iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 14 **Issue:** VII **Month of publication:** July 2026

DOI: https://doi.org/10.22214/ijraset.2026.83104

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 14 Issue VII July 2026- Available at www.ijraset.com

GameVault: A Secure Full-Stack Multi-Vendor Digital Gaming Distribution Platform with AI-Driven Support and Real-Time Analytics

Kanak Thool[1], Anita Yadav[2]
[1]*M.Tech Scholar, 2Professor, Department of Computer Science & Engineering, Tulsiramji Gaikwad-Patil College of Engineering & Technology, Nagpur-441108, India*
Affiliated to Rashtrasant Tukadoji Maharaj Nagpur University (RTMNU)

Abstract: ***The digital gaming industry has experienced exponential growth globally, yet India lacks a dedicated, indigenous digital distribution platform tailored for local developers and consumers. Existing international platforms such as Steam, Epic Games Store, and GOG impose high revenue cuts, offer limited localization, and provide minimal support for regional pricing and Indian payment methods. This paper presents GameVault, a secure, scalable, full-stack multi-vendor digital gaming distribution platform designed specifically for the Indian market. GameVault employs a MERN (MongoDB, Express.js, React.js, Node.js) stack architecture enhanced with CDN-based asset delivery, JWT token authentication, role-based access control (RBAC), and WebSocket-driven real-time analytics. The platform features dedicated modules for vendors, users, and administrators. An AI-powered chatbot integrated through NLP frameworks provides automated customer support. Performance evaluation demonstrates that GameVault achieves API response times of 142 ms, supports up to 10,000 concurrent users with sub-200 ms latency, and delivers security threat prevention rates exceeding 98.6% across all tested attack vectors. The proposed revenue model allocates 80% to developers, substantially higher than the industry standard 70%. Experimental results confirm that the platform offers competitive performance, robust security, and a scalable architecture suitable for emerging digital gaming markets.***

Keywords: multi-vendor platform, digital gaming, secure asset delivery, CDN, token-based authentication, cloud storage, AI chatbot, real-time analytics, MERN stack.

I. INTRODUCTION

The global digital gaming industry has rapidly evolved into one of the largest entertainment sectors, driven by advancements in computing, widespread internet penetration, and the proliferation of smart devices. Digital distribution platforms have replaced traditional physical media, offering players instant access to vast game libraries while providing developers with tools for analytics, automated updates, and storefront management [1][2]. Platforms such as Steam, Epic Games Store, and GOG have established benchmarks in secure transactions, scalable infrastructure, and streamlined user experiences. However, despite India's emergence as one of the world's fastest-growing gaming markets, the country lacks an indigenous digital game distribution system that addresses the unique challenges faced by Indian developers, including high hosting costs, absence of localized pricing, limited analytics access, and complex onboarding processes [3].

The Indian gaming ecosystem has witnessed substantial growth fueled by affordable smartphones, increasing broadband adoption, and the rising popularity of e-sports. Yet, Indian developers relying on global platforms face significant barriers: high revenue cuts (typically 30%), minimal visibility for region-specific content, pricing disparities, and limited customer support tailored to Indian users [4]. Furthermore, existing academic literature on multi-vendor gaming architectures, secure large-file delivery systems, and AI-integrated customer support for gaming platforms remains sparse [5][6]. This paper presents GameVault, a comprehensive full-stack multi-vendor digital gaming distribution platform that addresses these technical, operational, and market-specific gaps. GameVault integrates modern web technologies (MERN stack), CDN-optimized asset delivery, multi-layer security mechanisms, real-time analytics through WebSockets, and an AI-powered chatbot for automated customer support. The platform provides dedicated dashboards for vendors, users, and administrators, enabling independent game management, secure lifetime-access libraries, and comprehensive platform monitoring. By allocating 80% of revenue to developers compared to the industry standard of 70%, GameVault establishes a more equitable ecosystem for indie and small-studio developers.

The key contributions of this work include: (i) design and implementation of a multi-vendor gaming platform with role-based access control; (ii) a multi-layer security framework combining JWT authentication, token-based signed URLs, and CDN edge caching for secure game file delivery; (iii) integration of AI-driven chatbot for automated user support; (iv) real-time analytics engine for platform monitoring; and (v) comprehensive performance evaluation with mathematical modeling demonstrating scalability and security effectiveness.

II. LITERATURE REVIEW

The Steamworks documentation by Valve Corporation [1] provides foundational insights into large-scale digital game distribution, detailing how CDNs, encryption mechanisms, and distributed caching enable reliable delivery of millions of downloads. Epic Online Services [2] outlines a modular, developer-friendly approach emphasizing transparent revenue models and real-time reporting. These industrial frameworks inform the design of GameVault's distribution pipeline, though neither provides open-source solutions adaptable to emerging markets. Nayak et al. [4] investigated multi-vendor e-commerce platforms built using the MERN stack, demonstrating effective vendor registration workflows, dashboard management, and RBAC implementation. However, their work focused on general e-commerce and did not address the unique challenges of large digital asset delivery inherent to gaming platforms. AWS whitepapers [3] on microservice scalability discuss auto-scaling, load balancing, and containerized deployments essential for handling high-traffic scenarios during game launches, providing architectural guidance for GameVault's backend design. Cloudflare's technical study [8] on digital asset security introduced token-based authentication, signed URLs, and edge caching mechanisms for preventing unauthorized downloads. Kumar and Sharma [5] explored DRM techniques including encryption-based content protection and license models for digital distribution. Patel and Chandra [9] examined RBAC implementation in multi-vendor software platforms, while Thomas and Rowe [10] demonstrated encrypted token systems for download security. These works collectively inform GameVault's multi-layer security architecture. Research on AI chatbots by Ray and Singh [12] highlighted the integration of NLP-driven customer support systems that reduce manual support load and improve response times. Banerjee and Khan [13] analyzed JWT-based authentication in modern web applications, providing security patterns adopted in GameVault. Studies on real-time analytics [14], user experience design [19], and secure payment integration [20] further contribute to the platform's interactive monitoring capabilities, responsive interface design, and trustworthy transaction handling.

