ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/25may537

Supply Chain Management System Using Hyperledger Fabric

S. Pragatheeshwar[1]; A. Charishma Reddy[2]; S. Mugilan[3]; G. Sumathi[4]

[1,2,3,4]Sri Venkateswara College of Engineering, - 602 117, Tamil Nadu, India

Publication Date: 2025/06/10

Abstract: This project introduces a blockchain-enabled supply chain management (SCM) system focused on enhancing traceability, data synchronization, and operational transparency in commodity exposure and tender handling. Leveraging Hyperledger Fabric (v2.5.0) as the core blockchain framework and Golang (v1.22.1) for backend development, the system replaces traditional, error-prone methods with a decentralized and tamper-proof architecture. The solution is deployed on AWS using services such as EC2, Lambda, API Gateway, Cognito, Step Functions, and SQS to ensure scalability, secure authentication, and asynchronous request processing. Frontend technologies include HTML5, CSS3, and JavaScript (ES2024), enabling dynamic and interactive dashboards for real-time data visualization. By integrating blockchain with cloud infrastructure, the system facilitates seamless communication among stakeholders, minimizes fraud, and improves decision-making. This architecture addresses the complexities of modern supply chains, ultimately offering a robust, transparent, and resilient platform for organizations aiming to optimize performance and hedge effectively against market uncertainties.

Keywords: *Blockchain, Supply Chain Management (SCM), Hyperledger.*

How to Site: S. Pragatheeshwar; A. Charishma Reddy; S. Mugilan; G. Sumathi (2025) Supply Chain Management System Using Hyperledger Fabric. *International Journal of Innovative Science and Research Technology,* 10(5), 4178-4186. https://doi.org/10.38124/ijisrt/25may537

I. INTRODUCTION

Supply chain management. While traditional systems such as spreadsheets and centralized databases have played a vital role in tracking commodities and tender processes, they face significant challenges in traceability, real-time synchronization, and data integrity. These limitations often lead to inefficiencies, delays, and errors across the supply chain. A promising solution to these issues is blockchain technology, which provides a decentralized, tamper-proof, and transparent platform for recording and managing supply chain data. Historically, managing commodity exposure involved manual processes and fragmented systems, which made collaboration between stakeholders difficult and prone to miscommunication. With the adoption of blockchain and cloud-based solutions, the industry is moving towards systems that ensure data consistency, enhance security, and support automated workflows. This project proposes a blockchain-based architecture using Hyperledger Fabric and AWS cloud services to streamline tender management, improve operational transparency, and enable more efficient decision-making in commodity supply chains. Blockchain is a distributed ledger technology that enables secure, transparent, and tamper-proof record-keeping. In the context of supply chain management, it offers improved traceability, data synchronization, and stakeholder collaboration. Hyperledger

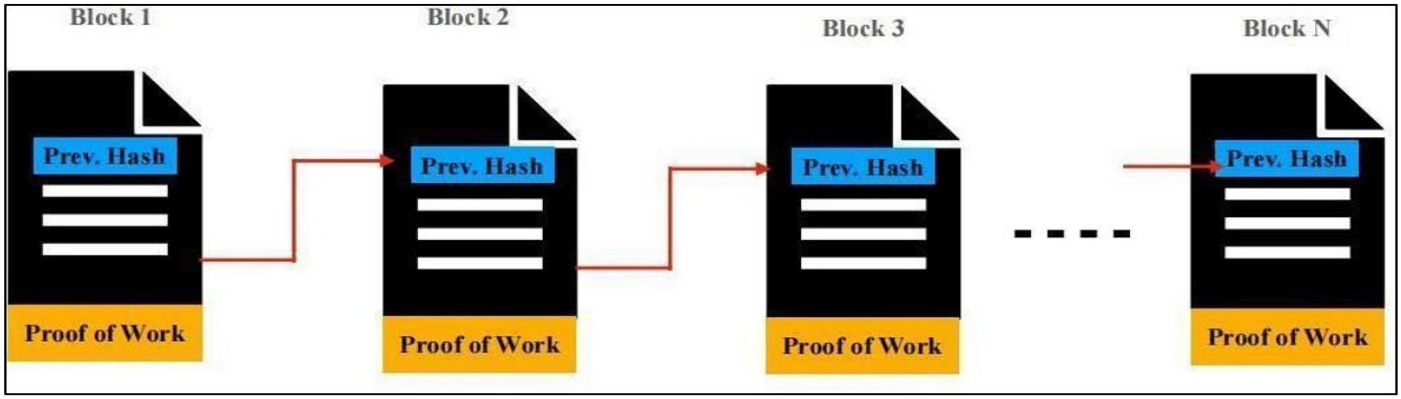

Fig 1 Visualization of Blockchain Block Structure

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/25may537

Fabric, an enterprise-grade blockchain framework, provides modularity, scalability, and support for 1 permissioned networks, making it suitable for complex enterprise applications Below are the essential components and dependencies required for implementing Hyperledger Fabric. Hyperledger Fabric is a permissioned blockchain framework developed under the Linux Foundation. It is designed for use in enterprise contexts and supports pluggable components such as consensus algorithms and membership services. It provides scalability, confidentiality, and fine-grained access control through channels and private data collections. The visualization of blockchain structure is depicted in Fig. 1.

A permissioned blockchain platform with a modular architecture that promotes security, scalability, and confidentiality, Hyperledger Fabric is intended for enterprise application. The logic for data transactions on the blockchain is defined by chain code, which are smart contracts written in Golang, Node.js, or Java. These transactions run on peer nodes in secure Docker containers. In order to promote communication among organizations, these peer nodes can be designated as anchor or committing peers, maintain the ledger, and approve transactions. The ordering service is made up of orderer nodes that use consensus techniques like Raft to sequence transactions into blocks.

