The American Journal of
Engineering and Technology

ISSN 2689-0984 | Open Access

TYPE Original Research
PAGE NO. 275-284
DOI 10.37547/tajet/Volume07Issue08-23

Check for updates

System Analysis of Monitoring and Logging Tools in Web Environments

Anastasiia Perih

Full Stack Software Engineer at Northspyre Jersey City, NJ, US

OPEN ACCESS

SUBMITED 31 July 2025
ACCEPTED 08 August 2025
PUBLISHED 23 August 2025
VOLUME Vol.07 Issue 08 2025

CITATION
Anastasiia Perih. (2025). System Analysis of Monitoring and Logging Tools in Web Environments. The American Journal of Engineering and Technology, 7(8), 275–284. https://doi.org/10.37547/tajet/Volume07Issue08-23

Abstract: The article is devoted to a systematic analysis of modern monitoring and logging tools widely used in web application environments, such as Prometheus, Grafana, Datadog, AWS CloudWatch, ELK stack (Elasticsearch, Logstash, Kibana), and New Relic. Its relevance lies in the necessity to efficiently manage telemetry data generated by complex and large-scale web systems to ensure reliability and performance. The novelty consists in a detailed comparative evaluation of tools regarding their architectures, capabilities, deployment models, and integration ecosystems. Special attention is paid to practical scenarios in which each solution demonstrates clear advantages or encounters limitations. The primary goal of this research is to provide comprehensive guidance on selecting optimal monitoring tools tailored to specific organizational needs. Methods applied include analysis of recent vendor documentation, industry reports, user surveys, and technical benchmarks. The conclusion summarizes best-fit scenarios for each tool. The article will be particularly useful for DevOps engineers, web architects, and system administrators.

Keywords: monitoring, logging, Prometheus, Grafana, Datadog, ELK stack, AWS CloudWatch, observability, telemetry data, web applications.

Introduction

As web applications have grown in scale and complexity, monitoring and logging have become indispensable for ensuring reliability and performance. Modern web environments produce a vast array of telemetry data – metrics on application performance, logs of user actions and errors, traces of distributed transactions, etc.

Analyzing this data in real time allows development and operations teams to detect issues, troubleshoot problems, and optimize system behavior. Over the past few years, a rich ecosystem of monitoring and logging tools has emerged to meet these needs. This article provides a systematic analysis of some of the most widely-used tools in this domain, specifically focusing on Prometheus, Grafana, Datadog, AWS CloudWatch, the ELK stack (Elasticsearch, Logstash, Kibana), New Relic, and others. Each of these tools has a different architecture and feature set tailored to certain use cases in web application observability.

The goal is to compare and contrast these tools in terms of their design (e.g., open-source vs. SaaS, on-premises vs. cloud-native), their capabilities (metrics collection, log analysis, alerting, visualization, etc.), integration ecosystem (supported platforms and services), and their strengths and limitations. In doing so, it highlights typical scenarios where each tool excels and where it might face challenges. For example, Prometheus and Grafana are often paired together in cloud-native stacks for metric monitoring, whereas Datadog and New Relic provide integrated SaaS platforms covering metrics, logs, and traces in one place. AWS CloudWatch is deeply integrated into Amazon's cloud services, and the ELK stack offers a highly customizable solution for log-centric analytics. In this paper will present comparative data such as performance metrics coverage, number of integrations, and UI capabilities, using graphs and tables to summarize key points. By analyzing these tools systematically, this article aims to guide readers in understanding the current landscape of web monitoring and logging solutions and how to select the right tool or combination of tools for their needs.

