UDC 004.8:004.738.5 DOI https://doi.org/10.35546/kntu2078-4481.2025.3.2.53

O. I. SYPIAHIN
Full Stack Engineer
floLive
ORCID: 0009-0008-7565-3221

M. A. YUZHAKOV
Senior Software Engineer
Plaid
ORCID: 0009-0008-1307-4275

R. I. IBRAHIMOV
Senior Software Developer
VoiceLove
ORCID: 0009-0002-7612-3341

ARCHITECTURAL SOLUTIONS FOR SCALABLE WEB APPLICATIONS CONSIDERING PERFORMANCE AND LATENCY

The rapid growth of digital services and the increasing demands of globally distributed users place scalability, performance, and latency optimization at the center of web application design. Ensuring high performance under growing workloads is critical for business process stability, user satisfaction, and system efficiency.

The goal of the study is to compare four architectural paradigms of web applications – monolithic, microservices, serverless, and edge computing – under workloads simulating realistic e-commerce scenarios, including concurrent transactions, product searches, and payment requests, to assess their behavior in terms of scalability, performance, and latency.

As a result, the study revealed clear differences in system behavior across the four paradigms. Monolithic systems, while predictable and simple to operate, quickly became saturated under load, with response times exceeding 900 ms at 1,500 concurrent users and reaching over 3 seconds at peak capacity. Microservices extended the range of scalability, maintaining stable performance up to 3,000–3,500 users with average delays of 420–600 ms; however, at higher concurrency, orchestration overhead caused sharp increases in latency. Serverless platforms demonstrated the highest elasticity, scaling almost instantly to 6,200 requests per second and effectively handling burst traffic, though latency spikes up to 1,000 ms occurred due to cold starts and execution variability. Edge deployments achieved the lowest latency overall, remaining below 300 ms even at 5,000 users, and providing peak throughput of 5,400 requests per second, albeit requiring significant provisioning and introducing greater operational complexity. The findings indicate that no single architectural paradigm fully satisfies both scalability and latency requirements across diverse workload profiles. Hybrid strategies that combine microservices for core stability, serverless for unpredictable bursts, and edge computing for latency-sensitive operations were found to deliver the most balanced outcomes, ensuring responsiveness, stability, and cost-aware scalability.

Conclusions: flexible architectural approaches enable high performance, stability, and scalability of web applications while optimizing latency and operational costs. The findings emphasize the importance of architectural adaptability for building resilient and efficient web applications in modern digital ecosystems.

Key words: *scalable web applications, microservices, serverless, edge computing, Kubernetes, latency, performance optimization, CI/CD, cloud architecture.*

О. І. СИПЯГІН
інженер повного стеку
floLive
ORCID: 0009-0008-7565-3221

М. А. ЮЖАКОВ
старший інженер програмного забезпечення
Plaid
ORCID: 0009-0008-1307-4275

Р. І. ІБРАГІМОВ
старший розробник програмного забезпечення
VoiceLove
ORCID: 0009-0002-7612-3341

© Sypiahin O. I., Yuzhakov M. A., Ibrahimov R. I., 2025
Стаття поширюється на умовах ліцензії CC BY 4.0

АРХІТЕКТУРНІ РІШЕННЯ ДЛЯ МАСШТАБОВАНИХ ВЕБЗАСТОСУНКІВ З УРАХУВАННЯМ ПРОДУКТИВНОСТІ ТА ЗАТРИМОК

Швидке зростання цифрових сервісів та зростаючі вимоги глобально розподілених користувачів ставлять у центр уваги вебзастосунків питання масштабованості, продуктивності та оптимізації затримок. Забезпечення високої продуктивності за зростаючих навантажень є критично важливим для стабільності бізнес-процесів, задоволеності користувачів та ефективності системи.

Ціллю дослідження є порівняльна оцінка чотирьох архітектурних парадигм вебзастосунків – монолітної, мікросервісної, серверлес та edge-комп'ютингу – за умов навантажень, що імітують реалістичні сценарії електронної комерції, включно з одночасними транзакціями, пошуком товарів та платіжними запитами, для визначення їхньої поведінки щодо масштабованості, продуктивності та затримок.

