

CampusCoin: A Blockchain-Based Smart Contract System for Automated Reward Distribution and Token-Based Payments on Academic Campuses

Mrs. M. VARALAKSHMI, ERANALA SRUJAN, BADAVATH UDAY

Assistant Professor, Dept. of Information Technology, MGIT, Hyderabad, India

B.Tech Student, Dept. of Information Technology, MGIT, Hyderabad, India

B.Tech Student, Dept. of Information Technology, MGIT, Hyderabad, India

Abstract - Rewards are small forms of recognition given to students when they take part in events or perform well on campus, and they keep students motivated and make their effort feel valued. In earlier days, rewards were given through paper certificates, medals, and entries in registers maintained by the faculty, and the same paper was the only proof a student had of having participated in anything. Once computers were introduced, most institutions switched to spreadsheets. The data is stored on a college server, in PDFs and in event portals where faculty enter the names in a file. Students receive certificates and by email after a few days or weeks. The problem with this existing system is that it's slow, easy to lose, and not transparent. A file can be edited later, the certificate may be misplaced and the student has no means of knowing that the prize was actually received from the institution. In this project, we have built CampusCoin, a blockchain-based campus reward platform where the faculty marks a student's participation and the system sends a token straight to the student's wallet, with the transaction recorded on a public ledger. These tokens can be used for the same items at campus vendors, such as canteens and stationery shops, will be able to access the rewards. The system uses an ERC-20 smart contract on an Ethereum-compatible network, with the frontend built in React, the backend in Node.js and Express, the database in PostgreSQL, and ethers.js handling all wallet operations. The advantages are that it is transparent, it gives immediate rewards, and it has one trusted record for the whole campus; the main limitation is that students must have a wallet such as MetaMask and a short time to get used to the platform and a short learning period to get used to it. In future, the platform can be extended to multiple colleges, made mobile-friendly, and connected with the existing ERP and event-management systems already used by institutions.

Key Words: Blockchain; Smart Contract; ERC-20; Campus Rewards; Ethereum; OpenZeppelin; Role-Based Access Control; Decentralised Application; Peer-to-Peer Payments; Digital Ledger

1. INTRODUCTION

The majority of academic institutions are still manually administering student rewards and campus payments. Event participation prizes are awarded on paper systems, faculty verification is based on personal records that can be lost or challenged, and there is no proper auditable trail for vendor transactions, which are mostly cash-based. These deficiencies result in three significant issues simultaneously: lack of transparency on who received what and when, there is no tamper-proof record that can be independently verified by multiple parties, and there is no proper mechanism to automate the transfer of rewards after faculty approval is made.

Blockchain technology is a suitable solution to these kinds of problems. A smart contract is a piece of code that runs automatically on a blockchain when certain conditions are met, logs all transactions on an

immutable ledger that is accessible to all participants, and validates rules with cryptographic logic instead of institutional rules. There are some previous studies that have explored the application of blockchain in academic credential verification [1], the storage of learning analytics [2], and campus record keeping [3], but there is a lack of systems that integrate reward distribution, vendor payments and a shared auditable ledger in a single platform for campus use.

This paper contributes the following:

- Fixed-supply token economy: ERC-20 smart contract (CAMP) with reward slabs for different positions (Winner, Runner-Up, Participant) that can be set per event, deployed on Ethereum-compatible networks.
- On-chain duplicate-payment guard: A guard that is implemented on-chain and prevents any

participation from being rewarded twice, using a participationRewarded mapping.

- Four role-based access control roles: Admin, Faculty, Student, and Vendor roles, with each role having access to only the operations that are relevant to them, using OpenZeppelin's AccessControl library.
- QR-based vendor settlement: A vendor payment flow based on QR code, where the payment is made in CAMP tokens and the event of the transfer is verified on-chain prior to updating the invoice status.
- Dual ledger design: The blockchain is the authoritative source of token balances, and the workflow state is stored in PostgreSQL, enabling rich relational queries without compromising on-chain integrity or paying unnecessary gas fees to store the workflow state on-chain.

2. RELATED WORK

There has been a significant amount of research at the intersection of blockchain and education, which can be grouped into three categories: credential verification, token-based incentives, and digital payments. Previously, it was not possible to have automated reward systems, but with the latest developments in smart contract technology, especially on the Ethereum platform, campuses can now develop self-executing reward systems.

