aims to contribute to a standardized and secure approach to vaccine distribution that transcends regional challenges. Overall, the project's scope is comprehensive, covering technical intricacies, performance evaluation, and specific challenges within vaccine supply chains on a global scale.

1.4 Organisation of Thesis

The groundwork has been laid for understanding the pivotal need and objectives of the Vaccine Supply Chain Management through Blockchain Technology project. Chapter 2 embarks on a comprehensive exploration, beginning with a literature review aimed at identifying

critical gaps in current research and technology related to vaccine supply chain management and blockchain technology. This review aims to establish the context and rationale for implementing a decentralized system to address transparency, traceability, and security concerns in vaccine distribution. Following this, Chapter 3 introduces the proposed system, detailing its design, methodology, and the tools utilized in its development. This chapter outlines the technical aspects of how blockchain technology is leveraged to create a transparent and accountable vaccine supply chain management system. Chapter 4 presents the results obtained from implementing the proposed system, analyzing findings in the context of the system's objectives. It delves into the performance, efficiency, and security enhancements achieved through blockchain integration, providing insights into the system's effectiveness in addressing the challenges of traditional vaccine supply chains. Finally, Chapter 5 concludes the document by summarizing key insights, implications, and reflecting on the significance of the findings. Chapter 6 outlines potential avenues for future research and development, focusing on extending the scope of the project and further advancing vaccine supply chain management through blockchain technology.

As the chapter concludes, it becomes evident that revolutionizing vaccine supply chains through the integration of blockchain technology is imperative.

Chapter 2

LITERATURE REVIEW

This chapter reviews the literature related to vaccine supply chain management, blockchain technology, and associated challenges in the pharmaceutical industry. It involves exploring existing studies and research articles to gain insights into these topics.

2.1 Existing System

The current vaccine supply chain management system operates within a framework reliant on centralized databases and manual processes. Typically overseen by a central authority, this system suffers from transparency issues due to limited real-time visibility into vaccine movements and relies on error-prone paper-based records and manual data entry for tracking. Centralized data storage also presents security vulnerabilities, with the potential for tampering or unauthorized access, jeopardizing public health through the introduction of counterfeit or compromised products. Moreover, the lack of robust traceability mechanisms complicates tracking vaccine batches from production to distribution and administration, hindering timely response efforts during recalls or disruptions. These challenges underscore the urgent need for a more efficient, transparent, and secure approach to vaccine distribution, with blockchain technology offering promising solutions through its decentralized, tamper-proof nature, enhancing transparency, traceability, and security throughout the supply chain.

2.2 System Study

L. Cui et al. proposed a Protecting Vaccine Safety: Improved, Blockchain-Based, Storage-Efficient Scheme [2]. The paper introduces a blockchain solution for bolstering vaccine safety, employing trace codes, transactions, and smart contracts. While tackling data reliability challenges, it encounters potential efficiency issues and cloud security concerns, prompting the need for additional enhancements to optimize performance in practical vaccine circulation scenarios.

T. Fatokun et al. proposed a Towards a Blockchain Assisted Patient Owned System for Electronic Health Records [3]. The objective is a patient-centric EHR system using blockchain for security, privacy, and interoperability. Patients control records ensuring secure data exchange.

The research includes functional and performance testing, contributing to enhanced data security, privacy, and interoperability in healthcare.

F. Jamil et al. proposed a Novel Medical Blockchain Model for Drug Supply Chain Integrity Management in a Smart Hospital [4]. Objectives include developing a secure drug supply chain record system with Hyperledger Fabric, enhancing transparency in smart hospitals, and providing a global solution to counterfeit pharmaceuticals. Features encompass secure record systems, network topology, smart contracts, access control, CRUD operations, and performance evaluation. Limitations include generalizability, scalability, regulatory considerations, security, privacy, and resource implications, with potential challenges in real-world implementation.

A. R. Nair et al. proposed FAIR, a blockchain-based approach for fair and secure vaccine distribution during pandemics [5]. Utilizing smart contracts and IPFS, it addresses challenges in healthcare supply chains. Limitations include potential complexities, costs, scalability issues, and integration challenges with existing systems and data privacy concerns.

I. T. Javed et al. proposed PETchain, a privacy-enhancing technology on a consortium blockchain [6]. PETchain empowers users with control over their data, stored securely on IPFS. It features a user-centric architecture, secure data storage, Trusted Execution Environment (TEE), auditable logs, and smart contract functionality. The objectives involve PETchain development, a smart contract for access control, and performance analysis.

A. Musamih et al. proposed a Blockchain-based solution for COVID-19 vaccine distribution [7] addressing transparency, traceability, and security. Features include accountability through Ethereum smart contracts, decentralized availability, protection against attacks, rigorous security analysis, and a comparative advantage over non-blockchain solutions. Limitations encompass scalability challenges, internet connectivity dependence, off-chain storage costs, smart contract customization complexity, and regulatory compliance hurdles.

S. Bhushan et al. proposed a Blockchain-powered vaccine supply chain system [8] to tackle distribution challenges, ensuring transparency, authenticity, and efficacy. Features include decentralized smart contracts, tamper-proof administration, IoT integration for temperature monitoring, and public reporting. Limitations involve technology adoption, regulatory hurdles, resource demands, and integration complexities with existing distribution systems.

The summary of the above study is provided in Table 2.1

Table 2.1: Literature Review

Sl No.	Title	Author	Objective	Features	Limitations
1	Protecting Vaccine Safety: An Improved, Blockchain-Based, Storage-Efficient Scheme (2022) [2]	L. Cui et al	<ul style="list-style-type: none"> • Tackle vaccine circulation reliability challenges. • Introduce a secure blockchain for enhanced vaccine safety and traceability in circulation. 	<ul style="list-style-type: none"> • Incorporates cloud technology for off-chain data storage and efficient data management. • Uses consortium blockchain to securely record vital vaccine circulation details. 	<ul style="list-style-type: none"> • Processing Speed Challenges in Packing Step • Computation and Storage Burden
2	Towards a Blockchain Assisted Patient Owned System for Electronic Health Records (2021) [3]	T. Fatokun et al	<ul style="list-style-type: none"> • Introduce a patient-centric EHR system using blockchain. • Enhance EHR system interoperability for secure data exchange. 	<ul style="list-style-type: none"> • Secure, consistent health records controlled by patients. • Patient-Centric EHR Web Portal 	<ul style="list-style-type: none"> • Bandwidth Overhead • Scalability Challenges
3	A Novel Medical Blockchain Model for Drug Supply Chain Integrity Management in a Smart Hospital (2019) [4]	F. Jamil et al	<ul style="list-style-type: none"> • Creating a secure drug supply chain with Hyperledger Fabric blockchain. • Handling counterfeit drugs in developing country pharmaceuticals. 	<ul style="list-style-type: none"> • Secure Drug Supply Chain Record System • Access Control and Permissions 	<ul style="list-style-type: none"> • Cost and Resource Implications • Real-world implementation challenges
4	FAIR: A Blockchain-based Vaccine Distribution Scheme for Pandemics (2021) [5]	A. R. Nair et al	<ul style="list-style-type: none"> • Address healthcare supply chain challenges. • Ensure Secure and Fair Distribution System. 	<ul style="list-style-type: none"> • Focus on trust and forecasting • Distinct working layers 	<ul style="list-style-type: none"> • Blockchain Implementation Challenges • Integrating blockchain into healthcare systems may pose challenges.

SI No.	Title	Author	Objective	Features	Limitations
5	PETchain: A Blockchain-Based Privacy Enhancing Technology (2021) [6]	I. T. Javed et al	<ul style="list-style-type: none"> • Introduce PETchain, enhancing privacy for service providers using user data. • Develop and implement PETchain as a user-centric privacy solution. 	<ul style="list-style-type: none"> • Auditable and Immutable Logs • User-Centric Architecture 	<ul style="list-style-type: none"> • Impact of Block-gas-limit • Block-time Variability
6	Blockchain-Based Solution for Distribution and Delivery of COVID-19 Vaccines (2021) [7]	A Musamih et al	<ul style="list-style-type: none"> • Propose a blockchain solution for COVID-19 vaccine distribution and delivery. • Address vaccine supply chain challenges to ensure data transparency, traceability, and trust. 	<ul style="list-style-type: none"> • Protection Against MITM Attacks • Accountability and Authorization 	<ul style="list-style-type: none"> • Limited Scalability • Cost of Off-Chain Storage
7	Blockchain Powered Vaccine Efficacy for Pharma Sector (2022) [8]	S. Bhushan et al	<ul style="list-style-type: none"> • Address vaccine distribution challenges: wastage, counterfeits, traceability, authenticity, and transparency. • Ensure proper transportation conditions within the cold chain supply for vaccines. 	<ul style="list-style-type: none"> • Decentralized smart contract-based vaccine monitoring system. • Rating system based on post-vaccination reactions in users. 	<ul style="list-style-type: none"> • Regulatory challenges may arise in implementing the system across global health-care systems. • The system may need substantial resources for ensuring vaccine data security and privacy.