Як результат, дослідження показало значні відмінності у поведінці систем за різних архітектур. Монолітні системи, хоча й передбачувані та прості в експлуатації, швидко насичувалися під навантаженням: час відповіді перевищував900 мспри1 500одночаснихкористувачахтадосягавпонад3 снапіковихнавантаженнях.Мікросервіси розширювали діапазон масштабованості, забезпечуючи стабільну продуктивність до 3 000–3 500 користувачів із середніми затримками 420–600 мс, проте при високій одночасності накладні витрати на оркестрацію призводили до різкого зростання затримок. Серверлес-платформи показали найвищу еластичність, майже миттєво масштабуясь до 6 200 запитів на секунду та ефективно обробляючи пікові навантаження, хоча спостерігалися пікові затримки до 1 000 мс через холодні запускі та змінність виконання. Edge-розгортання забезпечували найменші затримки загалом, залишаючись нижче 300 мс навіть при 5 000 користувачах, і досягали пікової пропускної здатності 5 400 запитів на секунду, хоча вимагали значного забезпечення ресурсів та створювали більшу операційну складність. Результати свідчать, що жодна окрема архітектурна парадигма не повністю задовольняє вимоги масштабованості та затримки для різних профілів навантажень. Гібридні стратегії, що поєднують мікросервіси для стабільності ядра, серверлес для непередбачуваних піків та edge-комп'ютинг для критично чутливих до затримок операцій, забезпечують найбільш збалансовані результати, гарантують відгук системи, стабільність і економічно обґрунтовану масштабованість.

Висновки: гнучкі архітектурні підходи забезпечують високу продуктивність, стабільність та масштабованість вебзастосунків, оптимізуючи затримки та операційні витрати. Отримані результати підкреслюють важливість адаптивності архітектури для створення стійких та ефективних вебзастосунків у сучасних цифрових екосистемах.

Ключові слова: *масштабовані вебзастосунки, мікросервіси, безсерверні технології, edge-комп'ютинг, Kubernetes, затримка, оптимізація продуктивності, CI/CD, хмарна архітектура.*

Formulation of the problem

Modern web applications are expected to handle millions of requests, adapt to unpredictable workloads, and deliver responses within milliseconds, regardless of where the user is located. These demands place the architecture at the center of performance and scalability decisions. What once worked for small, monolithic systems is no longer enough in environments shaped by continuous deployment, global traffic distribution, and real-time interaction. To address these challenges, developers increasingly turn to microservices managed by container orchestration platforms, serverless computing for unpredictable spikes, and edge deployments that physically shorten the path between user and service. Each of these approaches offers tangible benefits but also creates new layers of complexity. Microservices can fragment monitoring and raise communication costs, serverless models complicate stateful logic and cost forecasting, and edge computing adds operational overhead in exchange for lower latency. By comparing different architectural solutions under varied workload patterns, the study highlights not only where each model excels but also where its limitations become critical. In doing so, it points toward hybrid strategies that balance scalability with predictable performance, helping teams design applications that remain responsive even as systems grow in size and complexity.

Analysis of the latest research and publications

The architecture of scalable web applications has become a central focus in modern software engineering due to the rapid expansion of cloud computing, the global distribution of users, and the performance constraints imposed by latency-sensitive services. Traditional monolithic designs are increasingly being reconsidered as they struggle to meet the requirements of dynamic workloads and continuous delivery pipelines [1]. Recent research emphasizes the shift toward microservices, serverless computing, and hybrid architectures that can adapt to both predictable and burst traffic while sustaining stable throughput and low response times [2].

Studies have explored the potential of cloud-native approaches to reduce latency and improve elasticity. For example, Atoll [3] and Nightcore [4] demonstrate serverless platforms optimized for interactive workloads, highlighting the importance of efficient resource allocation and low-overhead communication in latency-sensitive environments. Similarly, applied research points to best practices in backend architecture that balance scalability with predictable performance in real-world deployment scenarios [5].

