Edge Computing for Web Applications: A State of the Art

Daniel Studera
it251507@ustp-students.at
University of Applied Science St.Pölten
Vienna, Austria

Abstract

This paper examines the use of edge computing to enhance modern web applications that require low latency and high interactivity. Traditional cloud architectures struggle with increasing traffic and long network distances, which can lead to congestion and reduce the quality of the user experience. Content Delivery Networks (CDNs) were an early step towards decentralisation because they cache content on servers closer to users, thereby shortening round-trip times. Edge computing builds on this concept by not only moving data, but also parts of the application logic, to nearby edge servers. This paper explains how serverless edge platforms execute event-driven functions at the network edge to enable faster, more dynamic page rendering and reduce latency, bandwidth requirements and energy usage on mobile devices. It also identifies open challenges for edge-based web applications, including limited resources on edge nodes, managing distributed application state and ensuring security and reliability across diverse infrastructures.

Keywords

Edge computing, Web applications, Content Delivery networks, Serverless computing, Web performance, Low-latency systems

ACM Reference Format:
Daniel Studera. 2025. Edge Computing for Web Applications: A State of the Art. In *Unpublished Manuscript, University of Applied Sciences St. Pölten.* ACM, New York, NY, USA, 5 pages.

1 Introduction

Edge computing has become a key technology for handling growing demands in low-latency data processing and real-time responsiveness. Unlike centralized cloud models, edge computing brings data processing and computation closer to end devices. This is particularly important for applications in the Internet of Things (IoT) and web environments, where responsiveness and localized control are essential. (Gupta et al., 2025, p. 94) (Batool and Kanwal, 2025, p. 1)

In theory, the fastest way is always to skip the data transportation over the internet and compute tasks directly on the device (Varghese et al., 2016, p. 21). However, for smaller devices like phones or Internet-of-Things devices the computational power may be too small and has to be sourced out to servers. But real-time applications may have preferred response times under 100 ms, where classic cloud infrastructure cannot provide responses fast enough. (Varghese et al., 2016, p. 21)

Web applications benefit from edge computing by reducing access time to distant cloud servers. Placing computing tasks at the network edge enables faster content delivery, dynamic page rendering and low-latency interactions. These benefits are crucial for modern, interactive web services. (Varghese et al., 2016, p. 21)

Bringing some of the computation physically closer to the user is part of a greater concept called the Content Delivery Network (CDN). It not only focuses on edge computing but has evolved over time into using AI techniques to predict traffic and optimize routes, 5G technologies and also serverless architectures. (Tyagi, 2025, p. 402)

This paper provides a state-of-the-art overview of edge computing for web applications, focusing on latency, CDN evolution and serverless edge platforms.

2 Background: Web applications and Latency

Today, Web-based Information Systems (WBIS) play a crucial role in many sectors such as healthcare, smart cities and industrial automation. However, traditional centralized cloud computing often struggles to meet the high performance requirements of these modern applications. The main problems are bandwidth limitations and long physical distance between a device and the cloud servers. To solve this, data processing must move closer to the source to improve responsiveness and efficiency. (Fazil et al., 2025, p. 1)

2.1 The Necessity of Low Latency

Using a centralized cloud introduces unavoidable delays because data has to travel a long distance. This is a major issue because mobile data traffic is exploding, with things like video streaming accounting for a huge portion of network load (Zhao et al., 2021, p. 1). Sending all this heavy traffic to central clouds causes congestion and wastes bandwidth (Varghese et al., 2016, p. 21). Furthermore, mobile devices often suffer from limited battery life. By processing tasks at the network edge instead of sending them deep into the cloud, we can not only reduce response time but also increase battery life (Javed et al., 2021, p. 16) (Zhao et al., 2021, p. 10). Ultimately, high latency hurts the Quality of Experience (QoE), which is often more important to users than simple technical metrics (Zhao et al., 2021, p. 2).