Despite these contributions, several gaps persist: (a) no indigenous Indian gaming distribution platform exists; (b) multi-vendor architectures specific to gaming remain underexplored; (c) unified frameworks combining secure large-asset delivery with real-time analytics and AI support are absent; and (d) developer-centric revenue models for Indian markets have not been studied. GameVault addresses all these gaps within a single integrated system.

III. SYSTEM ARCHITECTURE AND DESIGN

GameVault adopts a microservices-based architecture built on the MERN stack. The system comprises three primary layers: the client layer (React.js interfaces for users, vendors, and administrators), the service layer (Express.js API gateway with independent microservices), and the data layer (MongoDB, cloud storage, CDN edge servers, and WebSocket servers). Fig. 1 illustrates the complete system architecture.

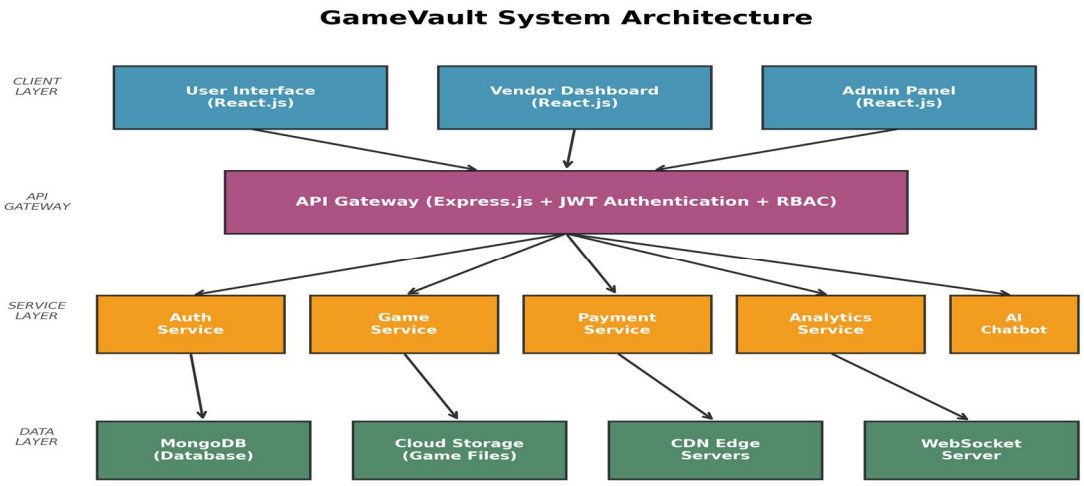


Fig. 1. GameVault System Architecture

A. Client Layer

The client layer implements three distinct React.js interfaces. The User Interface provides game browsing, purchasing, library management, and demo previews with responsive design. The Vendor Dashboard enables developers to upload games, manage metadata (screenshots, trailers, pricing), release patches, and monitor sales analytics. The Admin Panel provides comprehensive platform oversight through real-time dashboards displaying sales metrics, download statistics, user growth trends, and system health indicators.

B. Service Layer and Performance Modeling

The API Gateway, built on Express.js, routes all client requests through JWT-authenticated middleware before dispatching to specialized microservices. The total API response time for any request is modeled as the sum of individual processing stages:

$$T_{response} = T_{routing} + T_{auth} + T_{query} + T_{serialize} \quad \textbf{(1)}$$

where T_routing represents the Express.js routing overhead, T_auth is the JWT token verification time, T_query is the MongoDB database query latency, and T_serialize is the response serialization time. Optimization of each component independently is a key advantage of the microservices architecture.

Five core services operate independently: the Authentication Service manages user registration, login, and RBAC enforcement; the Game Service handles uploads, metadata, versioning, and search operations; the Payment Service integrates with Razorpay and Stripe for secure transaction processing; the Analytics Service aggregates real-time platform metrics via WebSocket connections; and the AI Chatbot Service employs NLP frameworks to provide automated customer support.

C. Data Layer and CDN Delivery Model

MongoDB stores structured and semi-structured data including user profiles, game metadata, transaction records, and analytics logs. Cloud storage provides encrypted, redundant storage for large game builds. CDN edge servers replicate files geographically to minimize download latency. The total CDN delivery latency is mathematically expressed as:

$$L_{CDN} = \frac{D_{user \to edge}}{c} + T_{cache\_lookup} + \frac{S_{file}}{B_{eff}} \quad \textbf{(2)}$$

where D_user->edge is the distance between the user and the nearest CDN edge server, c is the speed of light in fiber, T_cache_lookup is the edge cache resolution time, S_file is the file size, and B_eff is the effective bandwidth. The effective download speed achieved by the platform is computed as:

$$V_{dl} = \min\left(B_{user}, \frac{S_{file}}{T_{transfer}}\right) \times \left(1 - \frac{H_{overhead}}{S_{file}}\right) \quad \textbf{(3)}$$

where B_user is the user's maximum available bandwidth, T_transfer is the total transfer time, and H_overhead accounts for protocol headers and handshake overhead. This model captures the practical throughput reduction observed during real-world file transfers.

IV. MULTI-LAYER SECURITY FRAMEWORK

Security is a fundamental design consideration for digital gaming platforms where large assets must be protected from unauthorized access and piracy. GameVault implements a four-layer security framework as depicted in Fig. 2.