2.3 Gap Identification

In the current vaccine supply chain management system, several significant gaps have been identified, highlighting areas where improvements are essential. Firstly, there is a notable lack of transparency and visibility, as stakeholders often face challenges accessing real-time information about vaccine movements and status. This opacity hinders decision-making and response efforts during supply chain disruptions. Additionally, the system lacks robust traceability

mechanisms, making it difficult to track vaccine batches accurately from production to distribution and administration. Without effective traceability, identifying the source of vaccine-related issues, such as recalls or quality control problems, becomes challenging, delaying response efforts and potentially endangering public health. Moreover, the centralized nature of data storage in the existing system raises concerns about data integrity and security, with vulnerabilities to tampering and unauthorized access compromising the reliability of vaccine records. Manual processes prevalent in the system also contribute to inefficiencies and errors, highlighting the need for automation and digitization to improve efficiency and accuracy. Furthermore, the system's susceptibility to counterfeit vaccines infiltrating the supply chain poses significant risks to consumer safety. Addressing these gaps through the implementation of blockchain technology could lead to transformative improvements in vaccine supply chain management, enhancing transparency, traceability, security, and efficiency, ultimately bolstering public health response capabilities.

Table 2.2: Detailed Study of Current and Proposed System

Current System	Proposed System
Limited real-time visibility into vaccine movements	Enhanced transparency with real-time tracking of vaccine movements
Lack of robust traceability mechanisms	Precise tracking of vaccine batches from production to distribution using blockchain technology
Centralized data storage prone to tampering and unauthorized access	Decentralized, tamper-proof blockchain technology ensuring data integrity and security
Manual processes leading to inefficiencies and errors	Automation and digitization improving efficiency and accuracy
Lack of robust verification mechanisms	Secure authentication and verification systems preventing counterfeit infiltration
Susceptibility to supply chain disruptions	Increased resilience to disruptions with blockchain technology ensuring continuity

Table 2.2 compares aspects of the current and proposed systems, highlighting improvements introduced by the blockchain-based solution.

Based on the findings from the literature review of the current vaccine supply chain management system and the identified shortcomings, the objective of this project is to develop a novel system leveraging blockchain technology. The current system operates within a framework reliant on centralized databases and manual processes, leading to issues such as limited trans-

parency, traceability, and security vulnerabilities. The primary goal is to develop and implement a blockchain-based solution that enhances the efficiency, transparency, and security of vaccine distribution. The project aims to provide real-time visibility into vaccine movements and status throughout the supply chain. By leveraging blockchain's decentralized nature, stakeholders will have access to transparent and immutable records, enabling informed decision-making and timely response efforts during disruptions.

Key objectives include:

Enhancing transparency and Traceability: Blockchain provides a transparent and tamper-resistant ledger, ensuring visibility into the entire vaccine supply chain. The technology enables precise tracking of vaccines from production to distribution, reducing the risk of counterfeit products entering the supply chain.

Data Integrity Assurance: By utilizing blockchain technology, the project aims to address concerns regarding data integrity and security in the existing system. Blockchain's tamper-proof nature and cryptographic techniques will help mitigate risks associated with data tampering and unauthorized access, ensuring the reliability of vaccine records.

Decentralized Trust: Decentralization eliminates the need for a central authority, fostering trust among stakeholders and enhancing the overall security of the supply chain.

Mitigating risks of counterfeit vaccines: The project aims to mitigate the risks of counterfeit vaccines infiltrating the supply chain by implementing blockchain-based solutions that enhance authenticity and traceability. Through features such as smart contracts and decentralized verification mechanisms, the project seeks to ensure the integrity of vaccine products and safeguard consumer safety.

The literature review section synthesizes key discoveries and highlights existing gaps in the domain of vaccine supply chain management through blockchain technology. Drawing on these insights, the project aims to leverage blockchain's capabilities, addressing these challenges and contributing to the development of a more secure and transparent vaccine supply chain.

Chapter 3

PROPOSED METHODOLOGY

The proposed system represents a significant advancement in addressing the challenges identified in the current system. It covers key features, materials, and methods employed in the development. With a focus on leveraging blockchain technology, the outlined system offers innovative solutions to enhance efficiency, security, and transparency in the targeted domain.

3.1 Features of Proposed System

The proposed system leverages blockchain technology to introduce a decentralized network of nodes, eliminating the need for a central authority and fostering transparency and trust among stakeholders. It enables real-time tracking of vaccine batches throughout their journey from production to distribution and administration, providing stakeholders with immediate access to critical information about the status and location of vaccines. Through the immutability of blockchain, the system ensures the integrity and security of vaccine records, making them tamper-proof and resistant to unauthorized changes or manipulation. Precise tracking and tracing capabilities allow stakeholders to verify the authenticity and journey of each vaccine, reducing the risk of counterfeit infiltration.

In the system utilizing Hyperledger Fabric with the Practical Byzantine Fault Tolerance (PBFT) algorithm, an unparalleled level of reliability and trust in vaccine information management is poised to be established. PBFT, renowned for its resilience against Byzantine faults, ensures the consistency and validity of transactions within the distributed network. This algorithm enables nodes to reach consensus even in the presence of potentially malicious actors or network disruptions, thereby fortifying the integrity of vaccine data stored on the blockchain. By employing PBFT alongside Hyperledger Fabric, the system not only guarantees the authenticity and immutability of vaccine records but also bolsters confidence in the transparency and security of the entire vaccination ecosystem.

The system ensures resilience during disruptions, fosters collaboration, scales efficiently, and enhances transparency, traceability, security, and efficiency. PBFT's consensus mechanism fortifies trust, paving the way for effective global health management.

3.2 Materials and Methods

This section provides a systematic blueprint for the development of a blockchain-driven vaccine supply chain management system. It aims to tackle the transparency, traceability, and security challenges prevalent in current vaccine distribution systems. This structured methodology offers a framework to design, implement, and integrate vital components for efficient vaccine management.

3.2.1 Methodology

The methodology for developing the vaccine supply chain management system via blockchain technology involves defining API endpoints and functionalities, structuring data models, developing middleware, integrating with Hyperledger, implementing authentication mechanisms, and conducting testing. Each stage ensures the system's efficiency, security, and reliability in managing vaccine-related information and transactions.

1. API Design

This stage involves defining the endpoints and functionalities that the system will provide through its API (Application Programming Interface). The defined functionalities include registering new vaccines, tracking vaccine batches, verifying vaccine authenticity, and retrieving vaccine information.

2. Data Models

In this phase, data structures are defined to organize and manage information related to vaccine batches. Attributes such as batch number, manufacturer, production date, etc., are identified and structured to facilitate efficient data management. Additionally, structures for transactions and blocks specific to the Hyperledger blockchain are defined to ensure compatibility and interoperability with the chosen blockchain platform.

3. Middleware Development

Middleware development involves implementing a layer of software, typically using a programming language like Golang, to expose the defined API endpoints. Libraries such as Gorilla Mux are utilized for routing and handling HTTP requests, ensuring smooth communication between the frontend and backend components of the system.

4. Hyperledger Integration

This step focuses on integrating the Golang middleware with the chosen Hyperledger

blockchain platform, such as Hyperledger Fabric or Composer. Smart contracts are implemented to define the business logic and rules governing vaccine transactions and authenticity verification on the blockchain.

5. Authentication and Authorization

Authentication mechanisms such as JWT (JSON Web Tokens) are implemented to secure the API endpoints and prevent unauthorized access. Roles and permissions are defined to regulate access to different functionalities within the system, ensuring data security and privacy.

6. Testing

Testing is conducted to validate the functionality, performance, and reliability of the developed system. Unit tests are developed to verify the correctness of individual API endpoints and middleware functions. Integration tests are performed to ensure seamless interoperability between the developed components and the Hyperledger blockchain platform.