2.2 Real-Time Interaction and Dynamic Content

The architecture of web applications has evolved significantly. Historically, websites relied on Server-Side-Rendering (SSR), where the server builds the complete page for every request (Vepsäläinen et al., 2023, p. 2). Later, developers shifted towards Client-Side-Rendering (CSR) and Single Page Applications (SPAs) to create more interactive experiences that feel like desktop applications without reloading the whole page (Vepsäläinen et al., 2023, p. 2).

Unpublished, St. Pölten, Austria
2025.

However, mobile devices often have limited computing power and may not handle some complex tasks on their own (Varghese et al., 2016, p. 21). Offloading these tasks to a distant cloud is often too slow. Edge computing helps by allowing dynamic content generation at the network edge, supporting new hybrid techniques like Incremental Static Regeneration (ISR) or the "Islands Architecture", which allow dynamic content to be loaded efficiently without rebuilding the whole site (Vepsäläinen et al., 2023, p. 3).

2.3 Latency Thresholds and Requirements

Real-time applications have strict limits on response times. Research shows that interactive applications, such as visual guiding systems, work best with a response time between 25ms and 50ms (Varghese et al., 2016, p. 20-21). Traditional cloud infrastructures are often too slow, with round-trip times reaching around 175ms (e.g. between Canberra and Berkeley) (Varghese et al., 2016, p. 21). In industrial or medical scenarios, such delays can cause fatal errors (Zhao et al., 2021, p. 2). To consistently achieve response times fast enough, computing tasks could be moved from the centralized cloud to edge nodes (Cao et al., 2020, p. 85716).

3 Content Delivery Network as a Precursor

Before understanding edge computing, the greater concept to grasp is the Content Delivery Network (CDN) and the necessity to provide content more efficiently (Vepsäläinen et al., 2023, p. 1,4). CDNs represent a significant technological precursor in this evolution, establishing the fundamental concept of decentralizing data to reduce latency and increase reliability (Vepsäläinen et al., 2023, p. 1,3-4).

3.1 Basic Idea and Historical Context

For a long time, websites were hosted on a central web server that served static content (Vepsäläinen et al., 2023, p. 1). In the early 1990s, when the internet and websites were an emerging technology, bandwidth-intensive content such as images and web pages were causing bandwidth congestion and brought up the use of basic web caching (Zhao et al., 2021, p. 4). With the explosion of multimedia traffic in the 21st century, especially videos, centralized server architectures became insufficient due to high latency when transmitting data over long geographical distances (Zhao et al., 2021, p. 4) (Gupta, 2024, p. 2).

The fundamental concept of a Content Delivery Network is to lower the physical distance by distributing content across a global network of servers located closer to the end user (Siidorow, 2024, p. 9-10). Instead of routing every request to a central origin server, CDNs deliver static assets, such as HTML documents or media files, from multiple geographically distributed points (Siidorow, 2024, p. 9-10). This architecture significantly reduces the Round-Trip Time (RTT) and reduces the load on the central servers as well as other parts of the network's infrastructure by minimizing redundant data transmission (Gupta, 2024, p. 2) (Siidorow, 2024, p. 9-10).

3.2 Cache Hierarchies and Infrastructure

The architecture of a CDN is based on the strategic deployment of surrogate servers, or edge nodes at the network's border (Vepsäläinen et al., 2023, p. 3-4) (Varghese et al., 2016, p. 20-21). These nodes function as proxy caches that replicate and store copies of popular content to maximize availability and access speed and minimize requests to far away central servers (Gupta, 2024, p. 5).

