Journal of
Computer Science and Software Development

Research Article Open Access

Case Studies and Best Practices from Leading Companies for Monitoring API Endpoint

Akshay Chandrachood*

Software Engineer, Frankfort, Kentucky, USA

***Corresponding Author:** Akshay Chandrachood, Software Engineer, Frankfort, Kentucky, USA, E-mail: akshay.chandrachood@gmail.com

Received Date: May 09, 2024 **Accepted Date:** June 09, 2024 **Published Date:** June 12, 2024

Citation: Akshay Chandrachood (2024) Case Studies and Best Practices from Leading Companies for Monitoring API Endpoint. J Comput Sci Software Dev 3: 1-9

Abstract

Tracking the API endpoints is now an integral part of modern applications' stability and performance checks. This paper analyzes case studies and best practices of successful companies across different sectors, and, therefore, readers will discover how monitoring strategies work. Crucial elements like monitoring, alerting, logging, timely updates, quick incident resolution, and instant messaging are given priority. By using these best practices and investing in advanced monitoring tools, organizations can improve the quality of their API monitoring and provide the apps with high reliability and practicality. As APIs are a core part of the architecture, endpoint monitoring is still needed to keep the systems stable, performance at high levels, and users satisfied.

Keywords: API; Monitoring Strategies; Best Practices; Timely Updates; Quick Incident Resolution; Alerting; Advanced Monitoring Tools; Reliability; Maintainability; User Satisfaction

Introduction

In the modern digital world, where app development is a rapid race, APIs (Application Programming Interfaces) have become the structural elements of widely accepted applications. It facilitates smooth communication and data transfer between many software modules, enabling businesses to develop advanced and integrated solutions. While APIs are increasingly reliant, reliability and performance assurance remain top concerns for business organizations. Even a brief disruption in API access can lead to significant financial losses, underscoring the critical importance of API endpoint monitoring. Today, API endpoint monitoring has become a critical factor that enables us to preemptively identify and solve problems and ensure unstoppable services and superb user experiences. This paper discusses case studies and best approaches from major API endpoint monitoring companies that have done it successfully. Seeking justification in our approaches, lessons learned, and success stories, we strive to enable organizations with helpful tips and usable recommendations as they look to expand their monitoring processes for APIs**.**

Case Studies

Case study 1: Netflix - Resilience through Comprehensive API Monitoring

Netflix, one of the global streaming giants that has a strong and scalable platform with which you can interact through an application programming interface. With millions of subscribers across the globe and new content being added to their API endpoints continuously, the integrity and performance of these endpoints are of utmost importance [1]. Netflix has developed a multi-layered API monitoring plan based on proactive problem detection and quick problem-solving methods.

A critical factor in Netflix's measurement strategy is its use of real-time metrics and dashboards. They can get specific information about API response time, error rates, and throughput, which helps them immediately spot problems about real-time bottlenecks and performance degradation. Alerts and thresholds will be shaped so that the engineering team at Netflix will get early notification when there are deviations from standard behavior or anomalies [2].

Netflix employs a sophisticated suite of tools and technologies to monitor its API endpoints, ensuring high performance and reliability in its distributed systems. A cornerstone of Netflix's monitoring strategy is Hystrix, a library that provides latency and fault tolerance by isolating points of access to remote systems and services, thereby preventing cascading failures and enabling resilience [3]. For real-time telemetry, Netflix uses Atlas, an in-house tool that collects and analyzes operational data, offering detailed insights through real-time metrics [2]. Additionally, Zuul, Netflix's gateway service, plays a pivotal role by managing API traffic with features like dynamic routing, monitoring, resiliency, and security [4]. Together, these tools create a robust monitoring ecosystem that supports Netflix's complex infrastructure, providing real-time analytics, fault isolation, and efficient API traffic management to maintain optimal performance and reliability of its API endpoints.