CouchDB is an optional state database that supports rich queries over JSON data stored in the world state. It allows for complex searches and range queries. Certificate Authority (CA) issues digital certificates for identities (users, admins, peers). Fabric CA or external CAs can be used to issue and manage cryptographic identities. Docker is used to containerize the various components of the Fabric network, including peer nodes, orders, CAs, and chain code. It ensures consistency across environments. Golang is one of the primary languages used to write chain code. It is chosen for its performance, security, and concurrency capabilities. Cryptographic Libraries Hyperledger Fabric relies heavily on cryptographic functions for signing, encryption, and identity validation. These are managed via libraries that implement standard cryptographic protocols like ECDSA. Network Configuration Files These include YAML or JSON files used to define the network topology, chain code policies, channel configuration, and identities.

- *Related Work*

Blockchain technology has emerged as a transformative force in supply chain management (SCM), addressing challenges such as lack of transparency, traceability, and trust among stakeholders. Hyperledger Fabric, developed under the Linux Foundation's Hyperledger project, has gained significant attention due to its modular, permissioned architecture, which is especially suited for enterprise applications. Its fine-grained access control, pluggable consensus mechanisms, and support for private channels distinguish it from public blockchains like Ethereum, making it ideal for sectors requiring privacy and compliance, such as supply chain, healthcare, and finance [1][3][5]

Several studies demonstrate the effectiveness of Hyperledger Fabric in enhancing transparency and traceability within supply chains. Vujičić et al. provide a comprehensive overview of blockchain technology, including Hyperledger Fabric, highlighting its potential to improve trust and accountability in multi-stakeholder environments [3]. Wang et al. discuss smart contract architectures and their applications in automating business processes and enforcing Service Level Agreements (SLAs) in logistics and supply chain operations [4].

TrackChain, introduced by Kuo et al., is a supply chain solution built on Hyperledger Fabric designed to verify drug provenance and combat counterfeiting in the pharmaceutical industry, where traceability is critical for regulatory compliance and public safety. Their work demonstrates how blockchain can record immutable transaction histories, ensuring transparency and verifiability by authorized participants [5].

Marchese and Tomarchio further explored SLA compliance assessment using blockchain, showing that Hyperledger Fabric can automate and monitor contractual obligations in real time, reducing disputes among supply chain partners [6]. Additionally, Sahoo and Baruah provide a comprehensive review of blockchain applications in modern supply chains, noting the scalability benefits of frameworks like HBasechainDB and the importance of integrating blockchain with big data technologies for enhanced analytics and decision-making [7].

Beyond food and pharmaceuticals, Hyperledger Fabric has been applied across various domains to improve supply chain integrity and security. Wickremasinghe et al. proposed integrating knowledge graphs with Fabric to enhance data interoperability among supply chain participants, enabling more effective collaboration [8]. Islam et al. introduced a privacy-preserving permissioned blockchain framework for Industrial IoT (IIoT) environments, utilizing differential privacy techniques to protect sensitive data while maintaining transparency.

Despite these advancements, practical deployment of Hyperledger Fabric-based supply chain systems faces challenges. Scalability remains a concern, especially as transaction volumes and participants grow. Interoperability with legacy systems and IoT integration require robust middleware and APIs. Real-time data synchronization is critical for timely decision-making but remains difficult to achieve at scale [5] [6].

Managing identities across multiple organizations and dynamic partners is complex, even with private channels and Membership Service Providers (MSPs). Smart contract (chaincode) maintenance to reflect evolving business rules demands continuous updates and version control. Researchers are exploring solutions such as off-chain computation, zero-knowledge proofs for enhanced privacy, and cross-chain communication protocols to improve interoperability and data confidentiality [2] [6]