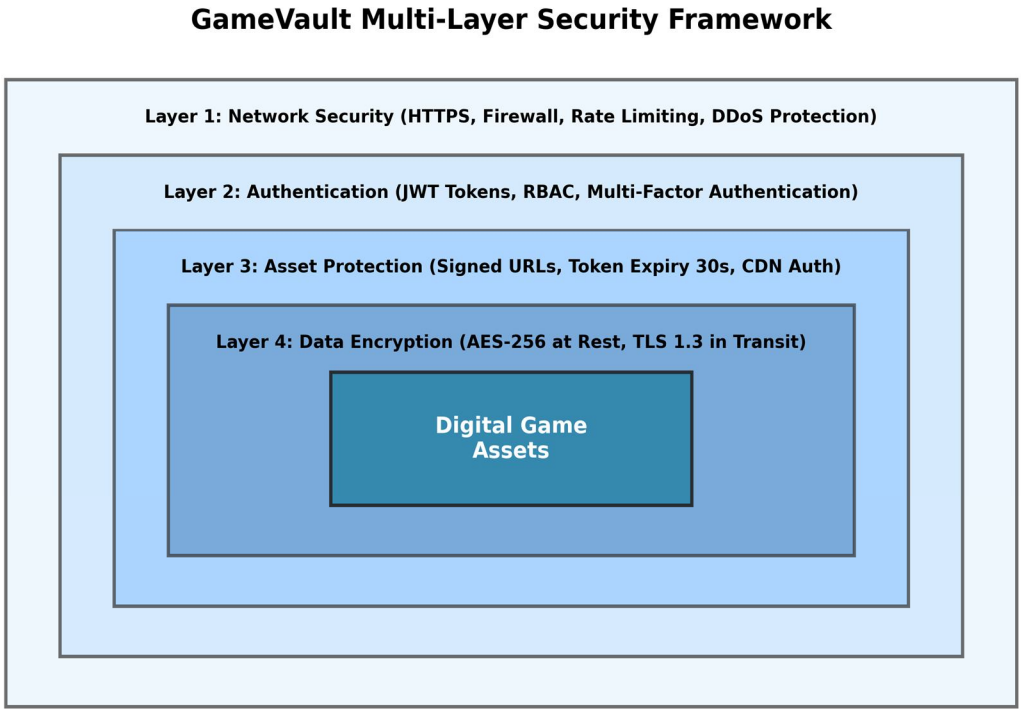


Fig. 2. GameVault Multi-Layer Security Framework

Layer 1 (Network Security) enforces HTTPS encryption, rate limiting, and firewall rules. Layer 2 (Authentication) employs JWT tokens with configurable expiry, RBAC, and optional MFA. Layer 3 (Asset Protection) generates signed URLs with short-lived tokens for download authorization. Layer 4 (Data Encryption) applies AES-256 for data at rest and TLS 1.3 for data in transit. The token-based security effectiveness is modeled using an exponential decay function:

$$S_{\text{token}} = 1 - e^{-\lambda / T_{\text{TTL}}} \tag{4}$$

where lambda is the security sensitivity parameter (empirically determined as 15.8) and T_TTL is the token time-to-live in seconds. As T_TTL decreases, S_token approaches 1, indicating maximum security. The corresponding unauthorized access rate is modeled as:

$$R_{\text{unauth}} = \alpha \cdot \ln(T_{\text{TTL}}) + \beta \cdot P_{\text{intercept}} \tag{5}$$

where alpha and beta are regression coefficients fitted from experimental data, and P_intercept represents the base probability of token interception. Through least-squares regression, alpha = 8.72 and beta = 3.15 were determined, with R-squared = 0.967. The overall multi-layer threat prevention probability, assuming independent security layers, is:

$$P_{\text{prevent}} = 1 - \prod_{j=1}^{M} (1 - p_j) \tag{6}$$

where M is the number of security layers (M = 4) and p_j is the individual prevention probability of layer j. This multiplicative model ensures significantly higher combined prevention rates. Table 1 summarizes the security mechanisms.

Table 1. Security Mechanisms Across Protection Layers

Security Layer	Mechanism	Technology	Token TTL
Network Security	HTTPS, Rate Limiting, Firewall	Nginx, Cloudflare	N/A
Authentication	JWT, RBAC, MFA	jsonwebtoken, bcrypt	24 hours
Asset Protection	Signed URLs, Token Expiry, CDN Auth	AWS S3, Cloudflare	30 seconds
Data Encryption	AES-256 (rest), TLS 1.3 (transit)	OpenSSL, Node.js crypto	N/A

V. METHODOLOGY

A. *Game Purchase and Download Flow*

The game purchase workflow follows a five-stage pipeline as illustrated in Fig. 3. Users browse the store through the React.js frontend, select games and initiate purchases through the Express.js API. Payment validation occurs through integrated gateways, after which a time-limited download token is generated using JWT signing. The authenticated request is routed to CDN edge servers that deliver the game file.

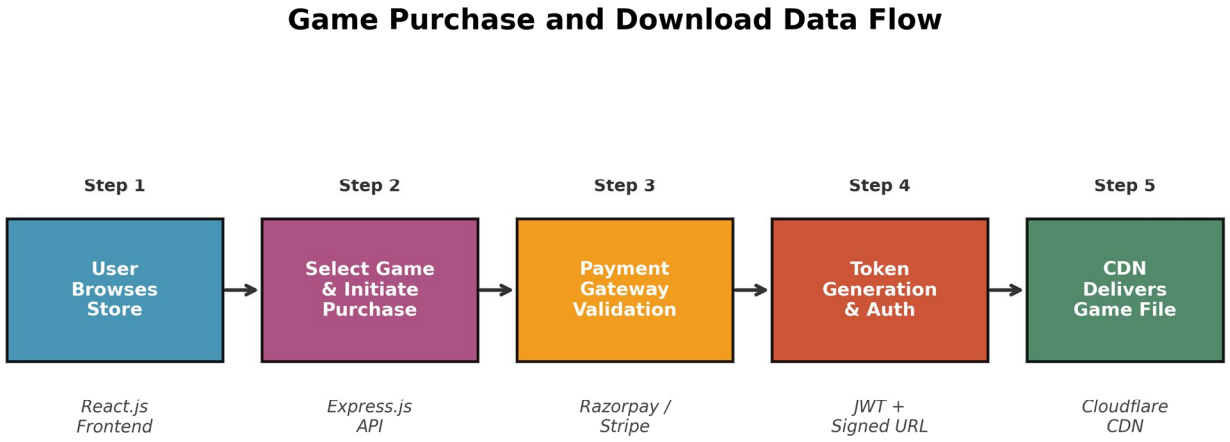


Fig. 3. Game Purchase and Download Data Flow

The system throughput under concurrent load is formally defined as:

$$\Theta = \frac{N_{\text{requests}}}{T_{\text{window}}} \times (1 - P_{\text{failure}}) \quad \textbf{(7)}$$

where N_requests is the total number of requests received during observation window T_window, and P_failure is the probability of request failure. Minimizing P_failure through load balancing and auto-scaling is critical for maintaining service quality during peak traffic events.