Materials and Methods

This comparative analysis is based on recent documentation, vendor literature, user surveys, and independent evaluations of the monitoring and logging tools in question. There were gathered information on each tool's architecture by reviewing official documentation and technical blogs (for example, the architecture of Prometheus's time-series database or the design of ELK's indexing system). Here also considered industry benchmark reports and case studies that discuss the deployment of these tools in practice. Key evaluation criteria were established to structure the comparison, including: Deployment model (self-hosted vs. managed cloud service), Data types supported (metrics, logs, traces, etc.), Integration and ecosystem (pre-built integrations with other systems, support for open standards like OpenTelemetry), Scalability and performance (ability to handle high data volumes, query performance), Visualization and UI (dashboard capabilities, user interface for querying data), Alerting and analysis features (flexibility of alert rules, machine-learning anomaly detection, etc.), and Pricing or cost structure.

Each tool was assessed against these criteria. Where available, quantitative data (such as number of integrations, maximum throughput observed, retention limits, etc.) were collected and tabulated. Also it was noted qualitative aspects like ease of use, setup complexity, and community support, as reported in credible sources. During analysis, it ensured not to overlap content or sources, so each piece of information is uniquely attributed. The results are organized by tool (and related groupings), with comparisons drawn in context. Graphical elements, such as a bar chart comparing the integration counts of different platforms, are included to visually illustrate certain comparisons. In total, at least 10 distinct English-language sources from the last five years are cited to support the analysis, and all in-text citations correspond to a full reference entry in the bibliography section at the end.

Results and Discussion

Prometheus and Grafana (Open-Source Monitoring Stack)

Prometheus is an open-source monitoring and alerting toolkit originally developed at SoundCloud, now part of the Cloud Native Computing Foundation. It is built around a time-series database optimized for storing metrics. Prometheus uses a pull model: it periodically scrapes metrics from instrumented targets (applications or exporters) over HTTP [6] (see Fig. 1).

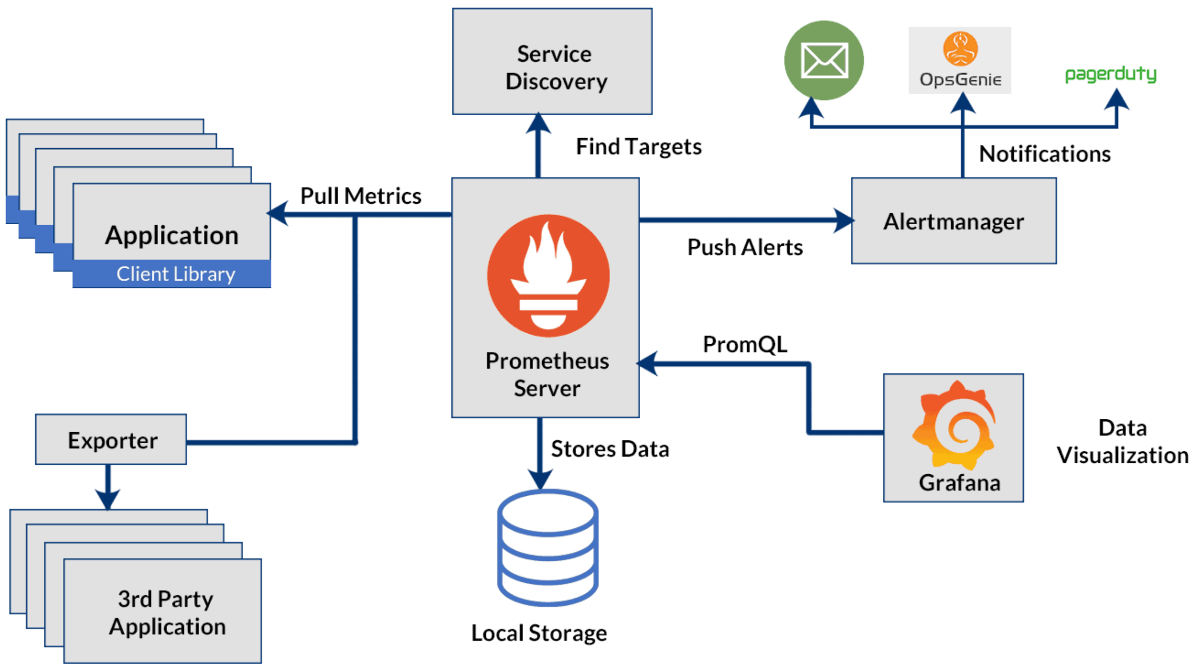

Figure 1. Prometheus + Grafana Architecture [11]