This methodology offers a structured approach to building a secure, transparent, and efficient vaccine supply chain management system using blockchain technology. Each stage is crucial for ensuring the system's successful implementation and deployment to tackle the challenges encountered in the current vaccine supply chain.

3.2.2 Algorithm

The Practical Byzantine Fault Tolerance (PBFT) algorithm serves as a consensus mechanism engineered to establish agreement within a distributed network of nodes, even when confronted with Byzantine faults—instances where nodes may exhibit arbitrary or malicious behavior. Crafted by Castro and Liskov in 1999, PBFT stands as a pioneering algorithm in the realm of distributed consensus. PBFT divides the network into a collection of replica nodes, typically comprising at least $3f+1$ replicas, where 'f' symbolizes the maximum number of faulty or malicious nodes the system can endure while retaining consistency. Each replica boasts a unique identifier and engages in communication with other replicas across the network. Through its design and operation, PBFT provides a robust framework for achieving consensus in the face of potential faults or adversarial behavior, marking a significant advancement in the field of distributed systems and consensus protocols.

The consensus process in PBFT involves several phases:

1. **Request:** A client sends a request to the network, specifying the operation it wants to be executed.
2. **Pre-Prepare:** The primary replica (leader) assigns a sequence number to the request and sends a pre-prepare message to other replicas, proposing the operation and sequence number.
3. **Prepare:** Upon receiving a pre-prepare message, each replica verifies the request and then broadcasts a prepare message to indicate acceptance of the proposed operation and sequence number.
4. **Commit:** Once a replica receives $2f+1$ prepare messages for the same sequence number and operation, it broadcasts a commit message to finalize the decision.
5. **Execute:** Replicas execute the operation specified in the request, applying it to their local state machine.

PBFT integrates checkpointing to boost efficiency and minimize message overhead. Checkpoint messages enable replicas to discard outdated messages and streamline the recovery process in the event of failures. This mechanism enhances the system's resilience and ensures smoother operation, especially during disruptions or system failures, contributing to a more robust and reliable consensus protocol.

PBFT manages leader failures via a view change mechanism. When a replica detects potential faults in the primary, it triggers a view change. This process facilitates the network's transition to a new primary replica, ensuring seamless continuation of the consensus process. By swiftly addressing leader failures, PBFT enhances the resilience and reliability of the distributed system, maintaining consistent operation.

PBFT exhibits remarkable fault tolerance, capable of withstanding up to f Byzantine faults while guaranteeing safety and liveness. Safety ensures that no two distinct nodes commit different transactions with the same sequence number, while liveness ensures that every correct node eventually commits a request. Furthermore, PBFT offers low latency and high throughput, rendering it ideal for applications demanding swift transaction finality and resilience against Byzantine faults.

Ultimately, the PBFT consensus algorithm provides robust solutions for achieving distributed agreement in a Byzantine fault-tolerant manner, making it suitable for various blockchain and distributed systems applications where trust and reliability are paramount.

3.2.3 System Architecture

Figure 3.1 illustrates the proposed framework, a blockchain-based system designed for vaccine supply chain management.

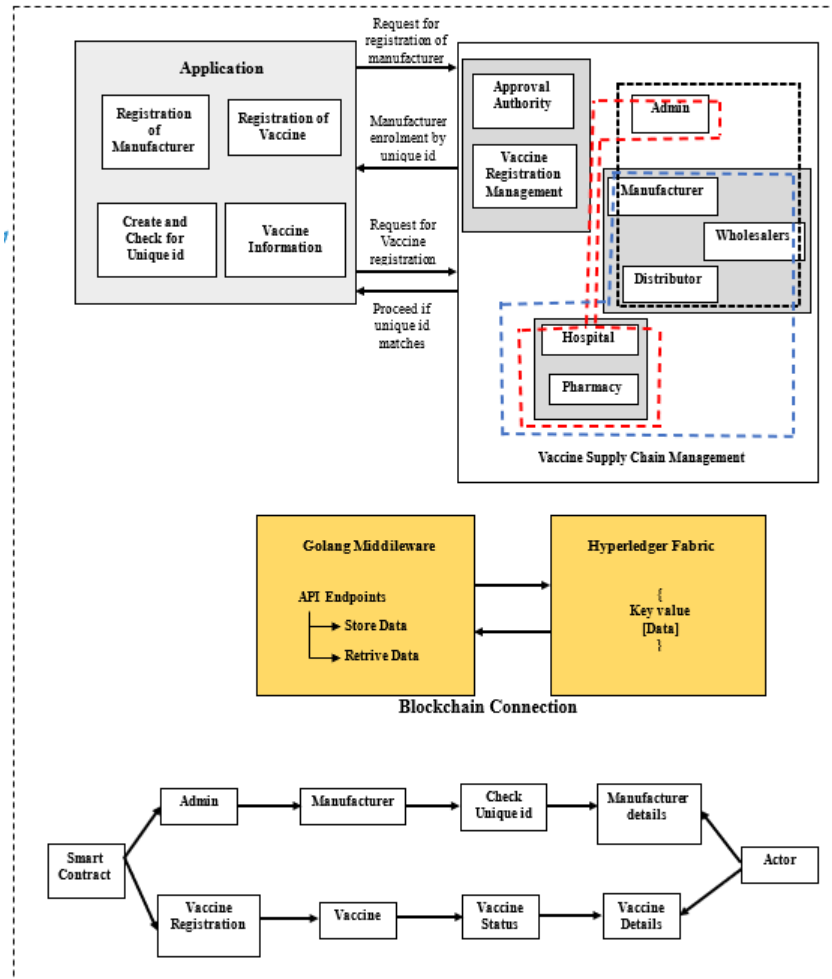


Figure 3.1: System Architecture

The framework involves various entities, including regulatory authorities, manufacturers, distributors, wholesalers, pharmacies/hospitals, and consumers. The diagram visually represents the flow of the entire process, with all information being stored on the blockchain to ensure immutability, transparency, and security of vaccine supply chain management. On the left side of the diagram, all the actors involved in the supply chain are listed, while on the right side, the specific activities of each actor are detailed. The bottom of the diagram contains the flow of the entire process, showcasing the movement of vaccines from the manufacturer to

the various entities involved, such as distributors, wholesalers, pharmacies/hospitals, and ultimately to the end consumer. The application of blockchain technology is facilitated by smart contracts, which handle the registration of the manufacturer and vaccine by assigning an offline unique address to the manufacturer. The framework ensures the integrity and immutability of vaccine data at every node, with information stored on the blockchain at each step to ensure its immutability and integrity.

It shows the proposed framework in which actors like regulatory authorities, manufacturers, distributors, whole salers, pharmacies/hospitals, and consumers take their part in vaccine supply chain management. The application of blockchain is based on smart contracts which are composed of the registration of the manufacturer and vaccine by giving an offline unique address to the manufacturer. If the unique address of the manufacturer matches, then that specific manufacturer can produce a specific vaccine. Distributors can deliver it to the wholesaler, from where it can be dispatched to hospitals/pharmacies, so consumers can buy and use it. On the left side, all the actors are mentioned in Fig. 3.1, and on the right side, the actor and their relevant activities are mentioned. The bottom of the diagram contains the flow of the entire process. In every step, all the information is stored on the blockchain which ensures the immutability, transparency, and security of vaccine supply chain management.

3.2.4 Database Schema

The database in Hyperledger Fabric allows for the storage of off-chain data, including sensitive information or large datasets unsuitable for on-chain storage due to scalability or privacy concerns.

Important tables

Table 3.1: User Login Table

Field Name	Datatype	Constraints
log_id	AutoField	PRIMARY KEY
username	CharField	
password	CharField	
role	CharField	

Table 3.1 stores user login information.

Table 3.2: Vaccine Table

Field Name	Datatype	Constraints
vacc_id	AutoField	PRIMARY KEY
name	CharField	
description	CharField	
composition	CharField	
pharmaceutical_form	CharField	
age	CharField	
method_of_administration	CharField	
contraindications	CharField	
precautions	CharField	

Table 3.2 stores essential vaccine information.

Table 3.3: Manufacturer Table

Field Name	Datatype	Constraints
manufacture_id	AutoField	PRIMARY KEY
company_name	CharField	
address	CharField	
licence_no	CharField	
phone_no	CharField	
login_id	IntegerField	FOREIGN KEY
status	CharField	

Table 3.3 contains essential details about vaccine manufacturers.