The distributed infrastructure represents a distinct shift from the traditional cloud computing model. While cloud computing relies on centralized data centers to gather resources and perform long-term, heavy data analysis, this centralization often introduces latency due to the physical distance to the data source. (Dong et al., 2020, p. 314, 316) (Cao et al., 2020, p. 85715) In contrast, the edge layer decentralizes operations by acting as an executor for real-time, small-scale data processing, while the cloud remains the global coordinator for tasks where high speeds are not a requirement (Dong et al., 2020, p. 318) (Cao et al., 2020, p. 85716). Therefore, edge computing and CDNs do not replace the cloud but supplement it and work together to form a "Cloud-Edge", where depending on latency and processing power requirements the tasks are either handled locally or forwarded to the central cloud (Dong et al., 2020, p. 315-316) (Fazil et al., 2025, p. 5) (Cao et al., 2020, p. 85717).

To manage the limited storage at these edge nodes efficiently, CDNs use sophisticated caching concepts like "Least recently used" (LRU), "Least frequently used" (LFU) and "First in first out" (FIFO) (Zhao et al., 2021, p. 12). In mobile network environments, these caching strategies can extend to caching directly at base stations to further reduce backhaul traffic and response time to improve the user's Quality of Experience (Zhao et al., 2021, p. 5,9).

3.3 Typical Request Flow

The typical flow of a request in an architecture that uses Content Delivery Networks differs significantly from the traditional communication between client and server. In a typical mobile edge caching model, content requests coming in from the user are first received by edge nodes located in the physically close environment of the user, rather than traveling directly to a far away central data center (Zhao et al., 2021, p. 8-9). To manage this traffic efficiently, CDNs use global load balancing mechanisms that assign each request to the closest available cache server (Dong et al., 2020, p. 318). In this process, the Domain Name System (DNS) redirects the request toward the nearest and most responsive cache node, based on the user's current network location (Zhao et al., 2021, p. 9).

A typical request flow could look like this:

(1) **Routing and Identification:** When a user requests content, the network identifies the optimal edge node for this request. This selection is handled by load-balancing algorithms that assign the request to the geographically nearest edge nodes (Zhao et al., 2021, p. 9) (Dong et al., 2020, p. 318)
(2) **Content Exploration and Cache Lookup** The edge node checks its local storage for the requested asset. If the content is not immediately available on the specific node, the system must search the network to determine the best way to retrieve it at the lowest cost. This is defined as the "content query problem" and may involve forwarding queries to neighboring user equipment or base stations before traveling the whole distance to the central network. (Zhao et al., 2021, p. 10-11)
(3) **Retrieval and Delivery**

- Cache Hit: If the file is available, it is delivered immediately to the user.
- Cache Miss: If the content is not available on the node, it is retrieved from higher-level servers, or in the worst case, from the central cloud. The file is then stored locally using replacement policies such as Least Recently Used (LRU) to manage limited storage, and finally served to the user. (Zhao et al., 2021, p. 10-12)

This mechanism not only minimizes latency but also significantly reduces the load on backhaul links by minimizing redundant data traffic to the core network. (Cao et al., 2020, p. 85720)

3.4 Advanced CDN Architectures

As the demand for real-time interactivity and dynamic content grows, the traditional model of simple caching servers has proven to be insufficient in some cases. CDN infrastructures have evolved into complex architectures that build the base for smart edge computing. This is shown by emerging strategies like Distributed, Hybrid and Multi-CDN which are designed to enhance scalability, reliability and performance under varying network conditions.

- **Distributed Architectures:** In modern CDNs, the goal is a highly distributed architecture where numerous edge servers are deployed across multiple Points of Presence (PoPs). This approach focuses on minimizing the physical path between the user and the content and significantly reducing latency and ensuring that high traffic volumes are handled locally rather than overwhelming central data centers. (Tyagi, 2025, 406-407,414)
- **Hybrid Architectures:** Hybrid architectures combine traditional on-premise edge servers with cloud-based CDN services. This approach helps organizations adjust their resources based on real-time demand. During peak traffic, additional workloads can be offloaded to the cloud, while normal operations continue on local infrastructure. This idea is based on the Cloud-Edge where both edge nodes and cloud systems work together. (Tyagi, 2025, 406-407)
- **Multi-CDN Strategies:** To ensure high availability and reduce the risk of vendor-specific outages, big enterprises increasingly use Multi-CDN strategies. In this model, the traffic is distributed across CDN services from multiple vendors based on real-time performance metrics, geographical location and cost. If a certain CDN encounters problems like congestion or latency in particular regions or globally, an AI-driven traffic management system can automatically reroute the users to better-performing providers. This hopes to ensure that big platforms like Amazon or Netflix won't encounter big outages and interrupted service. (Tyagi, 2025, 406-407,414)