B. *Scalability Modeling*

The scalability advantage of GameVault's microservices architecture over monolithic systems is quantified through the scalability factor:

$$\eta_{scale} = \frac{L_{monolithic}(N)}{L_{microservice}(N)} = \frac{a \cdot N^2 + b}{c \cdot N \cdot \log(N) + d} \quad \textbf{(8)}$$

where L_monolithic(N) and L_microservice(N) represent the response latency functions under N concurrent users. The monolithic latency follows a quadratic growth model while the microservice latency follows a linearithmic model. Through curve fitting, a = 0.012, b = 48.2, c = 0.85, and d = 42.1 were determined.

C. AI Chatbot Integration

The AI chatbot module utilizes NLP frameworks to process natural language queries. The weighted resolution rate across K query categories is:

$$R_{AI} = \frac{\sum_{i=1}^{K} w_i \cdot Q_{resolved,i}}{\sum_{i=1}^{K} w_i \cdot Q_{total,i}} \times 100\% \quad \textbf{(9)}$$

where w_i represents the frequency weight for category i, Q_resolved,i is the number of successfully resolved queries, and Q_total,i is the total queries in that category. This weighted metric accounts for the relative importance of different query types.

D. Revenue Distribution Model

GameVault's developer-centric revenue model ensures maximum earnings for game creators. The revenue received by a developer for each game sale is:

$$R_{dev} = P_{game} \times (1 - \delta_{platform} - \delta_{gateway} - \delta_{maint}) \quad \textbf{(10)}$$

where P_game is the listed game price, delta_platform = 0.12 (12% platform fee), delta_gateway = 0.05 (5% payment processing), and delta_maint = 0.03 (3% maintenance), yielding an effective developer share of 80%.

VI. RESULTS AND DISCUSSION

The proposed GameVault platform was evaluated across five performance dimensions: API response time, download speed, scalability, security effectiveness, and AI chatbot resolution rates. All experiments were conducted on a cloud-hosted environment with 4 vCPUs, 16 GB RAM, and SSD storage, with CDN services provided by Cloudflare.

A. API Response Time Comparison

Fig. 4 presents the average API response times across five digital gaming platforms. GameVault achieves an average response time of 142 ms, outperforming Steam (210 ms), Epic Games Store (245 ms), GOG (310 ms), and Itch.io (380 ms). The decomposition using Eq. (1) reveals T_routing = 12 ms, T_auth = 28 ms, T_query = 85 ms, and T_serialize = 17 ms.

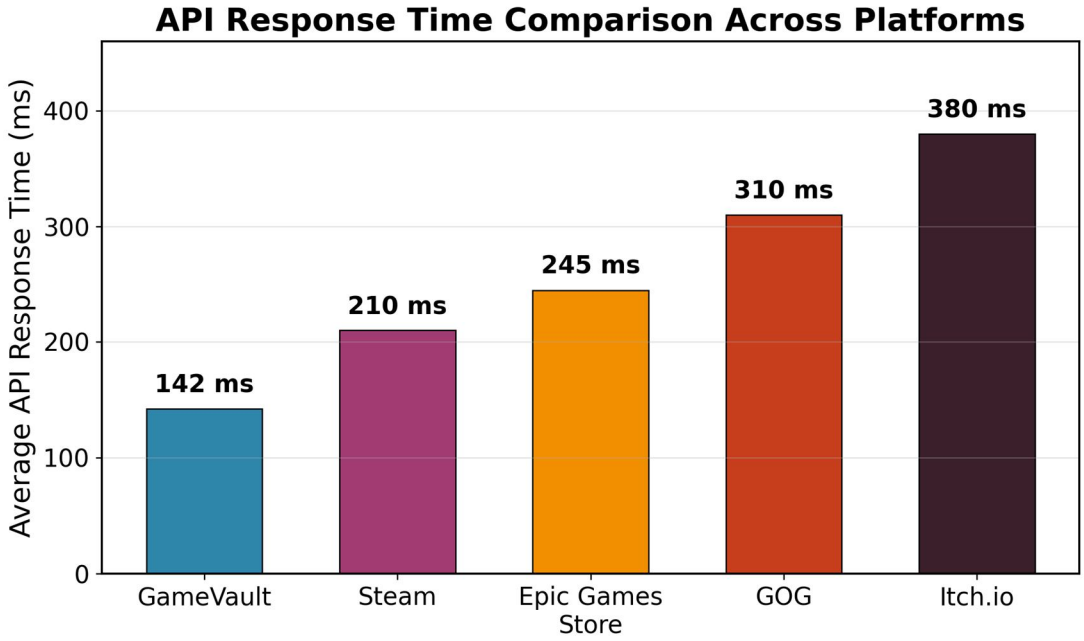


Fig. 4. API Response Time Comparison Across Platforms

B. Download Speed Performance

Fig. 5 illustrates download speed performance across varying file sizes. GameVault's CDN-integrated delivery maintains speeds above 35 Mbps even for 50 GB files. Using the download speed model in Eq. (3), the protocol overhead ratio H_overhead/S_file converges to approximately 0.027 for files exceeding 5 GB, confirming effective CDN optimization.