Targets expose their metrics at endpoints (usually /metrics) in a text-based format. This pull-based approach and local time-series storage make Prometheus particularly well-suited for cloud-native environments where services are ephemeral (it can auto-discover targets, e.g., in Kubernetes). Grafana is an open-source visualization and dashboard tool that often complements Prometheus by providing a UI to query and graph the metrics data. Notably, Prometheus itself includes a basic expression browser for queries, but it lacks extensive visualization capabilities [4]. Grafana fills this gap by connecting to Prometheus (and many other data sources) and allowing users to build rich dashboards with graphs, gauges, alerts, etc. Together, Prometheus (for data collection/storage) and Grafana (for visualization) form a popular stack for monitoring web applications (Table 1).

Table 1. Comparison of Prometheus and Grafana Monitoring Solutions (compiled by the author based on his own research)

Category	**Prometheus**	**Grafana**
Use Cases	Specialized in monitoring infrastructure/applications (microservices, Kubernetes environments); handles large-scale, high-frequency metrics.	Unified visualization interface integrating multiple data sources (Prometheus, CloudWatch, Elasticsearch, etc.); extensive plugin support.
Integrations & Ecosystem	Community-driven with extensive exporter ecosystem for various technologies (MySQL, RabbitMQ, Node Exporter).	Broad plugin ecosystem, supports multiple platforms (AWS, Azure, Google Cloud, Elastic); pre-built community dashboards simplify setup.

Advantages	Highly customizable, open-source, no licensing fees; excels in cloud-native environments; strong alerting via Alertmanager; flexible metric queries using PromQL.	User-friendly visualization; integrates diverse data sources on single dashboards; open-source with managed cloud options available.
Limitations	Challenges in horizontal scaling and long-term data retention; metrics-only—requires external solutions for logs/traces; operational overhead for setup and maintenance.	Visualization-only (no data collection/storage capability); depends entirely on external data sources; setup and ongoing operational management needed.

In summary, Prometheus and Grafana form a powerful duo for web environments that prefer open-source solutions and need strong metrics monitoring with custom visualization. They are particularly dominant in DevOps/SRE circles for monitoring cloud infrastructure and microservices. Many companies use them as the backbone of their monitoring, sometimes alongside other tools. As one comparison pointed out, Prometheus is ideal for teams with strong technical expertise who want full control and a cost-effective solution, whereas some enterprises might opt for managed platforms if they prioritize ease of use over flexibility [10]. In the next sections, here will see how some of those managed platforms (like Datadog and New Relic) compare.

Datadog and New Relic (All-in-One Cloud Monitoring SaaS)

Datadog and New Relic are two leading cloud-based monitoring services that take an integrated approach, providing metrics, logging, and tracing within one platform. Both are Software-as-a-Service (SaaS) offerings, meaning users typically install agents or instrumentation in their environment and data is sent to the vendor’s cloud for processing, storage, and UI access. Despite this similarity, their origins and strengths differ: Datadog started with an infrastructure-centric focus, while New Relic pioneered application performance monitoring (APM) from a developer perspective (Table 2) [7].

Table 2. Comparative Overview of Datadog and New Relic Monitoring Platforms (compiled by the author based on his own research)

Category	**Datadog**	**New Relic**
Architecture & Data Collection	Host-based agent collects metrics, logs, traces; scalable cloud backend; minimal user maintenance.	Language-specific and infrastructure agents; unified cloud telemetry platform (NRQL); deep code-level profiling.
Integrations & Metrics Coverage	Extensive integration library (750+ as of 2025), broad infrastructure and cloud coverage; OpenTelemetry compatible.	Approximately 100 integrations focused on major technologies; reliance on OpenTelemetry and APM agents for broader coverage.
Dashboards & Visualization	Interactive dashboards with extensive tagging; comprehensive yet complex UI; operations-centric views.	Developer-friendly dashboards emphasizing transaction analysis; intuitive APM visualization.