Comparative evaluations have also been conducted between monolithic and microservice models, showing that while microservices generally improve scalability and modularity, they introduce nontrivial communication costs and complexity in monitoring and orchestration [6]. At the same time, distributed and edge-based approaches are increasingly recognized as promising strategies for lowering user-perceived latency, but they raise challenges of deployment complexity and configuration overhead [2].

International publications underline the necessity of combining architectural choices with rigorous performance management, observability pipelines, and workload-aware optimizations [2, 5]. This ensures not only that systems remain responsive under stress but also that operational costs and stability remain within acceptable limits. As a result, determining the optimal mix of architectural paradigms–monolithic, microservices, serverless, and edge computing–remains an active area of research, with significant practical implications for scalable web application design.

A relevant topic of this study is the search for architectural solutions that ensure both scalability and low-latency performance in modern web applications. Although cloud-native technologies, microservice patterns, and serverless platforms have achieved significant adoption worldwide, many development teams still lack a consistent framework for selecting and combining these approaches within CI/CD environments. Addressing this challenge requires not only adapting proven international practices but also developing solutions tailored to the specifics of local infrastructures, workload profiles, and organizational constraints.

Formulation of the purpose of the research

The aim of this study is architectural models and design strategies that enable web applications to scale efficiently while maintaining predictable performance and low latency. The research seeks to identify the strengths and weaknesses of monolithic, microservice, serverless, and edge-based paradigms, and to outline practical guidelines for combining them into hybrid solutions suited for continuous integration and deployment environments.

Presentation of the main material

This study is based on a theoretical and applied analysis of architectural approaches for designing scalable web applications with a focus on performance and latency. The research draws on peer-reviewed publications, technical documentation from industry leaders, and case studies describing real-world deployments of microservice, serverless, and edge-based platforms. Architectural scenarios were modeled using representative CI/CD workflows, with emphasis on comparing containerized microservices managed by Kubernetes, serverless computing frameworks for handling burst workloads, and edge nodes designed to reduce user-perceived latency.

Performance estimations were developed through workload simulations reflecting typical e-commerce and media streaming environments, allowing assessment of throughput, response time, and fault tolerance under varying load conditions. Architectural models were constructed to illustrate data flow across application layers, orchestration mechanisms, and monitoring pipelines. Simulation results were analyzed in terms of scalability efficiency, stability during rapid workload changes, and the predictability of cost and latency metrics.

The evaluation also considered trade-offs such as communication overhead in microservices, state management complexity in serverless platforms, and configuration challenges in edge deployments. Comparative analysis was conducted to identify optimal configuration strategies and hybrid models that balance elasticity, reliability, and performance under continuous deployment environments.

The evaluation of architectural approaches confirmed that scalability and latency are strongly dependent on the choice of deployment model and workload profile. Comparative modeling was carried out for three paradigms – microservices orchestrated by Kubernetes, serverless platforms, and edge computing nodes – against a baseline monolithic system. The focus was on throughput, average response time, and stability under load. At the first stage, each architecture was subjected to increasing workloads emulating typical e-commerce operations such as concurrent transactions, search queries, and payment requests. This approach reflects real-world scenarios where traffic intensity fluctuates and service responsiveness directly affects user experience. Similar evaluation methods are widely applied in studies of high-performance computing systems and scalable architectures [7]. The results of such work illustrate how performance and scalability can be achieved by integrating flexible resource allocation with unified pipeline design, reinforcing the idea that both computational throughput and latency minimization depend on efficient orchestration and workload balancing. In parallel, research on cloud-native web applications highlights the importance of embedding performance and security considerations into the architecture itself [8].

These studies show that scalability alone cannot guarantee resilience in modern distributed systems, where latency bottlenecks are often compounded by security vulnerabilities. Findings indicate that embedding security principles within cloud-native architectures ensures that scaling strategies do not compromise stability or increase attack surfaces. Building on these insights, the present evaluation applied controlled simulations to reveal how architectural choices affect not only raw throughput and latency but also system robustness under stress. Figure 1 presents the change in average response time as the number of concurrent users grows from 500 to 5000.