Table 3.4: Vaccinerequest Table

Field Name	Datatype	Constraints
vaccreq_id	AutoField	PRIMARY KEY
vacc_id	IntegerField	FOREIGN KEY
manufacture_id	IntegerField	FOREIGN KEY
vaccine_name	CharField	
date	CharField	
required_documents	CharField	
additional_details	CharField	
stock	CharField	
status	CharField	

Table 3.4 stores the vaccine requests made by the manufacturer to the system for approving vaccine manufacturing.

3.2.5 System Specification

System specification provides an overview of the software or application including what it should do and what its parameters are, how it will interact with its environment, end users, hardware and software requirements.

Hardware Specification

- CPU: Intel Core i5 or higher
- RAM: 8 GB or higher
- Hard disk: 512 GB or higher
- Input Device: Mouse, Keyboard
- Output Device: Monitor with 1280 x 720 resolution, Printer

Software Specification

- Blockchain Development Framework: Hyperledger Fabric
- Smart Contract Development: Golang (Go)

- Database: MySQL
- IDE: Visual Studio Code
- Containerization Platform: Docker
- Front-end Framework: Django

3.2.6 Software Tools

1. Hyperledger Fabric

Hyperledger Fabric [9] serves as the foundational framework for constructing a permissioned blockchain network tailored precisely to the unique needs of the vaccine supply chain management project. Through its customizable access control mechanisms, the platform empowers administrators to finely manage permissions for network participants, ensuring precise control over data access and operations. Furthermore, Hyperledger Fabric boasts robust security features, including private transactions and endorsement policies, which collectively safeguard the privacy and integrity of data circulating within the blockchain network. These features culminate in a secure and efficient infrastructure, vital for maintaining trust and reliability in the vaccine supply chain ecosystem.

2. Docker

Docker [10] plays a pivotal role in streamlining the development and deployment processes within the vaccine supply chain management system, particularly in conjunction with Hyperledger Fabric. By containerizing applications and their dependencies, Docker ensures consistency across various development and deployment environments. This consistency is crucial for maintaining uniformity and predictability in the system's behavior. Furthermore, Docker's ability to package software into lightweight containers simplifies sharing and running applications across different systems, fostering seamless collaboration among developers and facilitating deployment across diverse infrastructure setups. Additionally, Docker optimizes resource utilization and offers scalability features, allowing the vaccine supply chain management system to efficiently scale as demand grows. These capabilities make Docker an ideal choice for deploying and managing the complex network infrastructure required by Hyperledger Fabric, ensuring smooth and efficient operation of the entire system.

3. **Golang (Programming Language)**

Golang [11] commonly referred to as Go, plays a pivotal role in the development of the vaccine supply chain management system, serving as the primary language for coding smart contracts and building the backend infrastructure. Its attributes, including simplicity, efficiency, and inherent support for concurrency, render it exceptionally suitable for crafting scalable and dependable blockchain applications. Moreover, Golang's statically typed nature and integrated garbage collection mechanism contribute to the robustness and maintainability of the codebase. These features streamline the development process, enabling the creation of intricate smart contracts and backend services essential for the vaccine supply chain management system.

4. **Django**

Django [12] is a high-level Python web framework known for its simplicity and rapid development capabilities. It features an MVC architecture, ORM for database interactions, built-in admin interface, URL routing, template engine, and security features. Its vibrant community and extensive ecosystem make it ideal for building scalable and secure web applications.

The proposed blockchain-based vaccine supply chain management system offers a transformative solution to address challenges in the current system. Leveraging Hyperledger Fabric and PBFT algorithm, it ensures transparency, security, and efficiency. With a structured methodology and robust software tools, the system promises resilience, scalability, and reliability in managing vaccine distribution.

Chapter 4

RESULTS AND ANALYSIS

In this chapter, significant achievements in transparency, security, and efficiency are highlighted. A detailed performance analysis of Hyperledger Fabric explores metrics such as throughput, latency, and scalability. Notably, transaction volume impacts latency and throughput, with different transaction types affecting network performance. This assessment provides insights into the platform's capabilities and scalability.

4.1 Results

Based on the comparison between the proposed system and the existing system, the project demonstrates significant advancements and achievements:

Immutability: The utilization of the SHA3-256 hashing algorithm ensures data remains unalterable, enhancing the security and integrity of vaccine-related information. SHA3-256 is a cryptographic hash function that transforms any data into a fixed-size, unique fingerprint.

Data Integrity: PBFT ensures data consistency by replicating the ledger across nodes. A leader broadcasts transactions, and replicas vote for validity. If a sufficient number of honest nodes agree on the validity of the transaction, it is considered committed to the ledger. This multi-step process with redundancy makes it highly resistant to data tampering or manipulation by malicious actors.

Transparency: The adoption of a shared ledger fosters real-time visibility into vaccine distribution processes, promoting transparency and accountability among stakeholders.

Traceability: Unique identifiers incorporated into the system enable precise tracking of vaccine batches throughout their journey, improving traceability and reducing the risk of counterfeit infiltration.

Security: Cryptographic features, such as public key encryption, enhance security by preventing unauthorized access to sensitive vaccine-related data, safeguarding against potential threats and breaches.

Decentralization: By leveraging a distributed ledger, the proposed system enhances system resilience, eliminating single points of failure and ensuring continuous operation even in the face of disruptions.

Efficiency: Smart contracts automate processes and reduce errors, streamlining vaccine supply chain management and improving overall efficiency.

Table 4.1: Outcomes Achieved

Feature	Mechanism	Benefits Achieved
Immutability	SHA3-256 Hashing Algorithm	Data Remains Unalterable
Data Integrity	PBFT Consensus Algorithm	Ensures Data Consistency
Transparency	Shared Ledger	Enables Real-time Visibility
Traceability	Unique Identifiers	Facilitates Precise Tracking
Security	Cryptographic Features	Prevents Unauthorized Access
Decentralization	Distributed Ledger	Enhances System Resilience
Efficiency	Smart Contracts	Automates Processes & Reduces Errors

The project successfully achieves its objectives of enhancing transparency, security, traceability, and efficiency in vaccine supply chain management through the implementation of blockchain technology, specifically Hyperledger Fabric, and associated cryptographic mechanisms and consensus algorithms.

4.2 Performance Analysis

The performance analysis of the Hyperledger Fabric platform focused on evaluating several key factors, including throughput, latency, and scalability. The evaluation was conducted using the Hyperledger Caliper benchmark tool, which allowed for the representation of multiple clients injecting workloads into the blockchain network. The performance evaluation considered varying transaction rates and transaction types.

Throughput, measured in terms of successful transactions per second (tps), was a crucial metric assessed in the performance analysis. The study found that the Hyperledger Fabric platform demonstrated robust throughput capabilities, supporting a significant number of transactions efficiently. However, the analysis revealed that as the transaction rate increased beyond a certain threshold, the blockchain throughput decreased, indicating potential scalability limitations under high transaction volumes.

Latency, representing the response time per transaction in seconds, was another important aspect evaluated. The study demonstrated that the Hyperledger Fabric platform exhibited low

latency, with response times in the order of a fraction of a second. This low latency is essential for ensuring quick transaction processing and maintaining a responsive blockchain network.

Scalability, referring to the platform’s ability to accommodate an increasing number of participants or users, was also assessed. The analysis indicated that Hyperledger Fabric demonstrated good scalability, supporting a high number of participants without compromising performance. However, the study highlighted the importance of optimizing network configurations and hardware resources to maintain scalability under varying transaction rates and volumes.

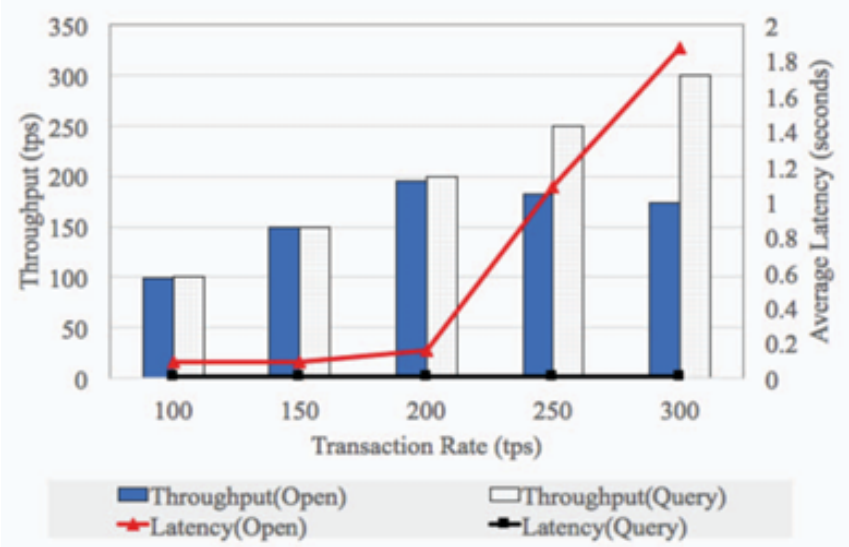


Figure 4.1: Impact of Transaction Rates and Transaction Types on the Blockchain Throughput and Latency

Figure 4.1 [13] illustrates that with "open" transactions, the network supported up to 200 tps without significant latency. Beyond 200 tps, throughput decreased, and latency increased. Conversely, for "query" transactions, the network handled 300 tps with minimal latency, indicating capacity for higher transaction rates.