Moreover, Netflix utilizes top-notch chaos engineering techniques to test the capability of their API endpoints [5]. They help Netflix's engineering team identify the source of performance issues. This detailed knowledge empowers the engineers at Netflix to fine-tune their APIs, reduce latency, and enhance overall system effectiveness. The organizations use a method of simulating errors and disruptions, which they manage in a controlled environment, to observe how the system reacts and improves. This quality assessment will help them identify vulnerabilities, improve security, and ensure a consistent user experience when unexpected issues arise [5].

Lessons learned from Netflix's API monitoring practices include the importance of immediate insight, testing, and a culture of continuous improvement [6]. Organizations can deliver API-driven applications efficiently and effectively by investing in maintenance solutions and establishing a good mindset.

Case Study 2: Stripe - Ensuring Payment API Reliability

Stripe, a renowned payment processing platform, largely relies on its API endpoints to ensure smooth finan-

cial flows for multifarious enterprises around the globe. Stripe's mission is to facilitate seamless payment transactions using API tools to safeguard the security and functionality of the payment process. They track the process of API requests and responses. By testing the payment API, Stripe can detect vulnerabilities before they reach real customers. They also allow them to diagnose and resolve problems, prevent outages, and keep systems running smoothly. Tools like Stripe Radar, Stripe Issuing, Stripe Sigma, and Stripe Analytics provide information about enforcement policies and track demand for different services and products. This is important because they help Stripe's engineering team identify the source of the performance issues. This microscopic knowledge empowers the engineers at Stripe to fine--tune their APIs, reduce latency, and enhance overall system effectiveness [7]. It is anticipated that 33% of companies that adopt payment APIs will experience a revenue boost of 10%, indicating a significant enhancement in their business activities [8].

Insights from Stripes' experience with API monitoring are critical. This helps implement correct test measures, comprehensive visibility, and ethical monitoring practices. Therefore, organizations need to adopt mechanisms that are effective and reliable in their payment systems [9]. This can be possible through creating virtual environments and gaining expertise in API performance [7]. This will, in turn, raise consumers' confidence and trust in the organization.

Case Study 3: Cerner – Ensuring Better Observability

Cerner Corporation, based in North Kansas City, Missouri, is a leading health information technology company. Specializing in electronic health records (EHR) and related technologies, Cerner's systems are widely adopted by hospitals and clinics globally. The company promotes seamless data exchange through standards, network connections, and nationwide exchanges, enhancing clinicians' insights into patient histories for better treatment outcomes. Cerner's interoperability solutions address medical information silos by integrating various data sources using FHIR®, HL7, and Cerner's proprietary API for efficient healthcare data operations [11][12].

Cerner Corporation employs several advanced tools and technologies for monitoring their APIs, particularly within their Cerner Millennium system. The primary tools they use include:

Goliath Technologies**:** Cerner uses Goliath's Application Monitoring for comprehensive performance tracking. This tool provides end-to-end visibility into the performance of Cerner Millennium, enabling monitoring through all application layers such as Citrix and third-party applications. Goliath's platform is designed to test performance from the end user's perspective, tracking tasks like login times, patient record access, billing processes, and lab test submissions. This helps in proactively identifying and resolving performance issues before they impact users [13].

eG Innovations**:** eG Enterprise is another key tool used by Cerner for IT performance monitoring. It offers unified monitoring for Docker containers and Kubernetes environments, which are integral to Cerner's infrastructure. This tool provides code-level visibility into applications, tracks resource usage, and identifies performance bottlenecks within the containerized environments. It integrates seamlessly with Kubernetes APIs to monitor the entire Kubernetes cluster without requiring agents, ensuring efficient and comprehensive monitoring [14].

The interface with eG Innovations, as depicted in Figure 1, includes a metric for page load time, demonstrating the detailed monitoring capabilities of the eG tool. Although specific information about API response time is not available on their website, it is likely that this metric is included within the tool itself but not shown in the sample images provided online. This assumption is based on the comprehensive nature of the monitoring features typically offered by eG Innovations.