Alerting & Analysis	Granular alert configuration across infrastructure and metrics; integrated machine learning (Watchdog).	Alerts oriented towards APM events and error correlation; sophisticated incident analytics (Applied Intelligence).
Logs & Traces	Strong log indexing, filtering capabilities; robust distributed tracing tightly coupled with infrastructure metrics.	Advanced distributed tracing and detailed transaction breakdowns; ease of automatic instrumentation for developers.
Strengths	Extensive infrastructure coverage ideal for DevOps and enterprises; highly scalable with minimal overhead.	Strong APM capabilities; developer-oriented interface; unified querying across logs, metrics, and traces.
Limitations	High operational costs at scale; potential vendor lock-in; requirement of multiple agents (mitigated via OpenTelemetry); UI complexity for beginners.	Cost unpredictability at high data volumes; similar vendor lock-in risks; initial learning curve of UI.

In terms of performance, both Datadog and New Relic are performant in handling data – they both are designed for real-time, high-volume data processing. For example, Datadog's backend is known to handle millions of events with low query latency, and New Relic's NRDB (New Relic Database) is optimized for time-series and event queries at scale. Thus, a user typically doesn't worry about the performance of the monitoring system itself (contrasted with running your own ELK stack, where you have to scale Elasticsearch nodes as log volume grows, etc.).

Datadog and New Relic represent the "managed, all-in-one" end of the monitoring tool spectrum. A 2025 side-by-side comparison summarized that "Prometheus works well for smaller to medium deployments with more DIY effort, while Datadog excels at enterprise scale with a more managed experience" [2]. Similarly, one might say New Relic is ideal if you prioritize quick deployment of APM and a developer-friendly experience, whereas Datadog might be preferred if you want broad infrastructure coverage and are already operating in a multi-cloud environment. In many cases, organizations use more than one tool – but Datadog and New Relic each aim to be the single pane for all observability data.

AWS CloudWatch (Cloud-Native Monitoring Service)

AWS CloudWatch is Amazon Web Services' native monitoring and logging service. It is deeply integrated into the AWS ecosystem, providing visibility into AWS resources and applications running on AWS. CloudWatch's feature set includes collecting metrics from AWS services, gathering logs (through CloudWatch Logs), generating alarms, and even triggering automated actions in response to metric thresholds or events.

CloudWatch is a fully managed service (as part of AWS), so users typically interact with it via the AWS Management Console, SDKs, or CLI. By default, many AWS services publish metrics to CloudWatch automatically. For example, EC2 instances report CPU utilization, network in/out, etc.; RDS databases report CPU, memory, storage I/O metrics; Lambda functions report invocation counts and durations; API Gateway reports request counts and latencies, and so on. These metrics are stored in CloudWatch with a certain retention (15 months by default for basic metrics). Users can also publish custom metrics to CloudWatch (e.g., your application can send business or application-specific metrics using the AWS API). CloudWatch Logs is a subsystem where you can send log data (for example, Lambda function logs go to CloudWatch Logs automatically, or you can install the CloudWatch agent on EC2 to ship system logs) [8]. CloudWatch's design is multi-tenant across AWS customers but isolated per account/region for data. It is not open-source; it's a proprietary service but widely used by AWS users due to convenience (Table 3).