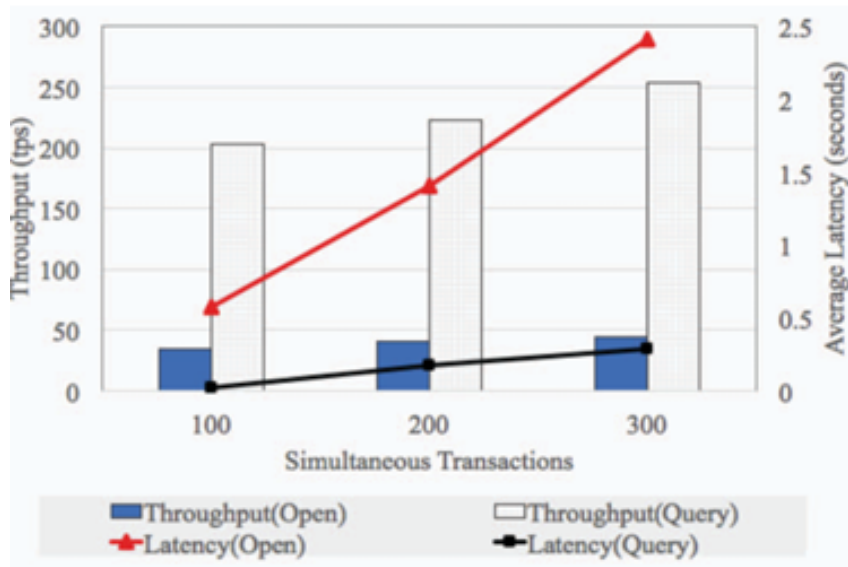


Figure 4.2: Impact of the Number of Participants on Throughput and Latency

In Figure 4.2 [13], both throughput and latency rise with increasing simultaneous transactions. For "open" transactions, throughputs were 34.6, 40.3, and 44.8 tps for 100, 200, and 300 simultaneous transactions, respectively, with increasing latency. Conversely, "query" transactions had higher throughput and lower latency due to their single-read operation, handling 200 simultaneous transactions without queuing, but queuing occurred at 300 transactions.

In general, the performance of the blockchain network is significantly impacted by an increase in simultaneous transactions, particularly affecting latency. Transaction type plays a crucial role in network performance. Additionally, while throughput remains relatively stable, latency increases as the system approaches its maximum capacity. The scalability assessment based on simultaneous transactions and transaction types reveals a direct correlation between increased transaction volume and its impact on both throughput and latency.

The project achieves its goals by implementing Hyperledger Fabric for transparency, security, and efficiency in vaccine distribution. Performance analysis confirms robust throughput, low latency, and good scalability, demonstrating the platform's suitability for real-world applications in vaccine supply chain management.

Chapter 5

CONCLUSION

The project successfully addresses the challenges faced by the pharmaceutical industry in vaccine supply chain management by implementing a decentralized system using Hyperledger Fabric. It achieves key milestones by enhancing transparency, security, traceability, and efficiency in vaccine distribution. By leveraging blockchain technology, the project ensures data integrity, mitigates counterfeit risks, and fosters trust among stakeholders. The project contributes to solving the problem of opaque and vulnerable vaccine supply chains by introducing a transparent and tamper-resistant ledger system. Through blockchain technology, it provides real-time visibility into vaccine movement, prevents data tampering, and reduces the risk of counterfeit infiltration. This ensures the authenticity and integrity of vaccine-related data, ultimately improving public health outcomes. However, limitations of the project may include the need for continuous optimization and adaptation to evolving technology and regulatory requirements. Additionally, ensuring widespread adoption and interoperability with existing systems may pose challenges. Despite these limitations, the project demonstrates significant progress in revolutionizing vaccine supply chain management through blockchain technology.

Chapter 6

FUTURE SCOPE

To extend the project's scope and enhance its effectiveness, several potential future directions can be explored, building upon the foundation laid by the implementation in Hyperledger Fabric. Firstly, real-world testing in larger environments is crucial to validate the framework's performance and feasibility, particularly focusing on aspects such as network scalability and user adoption rates. Enhancing user interaction by upgrading feedback systems and potentially integrating customer reputation mechanisms would improve reliability and ensure a seamless experience for stakeholders. Integration of cold chain processes, utilizing sensors for efficient vaccine transport and storage, would enhance the integrity and safety of vaccines throughout the supply chain. Continuously refining performance metrics, such as transaction speed, can optimize overall system efficiency. Adapting the framework to diverse global scenarios and addressing region-specific challenges in vaccine distribution is essential for its broader applicability. Additionally, improving interoperability with other healthcare technologies and standards would facilitate seamless integration with existing infrastructure. Strengthening data privacy and security measures to comply with emerging regulations and protect patient information is imperative. Furthermore, analyzing the impact of block time variations on network performance would provide insights for enhancing scalability and efficiency. These potential future directions and areas for further research or development will contribute to the framework's effectiveness and relevance in real-world applications, ultimately improving vaccine supply chain management.

REFERENCES

- [1] Muhammad Rehman et al. “A cyber secure medical management system by using blockchain”. In: *IEEE Transactions on Computational Social Systems* (2022).
- [2] Laizhong Cui et al. “Protecting vaccine safety: An improved, blockchain-based, storage-efficient scheme”. In: *IEEE Transactions on Cybernetics* (2022).
- [3] Tomilayo Fatokun, Avishek Nag, and Sachin Sharma. “Towards a blockchain assisted patient owned system for electronic health records”. In: *Electronics* 10.5 (2021), p. 580.
- [4] Faisal Jamil et al. “A novel medical blockchain model for drug supply chain integrity management in a smart hospital”. In: *Electronics* 8.5 (2019), p. 505.
- [5] Anuja R Nair, Rajesh Gupta, and Sudeep Tanwar. “FAIR: A blockchain-based vaccine distribution scheme for pandemics”. In: *2021 IEEE Globecom Workshops (GC Wkshps)*. IEEE. 2021, pp. 1–6.
- [6] Ibrahim Tariq Javed et al. “PETchain: A blockchain-based privacy enhancing technology”. In: *IEEE Access* 9 (2021), pp. 41129–41143.
- [7] Ahmad Musamih et al. “Blockchain-based solution for distribution and delivery of COVID-19 vaccines”. In: *Ieee Access* 9 (2021), pp. 71372–71387.
- [8] Shashi Bhushan et al. “Blockchain Powered Vaccine Efficacy for Pharma Sector.” In: *Computational & Mathematical Methods in Medicine* (2022).
- [9] Nitin Gaur et al. *Blockchain with hyperledger fabric: Build decentralized applications using hyperledger fabric 2*. Packt Publishing Ltd, 2020.
- [10] GeeksforGeeks. *Containerization using Docker*. Sept. 2019. URL: <https://www.geeksforgeeks.org/containerization-using-docker/>.
- [11] Alan AA Donovan and Brian W Kernighan. *The Go programming language*. Addison-Wesley Professional, 2015.
- [12] GeeksforGeeks. *Django Basics*. Feb. 2020. URL: <https://www.geeksforgeeks.org/django-tutorial/>.
- [13] Murat Kuzlu et al. “Performance analysis of a hyperledger fabric blockchain framework: throughput, latency and scalability”. In: *2019 IEEE international conference on blockchain (Blockchain)*. IEEE. 2019, pp. 536–540.

