Prosiding Seminar Nasional Penelitian dan Pemberdayaan Masyarakat 2025
Jakarta, 12-13 November 2025

DOI http://dx.doi.org/10.36722/psn.v5i1.5053

[SNP – 43]

Design and Implementation of a Blockchain-based Shipment Tracking Proof-of-Concept for Academic Application

Andi Arniaty[1*], Riri Safitri[1], Winangsari Pradani[1], Nurbojatmiko[1] Abdullah[1]

[1]Informatics Departement, Faculty of Science & Technology, Universitas Al-Azhar Indonesia,
Jl. Sisingamangaraja, Kebayoran Baru, DKI Jakarta, 12110.

Penulis untuk Korespondensi/E-mail: andi.arniaty@uai.ac.id

Abstract – **The demand for secure, transparent shipment tracking is rising as digital logistics advances. Traditional centralized tracking systems face issues such as data delays and manipulation, which reduce trust and accuracy. This research aims to create a blockchain-based shipment-tracking system that serves as both a prototype and an educational tool for students. The system features a user interface built with Vue.js and utilizes a middleware layer with Node.js and Express.js. A smart contract in Solidity is deployed on a local Ethereum network using the Hardhat framework. This prototype allows students to engage with decentralized application development and explore interactions among front-end components, middleware, and smart contracts. Each transaction is recorded immutably, with an average gas consumption of about 175,000 units and execution times under 2 seconds, while preventing duplicate shipments. This research offers a functional shipment-tracking model that provides transparent data recording and serves as an effective framework for blockchain education, bridging theoretical concepts with practical applications in higher education.**

Keywords *– Academic Application, Blockchain, Shipment Tracking, Smart Contract, Solidity.*

INTRODUCTION

The development of digital technology has created a demand for a more transparent, efficient, and real-time auditable logistics system. Conventional tracking systems that rely on centralized databases still encounter several issues, such as delays in status updates, human errors in data entry, and the potential for information manipulation by internal parties [1]. Dependence on a single central server makes these systems vulnerable to cyberattacks and data loss due to system failures, resulting in a single point of failure. Blockchain technology presents a solution by offering a decentralized ledger approach, where all transactions are stored in interconnected blocks and verified by a distributed network. Each block is secured using a cryptographic algorithm that ensures data integrity and immutability [2].

In the logistics context, blockchain can document every change in the status of a shipment with a transparent digital trail. This allows all parties involved, senders, service providers, and recipients, to independently verify the shipment's status without needing to trust a single entity [3]. In addition to enhancing transparency, implementing blockchain offers advantages in security and audit efficiency. Using smart contracts, business processes such as the creation of shipping data, status updates, and receipt verification can be automated based on the logic programmed into the contract. This reduces the need for manual intervention and minimizes the risk of data manipulation common in traditional systems [4].

However, applying blockchain technology in logistics requires complex infrastructure and can

incur high costs for large-scale implementation. This scenario makes an academic prototype-based approach particularly relevant for introducing blockchain concepts to students through realistic, secure, and scalable system simulations and experimentation. In an academic setting, students not only learn the theory of cryptography and distributed ledger technology but also gain insights into how smart contracts interact with user interfaces and middleware in real-world applications [5].

Numerous studies highlight blockchain technology's potential in logistics and supply chain systems, enhancing transparency, security, and efficiency. A notable example is TradeLens, developed by IBM and Maersk [6], which digitizes international shipping documents and connects supply chain stakeholders. However, its closed-source nature limits academic use.

Research by Casino [7] presents a blockchain-based traceability model for the food supply chain to ensure product authenticity and safety. Similarly, the VeChain Foundation [8] combats counterfeiting in pharmaceuticals and luxury goods using a QR-based verification system. Most existing research focuses on large-scale commercial contexts, often involving complex infrastructure and high costs.

While some studies have examined blockchain in education, they generally involve simple simulations or conceptual frameworks without full technical implementations. For instance, Delgado. VE et al. [9] propose a smart contract simulation framework, but lack user interface integration. Ullah N et.al. [10] stress hands-on learning, but haven't created a complete prototype.

In Indonesia, studies like those by Solihin [11] and Safitri & Huda [12] explore blockchain in supply chains. Still, they focus less on fully implementing shipment tracking systems that merge frontend and backend components with smart contracts. This research addresses that gap by developing a comprehensive blockchain-based shipment tracking system.

These research identifies a significant gap: the absence of replicable, full-stack blockchain prototypes that can function as practical learning tools while demonstrating actual transaction behavior, system integration, and performance metrics. To address this gap, the research develops a blockchain-based shipment-tracking system that uses Vue.js for the frontend, Node.js and Express.js as middleware, and Solidity smart contracts deployed with the Hardhat framework.

Hardhat is chosen for its flexibility as a local Ethereum development environment, providing detailed insights into transaction behavior, including gas usage, block generation, and error tracing. It also facilitates rapid testing without the complications of deploying to a public network. Moreover, Hardhat provides built-in debugging tools and a customizable node, making smart contract testing simpler and more efficient, and thus highly suitable for academic prototyping and instructional purposes [13].

The objective of this research is to design and implement a comprehensive blockchain-based shipment tracking system that integrates the interface layer, middleware, and smart contracts. Additionally, the research aims to validate the system's functionality within the Hardhat development environment, evaluate its performance using metrics such as gas usage, block number tracing, and execution time, and assess its potential as a project-based learning tool for students studying blockchain application development.

RESEARCH METHOD

This study employs a prototype-based development methodology commonly used in academic and experimental software engineering research. The aim is not only to create a functional system but also to validate concepts and enhance hands-on learning. This methodology focuses on incremental development, iterative refinement, and evaluation through functional testing and system tracing within a blockchain environment. Such an approach is particularly suitable for developing blockchain-based applications, as it enables validating transaction behavior, verifying smart contract correctness, and assessing user interaction flows before deploying the application in production environments.

The methodology is structured into five key stages: Requirement Analysis, System Design, Smart Contract Development, Frontend and Backend Integration, and Testing and Evaluation, as illustrated in figure 1.

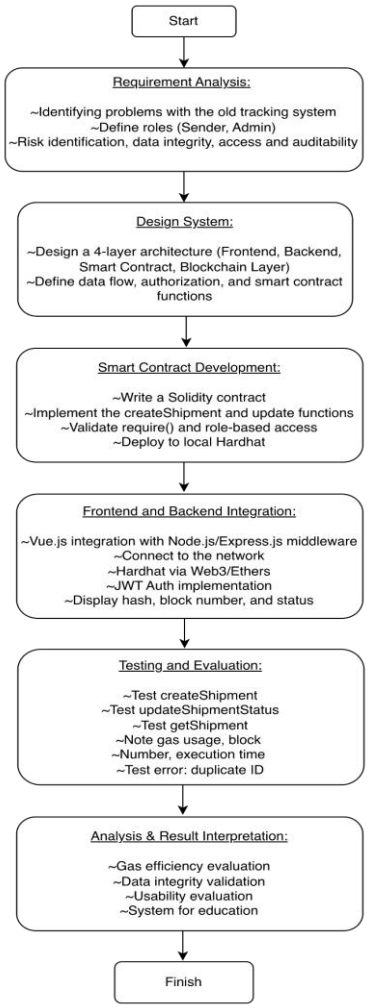

Figure 1. Research Stage

Requirement Analysis

A thorough analysis of existing shipment tracking workflows identified key issues with traditional centralized logistics systems, such as data integrity violations, uncontrolled access to information, limited stakeholder traceability, and a lack of verifiable audit trails. These systems often rely on trust-based updates with minimal cryptographic validation, leading to manipulation, delayed reporting, and disputes.

To address these concerns, the new system is built around two essential roles for verifiable shipment tracking: (1) Sender: Initiates the shipment, entering package details, including the receiver's identity and item metadata. (2) Administrator (Admin): Also responsible for entering shipment details.

Roles act as smart contract gatekeepers in Solidity with the logic `require (`msg.sender == roleAddress`), ensuring that critical actions are associated with authorized Ethereum addresses. By initially excluding the courier as a distinct role, we reduce the trust surface and architectural complexity, allowing focused testing on essential aspects like immutability and authorization. The courier role will be added in future iterations, potentially incorporating features such as GPS tracking and IoT sensors.

This streamlined model supports a controlled environment for prototyping a decentralized architecture while allowing for future role expansion and logistics integration.

Design System

The architecture is composed of four modular and dynamic layers: the Frontend Layer, Backend Layer, Smart Contract Layer, and Blockchain Infrastructure Layer. Each layer is essential for ensuring the integrity, security, and auditability of the system.

The Frontend Layer effectively captures user inputs and facilitates real-time interactions with shipment

data. Developed using Vue.js, it incorporates MetaMask integration for secure and decentralized blockchain transaction signing, ensuring identity verification.

The Backend Layer, built with Node.js and Express.js, manages user authentication through JSON Web Tokens (JWTs), oversees business logic, and communicates with the blockchain via Web3.js. It also handles transaction logging and provides an abstraction layer for seamless interaction with the smart contract.

The Smart Contract Layer, implemented in Solidity, encodes the fundamental business rules governing shipment tracking, including creation, state updates, and auditability. It enforces stringent access control based on the user's role, either sender or administrator, guaranteeing that all interactions are validated through cryptographic methods.

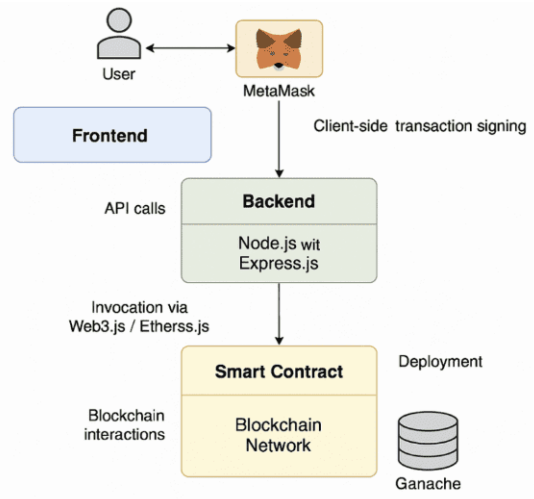

Figure 2. Architecture System

Smart Contract Development

The smart contract development phase involved deploying a smart contract to the local Ethereum network using the Hardhat framework. In this study, the contract was successfully deployed at the address `0x5FbDB2315678afecb367f032d93F642f64180aa3`.

All transactions were executed using the primary sender account, `0xf39Fd6e51aad88F6F4ce6aB8827279cffFb92266`, which served as the system administrator. Whenever a smart contract function such as `createShipment` or `updateShipmentStatus` was executed, the network generated transaction data, including the transaction hash, block number, and gas used. This data serves as digital proof that the transaction was successfully recorded on the blockchain.

To ensure the reliability of business logic validation, error handling tests were conducted by entering the same shipment ID multiple times. The test results showed that the system rejected duplicate transactions and displayed the error message "*Shipment already exists*." These findings demonstrate that the internal verification mechanism implemented via the `` `require()` `` function in the smart contract works as intended to prevent duplicate data entry. Thus, the system successfully upholds the integrity of shipment data and validates the security logic implemented in the smart contract.

Frontend and Backend Integration

In addition to deploying and validating the fundamental functions of the smart contract, the implementation phase involved establishing a complete integration between the blockchain layer (on-chain) and the web application layer (off-chain). This integration was achieved through a middleware service developed using Node.js and Express.js, which acts as a secure communication bridge between the Vue.js-based frontend and the smart contract running on the local Hardhat Ethereum network. The middleware is responsible for request routing, data validation, and smart contract interactions, utilizing Ethers.js or Web3.js. This ensures that all blockchain transactions adhere to a controlled and verifiable workflow.

To maintain strong access control, the middleware incorporates a JSON Web Token (JWT) authentication mechanism. Only authenticated administrators are permitted to invoke sensitive contract functions, such as `createShipment` and `updateShipmentStatus`. This layered authorization strategy ensures that, although the underlying ledger is decentralized, permissions and role management are consistently enforced at the application level. This prevents unauthorized or malicious contract calls while maintaining the system's transparency.

The user interface is intentionally designed to be accessible to non-technical users, featuring a structured form for entering shipment data, including recipient name, address, status, and delivery time. When the user clicks the "Create Shipment" button, the frontend sends an authenticated API request to the backend middleware. After validating the JWT and verifying the user's permissions, the backend forwards the

request to the appropriate smart contract function on the blockchain. Once the transaction is mined and confirmed, the blockchain returns cryptographic metadata such as the transaction hash, block number, and gas usage which is then displayed on the interface as immutable digital proof of the shipment record.

Overall, the implementation process demonstrates the successful integration of all system components: frontend, middleware, and blockchain layer into a seamless end-to-end pipeline. This cohesive architecture allows users to initiate, update, and track shipment data with real-time transparency and immutable audit trails. From an academic perspective, this integration provides students with a concrete and practical example of how blockchain-based systems are engineered, helping them understand smart contract interactions, transaction flows, and security enforcement within a modern full-stack application environment.

Testing and Evaluation

Testing was conducted to ensure the shipment tracking system could accurately execute all smart contract functions. Each function was tested sequentially, including contract deployment, shipment data creation (*createShipment*), shipment status updates (*updateShipmentStatus*), and data retrieval (*getShipment)*. Important parameters such as transaction hashes, block numbers, and gas consumption were recorded, with results summarized in Table 1.

Table 1. Transaction Test Results on Local Blockchain

Function	Transaction Hash	Block #	Gas Used	Status
Deploy Contract	0x31378c84…	1	960,843	Success
Create Shipment	0xd9302669…	2	175,066	Success
Update Shipment Status	0x55ee8d0b…	4	175,114	Success
Get Shipment	–	5	35,858	Read OK
Duplicate Check	–	3	31,671	“Shipment already exists”

The tests confirmed that each key function in the smart contract was successfully executed and recorded on the blockchain. The createShipment transaction produced a unique transaction hash (`0xd9302669…`) and resulted in a new block being added with block number 2 and gas consumption of 175,066 units. The updateShipmentStatus function demonstrated similar efficiency with gas usage of 175,114 units.

In contrast, the *getShipment* function consumed significantly less gas at 35,858 units since it is *read-only*. The data duplication test returned the error "*Shipment already exists*" in block number 3, confirming the effectiveness of the contract's validation mechanism.

Overall, the system meets essential criteria for a good blockchain implementation: transaction integrity and resource efficiency. The transaction log provides valuable insights for students, illustrating the impact of smart contract instructions on gas consumption and block addition, serving as both evidence of technical success and a learning tool in blockchain education.

RESULTS & DISCUSSION

Transaction Performance Analysis

System testing was conducted on a local Hardhat network (localhost:8545) using the default Ethereum Virtual Machine (EVM) configuration. The results showed that all smart contract functions executed successfully, with five transactions recorded in blocks 1 through 5, covering the deployment and operation of the contract.

The first transaction (Block #1) involved contract deployment, consuming 960,843 gas units. The second transaction (Block #2) executed the *createShipment()* function, using 175,066 gas units and successfully recording new shipment data with a transaction hash of `0xd9302669`. Block #3 confirmed the duplicate validation mechanism, generating an error message "S*hipment already exists*," validating the correct operation of the *require()* function.

The fourth transaction (Block #4) updated the shipment status with 175,114 gas units, while the fifth transaction (Block #5) executed the *getShipment()* function, consuming only 35,858 gas units due to its read-only nature. The average transaction time was under 2 seconds.

Table 2. Blockchain System Transaction Performance on Hardhat Local Network

No	Contract Function	Operation Type	Gas Used	Average Execution Time (seconds)	Status	Description
1	Deploy Contract	Write	960,843	2.10	Success	Contract Initialization Successful
2	createShipment	Write	175,066	1.84	Success	Shipment data input
3	updateShipmentStatus	Write	175,114	1.87	Success	Status update successful
4	getShipment	Read	35,858	0.64	Read OK	Data read from blockchain
5	Duplicate Check	Write	31,671	1.92	Failed	"Shipment already exists"

These results indicate that the Solidity contract is efficient and free from unnecessary complexity. The stability of gas usage for write functions and efficiency for read functions confirm the implementation standards for academic prototypes. Additionally, the error-handling test in Block #3 demonstrated robust business logic integrity.

A simulation with 10 parallel *createShipment* transactions suggested linear scalability, estimating total gas consumption at around 1.75 million units without significant cost spikes. While this has not been empirically tested on a public testnet, it lays the groundwork for further research on contract performance in environments like the Goerli Testnet or Polygon Mumbai, and for exploring gas optimization.

As shown in Table 2, the average gas consumption for write transactions is approximately 175,000 units, with execution times under 2 seconds. The consistent gas usage indicates that the Solidity code's structure is optimal, with no redundant instructions leading to unnecessary execution. In contrast, read operations consume significantly less gas since they do not alter the blockchain state.

Furthermore, the intentionally failed duplicate check test confirms that the contract validation mechanism (the `require()` function) operates as intended, effectively preventing duplicate entries. This demonstrates the security of the business logic in place.

In summary, this analysis highlights the effective implementation of a blockchain-based delivery tracking system, showing its potential as a real-world academic model for teaching blockchain efficiency, security, and scalability.

Comparison with Non-Blockchain Systems

To evaluate the effectiveness of the blockchain approach, we conducted a conceptual comparison with a centralized database-based tracking system. Conventional systems tend to allow for faster read and write transactions, averaging less than one second, because they do not require network verification [14]. However, they have significant drawbacks in terms of data integrity and audit transparency. In traditional systems, administrators with high access rights can modify or delete shipment data, creating opportunities to manipulate product statuses [2].

In contrast, the proposed blockchain-based system guarantees that every successfully recorded transaction is immutable, meaning it cannot be changed or deleted once confirmed on the ledger. The duplication error test, which returned the message "Shipment already exists," demonstrated that the smart contract effectively executed the integrity validation mechanism automatically, without needing third-party intervention. Although the blockchain system may be slightly slower than a traditional database, it provides significantly greater data reliability, auditability, and security features

that are critical for an effective shipment tracking system [15].

Security Analysis and Limitations

This system is designed with a three-layer security principle including, On-chain security: Business logic validation is performed directly by the smart contract using the *`require()`* function. This helps prevent duplication and unauthorized transactions. Off-chain security: JWT (JSON Web Token) authentication is employed to ensure that only verified users can access administrative functions, providing write access only to authorized personnel. Transport security: The HTTPS protocol is used to encrypt communications between the user interface and the middleware server, ensuring data integrity during transmission.

However, several security limitations have been identified. Personal data, such as recipient names and addresses, is stored in plain text within the contract, making it less suitable for public implementation. To address this, future developments could utilize an off-chain data storage approach, where sensitive information is stored in an encrypted database while the blockchain retains only hash references.

Additionally, this system currently operates on a local Hardhat network. As such, it has not yet tested fully decentralized security aspects, such as *Byzantine fault* tolerance or defenses against *Sybil attacks*, which are commonly seen on public networks.

Pedagogical Analysis

This system marks a significant advancement in Project-Based Learning (PBL) by featuring a three-layer architecture: frontend, backend, and blockchain allowing students to gain exposure to full-stack decentralized application (dApp) development [16],[17]. Students engage in the full blockchain development cycle, including designing and coding smart contracts in Solidity, implementing middleware for API communication using Node.js and Ethers.js, and integrating the user interface with Vue.js to display transaction results. This aligns with recent studies emphasizing the importance of hands-on blockchain development in higher education to bridge the gap between theory and real-world implementation [18].

In lab sessions, students learn to interpret technical data such as transaction hashes and gas consumption, understanding the link between contract complexity and execution costs on a blockchain network. Previous studies indicate that exposing learners to blockchain metrics and transaction behaviors improves their understanding of distributed consensus, computational cost models, and decentralized application logic [19].

This approach aligns with current trends in informatics education, emphasizing experiential learning in distributed technologies. Educators can design practical assignments, such as analyzing gas efficiency in contract functions, modifying contracts with additional validation, comparing performance between local networks and public testnets, and analyzing security implications [20].

Overall, this research showcases a technical implementation alongside an inspiring educational model for applied blockchain learning. It paves the way for integration into courses on Blockchain Technology, Software Engineering, Distributed Systems, and other informatics programs eager to embrace decentralized application development in their curricula.

CONCLUSION

This research successfully designed and implemented a blockchain-based shipment tracking system using smart contracts on a local Ethereum network (Hardhat Network). Test results indicate that the system's main functions: createShipment, updateShipmentStatus, and getShipment, operate efficiently and reliably, with an average gas consumption of 175,000 units and an execution time of under two seconds per transaction.

The sequence of blocks from Block #1 to Block #5 confirms that all transactions have been validated and permanently recorded in the blockchain, creating an immutable ledger, with no detected execution errors or logical anomalies.

In terms of functionality, this system illustrates how blockchain technology can enhance transparency, security, and data reliability in the shipment tracking process. The contract validation mechanism, which prevents duplicate entries, demonstrates that the business logic functions as intended and safeguards against data manipulation.

Furthermore, the test results show that the smart contract design has a stable and efficient code structure, making it a tangible example of applying gas optimization principles in an academic context.

The primary contribution of this research lies in the application of a blockchain system as a practical learning model (academic prototype) that connects cryptographic theory, distributed programming, and smart contract-based application development. This system can be utilized as an educational tool in informatics laboratories to introduce concepts such as decentralization, data integrity, and contract efficiency.

Future developments could focus on testing the system on public networks like the Goerli Testnet or Polygon, as well as integrating it with IoT technology to automatically record physical tracking data. Additionally, further research could investigate off-chain encryption models to protect sensitive data and analyze system performance under high transaction loads.

In summary, the developed system serves not only as a proof-of-concept for blockchain implementation in logistics but also as an adaptive learning platform for students to understand modern distributed technologies and the challenges of their real-world applications.