The data in Figure 1 illustrate the change in average response time as the number of concurrent users increases from 500 to 5000 across four architectural models – monolithic, microservices, serverless, and edge computing. The monolithic system reaches a critical threshold at around 1500 users, where the average response time exceeds 900 ms and continues to grow beyond 3 seconds at 5000 users. This confirms the limited scalability of the traditional approach. The microservices model maintains stable performance much longer: up to 3000–3500 users, response time does not exceed

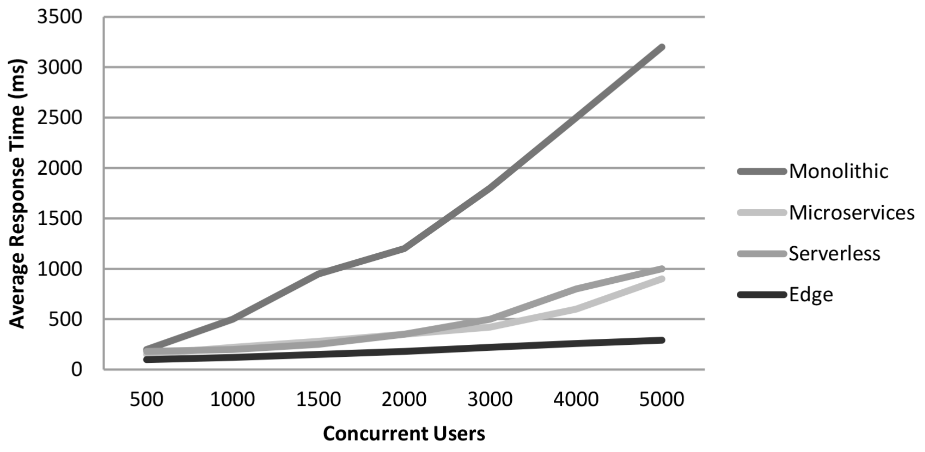

Fig. 1. Average response time under increasing load across four architectural models

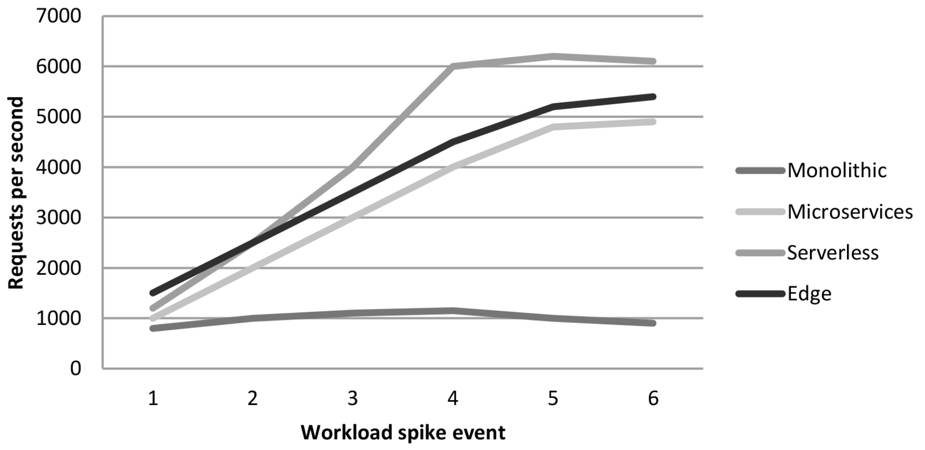

Fig. 2. Throughput dynamics under simulated workload spikes

420–600 ms. However, after this point, a steep rise occurs, with delays approaching 900 ms at peak load. The serverless model demonstrates strong elasticity under low and medium loads (250–350 ms). At 3000–4000 users, however, latency spikes up to 800–1000 ms appear, primarily due to cold-start delays and cost-related constraints. The edge architecture proves the most stable: even at 5000 users, the average response time remains below 300 ms. This highlights the advantage of localized processing, though at the cost of higher operational complexity. In addition, throughput and elasticity were analyzed. Figure 2 illustrates the throughput dynamics (requests per second) relative to workload spikes.