These advanced architectures show the significance of transitioning from CDNs as passive content repositories to active intelligent delivery platforms.

4 Edge Computing for Web Applications

While Content Delivery Networks have successfully decentralized the storage of static assets, the modern web requires the decentralization of application logic. Edge computing addresses this by shifting computational tasks from centralized cloud data centers to the edge of the network, closer to the end-user.

4.1 Definition and Operational Principle

Edge computing is defined as a distributed computing structure that brings computation and data storage closer to the location where it is needed, and therefore improves response times and saves bandwidth (Fazil et al., 2025, p. 1). Contrary to traditional cloud computing, where data is transmitted to distant data centers for processing, edge computing uses resources at the edge of the network, e.g. base stations, routers or micro data centers to execute application logic (Cao et al., 2020, p. 85715) (Varghese et al., 2016, p. 20).

In the specific context of web applications, it is often defined as Serverless Edge Computing. Here, developers deploy event-driven functions (Function-as-a-Service or FaaS) that run on edge nodes (Batool and Kanwal, 2025, p. 1). Platforms like AWS Lambda@Edge or Cloudflare Workers enable the execution of these functions in response to events (e.g. HTTPS requests) directly at edge nodes (Javed et al., 2021, p. 7-8) (Siidorow, 2024, p. 16,19-20). The serverless model on the edge is especially helpful for web applications, because it reduces the need for always-on servers, instead starting up containers only when requests occur, which optimizes resource usage and costs (Javed et al., 2021, p. 2).

4.2 Key Benefits: Latency, Bandwidth and Real-Time Processing

As already stated, the primary advantage of using edge computing in web development is the necessity to overcome the physical limitations of centralized cloud architecture.

- **Latency Reduction** By processing requests at the edge, the round-trip time (RTT) to the origin server is significantly reduced (Fazil et al., 2025, p. 1) (Varghese et al., 2016, p. 21). Research shows that in real-time applications such as gaming or augmented reality using a distant cloud poses serious latency problems due to geographical location (Varghese et al., 2016, p. 21).
- **Bandwidth Efficiency** Edge computing takes a lot of load off the central server infrastructure by processing data locally. Instead of transmitting lots of raw data to the cloud, edge nodes can filter, aggregate, or compress data and forward only relevant data to the central infrastructure (Varghese et al., 2016, p. 21). This is especially critical for bandwidth-heavy content like video streaming or Augmented and Virtual Reality applications where a lot of congestion can be prevented by processing on the edge of the network (Gupta, 2024, p. 3).
- **Real-Time Capabilities** Because edge computing is close to the data source, it can process information with much lower delay. This reduced transmission time enables real-time responses for the user and supports fast, context-aware decisions in web applications. (Fazil et al., 2025, p. 1-2)

4.3 Beyond the CDN: From Caching to Computing

Traditionally, Content Delivery Networks focused on caching static assets to reduce latency and origin server load (Vepsäläinen et al., 2023, p. 1) (Tyagi, 2025, p. 401-402). Edge computing evolves this model by transforming edge nodes from passive caches into active execution environments used to process dynamic content (Vepsäläinen et al., 2023, p. 4) (Tyagi, 2025, p. 411). This shift enables programmable capabilities where data analysis, security filtering and content generation happen at the edge of the network (Tyagi, 2025, p. 411) (Cao et al., 2020, p. 85715). Therefore, web applications can implement dynamic rendering strategies like server-side rendering (SSR) or incremental static generation (ISR), directly at the edge to optimize the Quality of Experience (QoE) for the end-user (Vepsäläinen et al., 2023, p. 1,3,5).