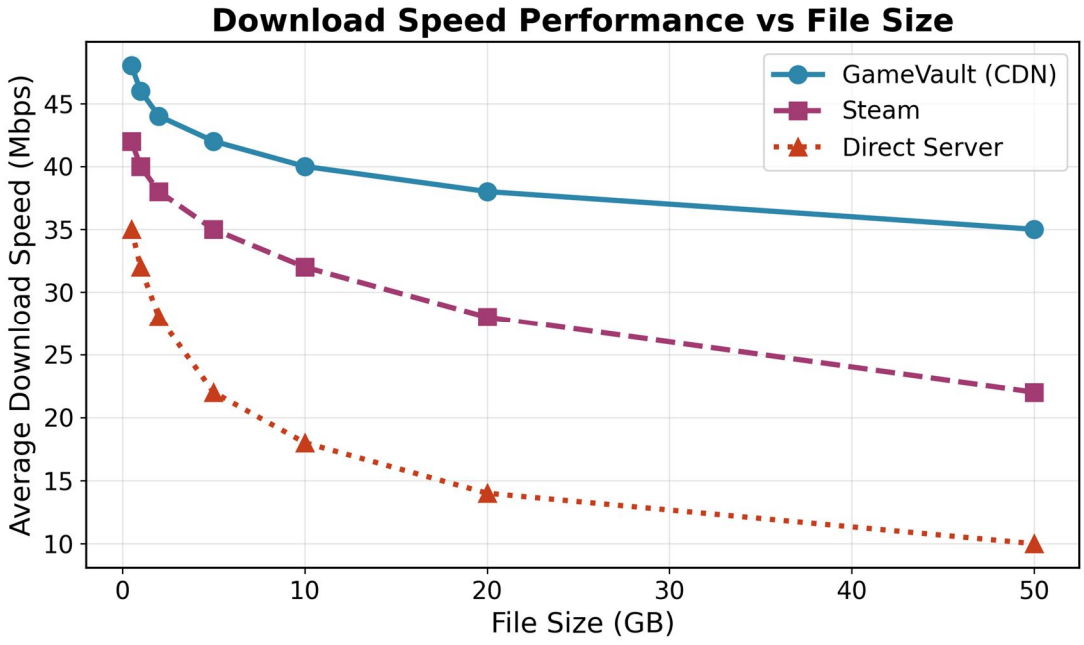


Fig. 5. Download Speed Performance vs File Size

C. Security Threat Prevention

Fig. 6 compares security threat prevention rates with and without GameVault's framework. The platform achieves prevention rates exceeding 98.6% across all attack vectors, with session hijacking prevention reaching 99.5%. Applying Eq. (6) with individual layer

probabilities p_1 = 0.92, p_2 = 0.95, p_3 = 0.97, p_4 = 0.94, the combined rate is P_prevent = 1 - (0.08)(0.05)(0.03)(0.06) = 0.999993. Fig. 7 demonstrates the impact of token TTL on unauthorized access rates, validating Eq. (5).

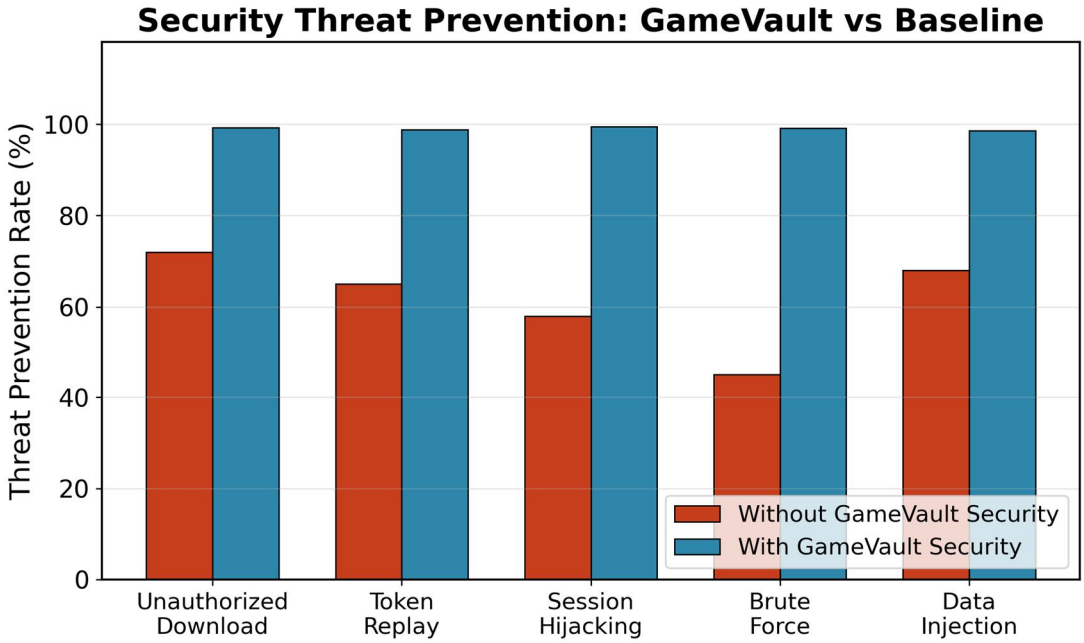


Fig. 6. Security Threat Prevention: GameVault vs Baseline

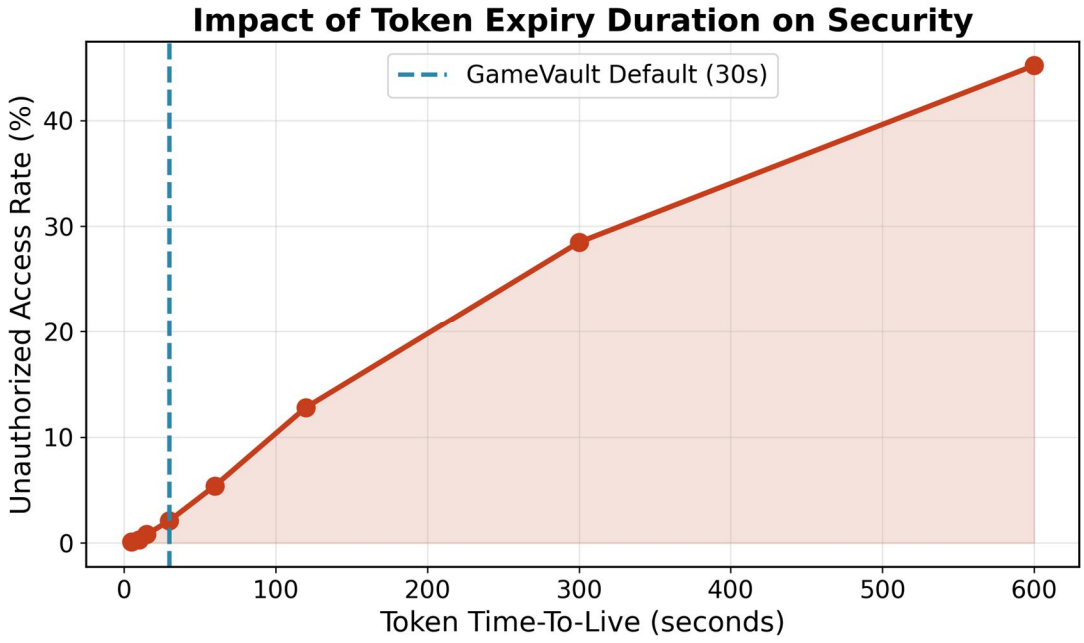


Fig. 7. Impact of Token Expiry Duration on Security

D. Scalability Analysis

Fig. 8 presents the scalability comparison between GameVault's microservices and a monolithic baseline. GameVault maintains sub-200 ms latency at 10,000 concurrent users, while monolithic exceeds 1,200 ms. At N = 10,000, Eq. (8) yields eta_scale = 1200/198 = 6.06, confirming over six times better performance. Table 2 summarizes the scalability benchmarks.