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/25may537

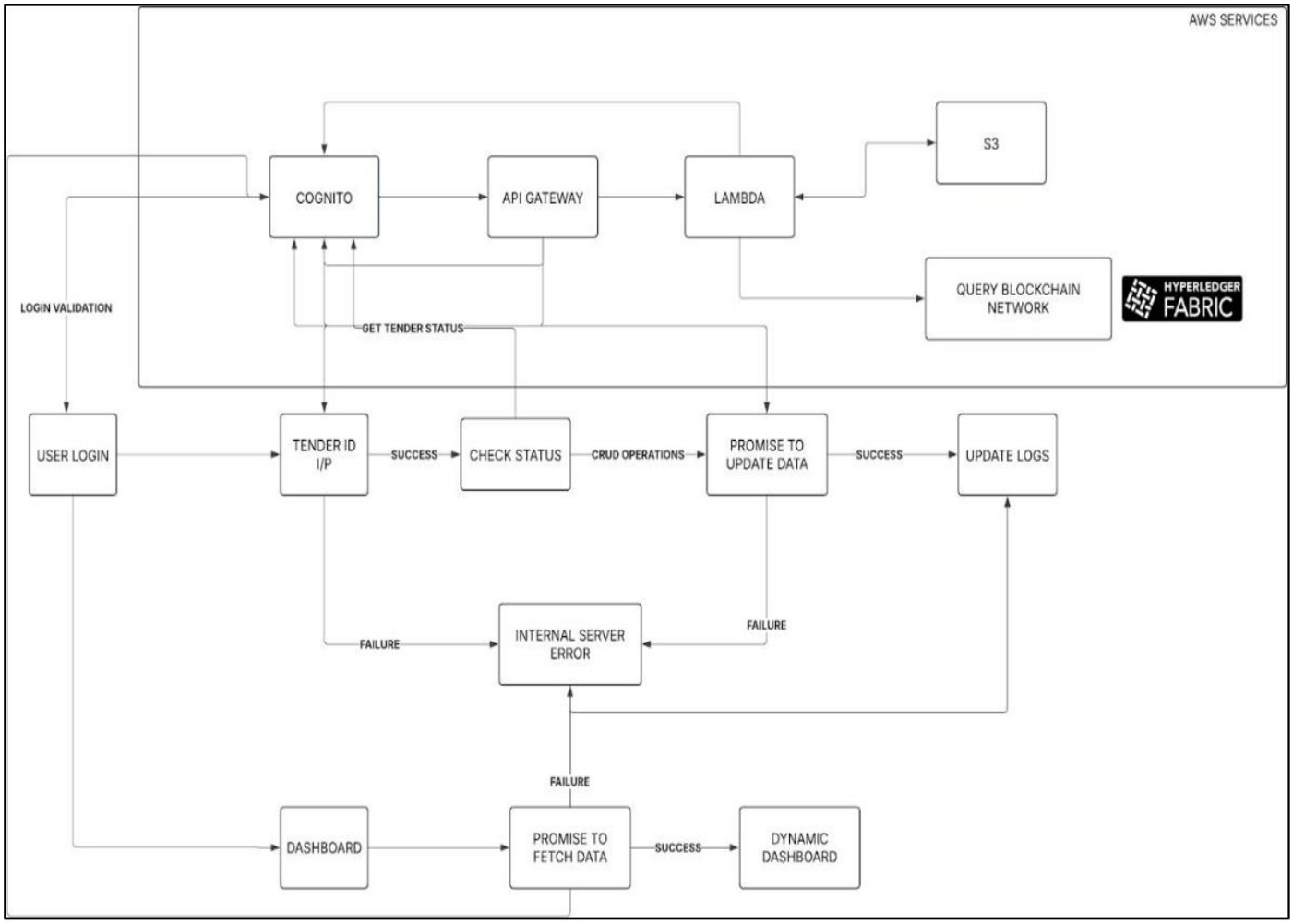

Fig 2 Blockchain-enabled Tender Management System

II. SYSTEM ARCHITECTURE FOR THE PROPOSED FRAMEWORK

The proposed system leverages Hyperledger Fabric integrated with AWS services to create a secure, decentralized supply chain solution for tender management. Cognito handles authentication, while API Gateway and Lambda facilitate communication with the blockchain network. Data is stored securely in S3, and transactions are validated through smart contracts. The system ensures real-time status updates, error handling, and dynamic dashboard visualization, enhancing transparency, security, and efficiency in tender processing. CRUD operations allow seamless data updates, while robust logging mechanisms maintain an immutable audit trail. The architecture ensures fault tolerance and scalability, reducing operational risks.

A. System Architecture

Fig. 2. depicts the architecture of a blockchain-based tender management system that integrates AWS services with Hyperledger Fabric to ensure security, transparency, and efficiency. User authentication is handled via AWS Cognito, allowing authorized access to the system. Once logged in, users input a Tender ID, triggering a status check. This request passes through API Gateway and Lambda, which queries the Hyperledger Fabric blockchain network for tender status while storing relevant data in AWS S3. If the process is successful, the system updates records via CRUD operations, logs changes, and updates the Dynamic Dashboard for real-time visualization. In case of failures, an Internal Server Error mechanism attempts to fetch valid data, ensuring system resilience. The architecture leverages blockchain for secure, tamper-proof transactions and AWS services for scalability, enabling automated logging,efficient data handling, and robust error management. This design enhances transparency, trust, and operational efficiency in the tendering process.

B. Technology Stack

Hyperledger Fabric serves as the underlying blockchain framework for the tender management system. As a permissioned blockchain, it ensures that only verified participants—such as government bodies, vendors, and auditors—can interact with the network. Hyperledger Fabric is designed for enterprise-grade use cases, offering modularity, scalability, and fine-grained access control. The blockchain ledger maintains tamper-proof records of all tender-related activities, including tender creation, bid submissions, approvals, and audits. This decentralized architecture eliminates the need for a central authority while ensuring transparency, trust, and data immutability across all stakeholders.

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/25may537

Chaincode in Hyperledger Fabric is equivalent to smart contracts in Ethereum. These are programs written in Go, JavaScript, or Java and define the business logic for the tender lifecycle. When a tender is created or a bid is submitted, chain code is invoked to validate and execute the transaction. For example, it ensures that no duplicate bids are submitted and verifies compliance with tender conditions. The chain code runs on endorsing peers and is executed only by authenticated users within the network, enforcing strict business rules with cryptographic guarantees. User Interaction User interaction is facilitated through a web-based portal built for different types of users for example: government officials, vendors, and auditors. Each user type is provided with role-specific interfaces for submitting, viewing, and approving tender data. The interface allows users to manage documents, track bidding status, and interact with the blockchain in real-time. Through secure login (via AWS Cognito), users can initiate transactions that are sent to the backend and recorded on the blockchain. The system is designed to be intuitive and accessible, even for non-technical users, ensuring wide usability across government departments. Node.js serves as the backend runtime environment, while Express.js provides the web server framework. These technologies handle the server-side logic of the application, including user session management, blockchain transactions, and RESTful APIs. Express routes are used to trigger chaincode functions such as creating tenders, submitting bids, and retrieving records. Node.js is well-suited for this application due to its non-blocking I/O model, which allows high concurrency—ideal for environments with multiple vendors and tenders running simultaneously.