4.4 Edge Functions and Runtimes

Edge logic is primarily implemented via Serverless Edge Computing or Function-as-a-Service (FaaS), which allows developers to deploy stateless functions without infrastructure management (Batool and Kanwal, 2025, p. 1-2). The runtimes used generally fall into two categories:

- **MicroVM-based:** Architectures like AWS Lambda@Edge utilize lightweight virtualization (e.g. Firecracker) to provide isolation and broad language support, though they can struggle with "cold start" latencies when initialized (Siidorow, 2024, p. 16-17).
- **Isolate-based:** Platforms like Cloudflare Workers or Deno Deploy use V8 isolates to run multiple functions within a single process. In this approach "cold starts" are eliminated and it reduces memory usage, but it is typically restricted to JavaScript and WebAssembly environments (Siidorow, 2024, p. 22-23,27-28).

These runtimes enable developers to execute custom code directly at the network periphery, allowing them to shape client requests and server responses, providing faster response times and new possibilities (Vepsäläinen et al., 2023, p. 1).

(1) **Dynamic Request Manipulation:** Functions can dynamically assemble web pages or tailor content in real-time based on the user's location or device type, rather than serving generic static resources (Gupta, 2024, p. 8).
(2) **Media Optimization** Edge functions can perform on-the-fly transformations of media assets, such as resizing, cropping or formatting images and videos to match the capabilities of the requesting device (Gupta, 2024, p. 8).
(3) **Security and Access Control:** Complex access control logic can be implemented directly at the edge to allow the system to validate requests without long round trips to the origin server (Gupta, 2024, p. 8).
(4) **Real-Time AI Inference:** Edge nodes are capable of running lightweight AI models to perform tasks such as real-time content analysis, automated content moderation, or media analytics closer to the end-user. (Gupta, 2024, p. 5).

4.5 Challenges and Limitations

Deploying web applications at the edge of the network can introduce challenges compared to using a centralized cloud.

(1) **Resource Constraints:** Edge nodes often possess limited computational power and memory compared to cloud data centers. Therefore, highly efficient algorithms and effective resource scheduling are needed (Batool and Kanwal, 2025, p. 14).
(2) **State Management:** Connecting to centralized databases from the edge can reintroduce latency due to the round-trip time required for queries. While data replication to the edge may be a solution, it introduces risks regarding data consistency and "replication lag", making it difficult to maintain a synchronized state across all nodes for real-time applications. (Siidorow, 2024, p. 29).
(3) **Security:** Since Edge Functions are usually limited to a small scope and need fewer privileges, they often have a significantly reduced attack surface compared to full applications in containerized or virtual environments (Siidorow, 2024, p. 28).
(4) **Cold Starts:** Cold starts may occur when a certain edge function is not used for a longer time. Depending on the runtime, cold starts can present a significant problem by causing delays during complex workloads such as AI functions, which negate the latency benefit of edge functions if they take longer than the original round trip time to the data center (Siidorow, 2024, p. 27)

5 Conclusion

This work demonstrates how edge computing builds upon the concepts of traditional content delivery networks (CDNs) to overcome the latency and bandwidth limitations of centralized cloud architectures. Modern web-based systems depend on real-time interaction, rich media, and personalization. In such scenarios, long network paths to distant data centers can quickly slow down applications. CDNs reduce this issue by caching static content at the edge; meanwhile, edge computing builds on this model by running parts of the application logic on servers close to the user.

For web applications, this shift enables faster handling of requests, more responsive dynamic content and an improved quality of experience, particularly on mobile devices or in scenarios that require a lot of bandwidth. However, serverless edge platforms also introduce new challenges. Edge nodes have limited computing and memory resources, and coordinating distributed state is more challenging. Cold starts and heterogeneous runtimes can further reduce performance if not managed effectively.

