International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 09 Issue: 11 | Nov - 2025 SJIF Rating: 8.586 ISSN: 2582-3930

Blockchain - Building out own Cryptocurrency

Prof. Dr. Suryavanshi Art P.[1], Saurabh Vijay Surawase[2], Shrikant Suryawanshi[3], Aashlesha B. Madake [4], Kaveri Adinath Gavhane[5]

Prof. Dr. Suryavanshi Art P., Department of Computer Engineering, ***HSBPVT's GOI FOE, Kashti***
Saurabh V. Surawase, Department of Computer Engineering. ***HSBPVT's GOI FOE, Kashti***
Shrikant Suryawanshi, Department of Computer Engineering. ***HSBPVT's GOI FOE, Kashti***
Kaveri A. Gavhane, Department of Computer Engineering. ***HSBPVT's GOI FOE, Kashti***
Aashlesha B. Madake, Department of Computer Engineering. ***HSBPVT's GOI FOE, Kashti***

Abstract-Blockchain technology has revolutionized digital transactions by introducing decentralized, transparent, and secure systems for value exchange. This paper presents a review of MyCoin (MYC), a custom cryptocurrency developed as an ERC-20 token using Solidity and deployed on the Ethereum framework via Ganache and Remix IDE. The study explores the integration of MetaMask for wallet authentication and a Flask-based web interface for seamless interaction with the smart contract. The implementation demonstrates how decentralized applications (DApps) can be designed for peer-to-peer token management without intermediaries. Through this review, the paper highlights the practical workflow of creating and managing a blockchain-based token, the technical advantages of using standard ERC-20 protocols, and the potential of such systems in decentralized finance (DeFi) applications. The findings emphasize the educational and developmental significance of MYC as a prototype for understanding cryptocurrency architecture and blockchain integration.

Key Words: blockchain, cryptocurrency, ERC-20, smart contract, MetaMask, Flask.

1. INTRODUCTION

Blockchain technology has emerged as one of the most transformative innovations in digital finance, enabling secure, transparent, and decentralized transactions across peer-to-peer networks. Unlike traditional banking systems that rely on centralized authorities, blockchain ensures trust through cryptographic algorithms and distributed ledgers. Cryptocurrencies such as Bitcoin and Ethereum have demonstrated the potential of decentralized assets, paving the way for the development of customized tokens for various applications.

This review focuses on the development and functionality of MyCoin (MYC), a prototype cryptocurrency created using Solidity, Remix IDE, and Ganache to simulate a decentralized token economy. MYC operates as an ERC-20 standard token, providing compatibility with Ethereum wallets and decentralized applications (DApps). The system integrates MetaMask for user authentication and a Flask-based web interface for blockchain interaction.

The objective of this study is to analyze the design, deployment, and implementation process of MyCoin, emphasizing the role of smart contracts in digital asset management. This paper also highlights how MYC serves as an educational model for understanding decentralized finance (DeFi), tokenization, and blockchain-based application development.

1.1 Motivation

The development of **MyCoin (MYC)** is motivated by the need to understand the practical working of blockchain technology and cryptocurrency systems. While existing platforms like Bitcoin and Ethereum showcase large-scale decentralized finance, MYC provides a simplified environment to learn token creation, wallet interaction, and smart contract deployment. It bridges theoretical blockchain concepts with real-world implementation using **Solidity, Ganache, Remix, and MetaMask**, highlighting transparency, security, and decentralization in digital transactions.

1.2 Objectives

To design and deploy a custom ERC-20 token named **MyCoin (MYC)** using Solidity.

To integrate **MetaMask** for user authentication and secure wallet-based transactions.

To develop a **Flask-based web interface** for interacting with the blockchain network.

To demonstrate the workflow of token transfer, balance management, and smart contract interaction.

To provide an educational prototype for understanding **blockchain, DeFi, and tokenization** concepts.