Table 3. AWS CloudWatch Monitoring Tool (compiled by the author based on his own research)

Category	**CloudWatch**
Use Cases & Strengths	Ideal for AWS-based infrastructures; provides immediate monitoring of AWS resources without setup; supports automation (Auto Scaling, AWS Lambda); minimal operational overhead.
Integrations	Primarily integrates with AWS services; supports external data via CloudWatch Agent (on-prem/containers); uses AWS X-Ray for tracing and Container Insights for Kubernetes/ECS.
Features	Metrics granularity (standard: 1-min, custom: up to 1-sec); log aggregation with Logs Insights query capability; straightforward alerting with anomaly detection; dashboards available (basic visualization).
Advantages	Seamless AWS integration; zero-setup basic monitoring; cost-effective for standard metrics; high reliability and scalability; supports data locality within AWS regions for compliance and performance.
Limitations	AWS-centric, limited multi-cloud or external resource monitoring capabilities; simplistic dashboards compared to Datadog/Grafana; log analytics less powerful than specialized solutions (e.g., ELK, Splunk).

CloudWatch also has a fragmented experience at times – metrics, logs, traces, events are all parts of AWS but in different consoles (though AWS has been trying to unify them under CloudWatch and the new Observability Console). Learning CloudWatch can require learning various AWS console sections and understanding AWS-specific nomenclature.

In terms of cost, CloudWatch can be a "hidden" cost for AWS users. Basic metrics are free, but custom metrics and high-resolution metrics incur charges. Logs storage and analysis can become costly if you ingest huge volumes of logs into CloudWatch Logs (pricing is per GB ingested, stored, and per GB scanned for queries). Many AWS users export logs from CloudWatch to external systems or use log agents to send directly to other systems to avoid high costs for log analysis on CloudWatch. So, while CloudWatch is convenient, at large scale the costs can add up, similar to other SaaS tools.

AWS CloudWatch is a solid choice for organizations primarily on AWS who need a straightforward way to monitor their cloud infrastructure and do not want to maintain a separate monitoring stack. Its deep integration with AWS is both its strength and its weakness – great if you are on AWS, not useful if you are not. It may not have all the bells and whistles (like advanced AI analytics or extensive custom integration library) that third-party tools have, but it covers the basics well [9]. In many cases, teams use CloudWatch in conjunction with other tools: for example, using CloudWatch Alarms for auto-scaling triggers and basic health, but exporting metrics to Prometheus or Datadog for more complex analysis and dashboards; or using CloudWatch for AWS service metrics but using New Relic for application APM. The choice often comes down to how hybrid the environment is and whether the organization is willing to invest in third-party monitoring or prefers to stick with AWS native tools. One source from a DevOps forum humorously advised, "don't rely on CloudWatch for anything outside AWS, and even within, some prefer dedicated tools", highlighting that CloudWatch's scope is somewhat narrow compared to specialized solutions [3]. Nonetheless, for AWS-centric web applications, CloudWatch provides a reliable foundation for monitoring and logging that requires minimal effort to get started.

ELK Stack (Elasticsearch, Logstash, Kibana) for Logging and Analytics

The ELK stack, now often called the Elastic Stack, refers to Elasticsearch, Logstash, and Kibana – three open-source projects (by Elastic NV) that together form a popular solution for log management and analytics in web environments. Often Beats (lightweight data shippers) are included as well (making it "ELKB" or simply "Elastic Stack"). This stack is widely used for centralized logging – aggregating logs from various sources, storing and indexing them, and providing search and visualization capabilities (Table 4).

Table 4. ELK Stack (Elasticsearch, Logstash, Kibana) for Logging and Analytics (compiled by the author based on his own research)

Category	**ELK Stack**
Architecture	Elasticsearch: Distributed search/indexing engine for logs. Logstash/Beats: Ingest, parse, transform logs; lightweight log collectors (Filebeat, Metricbeat). Kibana: UI for visualization, search, dashboards.
Use Cases	Centralized logging for web/server environments; log analytics; troubleshooting and trend analysis; security (SIEM); business analytics. Complements metrics-focused tools.
Advantages	Highly customizable; powerful search capabilities; scalable (handles large log volumes); open-source flexibility; rich plugin/community ecosystem; independent component scaling.
Integrations	Extensive integrations via Logstash and Beats (Nginx, Apache, Kubernetes, Docker, AWS); supports external visualization tools (Grafana); pre-built modules simplify setup.
Strengths	Fast, interactive log analysis; granular data control (retention, schemas); robust community and support ecosystem; extends beyond logging (APM, security).
Limitations	Operational complexity and maintenance overhead; expert knowledge required for scaling Elasticsearch clusters; potential for high storage costs without enforced retention policies; weaker in metric-alerting compared to specialized tools; steep learning curve for Kibana UI.