Overall, current research indicates that the cloud and the edge should be used together. Content Delivery Networks (CDNs) and serverless edge platforms handle latency-sensitive tasks near the user, while centralized cloud systems remain important for computation-intensive workloads and long-term data storage. In the future, CDN and edge strategies will become increasingly prominent, as real-time services, IoT deployments, and rich media applications continue to grow and push performance requirements beyond what centralized cloud architectures can reliably provide.

Acknowledgments

Parts of the wording of this manuscript were supported by generative AI tools for language editing. All scientific content, structure and conclusions were created and verified by the author.

References

Iqra Batool and Sania Kanwal. 2025. Serverless Edge Computing: A Taxonomy, Systematic Literature Review, Current Trends and Research Challenges. arXiv:2502.15775 [cs.NI] https://arxiv.org/abs/2502.15775

Keyan Cao, Yefan Liu, Gongjie Meng, and Qimeng Sun. 2020. An Overview on Edge Computing Research. *IEEE Access* 8 (2020), 85714–85728. doi:10.1109/ACCESS.2020.2991734

Yunqi Dong, Jiujun Bai, and Xuebo Chen. 2020. A Review of Edge Computing Nodes Based on the Internet of Things. In *Proceedings of the 5th International Conference on Internet of Things, Big Data and Security (IoTBDS)*. 313–320. doi:10.5220/0009407003130320

A. W. Fazil, A. Ghairat, and A. J. Kohistani. 2025. Advancing Web-Based Information Systems Performance via Edge Computing: A Comprehensive Systematic Review. *GAME* 2, 4 (2025), 1–20. doi:10.29103/game.v2i4.24189

Ragini Gupta, Claudiu Danilov, Josh Eckhardt, Keyshla Bernard, and Klara Nahrstedt. 2025. Characterizing Container Performance in Edge Computing. In *Proceedings of the ACM SIGCOMM 2025 Posters and Demos* (Coimbra, Portugal) *(ACM SIGCOMM Posters and Demos '25)*. Association for Computing Machinery, New York, NY, USA, 94–96. doi:10.1145/3744969.3748438

Sachin Gupta. 2024. Enhancing Content Delivery with Edge Computing in Media and Entertainment. Zenodo. doi:10.5281/zenodo.13933556

Hamza Javed, Adel N. Toosi, and Mohammad S. Aslanpour. 2021. Serverless Platforms on the Edge: A Performance Analysis. arXiv:2111.06563 [cs.DC] https://arxiv.org/abs/2111.06563

Mikael Siidorow. 2024. Survey of Serverless Edge Computing for Web Applications. doi:10.13140/RG.2.2.13600.39680

Anuj Tyagi. 2025. Optimizing digital experiences with content delivery networks: Architectures, performance strategies, and future trends. *World Journal of Advanced Research and Reviews* 7, 2 (01 2025), 401–417. doi:10.30574/wjarr.2020.7.2.0230

Blesson Varghese, Nan Wang, Sakil Barbhuiya, Peter Kilpatrick, and Dimitrios Nikolopoulos. 2016. Challenges and Opportunities in Edge Computing. In *2016 IEEE International Conference on Smart Cloud (SmartCloud)*. IEEE, 20–26. doi:10.1109/SmartCloud.2016.18

Juho Vepsäläinen, Arto Hellas, and Petri Vuorimaa. 2023. Implications of Edge Computing for Static Site Generation. arXiv:2309.05669 [cs.HC] https://arxiv.org/abs/2309.05669

Yuhan Zhao, Wei Zhang, Longquan Zhou, and Wenpeng Cao. 2021. A Survey on Caching in Mobile Edge Computing. *Wireless Communications and Mobile Computing* 2021 (2021), 1–21. doi:10.1155/2021/5565648