APPENDIX

Sample Code

Main

```
1 package main
2
3 import (
4     "fmt"
5     "rest-api-go/web"
6 )
7
8 func main() {
9     //Initialize setup for Org1
10    cryptoPath := "../test-network/organizations/peerOrganizations/
11        org1.example.com"
12    orgConfig := web.OrgSetup{
13        OrgName:      "Org1",
14        MSPID:         "Org1MSP",
15        CertPath:      cryptoPath + "/users/User1@org1.example.com/msp/
16            signcerts/User1@org1.example.com-cert.pem",
17        KeyPath:      cryptoPath + "/users/User1@org1.example.com/msp/
18            keystore/",
19        TLSCertPath:  cryptoPath + "/peers/peer0.org1.example.com/tls/ca.
20            crt",
21        PeerEndpoint: "localhost:7051",
22        GatewayPeer:  "peer0.org1.example.com",
23    }
24
25    orgSetup, err := web.Initialize(orgConfig)
26    if err != nil {
27        fmt.Println("Error initializing setup for Org1: ", err)
28    }
29    web.Bootup(web.OrgSetup(*orgSetup))
30 }
```

Smartcontract

```
1 package chaincode
2
3 import (
4     "encoding/hex"
5     "encoding/json"
6     "fmt"
7     "strconv"
8
9     "golang.org/x/crypto/sha3"
10
11     "github.com/hyperledger/fabric-contract-api-go/contractapi"
12 )
13
14 // SmartContract provides functions for managing vaccine supply chain
15 type SmartContract struct {
16     contractapi.Contract
17 }
18
19 type VaccineInfo struct {
20     VaccineID          string          `json:"VaccineID"`
21     VaccineName        string          `json:"VaccineName"`
22     AgeGroup           string          `json:"AgeGroup"`
23     ContraIndications string          `json:"ContraIndications"`
24     MethodOfAdministration string      `json:"MethodOfAdministration"`
25     VaccineDescription string          `json:"VaccineDescription"`
26     PharmaceuticalForm string          `json:"PharmaceuticalForm"`
27     Precautions         string          `json:"Precautions"`
28     ManufacturerId      string          `json:"ManufacturerId"`
29     VaccineExpiryDetail VaccineExpiryDetail `json:"VaccineExpiryDetails"`
30 }
```

```

30  DistributorId          string          `json:"DistributorId"`
31  HospitalId            string          `json:"HospitalId"`
32  PreviousBlockHash     string          `json:"PreviousBlockHash"
    " `
33  BlockHash              string          `json:"BlockHash"`
34 }
35
36 type VaccineExpiryDetail struct {
37     ManufacturingDate string
38     ExpiryDate         string
39 }
40
41 func createBlockHash(content string) string {
42
43     // Create a new SHA3-256 hasher
44     hasher := sha3.New256()
45
46     // Write the content to the hasher
47     hasher.Write([]byte(content))
48
49     // Get the finalized hash
50     hashBytes := hasher.Sum(nil)
51
52     // Convert the hash to a hexadecimal string
53     hashString := hex.EncodeToString(hashBytes)
54
55     // Print the hash
56     fmt.Println("SHA3-256 hash:", hashString)
57
58     return hashString
59 }
60
61 func (s *SmartContract) AddVaccine(ctx contractapi.
    TransactionContextInterface, vaccineId string, vaccineName string,
    ageGroup string, contraIndications string, methodOfAdministration

```

```

    string, vaccineDescription string, pharmaceuticalForm string,
    precautions string) error {
62 exists, err := s.CheckVaccineExists(ctx, vaccineId)
63 if err != nil {
64     return err
65 }
66 if exists {
67     return fmt.Errorf("the vaccine Id %s already exists", vaccineId)
68 }
69 integerValue, err := strconv.Atoi(vaccineId)
70 if err != nil {
71     fmt.Println("Error:", err)
72     return err
73 }
74 previousId := strconv.Itoa(integerValue - 1)
75 previousExists, err := s.CheckVaccineExists(ctx, previousId)
76 if err != nil {
77     return err
78 }
79 var previousBlockHash string
80 if previousExists {
81     vaccineInfoBytes, err := ctx.GetStub().GetState(previousId)
82     if err != nil {
83         return fmt.Errorf("failed to read VaccineInfo with ID %s: %v",
previousId, err)
84     }
85
86     if vaccineInfoBytes == nil {
87         return fmt.Errorf("VaccineInfo with ID %s does not exist",
previousId)
88     }
89
90     // Unmarshal the current state into a struct
91     var existingVaccineInfo VaccineInfo
92     err = json.Unmarshal(vaccineInfoBytes, &existingVaccineInfo)

```

```

93     if err != nil {
94         return fmt.Errorf("error unmarshalling VaccineInfo JSON: %v",
err)
95     }
96     previousBlockHash = existingVaccineInfo.BlockHash
97 }
98
99 //blockHash := createBlockHash(vaccineId + vaccineName +
vaccineType + manufacturer)
100
101 vaccineDetails := VaccineInfo{
102     VaccineID:          vaccineId,
103     VaccineName:        vaccineName,
104     AgeGroup:           ageGroup,
105     ContraIndications:  contraIndications,
106     MethodOfAdministration: methodOfAdministration,
107     VaccineDescription: vaccineDescription,
108     PharmaceuticalForm: pharmaceuticalForm,
109     Precautions:        precautions,
110     PreviousBlockHash:  previousBlockHash,
111 }
112 vaccineDetailsJSON, err := json.Marshal(vaccineDetails)
113 if err != nil {
114     return err
115 }
116
117 return ctx.GetStub().PutState(vaccineId, vaccineDetailsJSON)
118 }
119
120 func (s *SmartContract) AddVaccineExpiryDetail(ctx contractapi.
TransactionContextInterface, vaccineID string, manufacturingDate
string, expiryDate string) error {
121 // Retrieve the current state of the VaccineInfo
122 vaccineInfoBytes, err := ctx.GetStub().GetState(vaccineID)
123 if err != nil {

```



```

124     return fmt.Errorf("failed to read VaccineInfo with ID %s: %v",
        vaccineID, err)
125 }
126
127 if vaccineInfoBytes == nil {
128     return fmt.Errorf("VaccineInfo with ID %s does not exist",
        vaccineID)
129 }
130
131 // Unmarshal the current state into a struct
132 var existingVaccineInfo VaccineInfo
133 err = json.Unmarshal(vaccineInfoBytes, &existingVaccineInfo)
134 if err != nil {
135     return fmt.Errorf("error unmarshalling VaccineInfo JSON: %v", err
        )
136 }
137 vaccineExpiryInfo := VaccineExpiryDetail{
138     ManufacturingDate: manufacturingDate,
139     ExpiryDate:        expiryDate,
140 }
141 existingVaccineInfo.VaccineExpiryDetail = vaccineExpiryInfo
142 existingVaccineInfo.BlockHash = createBlockHash(existingVaccineInfo
    .VaccineID + existingVaccineInfo.AgeGroup + existingVaccineInfo.
    VaccineName + existingVaccineInfo.ContraIndications +
    existingVaccineInfo.DistributorId + existingVaccineInfo.HospitalId
    + existingVaccineInfo.HospitalId + existingVaccineInfo.
    ManufacturerId + existingVaccineInfo.MethodOfAdministration +
    existingVaccineInfo.PharmaceuticalForm + existingVaccineInfo.
    Precautions + existingVaccineInfo.VaccineDescription)
143 // Marshal the updated VaccineInfo back to JSON
144 updatedVaccineInfoBytes, err := json.Marshal(existingVaccineInfo)
145 if err != nil {
146     return fmt.Errorf("error marshalling updated VaccineInfo: %v",
        err)
147 }

```

```

148
149 // Update the VaccineInfo state on the ledger
150 err = ctx.GetStub().PutState(vaccineID, updatedVaccineInfoBytes)
151 if err != nil {
152     return fmt.Errorf("error updating VaccineInfo on the ledger: %v",
153         err)
154 }
155 return nil
156 }
157
158 func (s *SmartContract) AddManufactureDetail(ctx contractapi.
159     TransactionContextInterface, vaccineID string, manufacturerId
160     string) error {
161     // Retrieve the current state of the Vaccine
162     vaccineInfoBytes, err := ctx.GetStub().GetState(vaccineID)
163     if err != nil {
164         return fmt.Errorf("failed to read VaccineInfo with ID %s: %v",
165             vaccineID, err)
166     }
167
168     if vaccineInfoBytes == nil {
169         return fmt.Errorf("VaccineInfo with ID %s does not exist",
170             vaccineID)
171     }
172
173     // Unmarshal the current state into a struct
174     var existingVaccineInfo VaccineInfo
175     err = json.Unmarshal(vaccineInfoBytes, &existingVaccineInfo)
176     if err != nil {
177         return fmt.Errorf("error unmarshalling VaccineInfo JSON: %v", err)
178     }
179
180     existingVaccineInfo.ManufacturerId = manufacturerId