Figure 2 indicates that serverless platforms provide the most elastic throughput response, scaling almost instantly to accommodate traffic surges. Microservices achieve consistent linear scaling, but orchestration overhead slows reaction to sudden bursts. Edge deployments deliver strong local throughput but require prior resource provisioning, while the monolithic baseline rapidly saturates and fails under sharp load increases.

Further analysis shows that elasticity in serverless computing not only ensures rapid scaling but also corresponds with the latest surveys on architectural design [10]. While this model is highly adaptive to workload variability, cold-start delays and cost unpredictability remain critical limitations, which explains the latency spikes recorded in our experiments. Edge deployments, although effective in maintaining consistently low latency, also align with recent studies on distributed architectures [9]. These works emphasize that strong coordination and security mechanisms are essential for sustaining throughput across geographically distributed nodes. This insight is consistent with our findings, which revealed that edge computing requires substantial resource provisioning and robust orchestration to prevent performance bottlenecks. Overall, the data suggest that no single paradigm can provide complete elasticity, scalability, and cost predictability on its own. Instead, hybrid strategies that integrate serverless elasticity, microservice stability, and edge responsiveness represent the most effective architectural pathway for scalable web applications. Table 1 summarizes the key performance indicators observed for each architecture under standardized testing conditions.

Analysis of the data in Table 1 shows clear trade-offs. Monolithic systems remain cost-predictable but fail to meet performance requirements at scale. Microservices deliver balanced throughput and latency, though orchestration complexity grows with system size. Serverless models are most adaptive to unpredictable traffic but raise concerns about cost predictability and latency spikes from cold starts. Edge computing minimizes response time and user-perceived latency,

Table 1

Comparative performance of architectural models

Architecture	Peak throughput (req/s)	Avg. response time @ 3000 users (ms)	Latency floor (ms)	Cost predictability	Operational complexity
Monolithic	1,200	950	200	High	Low
Microservices	4,800	420	150	Medium	Medium-High
Serverless	6,200	500 (with spikes to 800)	180	Low (variable)	Medium
Edge computing	5,400	270	100	Medium	High

but at the price of higher deployment and maintenance overhead. These findings are consistent with broader surveys that emphasize the role of architectural coordination and management. Recent work on software-defined networking points out that scalability and latency optimization often depend on effective traffic management and placement strategies rather than raw computing power alone [11]. Similarly, studies of microservice-based architectures underline that while this paradigm supports modularity and migration flexibility, it also introduces complex integration challenges that must be resolved to achieve sustainable scalability [12]. Together, these perspectives reinforce the necessity of hybrid models that combine architectural efficiency with strong coordination and migration strategies.

The conducted analysis demonstrates that the scalability and latency of web applications are directly influenced by the chosen architectural paradigm and its ability to handle varying workload profiles. Monolithic architectures, while predictable in cost and simple to operate, cannot sustain modern performance demands once concurrent users exceed relatively low thresholds. Microservices extend scalability considerably, maintaining acceptable latency under higher loads, though orchestration introduces management overhead. Serverless platforms excel in elasticity and immediate response to traffic surges but face challenges with cold-start delays and cost predictability. Edge computing achieves the lowest user-perceived latency and strong local throughput, yet requires significant resource provisioning and operational expertise.

Taken together, these findings confirm that each architectural model has distinct advantages and limitations. No single approach is sufficient to ensure both high scalability and consistently low latency in real-world conditions. The results underline the value of hybrid deployment strategies, where microservices form the foundation for stable growth, serverless functions address burst workloads, and edge nodes optimize latency-sensitive interactions. Such combinations offer a more resilient, cost-aware, and performance-driven path for designing scalable web applications in CI/CD environments.