In comparisons, it's noted that ELK is flexible but requires more management, whereas Datadog is easier but costly. Adservio (2023) compared Datadog vs ELK and pointed out that while ELK is free to use (open-source), the effort to implement and maintain it, plus possibly needing to pay for third-party support or hosting, is a consideration [1]. They also mention ELK's high availability – being distributed, it can be made fault-tolerant, though that again is up to the implementer to configure (replication, multiple nodes, etc.).

Resource intensive – running an ELK cluster for large environments can require a lot of RAM and disk. Each log entry is indexed and stored, which can triple or more the raw data size once in Elasticsearch (index + raw + replicas). So hardware needs can be significant.

The ELK stack remains a cornerstone for logging in many web setups due to its proven capability to ingest and search logs effectively. It gives organizations full control over their log data and analytics. Its advantages are most realized when you have the capacity to manage it or when custom requirements (like very specific parsing, self-hosting due to compliance, or integrating logs with other custom data in Elasticsearch) are present. For companies heavily invested in open-source or needing on-premises solutions, ELK is often the go-to choice (versus sending data out to a third-party cloud). Elastic has also been evolving ELK into a more complete observability stack (with APM and uptime monitoring solutions, etc.), but in this article focus on its core logging strength.

A summary in a dev guide listed it as “open-source, flexible, highly customizable” with the cons being “requires setup and management, can be complex to configure and resource-intensive” [5]. This aligns with this analysis. Many teams pair ELK for logs with Prometheus/Grafana for metrics to cover both bases using open-source tools. Others might use ELK solely for logs and use a service for metrics. The decision often hinges on in-house expertise and the value of customization versus convenience.

Other Notable Tools (Splunk, Grafana Loki, etc.)

Beyond the primary tools requested (Prometheus, Grafana, Datadog, CloudWatch, ELK, New Relic), it’s worth briefly noting a couple of other popular monitoring/logging solutions and where they fit (Table 5):

Table 5. Comparison of Alternative Log Management and Monitoring Solutions (compiled by the author based on his own research)

Solution	**Pros & Cons**
Splunk	Commercial log management; powerful search (SPL), robust analytics, extensive enterprise features; popular for SIEM/security logging. High cost, steep learning curve, complex yet scalable architecture.
Grafana Loki	Open-source, lightweight logging system; minimal indexing (labels only), cost-effective storage; integrates seamlessly with Grafana via LogQL. Easier setup but less powerful queries compared to Elasticsearch.
Graylog	Open-source log management based on Elasticsearch; streamlined setup and simplified UI tailored specifically for logs. Enterprise support available; less complex than ELK but still flexible.
Dynatrace, AppDynamics	SaaS APM platforms comparable to New Relic; Dynatrace has strong AI-driven tracing and low configuration overhead; AppDynamics (Cisco) emphasizes enterprise-grade application monitoring.
Prometheus + Alertmanager vs. Alerting Services	Prometheus Alertmanager provides alerting integrated with Prometheus metrics. Alternative built-in solutions exist (Datadog, New Relic alerts, AWS CloudWatch Alarms). External incident-response integrations (PagerDuty, Grafana Cloud Alerting) commonly used.

Given the question scope, the above tools (Prometheus, Grafana, Datadog, CloudWatch, ELK, New Relic) cover a broad range of the monitoring/logging spectrum.