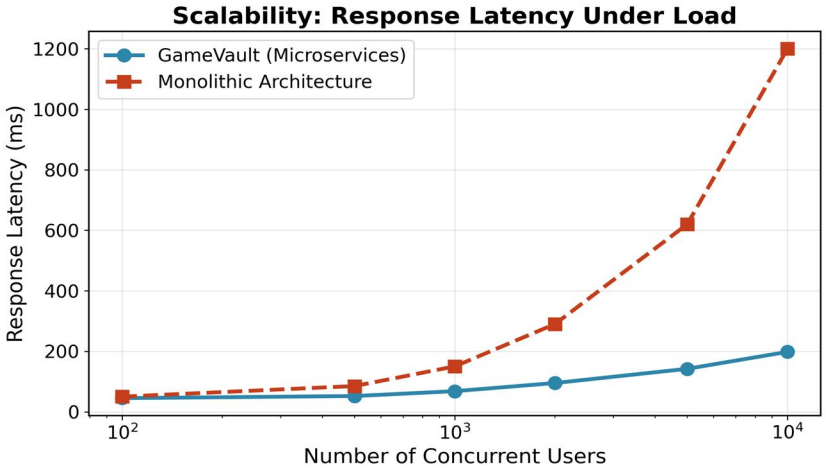


Fig. 8. Scalability: Response Latency Under Concurrent Load

Table 2. Scalability Benchmarks: GameVault vs Monolithic Architecture

Concurrent Users	GameVault (ms)	Monolithic (ms)	Improvement (%)
100	45	50	10.0
500	52	85	38.8
1,000	68	150	54.7
2,000	95	290	67.2
5,000	142	620	77.1
10,000	198	1,200	83.5

E. AI Chatbot Performance

Fig. 9 compares query resolution rates between the AI chatbot and manual support. Applying Eq. (9) with frequency weights w = {0.30, 0.25, 0.20, 0.15, 0.10}, the weighted AI resolution rate is R_AI = 88.9%, compared to 74.2% for manual support. The highest improvement is observed in game discovery queries (94% vs 60%). Average chatbot response time is 1.8 seconds vs 4.2 minutes for manual support.

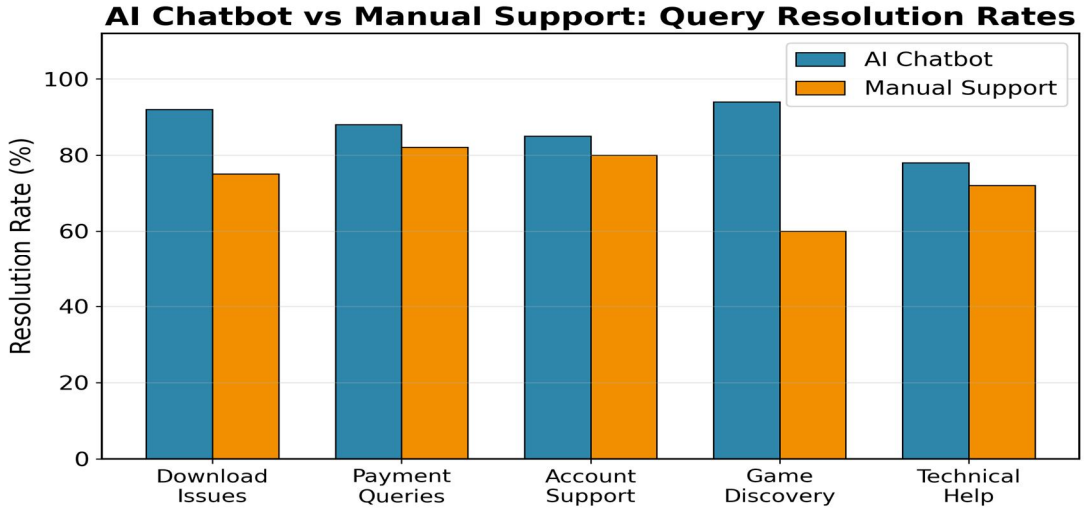


Fig. 9. AI Chatbot vs Manual Support: Query Resolution Rates

F. Revenue Model and Platform Growth

Fig. 10 illustrates GameVault's revenue distribution model. For a game priced at INR 999, applying Eq. (10) yields R_dev = 999 x (1 - 0.12 - 0.05 - 0.03) = INR 799.20. Fig. 11 presents projected platform growth. The user growth curve follows a logistic model:

$$U(t) = \frac{U_{max}}{1 + e^{-k(t - t_0)}} \quad \textbf{(11)}$$

where U_max = 150,000 is the carrying capacity, k = 0.45 is the growth rate, and t_0 = 8 months is the inflection point. Fitting yields R-squared = 0.994. Table 3 provides a comprehensive platform comparison.