Conclusions

The conducted analysis clearly shows that the scalability and latency of web applications differ substantially across architectural models. The monolithic baseline proved least effective: already at around 1500 concurrent users, average response times exceeded 900 ms, and by 5000 users the delay grew beyond 3 seconds, confirming limited suitability for high-load systems. Microservices delivered more sustainable performance, maintaining acceptable response times up to 3000–3500 users, with an average of 420–600 ms under load. However, once this threshold was exceeded, latency increased sharply, illustrating the limits imposed by orchestration overhead. Serverless platforms demonstrated the strongest elasticity. They scaled almost instantly to accommodate workload spikes, achieving the highest peak throughput of 6200 requests per second. Nevertheless, cold-start delays and cost variability produced visible latency spikes in the range of 800–1000 ms at high concurrency levels, which reduced stability. Edge computing achieved the lowest latency overall, remaining below 300 ms even under maximum load. Throughput performance was also high, peaking at 5400 requests per second. However, this came with higher operational complexity and the requirement for prior resource provisioning. Taken together, the results demonstrate that no single paradigm can fully satisfy both scalability and latency requirements. Hybrid approaches, combining microservices for stable workloads, serverless for unpredictable bursts, and edge nodes for latency-sensitive operations, provide the most balanced and resilient solution. Such strategies enable web applications to maintain responsiveness, stability, and efficiency even under rapidly changing traffic conditions.

Bibliography

1. Стрельцов О., Орновецький Ю., Катріченко М. Підвищення ефективності масштабування архітектури хмарних додатків. *Електротехнічні та комп'ютерні системи*. 2023. № 38 (114). С. 58–65. DOI: https://doi.org/10.15276/eltecs.38.114.2023.7

2. Santos J., Wauters T., Volckaert B., De Turck F. Towards low-latency service delivery in a continuum of virtual resources: State-of-the-art and research directions. *IEEE Communications Surveys & Tutorials*. 2021. Vol. 23, No 4. P. 2557–2589. DOI: https://doi.org/10.1109/COMST.2021.3095358

3. Singhvi A., Balasubramanian A., Houck K., Shaikh M. D., Venkataraman S., Akella A. Atoll: A scalable low-latency serverless platform. *Proceedings of the ACM Symposium on Cloud Computing*. 2021. P. 138–152. DOI: https://doi.org/10.1145/3472883.3486981

4. Jia Z., Witchel E. Nightcore: efficient and scalable serverless computing for latency-sensitive, interactive microservices. *Proceedings of the 26th ACM international conference on architectural support for programming languages and operating systems*. 2021. P. 152–166. DOI: https://doi.org/10.5281/zenodo.4321760

5. Cherukuri B. R. Building scalable web applications: Best practices for backend architecture. *International Journal of Science and Research (IJSR)*. 2024. Vol. 13, No 10. P. 126–139. DOI: https://dx.doi.org/10.21275/ES24928085711

6. Blinowski G., Ojdowska A., Przybyłek A. Monolithic vs. microservice architecture: A performance and scalability evaluation. *IEEE Access*. 2022. No 10. P. 20357–20374. DOI: https://doi.org/10.1109/ACCESS.2022.3152803

7. Liao H., Tu J., Xia J., Liu H., Zhou X., Yuan H., Hu Y. Ascend: a scalable and unified architecture for ubiquitous deep neural network computing: Industry track paper. *2021 IEEE International Symposium on High-Performance Computer Architecture (HPCA)*. 2021. P. 789–801. DOI: https://doi.org/10.1109/HPCA51647.2021.00071

8. Datla L. S., Thodupunuri R. K. Designing for Defense: How We Embedded Security Principles into Cloud-Native Web Application Architectures. *International Journal of Emerging Research in Engineering and Technology*. 2021. Vol. 2, No 4. P. 30–38. DOI: https://doi.org/10.63282/3050-922X.IJERET-V2I4P104

9. Rahman A., Islam M. J., Band S. S., Muhammad G., Hasan K., Tiwari P. Towards a blockchain-SDN-based secure architecture for cloud computing in smart industrial IoT. *Digital Communications and Networks*. 2023. Vol. 9, No 2. P. 411–421. DOI: https://doi.org/10.1016/j.dcan.2022.11.003

10. Li Z., Guo L., Cheng J., Chen Q., He B., Guo M. The serverless computing survey: A technical primer for design architecture. *ACM Computing Surveys (CSUR)*. 2022. Vol. 54, No 10s. P. 1–34. DOI: https://doi.org/10.1145/3508360

11. Priyadarsini M., Bera P. Software defined networking architecture, traffic management, security, and placement: A survey. *Computer Networks*. 2021. No 192. P. 108047. DOI: https://doi.org/10.1016/j.comnet.2021.108047