Turkanovic et al. [1] introduced a new credit transfer system for higher education based on blockchain (EduCTX) that allows for the transfer of academic records from one HE institution to another in a portable and tamper-proof way. While EduCTX plays a crucial role in establishing institutional trust on-chain, it is primarily geared towards certificates rather than dynamic reward flows and token economies. Alammary et al. [2] also did a systematic review of the use of blockchain in education, and categorized the applications into four use cases: credential storage, access control to resources, payment management, and smart certificates. They discovered that the events that are recognized as participation events in the existing systems do not have automatic reward triggering.

Ocheja et al. [3] suggested an architecture for learning analytics data management without any monetary incentives, but with the help of blockchain technology. They show the data-custody aspect of blockchain, but not on-chain token transfers or vendor payment integration.

Blockchain Architecture and Scalability: Zheng et al. [4] gave a comprehensive overview of the blockchain architecture, consensus mechanisms, and open research challenges. Their analysis of performance and scalability trade-offs influenced the choice of using a local Ethereum-compatible network for development, along with implementing a fixed-supply token model to make on-chain state management easier.

Reward Systems: Tran et al. [5] talked about the reward systems based on blockchain that can give rewards to community members for their peer-review activities. Their work supports the design principle of token scarcity, which is also adopted in the campus incentive model in this paper.

Smart Campus Token Economies: Khanna et al. [6] surveyed the implementation of token economies in smart campus environments and concluded that most of the implementations have more than one payment rail, that is, rewards and vendor payments are in separate and disconnected systems. CampusCoin solves this problem with the CAMP token, which will be the universal currency for distributing rewards and settling vendors on campus.

Security and Access Control: Raimundo and Rosario [7] presented the topic of blockchain in Higher Education from the information security point of view and highlighted the need for role-based access control for academic platforms that have multiple users and roles. This finding resulted in the use of OpenZeppelin's AccessControl module, which allows for more granular and transparent privilege delegation throughout the platform than a single-owner approach.

Research Gap: The current literature is focused on certain aspects such as credential storage, learning analytics, and token incentives. However, there is still a long way to go before a single platform can combine event-driven reward distribution, real-time vendor payments and a shared public ledger. CampusCoin platform provides a comprehensive, scalable and integrated solution to the campus reward and payment problem to address these challenges.

3. METHODS AND MATERIALS

The proposed system will try to provide an end-to-end blockchain solution that will automate the campus reward process and the vendor payment lifecycle in one application. It has been built to enable structured transfers of tokens, real-time verification on-chain and continuous performance monitoring via modern web technologies and smart contracts on Ethereum. It features four layers of architecture: frontend interface,

backend processing layer, smart-contract layer, and relational database layer, all of which help to ensure a seamless user experience and efficient system performance.

3.1 Smart Contract Layer

The CampusCoin smart contract has been written in Solidity ^0.8.19 and is based on OpenZeppelin's ERC20 and AccessControl contracts. The CAMP tokens are minted to the treasury wallet during deployment and the mint() function is not exposed after deployment. All rewards and payments are done in the form of treasury transfers, meaning that the total number of tokens is provably constant from the first block. The two administrative functions in the reward economy are setEventRewards(eventId, winnerWei, runnerUpWei, participantWei), which is called when an event is created and stores the reward amount per position on-chain, and payParticipationReward(eventId, participationId, position), which is called when the faculty confirms the participation and transfers the reward from the treasury to the student. Before a transfer is made, there are three checks that are performed: event rewards must be set, participationRewarded[participationId] mapping must be false, and the treasury wallet must have enough CAMP balance.

3.2 Frontend Interface Development

The system's user interface has been created with React 18 and custom CSS for styling. There are four dashboards - Administrator, Faculty, Student and Vendor - each of which shows only the operations relevant to the respective role. The communication with the wallet is done via MetaMask and ethers.js v5.7's Web3Provider. The useWallet custom React hook is used to connect to a wallet, validate chain ID, and verify the wallet role with hasRewardManagerRole(). If a faculty wallet has the on-chain REWARD_MANAGER role, then reward transactions are signed directly in the browser with MetaMask, otherwise the backend signer signs the transaction on behalf of the faculty user.

3.3 Backend System Integration

The backend is a single file Node.js / Express application with about 2100 lines of code, and uses ethers.js v5.7 for all on-chain interactions. When the backend is started, it loads the contract ABI from the coinABI.json file, attaches a JsonRpcProvider to the configured Ethereum node and creates a Wallet with the BACKEND_PRIVATE_KEY environment variable that will be used to sign on-chain transactions.