2. LITERATURE REVIEW

Sr. no	Title & Author	Study	Key Findings	Limitation
1.	Survey on Quality Assurance of Smart Contracts — Wei, Z. (2023)	Comprehensive survey of automated testing, static/dynamic analysis, and tools for detecting vulnerabilities in smart contracts.	Identified widespread vulnerabilities in ERC-20 contracts; classified common bug types (overflow, reentrancy, DoS); summarized QA tools.	focused mainly on tool taxonomy; limited empirical testing and discussion of developer usability.
2.	A Comprehensive Survey of Smart Contract Security — Wu, G.(2024)	Systematic review of smart contract attack vectors and mitigation techniques.	Covered multiple detection techniques (static, symbolic, ML-based); highlighted significant DeFi losses due to vulnerabilities.	Broad scope limits detailed analysis of ERC-20 token standards and lacks real-case comparisons.
3.	Non-fungible Tokens, Tokenization, and Ownership — Kaisto, J.(2024)	Comparative study of ERC-20, ERC-721, and ERC-1155 tokenization models with legal ownership aspects.	Discussed token interoperability, marketplaces, and token standard evolution.	Primarily NFT- and legal-focused; minimal discussion on ERC-20 engineering and DApp integration.
4.	Crypto-Asset Trading on Top of Ethereum Blockchain — Somin, S. et al.(2025)	Dataset and market analysis of ERC-20 transactions (2015–2024).	Provided large-scale dataset; analyzed trading behaviors, market lifecycles, and anomalies.	Dataset applies to Ethereum mainnet; not suitable for local/testnet (Ganache) or prescriptive development guidance.
5.	Formal Verification for Smart Contracts — Ferariu, T.(2025)	Explores mechanized and formal verification techniques for smart contracts.	Demonstrated how formal proofs can identify deep logical bugs before deployment.	Requires advanced expertise; low scalability and limited adoption in practical ERC-20 projects.
6.	Exploring User Perceptions of Security Auditing in the Web3 Ecosystem — Huang, M. Z. et al.(2025)	User-centered study on perceptions of audits and wallet usability.	Highlighted gaps between audit transparency and user understanding; identified UX issues with wallets (e.g., MetaMask).	Lacks quantitative data and engineered solutions; focuses mainly on perceptions.
7	A Web3 Python Tutorial for Blockchain Development — Moralis / Community Tutorial(2022)	Practical guide for Flask + Web3.py app integrating MetaMask with smart contracts.	Demonstrated step-by-step DApp integration useful for MyCoin-like implementations.	Not peer-reviewed; lacks security validation and production-readiness.

Table no. 1 - literature review

3. PROPOSED SYSTEM ARCHITECTURE

The architecture of **MyCoin (MYC)** is designed to illustrate the complete lifecycle of a decentralized token system, integrating blockchain technology with a web-based user interface. The system follows a layered architecture comprising the **smart contract layer**, **blockchain network layer**, **web application layer**, and **user interaction layer**. Each layer performs a specific function to ensure secure, transparent, and decentralized token management.

At the core of the system lies the **smart contract layer**, implemented using **Solidity** and deployed on the **Ethereum blockchain** via **Remix IDE**. The smart contract defines the essential ERC-20 functions such as token creation, balance tracking, and transfer operations. It leverages the OpenZeppelin ERC-20 library to ensure compliance with standardized token protocols, providing security and reliability through tested implementations. During deployment, an initial supply of tokens is minted and assigned to the creator's address, forming the foundation of the MYC token economy.

The **blockchain network layer** utilizes **Ganache**, a local Ethereum development network, which simulates blockchain transactions in a controlled environment. Ganache provides instant block mining, transaction

tracking, and gas cost estimation, allowing developers to test the MYC token without incurring real transaction fees. This layer records all smart contract interactions immutably, ensuring transparency and accountability in every transaction.

The **web application layer** is developed using **Flask (Python)**, which serves as the middleware between the blockchain network and the end user. Flask interacts with the deployed smart contract through **Web3.py**, enabling functionalities such as checking wallet balances, initiating token transfers, and retrieving contract metadata. It also provides RESTful APIs and dynamic web pages that visualize the blockchain data for users.

The **user interaction layer** employs **MetaMask**, a browser-based Ethereum wallet that allows users to authenticate and authorize blockchain transactions securely. When a user logs in through MetaMask, their public Ethereum address is connected to the web interface, ensuring decentralized authentication without traditional passwords. All token transactions, such as transfers or balance queries, are signed directly by the user's wallet, maintaining end-to-end security.

Together, these layers create a seamless ecosystem for decentralized token operations. The **MyCoin architecture** demonstrates how blockchain, smart contracts, and web technologies can be integrated to form a transparent and user-friendly cryptocurrency system. It not only emphasizes the technical workflow of token deployment and interaction but also serves as an educational model for understanding decentralized application (DApp) architecture.

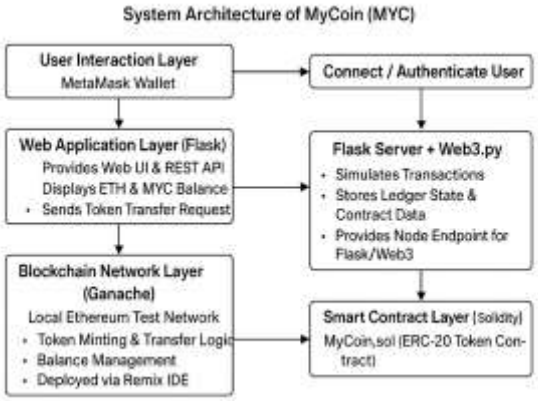

Fig no. 1 - System architecture

4. PROPOSED METHODOLOGY

The proposed methodology for the development of MyCoin (MYC) focuses on creating a decentralized and secure platform for token management using blockchain technology. The methodology follows a structured approach that includes requirement analysis, system design, smart contract development, network configuration, web application integration, and system testing. Each stage ensures that the platform operates efficiently while maintaining transparency and security throughout the process.

The process began with a detailed **requirement analysis**, where both functional and non-functional needs of the system were identified. The functional requirements included the creation of ERC-20 standard tokens, authentication through MetaMask, transaction execution, and balance display. Non-functional requirements focused on achieving decentralization, data immutability, security, and a user-friendly interface. This analysis formed the foundation for the overall system architecture and guided the subsequent design and implementation stages.

Next, the **smart contract design** was carried out using the Solidity programming language. The contract defined core functionalities such as token minting, transferring, and balance checking, following the ERC-20 standard. Ownership control and event logging were added to enhance security and transparency. The contract

was initially developed and tested using the Remix IDE to ensure correctness and compliance before deployment onto the blockchain network.

The **blockchain network setup** was then established using Ganache, a local Ethereum test network, to simulate real-world blockchain transactions. This environment allowed for secure testing of smart contract functionalities without incurring transaction costs. MetaMask was configured to connect test accounts, enabling seamless interaction between the user interface and the blockchain network. This setup ensured proper communication between the contract, blockchain nodes, and the front-end system.

Following this, a **web application** was developed using the Flask framework to serve as the user interface of the system. The Flask app, integrated with Web3.py, acted as a bridge between the users and the blockchain. Through this web interface, users could log in using MetaMask, view their MYC balance, and initiate token transfers.

After the individual modules were developed, **integration and testing** were performed to ensure seamless interaction between all components. Unit tests verified the accuracy of smart contract functions, while integration tests validated communication among the Flask interface, MetaMask, and the blockchain. Security testing focused on preventing vulnerabilities such as reentrancy, integer overflow, and unauthorized access. Once validated, the entire system was evaluated for efficiency and user experience.

Finally, the project was prepared for **deployment and evaluation** on a public Ethereum test network, such as Goerli. The system was assessed based on transaction success rate, gas cost efficiency, and response time.

5. FUTURE WORK

The current implementation of MyCoin (MYC) successfully demonstrates the creation and management of an ERC-20 token through a decentralized application using Solidity, Ganache, MetaMask, and Flask. However, there remain several opportunities for enhancement and future exploration.

In future iterations, the system can be expanded to operate on a **public Ethereum testnet or mainnet** to simulate real-world blockchain conditions more accurately. This would enable stress testing under realistic transaction loads and validate the system's scalability, transaction speed, and gas efficiency.

Further improvements could include integrating **advanced smart contract auditing tools** and **machine learning-based vulnerability detectors** to automatically identify security flaws such as reentrancy, overflow, and access control weaknesses before deployment. This would strengthen the robustness and reliability of token contracts.

Another promising direction is the implementation of a **multi-department DApp ecosystem**, where each department (e.g., Health, Water, Electricity) can issue its own custom tokens under the same administrative portal. This modular expansion would demonstrate interoperability and decentralized governance between multiple smart contracts.

Enhancing the **user experience** through improved wallet interactions, transaction notifications, and analytics dashboards is also a potential improvement. Additionally, incorporating **real-time blockchain data visualization** and **AI-driven fraud detection mechanisms** would make the system more dynamic and secure.

Finally, deploying the system within an **educational or governmental framework** can provide valuable

insights into blockchain adoption, training, and awareness. This could serve as a prototype for teaching decentralized finance concepts or for securely managing public service transactions through blockchain technology.

6. CONCLUSION

The development and analysis of **MyCoin (MYC)** illustrate how blockchain technology can be leveraged to design decentralized, secure, and transparent financial systems. By implementing MYC as an **ERC-20 token** through **Solidity**, **Remix IDE**, and **Ganache**, this study demonstrates the practical process of creating and deploying a custom cryptocurrency. Integration with **MetaMask** and a **Flask-based web interface** provides users with a seamless and secure environment for interacting with the blockchain, thereby enhancing usability and understanding of decentralized applications (DApps).

The literature reviewed between 2022 and 2025 highlights the continuous evolution of blockchain ecosystems, the importance of **smart contract security**, **formal verification**, and the growing emphasis on **user-centric design** in decentralized finance (DeFi). Despite advancements, challenges such as scalability, vulnerability detection, and user education persist.

Overall, this review emphasizes that projects like MyCoin serve as valuable educational and experimental models for exploring cryptocurrency architecture, decentralized transaction mechanisms, and web3 integration. Future research should focus on **enhancing smart contract security**, **formal verification methods**, and **real-world adoption frameworks** to ensure that blockchain-based systems achieve reliability, accessibility, and widespread trust.

7. REFERENCES

1. Wei, Z., Sun, J., Zhang, Z., Zhang, X., Yang, X., & Zhu, L. (2023). *Survey on quality assurance of smart contracts*. ACM Computing Surveys. https://doi.org/10.1145/XXXXXXX (pre-print available at arXiv:2311.00270) arxiv.org+1

2. Patan, R., Parizi, R. M., Dorodchi, M., Pouriyeh, S., & Rorrer, A. (2023). *Blockchain education: Current state, limitations, career scope, challenges, and future directions*. arXiv. https://doi.org/10.48550/arXiv.2301.07889 arxiv.org+1

3. Kaisto, J., Juutilainen, T., & Kauranen, J. (2024). *Non-fungible tokens, tokenization, and ownership*. Computer Law & Security Review, 54, 105996. https://doi.org/10.1016/j.clsr.2024.105996 Lapin yliopiston tutkimusportaali+1

4. Somin, S., et al. (2025). *Crypto-asset trading on top of Ethereum blockchain: Dataset & analysis*. Nature Scientific Data. [Details to verify: volume/page].

5. Ferariu, T. (2025). *Formal verification for smart contracts*. FMBC Proceedings. [Details to verify: volume/page].

6.Huang, M. Z., Jiang, R., Sharma, T., & Wang, K. Y. (2025). *Exploring user perceptions of security auditing in the Web3 ecosystem*. In NDSS Symposium 2025. https://doi.org/10.14722/ndss.2025.240775 NDSS Symposium+1