It is common to use multiple in tandem: for instance, use Prometheus+Grafana for infrastructure metrics, ELK for logs, and maybe New Relic for deep application performance on critical services. However, there is a trend towards unification (the so-called “observability” trend) where vendors push using one platform for all telemetry. Datadog and New Relic exemplify that approach (metrics, logs, traces in one). Elastic is also trying to do that within the open-source model (adding

APM and metrics to ELK). Grafana is integrating metrics, logs, and traces (offering Grafana for visualization, Prometheus for metrics, Loki for logs, Tempo for tracing).

Thus, when choosing tools, teams must consider factors like: existing ecosystem fit (e.g., if heavily AWS, CloudWatch might suffice or at least be the starting point), cost vs. control, scale of data, team expertise, and specific feature needs (like need for strong log analysis = ELK/Splunk, need for quick deploy APM = New Relic, etc.).

Conclusion

Modern web applications require robust monitoring and logging to ensure performance, reliability, and security. In this comparative analysis, examined a range of popular tools – from open-source stacks like Prometheus/Grafana and ELK, to cloud-native services like AWS CloudWatch, to integrated SaaS platforms such as Datadog and New Relic. Each tool brings a unique approach: Prometheus excels at real-time metrics and alerting with a cloud-native, pull-based design; Grafana provides a flexible, technology-agnostic visualization layer; the ELK stack offers powerful log aggregation and search with complete customizability; CloudWatch integrates seamlessly with AWS environments; Datadog delivers an all-in-one managed solution with extensive integrations and minimal setup; and New Relic specializes in in-depth application performance monitoring with a developer-friendly lens.

There is no one-size-fits-all winner – the "best" choice depends on an organization's priorities. If control over data and open-source flexibility are most important, a combination of Prometheus (for metrics) and ELK (for logs) backed by Grafana might be ideal, provided the team can invest the effort to manage it. If ease of deployment and breadth of features are paramount (and budget allows), Datadog or New Relic can quickly provide comprehensive observability across an application's stack with little maintenance. AWS-centric teams might leverage CloudWatch as a cost-effective starting point, augmenting it with other tools as needed to cover gaps (for example, using Kibana for more advanced log analysis or Prometheus for container metrics).

This analysis highlighted that these tools often complement rather than completely replace one another. In practice, many organizations integrate multiple solutions to play to each's strengths – for instance, using Grafana to unify views of Prometheus metrics and ELK logs side by side. What's crucial is establishing a system of record for each type of data: a reliable source for metrics, another for logs, and ensuring they can be correlated (either via a unified platform or good linking practices). All the reviewed tools support such correlations to varying degrees, and emerging standards like OpenTelemetry are making it easier to tie metrics, logs, and traces together across different platforms.

In summary, the landscape of monitoring and logging tools in web environments can be viewed along a spectrum of control vs. convenience: on one end, open-source/self-hosted solutions offer maximum control and integration at the cost of manual effort (Prometheus, Grafana, ELK), and on the other end, managed services offer convenience and unified functionality at a monetary cost (Datadog, New Relic, etc.). AWS CloudWatch lies somewhere in between, tailored for AWS users with moderate needs. Architecture: Each tool's architecture (whether it's Prometheus's single-node TSDB or Datadog's cloud-based data pipeline) influences its best use cases. Scenarios of use: Prometheus+Grafana shine in cloud-native, Kubernetes-heavy scenarios; ELK/Splunk shine when log analysis depth is needed; Datadog/New Relic shine when a fast, turnkey solution is needed across a diverse infrastructure; CloudWatch shines for AWS-only, simpler setups. Advantages and limitations: were discussed in detail for each, and summarized in comparative tables and figures (such as integration support and pros/cons).