The four different roles are enforced with the help of JWT claims being validated against the PostgreSQL users table, via the role-based middleware requireActor(). All the token balance queries and all the reward disbursements are going through the smart contract, and all the workflow state, which is the event records, the participation statuses and the invoice lifecycle, are stored in PostgreSQL via a pg connection pool.

3.4 Database and Relational Layer

The workflow information is kept in five main tables: users, events, participations, invoices and transactions in PostgreSQL. The transactions table supports idempotent transaction recording and has a composite unique index (tx_hash, chain_id, coin_address). The ON CONFLICT DO NOTHING clause will prevent duplicate ledger entries if the same on-chain transaction is observed by the polling worker more than once, which can lead to duplicate entries.

3.5 Vendor Payment Module

The vendor generates an invoice for the customer, and the payment is settled in CAMP tokens. The backend is generating a base64-encoded QR payload which contains the invoice ID, the vendor wallet address as well as the token amount. The student then scans the QR code, which causes a regular ERC-20 transfer() call to be sent from the student's wallet to the vendor's wallet. The backend receives the transaction hash from the frontend and then calls verifyTransfer(). This function is reading the Ethereum transaction receipt directly from the node and is validating the Transfer event log against the expected recipient and amount, before updating the invoice status to PAID. User-submitted input is never trusted for any type of settlement decision.

3.6 Role-Based Access Control

The OpenZeppelin AccessControl module controls three on-chain roles: DEFAULT_ADMIN_ROLE (granted to the deployer or the treasury wallet) that can be granted or revoked by the DEFAULT_ADMIN_ROLE; REWARD_MANAGER (granted to the backend signer wallet, and optionally to the faculty wallets) that is required to call setEventRewards() and payParticipationReward(); and the standard ERC-20 token holder capability, which allows any Ethereum address to receive and transfer CAMP. In addition to these on-chain roles, the application layer has four different roles: Admin, Faculty, Student and Vendor, which are enforced with

a second layer of permission control on the API endpoints via the `requireActor()` middleware.

3.7 System Workflow and Process

1. The user logs in to the platform using the assigned credentials.
2. The backend validates the credentials against the users table and issues a JWT bearer token for the session.
3. The Administrator creates an event by entering the title, the date, and the per-position reward amounts.
4. The backend calls `setEventRewards()` on-chain in order to record the reward slabs alongside the off-chain event metadata.
5. The Faculty marks a student's participation in the event with a position (Winner / Runner-Up / Participant).
6. The Faculty then verifies the participation, which triggers a call to `payParticipationReward()`.
7. The smart contract checks the duplicate-payment guard and, if it is allowed, transfers the CAMP from the treasury to the student.
8. The student is able to see the updated CAMP balance inside the dashboard through MetaMask.
9. The Vendor creates an invoice by entering the amount in CAMP, and the backend generates a QR code.
10. The student scans the QR and signs the ERC-20 transfer through MetaMask.
11. The backend verifies the transaction receipt and marks the invoice as PAID.
12. All the on-chain activity is being recorded inside the PostgreSQL transactions table for later auditing.

4. EXPERIMENTAL STUDY

This section describes the experimental setup and the testing that was carried out on the proposed CampusCoin system. The implementation has been tested to gauge user interactions, reward disbursement, secure execution of on-chain transactions and the functionality of all four role panels. The interactions between the React front-end and the Express back-end, between the smart contract on the Ethereum compatible network and the PostgreSQL database have also been examined to guarantee the overall system's reliability and efficiency.

The proposed system is developed using the latest web technologies. The frontend is created with React 18, the backend with Node.js and Express 4, the smart contract with Solidity and Hardhat v2.22, OpenZeppelin Contracts v4.9.2, and the database is

PostgreSQL 15. The system can be run on three target networks which are selectable through the `DEMO_NETWORK` environment variable - Ganache (port 7545, chain ID 1337) for GUI-based local development, Hardhat node (port 8545, chain ID 31337) for automated CI testing, and Sepolia testnet for public demonstration purposes. The deploy script automatically grants the `REWARD_MANAGER` role to the backend wallet address, and also prints the address of the contract to be added to the `.env` file.