The Fabric SDK provides the necessary APIs to interact with the blockchain network from the Node.js backend. It is responsible for submitting transactions, querying the ledger, and managing digital identities. The SDK supports cryptographic signing, channel interactions, and communication with peer/orderer nodes. Through this SDK, the backend can seamlessly integrate blockchain operations into the application workflow, ensuring that all user actions are recorded immutably. React.js is used to build the frontend interface of the system. Its component-based architecture allows for modular development of features such as tender dashboards, bid forms, status tracking, and audit views. React's state management and efficient rendering (via the virtual DOM) ensure a responsive user experience, even as data updates in real-time. It integrates with REST APIs and AWS services, enabling secure, dynamic interaction with blockchain data. AWS Cognito handles user authentication and authorization. It provides OAuth 2.0, JWT-based tokenization, and Multi-Factor Authentication (MFA). Each authenticated user is mapped to a specific blockchain identity, ensuring that actions like bidding or tender creation are tied to verifiable entities. Cognito enhances security and helps enforce access control at both the API and blockchain levels. AWS Lambda powers the serverless execution of certain backend workflows. It handles event-driven logic such as tender deadline enforcement, document scanning, or automatic status updates. These 18 functions can be triggered by user actions or time-based rules without provisioning any servers, thus reducing costs and complexity. Amazon API Gateway API Gateway serves as the interface layer between the frontend and backend. It handles thousands of concurrent API requests, applies rate limiting, validates JWTs from Cognito, and forwards requests securely to Lambda or the Node.js backend. It abstracts the infrastructure and ensures secure access to blockchain functions via REST endpoints. Amazon S3 is used to store tender-related documents and metadata, such as bidding files, approvals, and audit reports. The documents are encrypted and version-controlled, ensuring data integrity and regulatory compliance. While sensitive data is recorded on the blockchain as a hash, the actual files are stored securely in S3 and linked via metadata. CouchDB acts as the state database for Hyperledger Fabric. It stores the current world state of all tender objects in JSON format, making them easily queryable. It works in tandem with the immutable blockchain ledger to allow fast read access for frontend views and audit tools. The system is deployed in containers using Docker, which packages the blockchain network, backend services, and APIs. Kubernetes orchestrates these containers, ensuring auto-scaling, fault tolerance, and high availability. It simplifies deployment across environments and ensures the system remains robust during peak bidding times. 19 Terraform is used for Infrastructure-as-Code (IaC), automating the provisioning of all AWS resources including Cognito, Lambda, S3, and API Gateway. It ensures repeatable, version-controlled infrastructure setups for development, testing, and production environments.

III. PERFORMANCE EVALUATION

A comprehensive analysis of the performance characteristics of our blockchain-based supply chain management system—specifically focusing on the tender creation operation in Hyperledger Fabric. The system is deployed and tested in two environments: a local setup (Windows 11, 8GB RAM, 512GB SSD) and a production-grade AWS infrastructure (utilizing Lambda, EC2, Step Functions, API Gateway, and Cognito). We examine the performance metrics with detailed theoretical justifications and visual comparisons.

A. Latency Comparison: Local vs AWS

Tender creation is a frequent and foundational transaction in the blockchain-based supply chain. It serves as a benchmark to measure the overall responsiveness of the system. The average time to create a tender in the local environment was recorded at 0.576ms, whereas the AWS environment achieved a faster completion time of 0.456ms. This comparison is visually presented in Fig. 3: Tender Creation Time Comparison, which clearly shows the superior performance of the cloud-hosted deployment. Additionally, when we simulated the transaction 100 times, we observed a consistent latency of 0.880ms on AWS compared to 0.889ms on local, as shown in Fig. 4: Latency Over 100 Tender Creations. Though the differences are minute in a single transaction, they scale significantly across thousands of operations in a production environment.