Ultimately, an effective monitoring and logging strategy may involve leveraging multiple tools and integrating them – for example, using Datadog for high-level monitoring but exporting logs to an ELK stack for long-term audit requirements, or using Prometheus for core metrics but feeding critical alerts into CloudWatch or an ITSM system. The key for engineering teams is to ensure that whichever combination is chosen, it meets the reliability needs (does not miss critical issues) and is adopted by the team (usability). When done right, these tools provide invaluable visibility: metrics to quantify system health, logs to explain the detailed events, and traces to map user transactions – all contributing to faster debugging, proactive performance tuning, and robust operations.

In conclusion, the system analysis of these monitoring and logging tools shows that while their feature sets overlap, each has particular strengths. A comparative summary might note: Prometheus/Grafana for customizable, low-cost metrics with strong community support; ELK stack for powerful, self-hosted log analytics; Datadog for a comprehensive, low-maintenance observability platform; New Relic for deep APM insights and full-stack visibility; AWS CloudWatch for native cloud integration and automation hooks; and Splunk/others for enterprise-grade log and security analytics. Organizations should assess their technical requirements, team capabilities, and budget to select the tool or combination of tools that best align with their monitoring goals. With the right toolset in place, teams can achieve end-to-end observability of their web applications – gaining the ability to detect issues early, pinpoint root causes quickly, and ultimately deliver more reliable and high-performance services to their users.

References

1. Adservio. Datadog vs. ELK [Electronic resource]. – 2023. – URL: https://www.adservio.fr/post/datadog-vs-elk (date of access: 03.04.2025).
2. Better Stack Team. Datadog vs. Prometheus: a side-by-side comparison for 2025 [Electronic resource]. – 2025. – URL: https://betterstack.com/community/comparisons/datadog-vs-prometheus/ (date of access: 03.04.2025).
3. Blackden C. Compare Datadog vs. New Relic for IT monitoring in 2024 / TechTarget [Electronic resource]. – 2024. – URL: https://www.techtarget.com/searchaws/tip/Compare-CloudWatch-vs-Datadog-and-New-Relic-for-AWS-monitoring (accessed: 03.04.2025).
4. DevOpsCube. Learn Prometheus Architecture: A Complete Guide [Electronic resource]. – 2023. – URL: https://devopscube.com/prometheus-architecture/ (accessed: 03.04.2025).
5. Hamilton G. The Ultimate Guide to ELK Log Analysis [Electronic resource]. – 2023. – URL: https://www.chaossearch.io/blog/ultimate-guide-elk-log-analysis (accessed: 03.04.2025).
6. Hu E. Prometheus vs. Datadog: A Complete Comparison Guide for 2025 / Indie Hackers [Electronic resource]. – 2025. – URL: https://www.indiehackers.com/post/prometheus-vs-datadog-a-complete-comparison-guide-for-2025-e3765fef19 (accessed: 04/03/2025).
7. Sematext Team. Datadog vs. New Relic: Comparison and Key Features [Electronic resource]. – 2024. – URL: https://sematext.com/blog/datadog-vs-new-relic/ (accessed: 04/03/2025).
8. SigNoz. Datadog vs CloudWatch – Key Differences [Electronic resource]. – 2022. – URL: https://signoz.io/blog/datadog-vs-cloudwatch/ (date of access: 03.04.2025).
9. Strapi. Comprehensive Guide to Top Monitoring and Logging Services [Electronic resource]. – 2023. – URL: https://strapi.io/blog/comprehensive-guide-to-top-monitoring-and-logging-services (date of access: 03.04.2025).
10. Veeramachaneni G., Edwards C. Our commitment to OpenTelemetry / Prometheus Authors [Electronic resource]. – 2024. – URL: https://prometheus.io/blog/2024/03/14/commitment-to-opentelemetry/ (date of access: 03.04.2025).
11. Hijaguna Darshan. Install Prometheus and Grafana on Kubernetes using Helm // K21Academy. – 2023. – September 18. – URL: https://k21academy.com/docker-kubernetes/prometheus-grafana-monitoring/ (date of access: 19.04.2025). – Text: electronic.