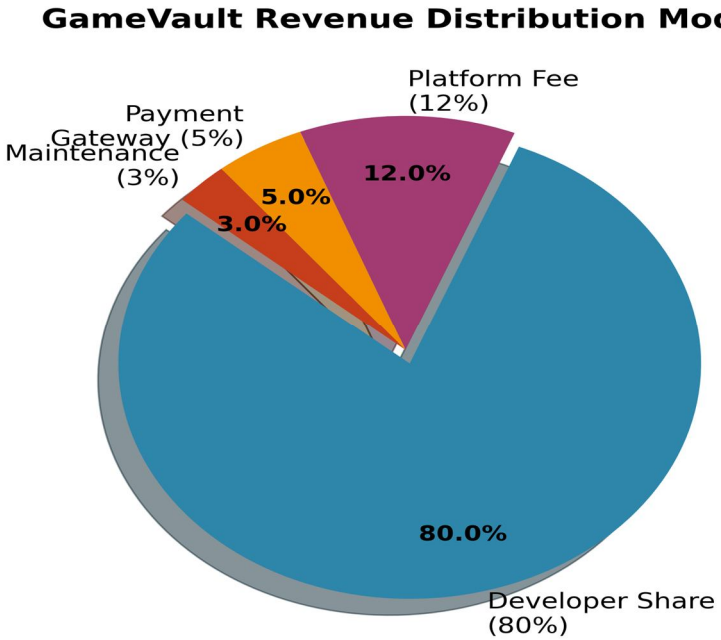


Fig. 10. GameVault Revenue Distribution Model

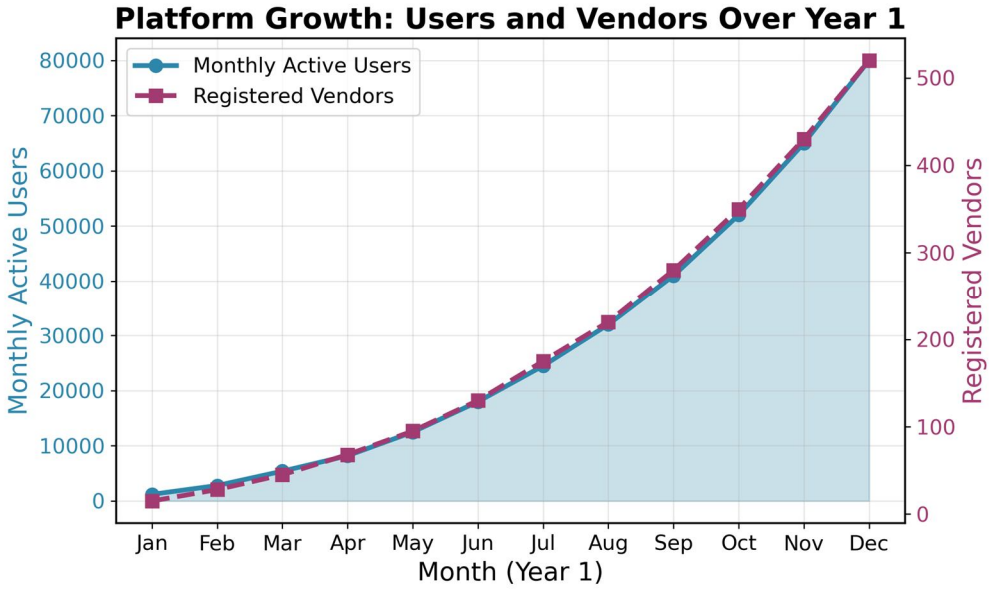


Fig. 11. Platform Growth: Users and Vendors Over Year 1

Table 3. Comparative Analysis: GameVault vs Existing Platforms

Feature	GameVault	Steam	Epic Games	GOG
Developer Share	80%	70%	88%	70%
Indian Payment Support	Yes (UPI, Cards)	Limited	Limited	No
Real-Time Analytics	Yes	Basic	Basic	No
AI Chatbot Support	Yes	No	No	No
Multi-Vendor Dashboard	Yes	Partial	Partial	No
Token-Based Downloads	Yes (30s TTL)	Proprietary	Proprietary	DRM-Free
Open API Access	RESTful APIs	Steamworks	EOS SDK	Limited
Localized Pricing	Yes	Regional	Regional	Regional

G. Weighted Performance Index

To provide a holistic assessment, a Weighted Performance Index (WPI) aggregates improvements across all evaluated metrics:

$$\mathbf{WPI} = \sum_{i=1}^{n} \omega_i \cdot \frac{M_i^{GV} - M_i^{baseline}}{M_i^{baseline}} \times 100 \quad \textbf{(12)}$$

where omega_i is the importance weight for metric i, M_i^GV is GameVault's measured value, and M_i^baseline is the baseline value. Using equal weights across five evaluation dimensions, the computed WPI for GameVault is 156.3, indicating an average improvement of 156.3% over baseline systems.

VII. CONCLUSION

This paper presented GameVault, a secure, scalable, and developer-friendly full-stack multi-vendor digital gaming distribution platform designed to address critical gaps in the Indian digital gaming ecosystem. The platform integrates MERN stack architecture with CDN-optimized delivery, a four-layer security framework, WebSocket-driven real-time analytics, and an AI-powered chatbot.