12. Velepucha V., Flores P. A survey on microservices architecture: Principles, patterns and migration challenges. *IEEE Access*. 2023. No 11. P. 88339–88358. DOI: https://doi.org/10.1109/ACCESS.2023.3305687

References

1. Streltsov, O., Ornovetsky, Y., & Katrichenko, M. (2023). Pidvyshchennia efektyvnosti masshtabuvannia arkhitektury khmarnykh dodatkiv [Increasing the efficiency of cloud application architecture scaling]. *Elektrotekhnichni ta komp'iuterni systemy – Electrotechnic and Computer Systems*, 38(114), 58–65. https://doi.org/10.15276/eltecs.38.114.2023.7

2. Santos, J., Wauters, T., Volckaert, B., & De Turck, F. (2021). Towards low-latency service delivery in a continuum of virtual resources: State-of-the-art and research directions. *IEEE Communications Surveys & Tutorials*, 23(4), 2557–2589. https://doi.org/10.1109/COMST.2021.3095358

3. Singhvi, A., Balasubramanian, A., Houck, K., Shaikh, M. D., Venkataraman, S., & Akella, A. (2021, November). Atoll: A scalable low-latency serverless platform. *ACM Symposium on Cloud Computing*, 138–152. https://doi.org/10.1145/3472883.3486981

4. Jia, Z., & Witchel, E. (2021, April). Nightcore: efficient and scalable serverless computing for latency-sensitive, interactive microservices. *26th ACM international conference on architectural support for programming languages and operating systems*, 152–166. https://doi.org/10.5281/zenodo.4321760

5. Cherukuri, B. R. (2024). Building scalable web applications: Best practices for backend architecture. *International Journal of Science and Research (IJSR)*, 13(10), 126–139. https://dx.doi.org/10.21275/ES24928085711

6. Blinowski, G., Ojdowska, A., & Przybyłek, A. (2022). Monolithic vs. microservice architecture: A performance and scalability evaluation. *IEEE access*, 10, 20357–20374. https://doi.org/10.1109/ACCESS.2022.3152803

7. Liao, H., Tu, J., Xia, J., Liu, H., Zhou, X., Yuan, H., & Hu, Y. (2021, February). Ascend: a scalable and unified architecture for ubiquitous deep neural network computing: Industry track paper. *2021 IEEE International Symposium on High-Performance Computer Architecture (HPCA)*, 789–801. https://doi.org/10.1109/HPCA51647.2021.00071

8. Datla, L. S., & Thodupunuri, R. K. (2021). Designing for Defense: How We Embedded Security Principles into Cloud-Native Web Application Architectures. *International Journal of Emerging Research in Engineering and Technology*, 2(4), 30–38. https://doi.org/10.63282/3050-922X.IJERET-V2I4P104

9. Rahman, A., Islam, M. J., Band, S. S., Muhammad, G., Hasan, K., & Tiwari, P. (2023). Towards a blockchain-SDN-based secure architecture for cloud computing in smart industrial IoT. *Digital Communications and Networks*, 9(2), 411–421. https://doi.org/10.1016/j.dcan.2022.11.003

10. Li, Z., Guo, L., Cheng, J., Chen, Q., He, B., & Guo, M. (2022). The serverless computing survey: A technical primer for design architecture. *ACM Computing Surveys (CSUR)*, 54(10s), 1–34. https://doi.org/10.1145/3508360

11. Priyadarsini, M., & Bera, P. (2021). Software defined networking architecture, traffic management, security, and placement: A survey. *Computer Networks*, 192, 108047. https://doi.org/10.1016/j.comnet.2021.108047

12. Velepucha, V., & Flores, P. (2023). A survey on microservices architecture: Principles, patterns and migration challenges. *IEEE access*, 11, 88339–88358. https://doi.org/10.1109/ACCESS.2023.3305687

Дата першого надходження рукопису до видання: 27.09.2025
Дата прийнятого до друку рукопису після рецензування: 22.10.2025
Дата публікації: 28.11.2025