A total of 20 structured test cases (TC-01 to TC-20) have been developed to test the system, covering the whole life cycle for each stakeholder role. The Administrator role was tested to ensure that new events can be created, per position reward slabs can be set up and the `REWARD_MANAGER` role can be added to faculty wallets. The Faculty role was tested to verify the ability to mark and confirm student participations. The Student role was tested to ensure that students are able to see the CAMP balance, check the history of rewards and pay vendor invoices using QR scanning. The Vendor role was tested for the creation of new invoices and viewing payment history.

The smart contract test suite verifies four properties of the smart contract: After deployment, the total supply is exactly 1,000,000 CAMP; Wallets with the `REWARD_MANAGER` role can call privileged functions; `payParticipationReward()` correctly transfers the per-position amounts when event rewards are configured; and a second call for the same participation ID reverts with the duplicate-guard error message.

The performance of concurrent transactions and consistency of the dual-ledger design were also tested. Several user accounts were established and different real-world campus scenarios were simulated. The authentication system was tested to make sure that only authorized users can access the system and related data. The transactions table of the PostgreSQL database was also verified to make sure that all on-chain transactions are properly recorded with the proper `tx_hash`, `chain_id` and `coin_address`. To ensure that the system does not accept maliciously altered transaction hashes, the `verifyTransfer()` function was tested by sending a modified transaction hash.

All test rounds were successful. The reward disbursement module was reliable, the outputs were structured, smart-contract transactions were confirmed with low latency and the vendor QR payment module efficiently handled end-to-end token transfers. The platform was capable of maintaining user session

information as well as participation history for later audit access. Moreover, the system ensured proper handling of user interactions and no data loss or data inconsistencies. Even in simulated retry scenarios, the on-chain authoritative balance was synced with the off-chain workflow state using the dual-ledger pattern. The experimental analysis showed that the blockchain and web modules were successfully integrated into the proposed system, and the system was able to effectively manage the campus reward and payment processes.

5. RESULTS

The outcome of the experimental evaluation demonstrates the effectiveness of the proposed system in providing a reliable and efficient reward and payment system for events, which integrates event reward management, faculty participation verification, student wallet integration, and the QR payment flow in a single environment. The results for the various test cases that were created for the system are summarised in Table 1.

Table -1: Functional Test Case Summary

Test Category	TCs	Pass	Fail
Authentication and Authorisation	3	3	0
Event Creation and On-Chain Configuration	3	3	0
Participation Marking and Reward	4	4	0
Duplicate Reward Prevention	2	2	0
Wallet Connection and Chain Validation	2	2	0
Vendor Invoice and QR-Based Payment	3	3	0
Contract Invariants and Treasury	3	3	0
Total	20	20	0

All 20 test cases passed without requiring modifications. The following are some of the key observations made during the test run, and the figures

are also included to show the operation of the system for different role panels.

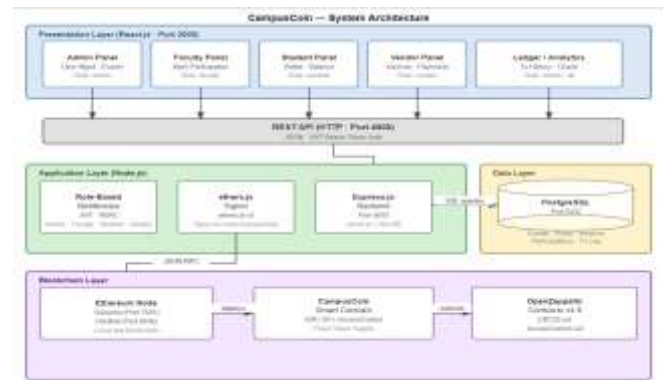


Fig -1: Four-Tier System Architecture of CampusCoin.

This figure presents a high-level view of the system architecture, where the four main components are the React frontend, the Express backend, the PostgreSQL database, and the Ethereum smart contract, all connected through HTTP REST, SQL and JSON-RPC interfaces.

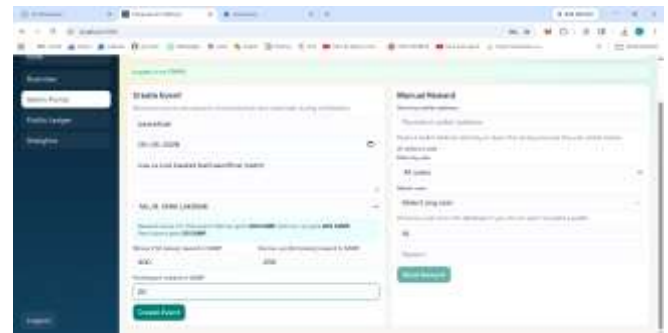


Fig -2: Administrator Dashboard for event creation and reward slab configuration.

This is the interface that the Administrator uses to set up a new event and add the reward amounts for each position in CAMP. Event reward configuration is persisted on-chain after submission using setEventRewards(), and the event metadata is stored in PostgreSQL off-chain.



Fig -3: Faculty Panel showing pending and verified participations.

This figure shows the process used by the faculty to mark and verify participations. When the Verify button

is clicked, if the faculty wallet has the REWARD_MANAGER on-chain role, the system calls payParticipationReward() through the faculty's MetaMask wallet; otherwise the transaction is executed through the backend signer wallet.



Fig -4: Student Dashboard with CAMP balance and reward history.

This figure shows the dashboard visible to the student, where the live CAMP token balance, the participation history and the on-chain transaction hashes are displayed. The student is able to view the reward details and also confirm the on-chain transfers using a public block explorer.

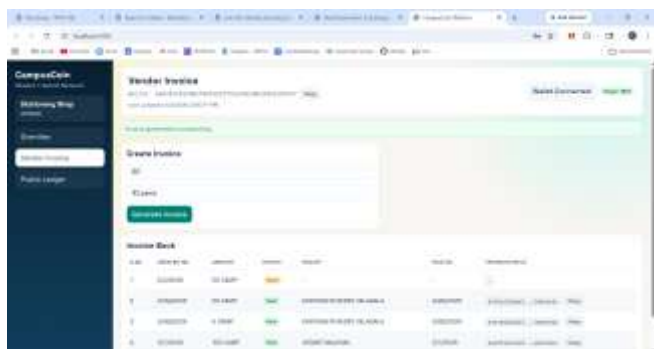


Fig -5: Vendor Invoice with QR Code for payment.

This figure shows the interface used by the vendor to create a new invoice for a single student. The system creates a QR code containing the invoice ID, vendor wallet address and the amount in CAMP. The student uses a scanning interface to scan the QR code and signs the ERC-20 transfer using MetaMask.

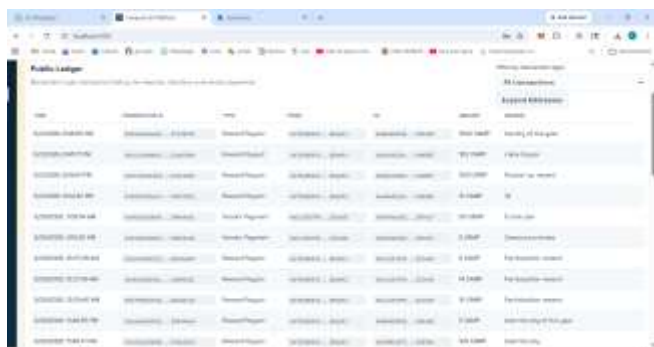


Fig -6: Public Ledger Panel with paginated transaction history.

The figure shows the ledger view, which is available to all stakeholders for auditing purposes, and the recent on-chain transactions, event references, wallet addresses and the timestamp information are displayed in a paginated table format with a CSV export option.

The other interesting outcome of these test runs was that if a second verification attempt was made for the same participation ID through the participationRewarded guard, the caller would get an HTTP 409 Conflict, resulting in a revert at the blockchain level. TC-14 (QR Payment Integrity) confirmed that the system was able to reject a tampered transaction hash that was sent to a different recipient address, leaving the invoice in Pending state and protecting the vendor from such attacks. TC-19 (Treasury Invariant) checked the no-inflation property of the system, and found that totalSupply() returned exactly 1,000,000 CAMP after eight reward payments and three vendor payments had already been made. TC-20 (Ledger Completeness) also verified that all on-chain transactions were recorded in the PostgreSQL transactions table using the proper identifiers, thus proving dual-ledger consistency.

The reward transactions were processed in the local Ganache network within 1-2 seconds. The 50-entry ledger page was able to return results in less than 10 milliseconds on the development machine with the PostgreSQL queries. The QR code generation and scan-to-confirm took less than 5 seconds. The results are based on a single-machine development environment. A comprehensive latency and throughput benchmarking study for production deployment on a public testnet or a Layer-2 network with concurrent load has been left for future work.