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/25may537

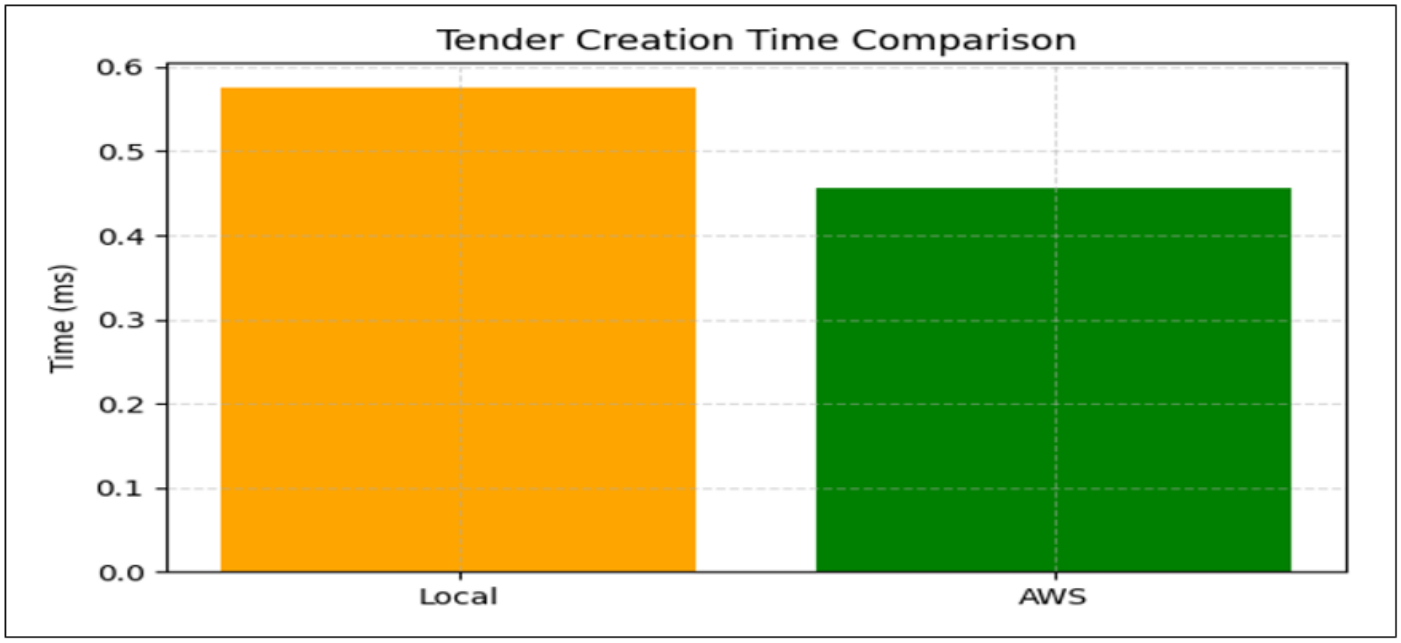

Fig 3 Tender Creation Time Comparison

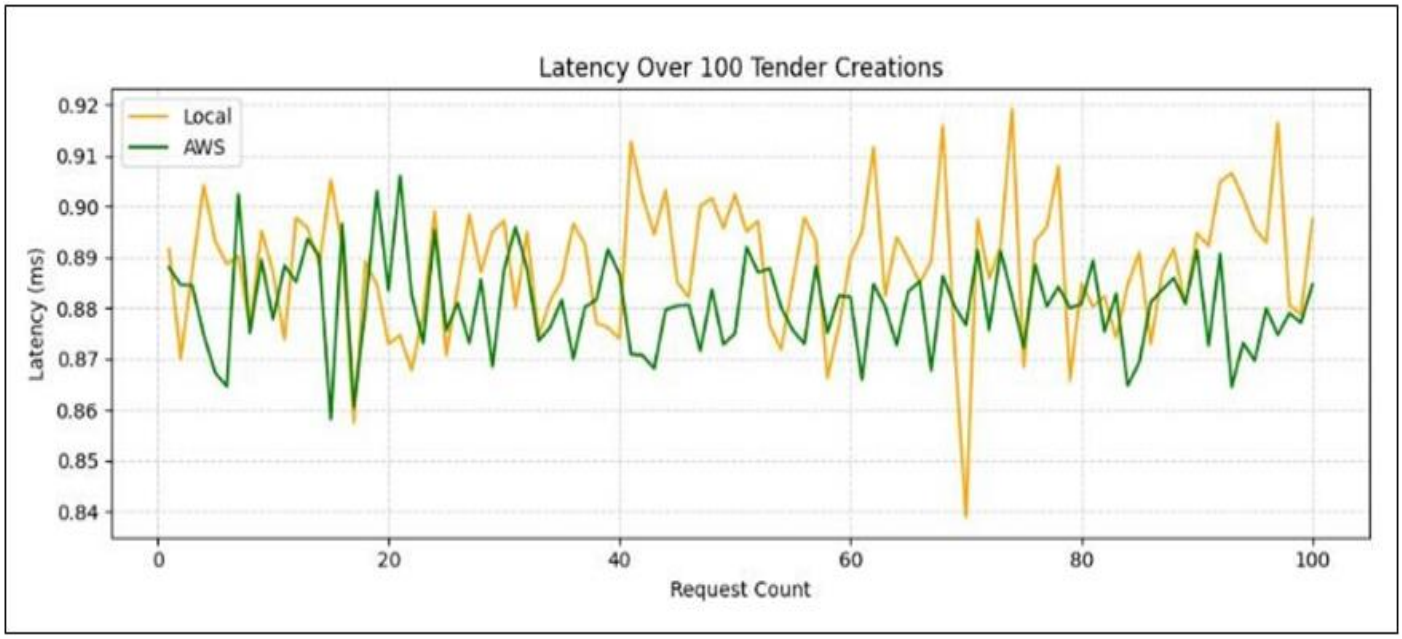

Fig 4 Latency Over 100 Tender Creations

B. Architectural Foundations Behind Latency Gaps

Resource Allocation and Execution Context The local machine, constrained by limited RAM and CPU, shows increased context-switching and I/O delays. In contrast, AWS offers high-performance computing (EC2), serverless execution (Lambda), and faster storage access, allowing optimized parallelism and faster data access. This directly reduces delays in API processing and ledger commit times.

C. Ledger Lookup Efficiency and Read Complexity

Hyperledger Fabric leverages a state database (LevelDB or CouchDB) for storing current key-value states. During tender creation, the system must verify that the tender ID doesn't already exist. This introduces a read operation whose efficiency depends on whether indexing is used.

In Unindexed Systems, lookup time Increases Linearly with Records:

$$T\_ \text{lookup} = \sum (i=1 \text{ to } n)\ f(i) \quad (1)$$

In indexed systems, lookup follows logarithmic growth:

$$T\_ \text{lookup} = O(\log n) \quad (2)$$

These models are illustrated in Fig. 5.: Ledger Lookup Time Complexity, which compares indexed vs unindexed lookup efficiency. Local environments often suffer from unindexed access due to configuration limitations or memory pressure, while AWS offers better support for indexing and memory-optimized reads, reducing overall read latency.

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/25may537

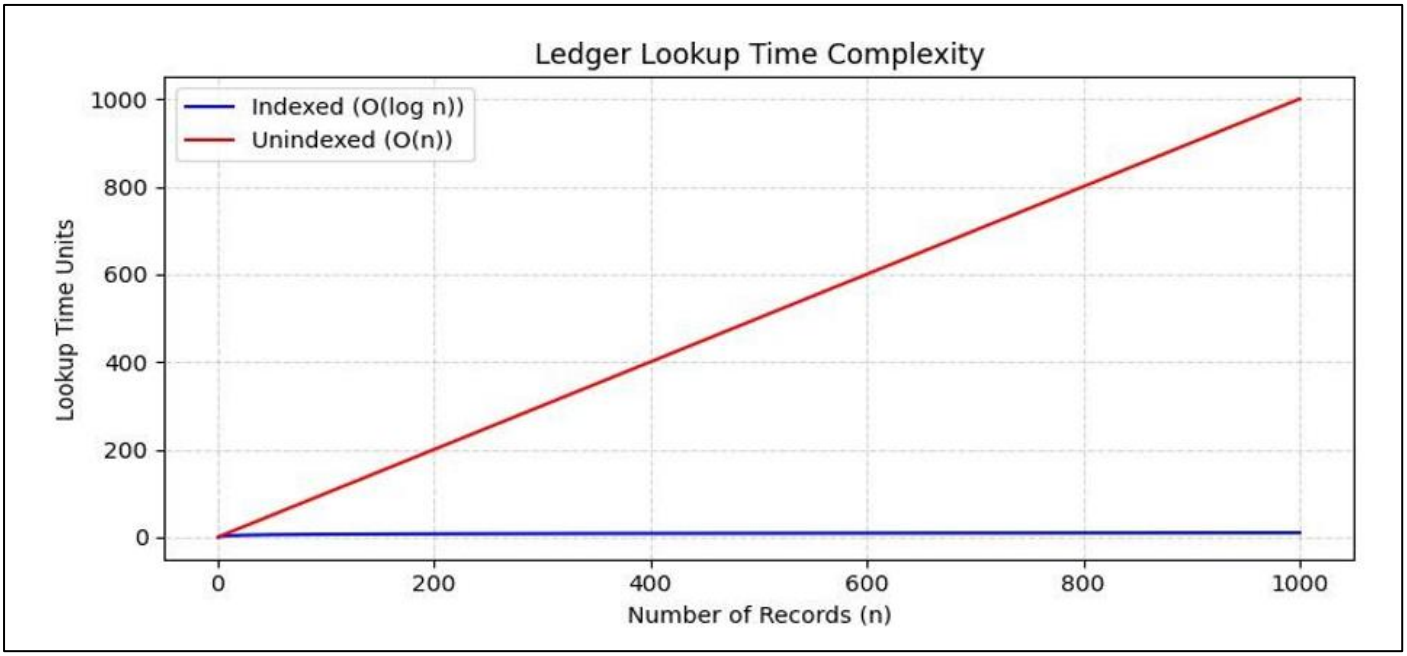

Fig 5 Ledger Lookup Time Complexity

D. Consensus and Replication Overhead

Hyperledger Fabric's consensus protocol (Raft) ensures fault tolerance and ordering across distributed peers. The time spent in consensus includes: Leader election Proposal broadcasting Log replicationCommit acknowledgment We model consensus cost using a continuous time.

$$T\_ \text{Consensus} = \int_0^c (l(t) + s(t))\, dt \quad (3)$$

Where:

.

l(t) = time-dependent communication delay

s(t) = block size increase or transaction load

As seen in Fig. 6: Consensus Overhead with Increasing Transactions, as the number of transactions increases, the consensus latency grows due to increased communication and data replication time. AWS mitigates this with better VPC networking and multi-core resource isolation, while the local machine, often constrained by Docker networks and single-threaded execution, faces amplified latency as load increases.

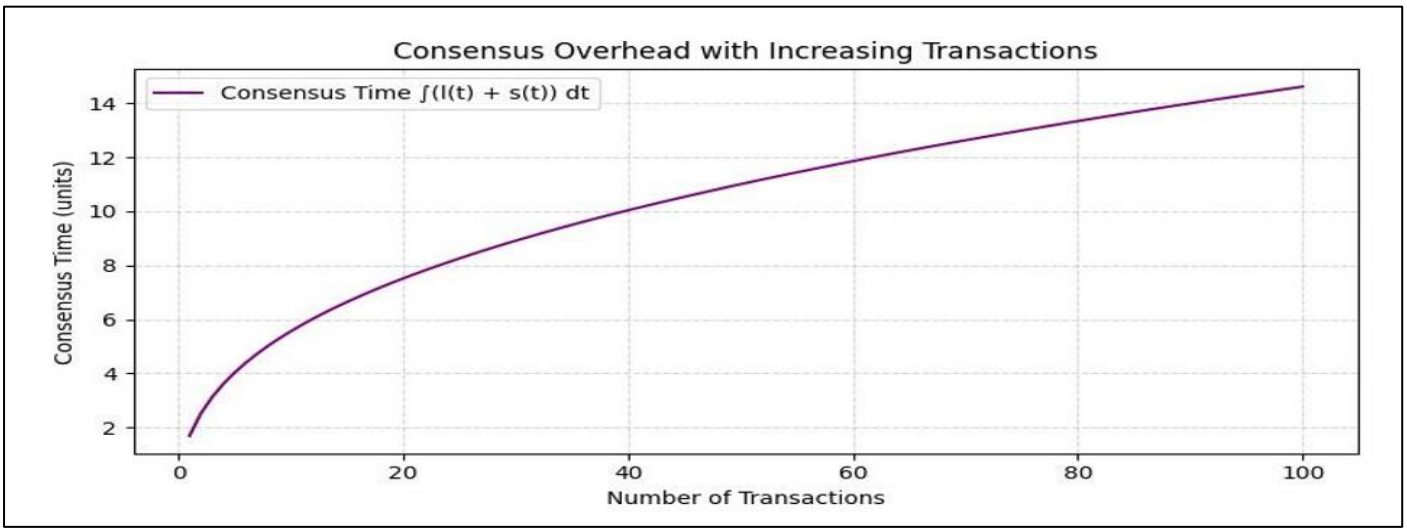

Fig 6 Consensus Overhead with Increasing Transactions

E. Gossip Protocol and State Synchronization

After the block is created, it must be distributed across all peers via Fabric's gossip protocol. The time taken for this peer-to-peer synchronization is represented as:

$$T\_ \text{gossip} = \int_0^R [P / k^t]\, dt \quad (4)$$

Where:

P = number of peers

k^t = spread factor per gossip round

R = number of rounds until full convergence

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/25may537

In local environments, the network bridging between Docker containers and OS-level constraints slows the convergence rate. In AWS, gossip propagation is expedited due to high-speed internal networking and dedicated peer containers or VMs.

F. Cryptographic Operations and Disk I/O

$$T\_ crypto = T\_ sign + T\_ verify + T\_ hash \quad (5)$$

These cryptographic checks ensure authenticity, non-repudiation, and integrity. In local setups, generic CPU execution without hardware acceleration introduces minor cryptographic delay. On AWS, this is often mitigated via Nitro enclaves, KMS-backed keys, and CPU instructions designed for cryptographic operations. Final ledger commits depend on block size and commit strategy:

$$T\_ block = (B\_ size / T\ x\_ rate) + B\_ timeout \quad (6)$$

In the local system, the combination of smaller block size thresholds and lower throughput causes frequent block commits with fewer transactions, thereby increasing total commit operations. AWS, with its batching optimizations, reduces such inefficiencies.

G. Overall Latency Model

All the above layers—lookup, gossip, consensus, cryptographic validation, and disk commits—can be synthesized into a single performance expression:

$$L\_ total = \sum_{i=1 \text{ to } n} f(i) + \int_0^R [P/k^t]\, dt + \int_0^c (l(t) + s(t))\, dt + T\_ sign + T\_ verify + T\_ hash + T\_ disk \quad (7)$$

This comprehensive formula encapsulates every technical factor affecting latency. The difference between AWS and local environments—even when minimal in single operations—becomes significantly magnified as we scale the number of tenders, making AWS the optimal choice for real-world deployment.

IV. RESULTS

The result is the successful development and deployment of a blockchain-based supply chain management system using Hyperledger Fabric, aimed at enhancing transparency, traceability, and efficiency across the supply chain lifecycle. By leveraging the permissioned blockchain features of Hyperledger Fabric, the system ensures that all transactions—such as product creation, shipment, and delivery—are recorded immutably, creating a secure and auditable ledger for all stakeholders involved. To enhance the system's scalability and integration with modern cloud infrastructure, AWS services were utilized extensively. Amazon Cognito provides secure user authentication and role-based access control, ensuring that only authorized participants can interact with the network. AWS Lambda enables seamless execution of backend logic without managing servers, while Amazon S3 offers durable storage for off-chain data such as documents or product images. Additionally, AWS Step Functions were used to coordinate complex workflows and ensure smooth, automated transitions between various stages of the supply chain process.

This integrated approach provides supply chain participants—including manufacturers, distributors, and retailers—with real-time visibility into the movement and status of goods. Every transaction is recorded as a new block, preserving the integrity of historical data and preventing unauthorized tampering. The system fosters trust and collaboration among stakeholders, reduces operational bottlenecks, and supports swift, data-driven decision-making.

V. CONCLUSION

The development of the blockchain-based tender management system represents a significant step forward in digitizing and decentralizing procurement within the supply chain sector. Leveraging Hyperledger Fabric, the platform ensures transparency, security, and integrity, addressing persistent issues like bid manipulation and lack of traceability. Smart contracts automate key functions such as tender publishing, bid evaluation, and contract awarding, reducing errors and administrative overhead while promoting fairness. A key innovation is the integrated hedging module, which proactively mitigates market risks through real-time analysis and automated strategies, stabilizing procurement outcomes. Built on AWS and guided by modular design, the system is scalable, extensible, and suited to diverse enterprise needs. Enhanced by decentralized identity, secure access control, and event-driven architecture, it stands as a robust, compliant solution. Ultimately, this system not only demonstrates the transformative potential of blockchain in tendering but also lays the groundwork for resilient, risk-aware, and financially efficient supply chains. As global procurement becomes increasingly complex, solutions like this will be vital in enabling greater operational agility, cost control, and stakeholder confidence. With continued enhancements, the platform is well-positioned to serve as a benchmark for next-generation digital procurement ecosystems. However, ongoing research is required to overcome challenges related to scalability, interoperability, and dynamic partner management. Integration of advanced privacy techniques, AI-based analytics, and cross-chain protocols presents promising future directions, motivating continued exploration of Hyperledger Fabric across various supply chain domains [4] [7].

FUTURE WORKS

Future enhancements to the blockchain-based tender management system with integrated hedging will focus on improving scalability, automation, and real-world readiness. A major upgrade involves automating infrastructure deployment using Terraform, enabling rapid and consistent provisioning of the Hyperledger Fabric network across cloud platforms like AWS. Integration of decentralized oracle services will provide real-time commodity and currency data, enhancing the system's ability to adapt hedging strategies to market changes. To reduce blockchain overhead, an off-chain data approach will be adopted, storing only essential metadata

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/25may537

on-chain while using solutions like IPFS or AWS S3 for transactional and document storage. The system will also implement Decentralized Identity (DID) standards for secure, verifiable identity management without relying on centralized authentication. Future updates will include analytics dashboards for monitoring bids, supplier performance, and hedging outcomes, as well as integration with ERP and SCM systems for seamless data flow. Enhanced compliance features, such as cryptographically timestamped audit logs and automated reporting tools, will ensure transparency and support regulatory requirements

REFERENCES

[1]. Gordon, W.J., Catalini, C.: Blockchain Technology for Healthcare: Facilitating the Transition to Patient-Driven Interoperability. (n.d.)

[2]. Eberhardt, J., Tai, S.: On or Off the Blockchain? Insights on Offchaining Computation and Data. (n.d.)

[3]. Vujičić, D., Jagodić, D., Randić, S.: Blockchain Technology, Bitcoin, and Ethereum: A Brief Overview. (n.d.)

[4]. Wang, S., Yuan, Y., Wang, X., Li, J., Qin, R., Wang, F.-Y.: An Overview of Smart Contract: Architecture, Applications, and Future Trends. (n.d.)

[5]. Kuo, T.-T., Kim, H.-E., Ohno-Machado, L.: Blockchain Distributed Ledger Technologies for Biomedical and Healthcare Applications. (n.d.)

[6]. Gramoli, V.: From Blockchain Consensus Back to Byzantine Consensus. (n.d.)

[7]. Sahoo, M.S., Baruah, P.K.: HBasechainDB—A Scalable Blockchain Framework on Hadoop Ecosystem. (n.d.)

[8]. Zhang, P., White, J., Schmidt, D.C., Lenz, G., Rosenbloom, S.T.: FHIRChain: Applying Blockchain to Securely and Scalably Share Clinical Data. (n.d.)

[9]. Kim, M.G., Lee, A.R., Kwon, H.J., Kim, J.W., Kim, I.K.: Sharing Medical Questionnaires Based on Blockchain. (n.d.)

[10]. Ekblaw, A., Azaria, A., Halamka, J.D., Lippman, A., Vieira, T.: A Case Study for Blockchain in Healthcare: "MedRec" Prototype for Electronic Health Records and Medical Research Data White Paper. (n.d.)

[11]. Zhuang, Y., Sheets, L.R., Chen, Y.W., Shae, Z.Y., Tsai, J.J.P., Shyu, C.R.: A Patient-Centric Health Information Exchange Framework Using Blockchain Technology. (n.d.)

[12]. Dagher, G.G., Mohler, J., Milojkovic, M., Marella, P.B.: Ancile: Privacy-Preserving Framework for Access Control and Interoperability of Electronic Health Records Using Blockchain Technology. (n.d.)

[13]. Dar, A.A., Alam, M.Z., Ahmad, A., Reegu, F.A., Rahin, S.A.: Blockchain Framework for Secure COVID-19 Pandemic Data Handling and Protection. (n.d.)

[14]. Hsieh, G., Chen, R.: Design for a Secure Interoperable Cloud-Based Personal Health Record Service. (n.d.)

[15]. Azarm, M., Backman, C., Kuziemsky, C., Peyton, L.: Breaking the Healthcare Interoperability Barrier by Empowering and Engaging Actors in the Healthcare System. Procedia Computer Science 113, 326–333 (2017)

[16]. Singh, A.P., et al.: A Novel Patient-Centric Architectural Framework for Blockchain-Enabled Healthcare Applications. IEEE Trans. Ind. Inform. 17(8), 5779–5789 (2021)

[17]. Castaldo, L., Cinque, V.: Blockchain-Based Logging for the Cross-Border Exchange of E-Health Data in Europe. In: International ISCIS Security Workshop, pp. 46–56. Springer, (2018)

[18]. Yue, X., Wang, H., Jin, D., Li, M., Jiang, W.: Healthcare Data Gateways: Found Healthcare Intelligence on Blockchain with Novel Privacy Risk Control. J. Med. Syst. 40(10), 218 (2016)

[19]. Fan, K., Wang, S., Ren, Y., Li, H., Yang, Y.: MedBlock: Efficient and Secure Medical Data Sharing via Blockchain. J. Med. Syst. 42(8), 136 (2018)

[20]. Wang, H., Song, Y.: Secure Cloud-Based EHR System Using Attribute-Based Cryptosystem and Blockchain. J. Med. Syst. 42(8), 152 (2018)

[21]. Zhang, X., Poslad, S., Ma, Z.: Block-Based Access Control for Blockchain-Based Electronic Medical Records (EMRs) Query in eHealth. In: IEEE Global Communications Conference, pp. 1–7. IEEE Press, (2018)

[22]. Jiang, S., Cao, J., Wu, H., Yang, Y., Ma, M., et al.: Blochie: A Blockchain-Based Platform for Healthcare Information Exchange. In: 2018 IEEE Int. Conf. on Smart Computing, Taormina, Italy, pp. 49–56. IEEE Press, (2018)

[23]. Hussein, A.F., ArunKumar, N., Ramirez-Gonzalez, G., Abdulhay, E., Tavares, J.M., et al.: A Medical Records Managing and Securing Blockchain-Based System Supported by a Genetic Algorithm and Discrete Wavelet Transform. Cogn. Syst. Res. 52, 1–11 (2018)

[24]. Zhu, L., Wu, Y., Gai, K., Choo, K.K.: Controllable and Trustworthy Blockchain-Based Cloud Data Management. Future Gener. Comput. Syst. 91, 527–535 (2019)

[25]. Huang, J., Qi, Y.W., Asghar, M.R., Meads, A., Tu, Y.C.: MedBloc: A Blockchain-Based Secure EHR System for Sharing and Accessing Medical Data. In: 2019 18th IEEE Int. Conf. on Trust, Security and Privacy in Computing and Communications, pp. 594–601. IEEE Press, (2019)

[26]. Zhang, X., Poslad, S.: Blockchain Support for Flexible Queries with Granular Access Control to Electronic Medical Records (EMR). In: 2018 IEEE Int. Conf. on Communications, Kansas City, MO, USA, pp. 1–6. IEEE Press, (2018)

[27]. Alshalali, T., M'Bale, K., Josyula, D.: Security and Privacy of Electronic Health Records Sharing Using Hyperledger Fabric. In: 2018 Int. Conf. on Computational Science and Computational Intelligence, Las Vegas, NV, USA, pp. 760–763. IEEE Press, (2018)

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/25may537

[28]. Jamoom, E., Yang, N., Hing, E.: Adoption of Certified Electronic Health Record Systems and Electronic Information Sharing in Physician Offices: United States, 2013 and 2014. NCHS Data Brief 236 (2016)