```

```

177
178 // Marshal the updated VaccineInfo back to JSON
179 updatedVaccineInfoBytes, err := json.Marshal(existingVaccineInfo)
180 if err != nil {
181     return fmt.Errorf("error marshalling updated VaccineInfo: %v",
182         err)
183 }
184 // Update the VaccineInfo state on the ledger
185 err = ctx.GetStub().PutState(vaccineID, updatedVaccineInfoBytes)
186 if err != nil {
187     return fmt.Errorf("error updating VaccineInfo on the ledger: %v",
188         err)
189 }
190 return nil
191 }
192
193 func (s *SmartContract) AddHospitalDetail(ctx contractapi.
194     TransactionContextInterface, vaccineID string, hospitalId string)
195     error {
196     // Retrieve the current state of the Vaccine
197     vaccineInfoBytes, err := ctx.GetStub().GetState(vaccineID)
198     if err != nil {
199         return fmt.Errorf("failed to read VaccineInfo with ID %s: %v",
200             vaccineID, err)
201     }
202     if vaccineInfoBytes == nil {
203         return fmt.Errorf("VaccineInfo with ID %s does not exist",
204             vaccineID)
205     }
206     // Unmarshal the current state into a struct
207     var existingVaccineInfo VaccineInfo

```

```

206     err = json.Unmarshal(vaccineInfoBytes, &existingVaccineInfo)
207     if err != nil {
208         return fmt.Errorf("error unmarshalling VaccineInfo JSON: %v", err
209     )
210     }
211     existingVaccineInfo.HospitalId = hospitalId
212
213     // Marshal the updated VaccineInfo back to JSON
214     updatedVaccineInfoBytes, err := json.Marshal(existingVaccineInfo)
215     if err != nil {
216         return fmt.Errorf("error marshalling updated VaccineInfo: %v",
217             err)
218     }
219     // Update the VaccineInfo state on the ledger
220     err = ctx.GetStub().PutState(vaccineID, updatedVaccineInfoBytes)
221     if err != nil {
222         return fmt.Errorf("error updating VaccineInfo on the ledger: %v",
223             err)
224     }
225     return nil
226 }
227
228 func (s *SmartContract) AddDistributorDetail(ctx contractapi.
    TransactionContextInterface, vaccineID string, distributorId
    string) error {
229     // Retrieve the current state of the Vaccine
230     vaccineInfoBytes, err := ctx.GetStub().GetState(vaccineID)
231     if err != nil {
232         return fmt.Errorf("failed to read VaccineInfo with ID %s: %v",
233             vaccineID, err)
234     }

```

```

235     if vaccineInfoBytes == nil {
236         return fmt.Errorf("VaccineInfo with ID %s does not exist",
            vaccineID)
237     }
238
239     // Unmarshal the current state into a struct
240     var existingVaccineInfo VaccineInfo
241     err = json.Unmarshal(vaccineInfoBytes, &existingVaccineInfo)
242     if err != nil {
243         return fmt.Errorf("error unmarshalling VaccineInfo JSON: %v", err
            )
244     }
245
246     existingVaccineInfo.DistributorId = distributorId
247
248     // Marshal the updated VaccineInfo back to JSON
249     updatedVaccineInfoBytes, err := json.Marshal(existingVaccineInfo)
250     if err != nil {
251         return fmt.Errorf("error marshalling updated VaccineInfo: %v",
            err)
252     }
253
254     // Update the VaccineInfo state on the ledger
255     err = ctx.GetStub().PutState(vaccineID, updatedVaccineInfoBytes)
256     if err != nil {
257         return fmt.Errorf("error updating VaccineInfo on the ledger: %v",
            err)
258     }
259
260     return nil
261 }
262
263 // VaccineExists check the ledger whether the vaccine respective to
    the particular id exists
264 func (s *SmartContract) CheckVaccineExists(ctx contractapi.

```

```

    TransactionContextInterface, vaccineId string) (bool, error) {
265 vaccineDetailsJSON, err := ctx.GetStub().GetState(vaccineId)
266 if err != nil {
267     return false, fmt.Errorf("failed to read from world state: %v",
        err)
268 }
269
270 return vaccineDetailsJSON != nil, nil
271 }
272
273 // ReadAsset returns the asset stored in the world state with given
    id.
274 func (s *SmartContract) GetVaccineDetails(ctx contractapi.
    TransactionContextInterface, vaccineId string) (*VaccineInfo,
    error) {
275 vaccineDetailsJSON, err := ctx.GetStub().GetState(vaccineId)
276 if err != nil {
277     return nil, fmt.Errorf("failed to read from world state: %v", err
        )
278 }
279 if vaccineDetailsJSON == nil {
280     return nil, fmt.Errorf("the vaccine details %s does not exist",
        vaccineId)
281 }
282
283 var vaccineDetails VaccineInfo
284 err = json.Unmarshal(vaccineDetailsJSON, &vaccineDetails)
285 if err != nil {
286     return nil, err
287 }
288
289 return &vaccineDetails, nil
290 }
291
292 func (s *SmartContract) GetAllVaccineDetails(ctx contractapi.

```

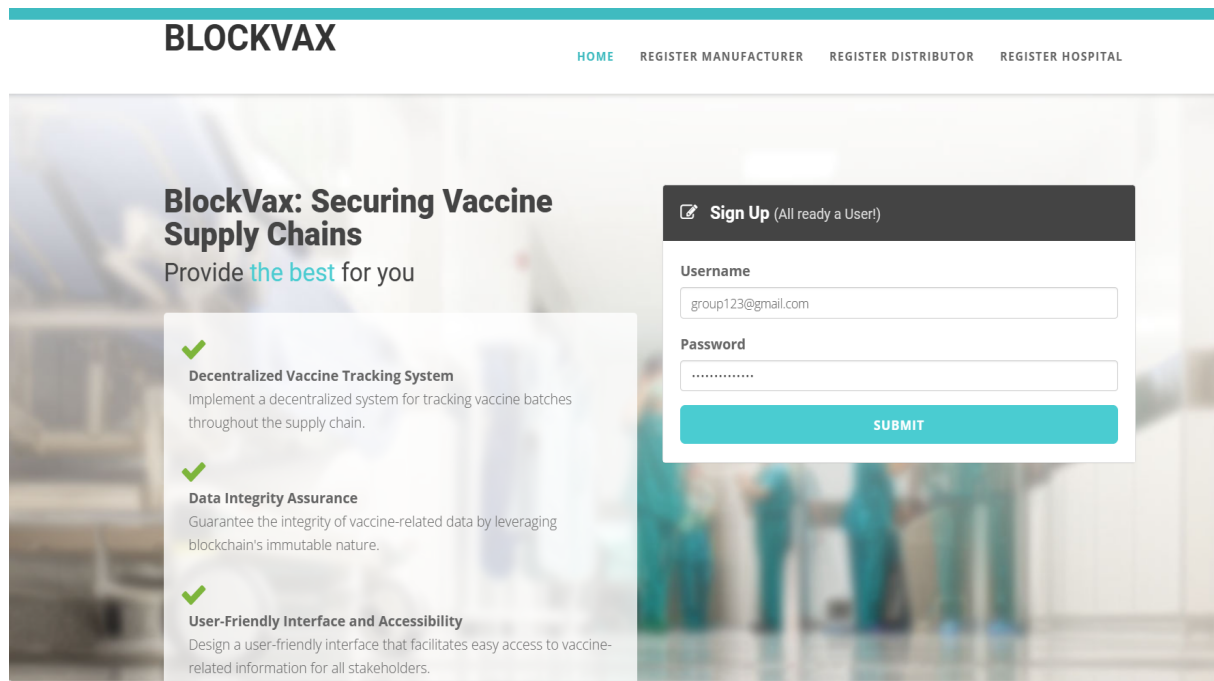
```

    TransactionContextInterface) ([]*VaccineInfo, error) {
293 // range query with empty string for startKey and endKey does an
294 // open-ended query of all assets in the chaincode namespace.
295 resultsIterator, err := ctx.GetStub().GetStateByRange("", "")
296 if err != nil {
297     return nil, err
298 }
299 defer resultsIterator.Close()
300
301 var vaccineDetails []*VaccineInfo
302 for resultsIterator.HasNext() {
303     queryResponse, err := resultsIterator.Next()
304     if err != nil {
305         return nil, err
306     }
307
308     var vaccineDetail VaccineInfo
309     err = json.Unmarshal(queryResponse.Value, &vaccineDetail)
310     if err != nil {
311         return nil, err
312     }
313     vaccineDetails = append(vaccineDetails, &vaccineDetail)
314 }
315
316 return vaccineDetails, nil
317 }