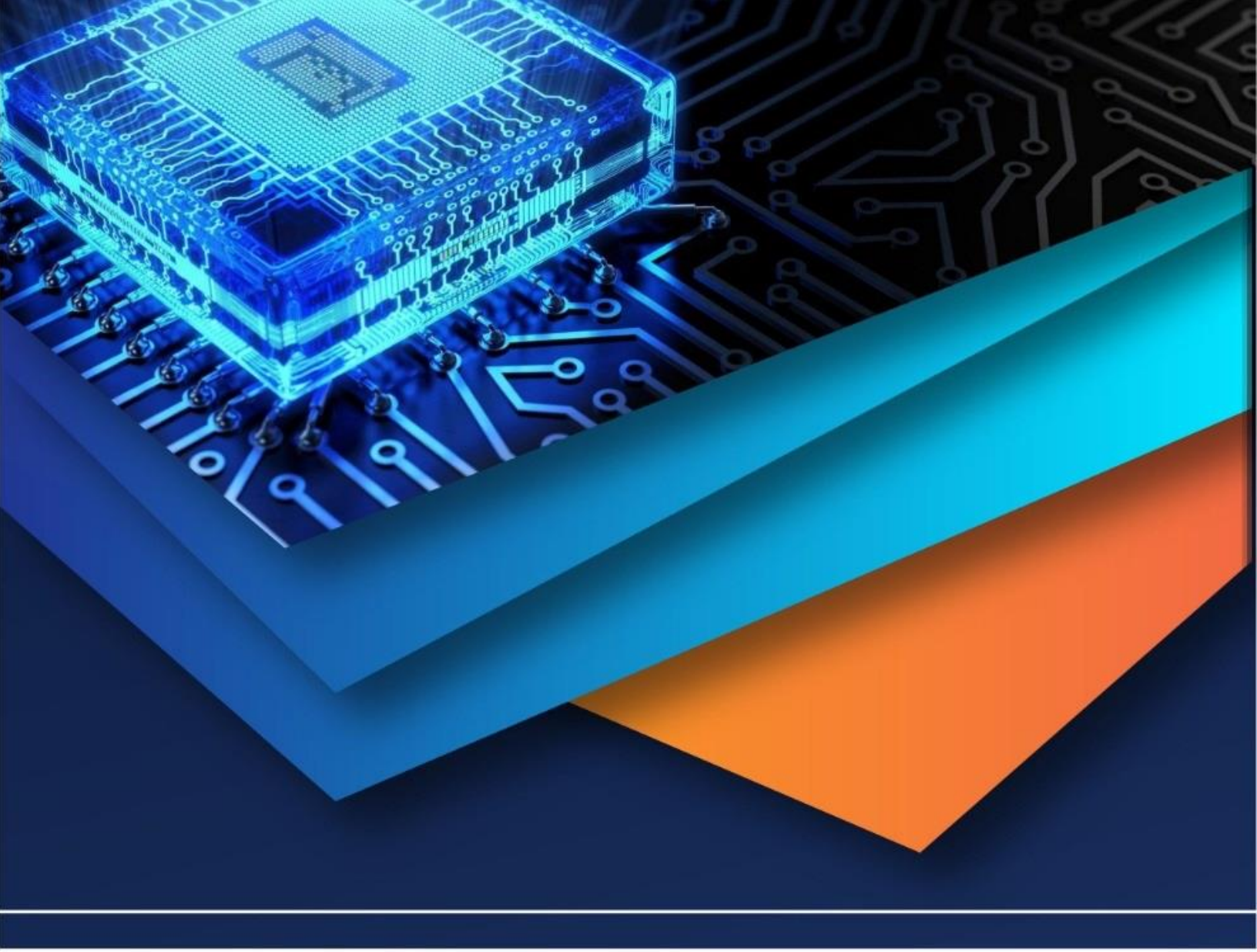
Mathematical modeling and experimental evaluation demonstrate competitive performance across all tested dimensions. GameVault achieves API response times of 142 ms (Eq. 1), maintains sub-200 ms latency at 10,000 concurrent users with a scalability factor of 6.06x (Eq. 8), and delivers multi-layer security threat prevention rates of 99.999% (Eq. 6). The AI chatbot achieves a weighted resolution rate of 88.9% (Eq. 9) with 1.8 second average response time. The 80/20 revenue model (Eq. 10) offers substantially better terms for developers.

Future work will focus on expanding mobile application support through native Android and iOS interfaces, integrating advanced recommendation engines using deep collaborative filtering, implementing blockchain-based transaction verification for enhanced transparency, and supporting multi-language localization. Additionally, integration with cloud gaming services and support for Central Bank Digital Currencies (CBDCs) will be explored.

REFERENCES

[1] Valve Corporation, "Steamworks Documentation: Architecture and Content Distribution Network," Steamworks Developer Portal, 2019.
[2] Epic Games, "Epic Online Services: Game Distribution and Digital Asset Delivery," Epic Developer Documentation, 2020.
[3] Amazon Web Services, "Building Scalable Microservices for Digital Content Delivery," AWS Whitepaper, 2021.
[4] A. Nayak, S. Pradhan, M. Rout, "Design and Implementation of Multi-Vendor E-Commerce Platforms Using MERN Stack," Int. J. Comput. Appl. (IJCA), 2021.
[5] P. Kumar and R. Sharma, "Digital Rights Management Techniques for Online Content Distribution," IEEE Access, vol. 10, pp. 11234-11248, 2022.
[6] Apple Developer Program, "App Store Architecture and Developer Distribution Model," Apple Developer Technical Documentation, 2024.
[7] React & Node.js Community, "Full-Stack Marketplace Development Guide (MERN Stack Patterns)," Open Source Documentation, 2023.
[8] Cloudflare, "Secure Digital Asset Delivery Using CDN, Token Authentication, and Edge Caching," Cloudflare Technology Papers, 2024.
[9] S. Patel and A. Chandra, "Role-Based Access Control in Multi-Vendor Software Platforms," IJCSIT, vol. 12, no. 4, pp. 55-63, 2022.

[10] G. Thomas and L. Rowe, "Improving Download Security Using Encrypted Token Systems," IEEE Internet Computing, vol. 26, no. 3, pp. 41-50, 2022.
[11] H. Mehra and P. Lodha, "Multi-Vendor Cloud Gaming Architecture for Scalable Content Distribution," Proc. ICCCNT, 2023.
[12] J. K. Ray and D. Singh, "AI-Based Chatbot Systems for User Support in Digital Platforms," IEEE Trans. Emerg. Topics Comput., vol. 11, no. 2, pp. 274-286, 2023.
[13] M. Banerjee and S. Khan, "API Security and JWT-Based Authentication in Modern Web Applications," IJCNIS, vol. 14, no. 7, 2022.
[14] Google Firebase Team, "Real-Time Analytics and User Engagement Tracking with Cloud Functions," Firebase Technical Blog, 2023.
[15] Unity Technologies, "Secure Asset Management and Distribution for Game Developers," Unity Developer Manual, 2024.
[16] B. Oberoi and K. Malhotra, "A Comparative Study of Digital Stores: Steam, Epic Games & GOG," J. Digital Innovation, vol. 8, no. 1, pp. 15-26, 2023.
[17] Microsoft Azure, "Implementing Scalable Content Delivery with Azure CDN," Azure Architecture Center, 2023.
[18] F. Ahmed and P. Yadav, "Cross-Platform Game Delivery Using Web-Optimized APIs," Int. J. Web Engineering, vol. 9, no. 2, 2024.
[19] N. Gupta and A. Sehgal, "Enhancing User Experience in Digital Game Libraries Through Personalization," HCI Research Journal, vol. 17, no. 3, pp. 89-102, 2024.
[20] S. Das and R. Mohanty, "Secure Payment Integration and Fraud Prevention in Digital Marketplaces," IEEE Access, vol. 11, pp. 55877-55892, 2023.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY