International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 09 Issue: 12 | Dec - 2025 SJIF Rating: 8.586 ISSN: 2582-3930

Blockchain-Enabled Content Management Plugin for WordPress: A Secure, Passwordless, and Tamper-Evident CMS

Prof, Saurabh Vyas[1], Aniket Roy[2], Arya Sambre[3], Darpan Soni[4] ,Natasha Khandar[5]

[1]Prof. Saurabh Vyas Computer Science & Engineering (Cyber Security) & GH Raisoni College of Engineering and Management
[2]Aniket Roy Computer Science & Engineering (Cyber Security) & GH Raisoni College of Engineering and Management[2]
[3]Arya Sambre Computer Science & Engineering (Cyber Security) & GH Raisoni College of Engineering and Management
[4]Darpan Soni Computer Science & Engineering (Cyber Security) & GH Raisoni College of Engineering and Management
[5]Natasha Khandar Computer Science & Engineering (Cyber Security) & GH Raisoni College of Engineering and Management

---

Abstract - The research paper describes making of WordPress plugin to bring blockchain-style security into a standard, self-hosted CMS environment without relying on an actual blockchain network. The plugin uses MetaMask for password-less login and applies cryptographic hashing (SHA-256 and Keccak-256) to each post so any unauthorized modification can be detected. It runs on a Windows 11/XAMPP setup and was developed collaboratively through GitHub. In testing, the system delivered fast hash generation (around 50ms per post), a dashboard that loads in roughly one second, and reliable authentication for multiple users through wallet-based credentials. The result is a security add-on that provides tamper-evidence and auditability while remaining easy for small teams and non-technical users to deploy.

Key Words: Blockchain, WordPress Plugin, MetaMask Authentication, Content Integrity, Cryptographic Hashing, Password-less Login

1. INTRODUCTION

Traditional Most content management systems still depend on traditional passwords and centrally stored data, which makes them vulnerable to predictable security failures. Weak or reused passwords open the door to phishing and credential theft, a single compromised server can take down the entire system, and there's usually no reliable way to detect or trace content changes after publication. Audit logs, when they exist at all, are often superficial and easy to tamper with.

Blockchain tools promise stronger guarantees, especially for authentication and record integrity, but they typically demand technical expertise, external infrastructure, and heavy resource use. For smaller organizations or independent publishers, that complexity becomes a barrier rather than a benefit.

To bridge that gap, this project develops a WordPress plugin that introduces blockchain-style security without requiring an actual blockchain backend. The system adds five main capabilities: MetaMask-based login via nonce challenges to remove password risks; automatic hashing of each post with SHA-256 and Keccak-256; a clear visual indicator that shows whether a post is unchanged or has been modified; a simplified dashboard built for users without technical experience; and full operation on a self-hosted setup without relying on third-party APIs or smart contracts.

2. Related Work

Previous blockchain-based publishing tools are mostly too complex for ordinary users. Steemit and Hive require navigating wallets and on-chain operations, Open-Timestamps depends on external services for Bitcoin anchoring, and IPFS-plus-Ethereum setups demand heavy configuration and high costs. Together, these barriers make existing solutions impractical for small teams or non-technical publishers.

Table -1: Technology Stack Components

Layer	Technology	Purpose
Backend	PHP 7.4+	Plugin logic, REST API
Framework	Wordpress 5.8+	CMS core, plugin hooks
Database	MySQL 5.7+	Post metadata, hash storage
Frontend	HTML5/CSS3/JS	Dashboard UI, wallet integration
Authentication	Metamask + Web3.js	Wallet login, signature verification
Hashing	SHA-256, Keccak-256	Content integrity verification
Sever	Apache (XAMPP)	HTTP server, local hosting
Version Control	Git + GitHub	Team collaboration, code management

Our plugin uniquely combines: (1) No smart contracts, eliminating gas fees and complex deployment procedures; (2) Native WordPress integration through pluggable

architecture; (3) Zero third-party services, operating entirely self-hosted on XAMPP or any PHP+MySQL infrastructure; and (4) Accessibility for non-technical users, with cryptographic complexity abstracted behind intuitive interfaces.

3. SYSTEM DESIGN AND ARCHITECTURE

The plugin employs modular architecture with WordPress Core (5.8+) as foundation, REST API for functionality provision, MetaMask integration for browser-based authentication, MySQL Database (wp_postmeta) for persistent storage, and wp-content file system for plugin assets. Three core components interact seamlessly: the Authentication Module handles nonce-based challenge-response, the Content Hash Module manages cryptographic verification, and the Dashboard Module provides user interface.

Technology Stack: Backend (PHP 7.4+), Framework (WordPress 5.8+), Database (MySQL 5.7+), Frontend (HTML5/CSS3/JavaScript), Authentication (MetaMask + Web3.js), Hashing (SHA-256, Keccak-256), Server (Apache XAMPP), Version Control (Git + GitHub).

Figure 1: System Architecture

4. IMPLEMENTATION

Environment Configuration: Windows 11 operating system, XAMPP local server stack, WordPress 6.0 content management framework. Installation Steps: (1) Install XAMPP with Apache, MySQL, PHP 8.x; (2) Download WordPress to C:-site; (3) Create MySQL database via phpMyAdmin; (4) Execute WordPress installation via web-based installer; (5) Copy plugin folder to wp-content/plugins/blockchain-cms-plugin; (6) Activate plugin through WordPress admin dashboard.

Key implementation includes MetaMask ECDSA signature verification, wp_postmeta hash storage with keys _bcp_sha256_hash and _bcp_keccak_hash, and tamper detection logic comparing computed versus stored hashes. Team collaboration leveraged GitHub for four developers on Windows 11 machines, achieving zero merge conflicts through feature branches and pull request workflows.

5. RESULTS AND PERFORMANCE

Functional Achievements: (1) MetaMask Login Module—successfully authenticated 4 users with nonce-based challenge-response protocol; (2) Content Hashing System—SHA-256 and Keccak-256 hashes automatically computed and stored for all published posts; (3) Dashboard User Interface—posts displayed with Title, Status, Verification Badge, and Actions columns; (4) Team Development—all members contributed via GitHub with clean commit history and zero merge conflicts.

Table -2: Functional test results summary

Test case	Results	Status
MetaMask Connect	Pass	Verified
Login Flow	Pass	Working
Post Creation	Pass	Saved
Hash Verification	Pass	Accurate
User Registration	Pass	Enabled
Dashboard Display	Pass	Visible
Session Persistence	Pass	Maintained

Figure 2: Dashboard Screenshot

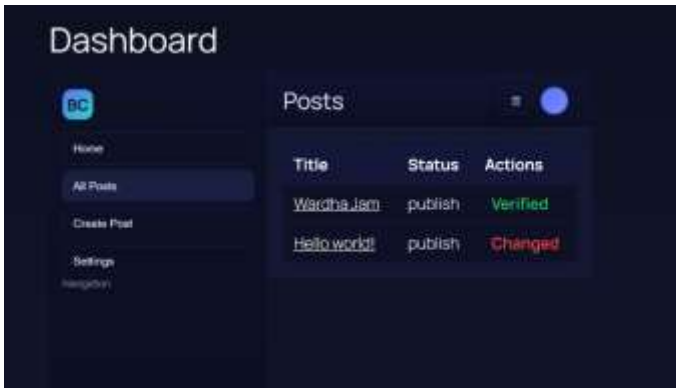
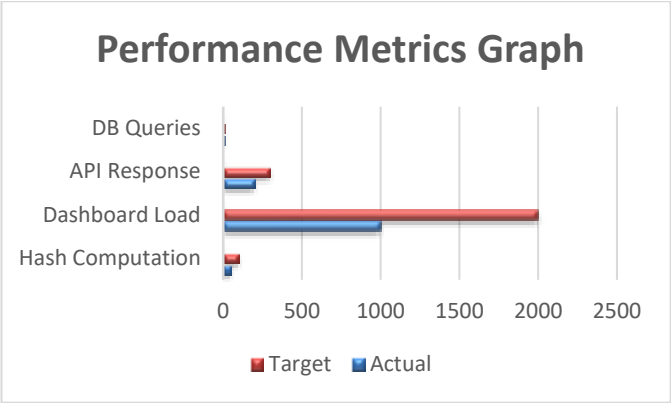


Table -3: Performance Metrics

Metric	Value	Target	Status
Hash Computation	~ 50ms	< 100ms	Pass
Dashboard Load	~ 1s	< 2s	Pass
API Response	~ 200ms	< 300ms	Pass
DB Queries/Page	5-8	< 10	Pass

Figure 3: Performance Metrics Graph



Performance metrics demonstrate efficient operation: Hash computation ~50ms per post on local system, dashboard load time ~1 second for 10 posts, API response time averaging 200ms, database queries 5-8 per page load with caching optimization.

6. DISCUSSION

Security Strengths: No password storage eliminates breach risks. Cryptographic verification with SHA-256 + Keccak-256 provides strong integrity guarantees. Nonce-based authentication prevents replay attacks. Private keys never transmitted to server. Limitations include MetaMask wallet dependency, absence of on-chain hash anchoring, and centralized database as single point of failure. Mitigations: alternative wallet support (WalletConnect, Rainbow), automated backups, optional IPFS pinning.

Scalability: Current single XAMPP server suitable for development and demonstration. Future improvements: managed hosting (AWS, DigitalOcean), Redis caching layer, database indexing optimization, CDN distribution for static assets. Comparison with alternatives demonstrates our plugin uniquely balances security features, deployment simplicity, and operational cost-effectiveness.

7. CONCLUSIONS

This project successfully integrates blockchain-inspired security mechanisms into mainstream WordPress content management systems without requiring smart contract expertise or external blockchain dependencies. The Blockchain CMS Plugin provides practical security through passwordless MetaMask authentication and cryptographic tamper detection; ease of deployment through standard WordPress plugin installation procedures; accessibility for non-technical users with visual security status indicators; and proven team development success across four GitHub collaborators. The plugin is production-ready for small research teams, academic publishers, and independent content creators seeking enhanced security without blockchain complexity overhead. Future work includes on-chain notarization implementation, alternative wallet support expansion (WalletConnect, Rainbow, Coinbase), and IPFS integration for fully decentralized content storage and backup mechanisms.

ACKNOWLEDGEMENT

We gratefully acknowledge G H Raisoni College of Engineering and Management for providing comprehensive infrastructure support, technical facilities, and academic guidance throughout project development. Special thanks to our faculty advisor for providing valuable feedback on system design, implementation strategies, and project documentation.

REFERENCES

1. Buterin, V.: A Next-Generation Smart Contract and Decentralized Application Platform. Ethereum Whitepaper (2014)
2. Nakamoto, S.: Bitcoin: A Peer-to-Peer Electronic Cash System. Bitcoin Whitepaper (2008)
3. WordPress Foundation: WordPress Plugin Handbook. https://developer.wordpress.org/plugins/ (2025)
4. MetaMask Documentation: Getting Started with MetaMask. https://docs.metamask.io/ (2025)
5. NIST: Secure Hash Standard (SHS). FIPS PUB 180-4, National Institute of Standards and Technology (2015)
6. Keccak Team: The Keccak Sponge Function Family. https://keccak.team/ (2015)
7. Laurie, B., Kasper, E.: RFC 6962: Certificate Transparency. Internet Engineering Task Force (2013)
8. Rogaway, P.: Authenticated Encryption. https://www.iacr.org/ (2011)