6. CONCLUSION

The CampusCoin platform outlined in this paper illustrates how a single blockchain-based solution can streamline and enhance transparency throughout the campus reward and payment process. It starts with the creation of the event, the verification by the faculty and finally the crediting to the student wallet and settlement with the vendor. It is a fixed-supply ERC-20 token system that provides on-chain duplicate-payment protection and role-based access control, meaning that it provides a tamper-proof environment without relying on any single administrator or central authority. Every transaction is written on the public ledger and the rules of payment are not only enforced through institutional convention, but also by cryptographic mechanisms via the smart contract.

The dual-ledger system can be used by institutions that require on-chain integrity and the ability to perform relational queries. The token state is stored on the blockchain and the workflow state is stored in the PostgreSQL database. This means that no extra gas is needed to store the metadata on-chain, but it's still auditable and consistent. It has been tested using 20 structured test cases and all the major flows have been validated successfully. These were the event lifecycle, duplicate-payment prevention, vendor payments through QR code and consistency of the ledger.

The system can be expanded in a number of ways. An alternative to the public chains that might be a constraint in institutional deployments is a permissioned consortium blockchain such as Hyperledger Besu. The content and the participation evidence of the event could also be stored on IPFS for decentralised storage. Moreover, the platform's support for Layer-2 solutions like Polygon or Optimism can also assist in reducing transaction latency and gas costs during busy periods at the campus. Additional future enhancements include mobile support and more sophisticated governance models to manage reward policies throughout the campus ecosystem.

REFERENCES

- [1] M. Turkanovic, M. Holbl, K. Kosic, M. Hericko, and A. Kamisalic, "EduCTX: A Blockchain-Based Higher Education Credit Platform," *IEEE Access*, vol. 6, pp. 5112–5127, 2018.
- [2] A. Alammary, S. Alhazmi, M. Almasri, and S. Gillani, "Blockchain-Based Applications in Education: A Systematic Review," *Applied Sciences*, vol. 9, no. 12, p. 2400, 2019.
- [3] P. Ocheja, B. Flanagan, H. Ueda, and H. Ogata, "Managing Lifelong Learning Records through Blockchain," *Research and Practice in Technology Enhanced Learning*, vol. 14, no. 1, pp. 1–19, 2022.
- [4] Z. Zheng, S. Xie, H. Dai, X. Chen, and H. Wang, "Blockchain Challenges and Opportunities: A Survey," *International Journal of Web and Grid Services*, vol. 14, no. 4, pp. 352–375, 2018.
- [5] T. T. Tran, T. L. Nguyen, and P. H. Le, "A Blockchain-Based Reward System for Collaborative Learning Communities," *IEEE Transactions on Learning Technologies*, vol. 16, no. 2, pp. 213–226, 2023.
- [6] A. Khanna, S. Jain, and R. Mehta, "Token Economies in Smart Campus Environments: A Survey and Future Directions," *Computers and Education: Artificial Intelligence*, vol. 7, p. 100215, 2025.
- [7] R. Raimundo and A. Rosario, "Blockchain System in the Higher Education," *European Journal of Investigation in Health, Psychology and Education*, vol. 11, no. 1, pp. 276–293, 2021.
- [8] M. Sharples and J. Domingue, "The Blockchain and Kudos: A Distributed System for Educational Record, Reputation and Reward," in *Proc. European Conference on Technology Enhanced Learning (EC-TEL)*, 2016, pp. 490–496.
- [9] OpenZeppelin, "OpenZeppelin Contracts v4.9.2: ERC20 and AccessControl," *GitHub Repository*, 2023. [Online]. Available: <https://github.com/OpenZeppelin/openzeppelin-contracts>
- [10] Ethereum Foundation, "Ethereum: A Next-Generation Smart Contract and Decentralized Application Platform," *Ethereum Whitepaper*, 2014.
- [11] S. Nakamoto, "Bitcoin: A Peer-to-Peer Electronic Cash System," *Bitcoin.org*, 2008.
- [12] A. Gala, "Blockchain Adoption in Higher Education: Challenges and Opportunities," *Journal of Educational Technology and Society*, vol. 24, no. 3, pp. 1–15, 2021.
- [13] V. Buterin, "A Next-Generation Smart Contract and Decentralized Application Platform," *Ethereum White Paper*, 2014.
- [14] W. Li, A. Sforzin, S. Fedorov, and G. Karame, "Towards Scalable and Private Industrial Blockchains," in *Proc. ACM Workshop on Blockchain, Cryptocurrencies and Contracts (BCC)*, 2017, pp. 9–14.