```

Project Screenshots

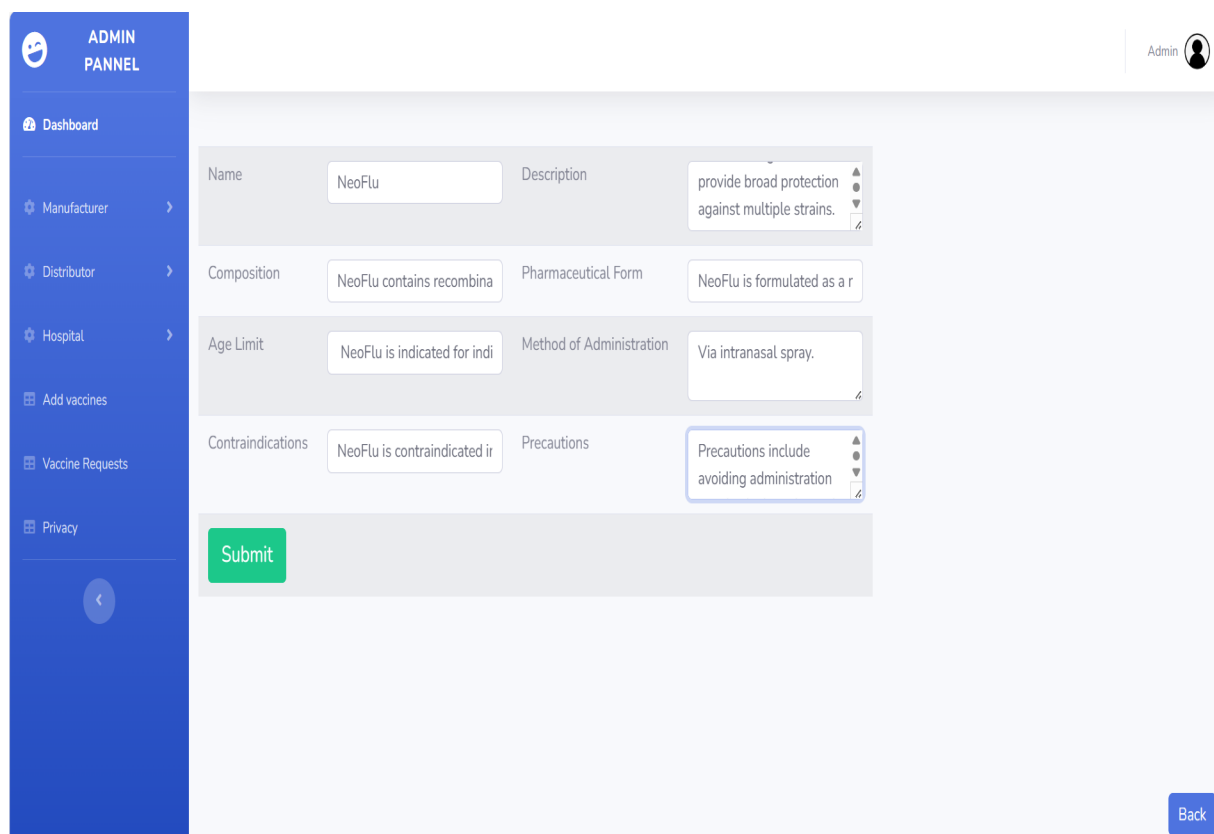
User Login



The screenshot shows the BlockVax user login page. At the top, the 'BLOCKVAX' logo is on the left, and navigation links for 'HOME', 'REGISTER MANUFACTURER', 'REGISTER DISTRIBUTOR', and 'REGISTER HOSPITAL' are on the right. The main content area has a background image of a hospital corridor. On the left, a section titled 'BlockVax: Securing Vaccine Supply Chains' with the tagline 'Provide the best for you' lists three features: 'Decentralized Vaccine Tracking System', 'Data Integrity Assurance', and 'User-Friendly Interface and Accessibility'. On the right, a 'Sign Up' form is displayed with fields for 'Username' (containing 'group123@gmail.com') and 'Password' (masked with dots), and a 'SUBMIT' button.

Figure 6.1: User Login

Add Vaccine



The screenshot shows the 'Add Vaccine' form in the BlockVax Admin Panel. The left sidebar contains navigation links: 'Dashboard', 'Manufacturer', 'Distributor', 'Hospital', 'Add vaccines', 'Vaccine Requests', and 'Privacy'. The main form area is titled 'ADMIN PANNEL' and includes a 'Back' button in the top right. The form fields are: 'Name' (NeoFlu), 'Description' (provide broad protection against multiple strains.), 'Composition' (NeoFlu contains recombina), 'Pharmaceutical Form' (NeoFlu is formulated as a r), 'Age Limit' (NeoFlu is indicated for indi), 'Method of Administration' (Via intranasal spray.), 'Contraindications' (NeoFlu is contraindicated ir), and 'Precautions' (Precautions include avoiding administration). A green 'Submit' button is at the bottom left of the form.

Figure 6.2: Add Vaccine

Blockchain Transaction

```
gopikagopikrishnan:/mnt/c/Users/gopik/MainProject_VaccinesupplyChainManagement/forensic-setup/fabric-samples/fabric-setup/rest-api-go> go run main.go
2024/04/15 10:31:20 Initializing connection for Org1...
2024/04/15 10:31:20 Initialization complete
2024/04/15 10:31:20 Server Started
2024/04/15 10:39:14 {20 Immunex Immunex is approved for individuals aged 18 and above Immunex is contraindicated in individuals with a history of severe allergic reactions to any component of the vaccine or a previous dose of Immunex. Immunex is administered via intramuscular injection. Immunex is a novel vaccine developed to bolster the body's immune response against a specific pathogen. Immunex is formulated as a sterile suspension for injection. Immunex should be used with caution in individuals with compromised immune systems, as the vaccine's efficacy may be reduced. { } }

--> Submit Transaction: AddVaccineInfo, add the given Vaccine info to the chain
Vaccine Info: {20 Immunex Immunex is approved for individuals aged 18 and above Immunex is contraindicated in individuals with a history of severe allergic reactions to any component of the vaccine or a previous dose of Immunex. Immunex is administered via intramuscular injection. Immunex is a novel vaccine developed to bolster the body's immune response against a specific pathogen. Immunex is formulated as a sterile suspension for injection. Immunex should be used with caution in individuals with compromised immune systems, as the vaccine's efficacy may be reduced. { } }
*** Transaction committed successfully
```

Figure 6.3: Data Stored on Blockchain

Bitbucket History

Bitbucket

Your work

Pull requests

Repositories

Projects

People

More

Create

mainproject

Source

Commits

Branches

Pull requests

Pipelines

Deployments

Jira issues

Security

Downloads

Repository settings

Gopika Krishnan. S / mainproject / mainproject

Commits

Search commits

All branches

Author	Commit	Message	Date
Gopika Krishnan. S	7555b3c	updated app.go	33 seconds ago
Gopika Krishnan. S	8045e71	updated smartcontract	2 minutes ago
Gopika Krishnan. S	bec1a00	database	19 hours ago
Gopika Krishnan. S	7bc9131	manage_authorization	4 days ago
Gopika Krishnan. S	ecfac08	lint-go.	5 days ago
Gopika Krishnan. S	bc05432	run-test-network-basic.sh	6 days ago
Gopika Krishnan. S	c1c0fc3	middleware.go	7 days ago
Gopika Krishnan. S	4286938	assetTransfer.go	2024-04-05
Gopika Krishnan. S	4b60ba0	main.go	2024-04-04
Gopika Krishnan. S	d411af5	network.sh	2024-04-02
Gopika Krishnan. S	deaec1d	CHAINCODE	2024-04-01
Gopika Krishnan. S	fb77c4	activate	2024-03-21

Figure 6.4: Bitbucket History