REFERENCES

[1] Nakamoto S. Bitcoin: A peer-to-peer electronic cash system. Available at SSRN 3440802. 2008 Aug 21. http://dx.doi.org/10.2139/ssrn.3440802

[2] Kshetri N. 1 Blockchain's roles in meeting key supply chain management objectives. International Journal of information management. 2018 Apr 1;39:80-9. https://doi.org/10.1016/j.ijinfomgt.2017.12.00

[3] Alqarni MA, Alkatheiri MS, Chauhdary SH, Saleem S. Use of blockchain-based smart contracts in logistics and supply chains. Electronics. 2023 Mar 11;12(6):1340. https://doi.org/10.3390/electronics12061340

[4] Geimer H, Vermeire P, van Ostaeyen L, Kapasi H. Blockchain in Logistics. Article. June. 2020. https://www.pwc.de/en/transport-und-logistik/blockchain-in-logistics.pdf?utm

[5] Mohammad A, Vargas S. Challenges of using blockchain in the education sector: A literature review. Applied Sciences. 2022 Jun 23;12(13):6380.

[6] IBM & Maersk. *TradeLens Platform Overview.* 2019. https://dgtr.de/wp-content/uploads/TradeLens-Blockchain.pdf

[7] Casino F, Kanakaris V, Dasaklis TK, Moschuris S, Stachtiaris S, Pagoni M, Rachaniotis NP. Blockchain-based food supply chain traceability: a case study in the dairy sector. International journal of production research. 2021 Oct 2;59(19):5758-70. https://doi.org/10.1080/00207543.2020.1789238

[8] VeChain Foundation. *Web3 for Better: Sustainable Blockchain Solutions* (Whitepaper). 2023. https://www.vechain.org/assets/whitepaper/whitepaper-3-0.pdf?utm

[9] Delgado-von-Eitzen C, Anido-Rifón L, Fernández-Iglesias MJ. Blockchain applications in education: A systematic literature review. Applied Sciences. 2021 Dec 12;11(24):11811. https://doi.org/10.3390/app112411811

[10] Ullah N, Mugahed Al-Rahmi W, Alzahrani AI, Alfarraj O, Alblehai FM. Blockchain technology adoption in smart learning environments. Sustainability. 2021 Feb 7;13(4):1801. https://doi.org/10.3390/su13041801

[11] Solihin MA, Nur D, Tungadi E, Yusri IK. Logistic supply chain management system modeling using blockchain. Jurnal Teknologi Elekterika. 2024 Nov 15;21(2):142-9. https://doi.org/10.31963/elekterika.v21i2.5110

[12] Safitri W, Huda M. Adoption of Blockchain Technology in Indonesian MSME Supply Chain Management (SCM). Jurnal Ekonika vol. 2023;8:2.

[13] Naik PG, Naik GR. Every Stuff You Need for Development of Decentralized App Using Blockchain Technology:(Covers Hardhat, React. js and Ethers. js). Shashwat Publication; 2023 Nov 20.

[14] Zheng Z, Xie S, Dai H, Chen X, Wang H. An overview of blockchain technology: Architecture, consensus, and future trends. In2017 IEEE international congress on big data (BigData congress) 2017 Jun 25 (pp. 557-564). Ieee.

[15] Crosby M, Pattanayak P, Verma S, Kalyanaraman V. Blockchain technology: Beyond bitcoin. Applied innovation. 2016 Jun;2(6-10):71.

[16] Saberi S, Kouhizadeh M, Sarkis J, Shen L. Blockchain technology and its relationships to sustainable supply chain management. International journal of production research. 2019 Apr 3;57(7):2117-35. https://doi.org/10.1080/00207543.2018.1533261

[17] Thomas JW. A review of research on project-based learning.2000.

[18] Prince MJ, Felder RM. Inductive teaching and learning methods: Definitions, comparisons, and research bases. Journal of engineering education. 2006 Apr;95(2):123-38. https://doi.org/10.1002/j.2168-9830.2006.tb00884.x

[19] Hapuarachchi T, Mapkar M, Rahouti M, Xiong K. Advancing Blockchain Learning in STEM Education Through A Comprehensive Hands-On Educational Approach. In 2024 IEEE Integrated STEM Education Conference (ISEC) 2024 Mar 9 (pp. 1-6). IEEE.

[20] Steiu MF. Blockchain in education: Opportunities, applications, and challenges. First Monday. 2020 Aug 24.