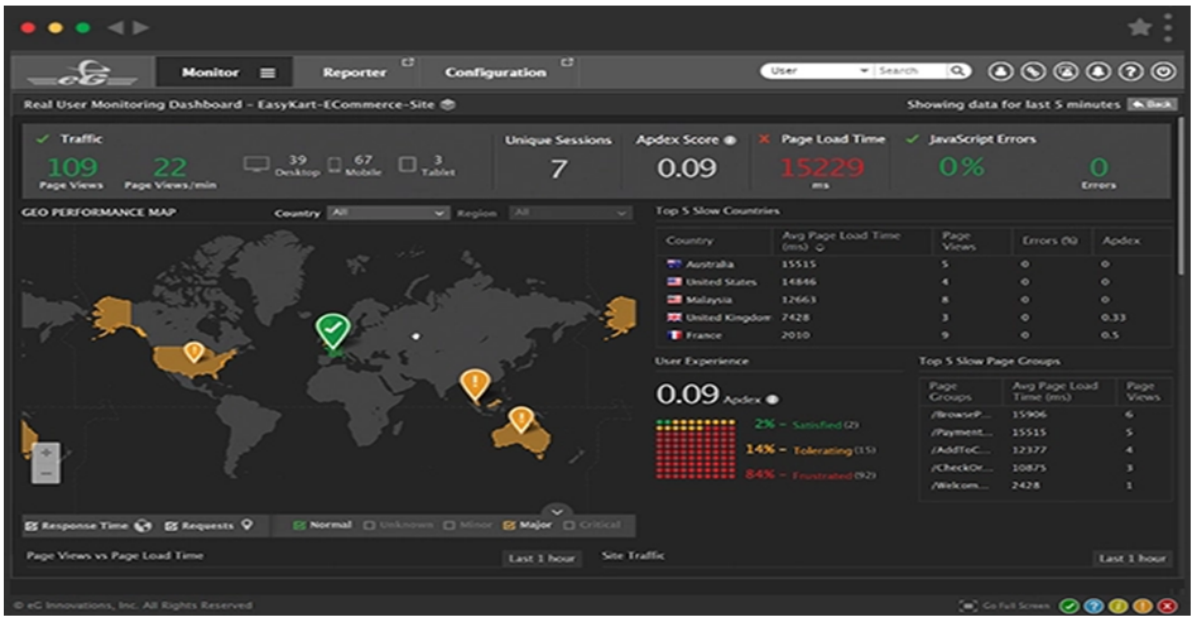

Figure 1: Cerner resource monitoring to user experience monitoring for better observability [15].

These technologies enable Cerner to maintain high availability and performance of their APIs, ensuring reliable and efficient service delivery for their healthcare solutions.

From Cerner's approach to API monitoring, several key learnings emerge. Firstly, we cannot overstate the importance of real-time monitoring. Having the ability to detect and address issues as they occur minimizes downtime and maintains the reliability of healthcare services. Secondly, the integration of automated alerts and detailed logging provides a comprehensive understanding of API performance, enabling swift troubleshooting and resolution. This serves as a model for other organizations aiming to achieve high standards of performance and reliability in API management.

Case Study 4: Twilio - Ensuring Communication API Availability

Twilio is a cloud communications platform that enables businesses to integrate voice, messaging, and video capabilities into their applications through APIs. With many customers depending on Twilio's end-to-end communication services, maintaining the quality and performance of these services is essential for customer satisfaction and operational success. To ensure this, Twilio employs various methods to monitor the health and functionality of its API endpoints. These methods include operational and analytical techniques to assess the performance of the APIs.

Twilio utilizes various Application Performance Monitoring tools for real-time monitoring of their API endpoints. These tools allow engineers to continuously track various performance metrics such as response time, error rates, and availability of API endpoints. With such tools, Twilio engineers gain detailed insights into the health and performance of their APIs through customizable dashboards and alerting mechanisms. This enables engineers to promptly identify and address any anomalies or performance degradation, ensuring the reliability and responsiveness of API services.

While Twilio's documentation doesn't explicitly state their use of specific monitoring tools like New Relic or Datalog, they do emphasize the importance of monitoring API performance [16]. These tools offer comprehensive monitoring capabilities, including real-time tracking of performance metrics, customizable dashboards, and alerting mechanisms, aligning closely with the requirements for monitoring Twilio's API endpoints effectively.

In addition to active monitoring, Twilio also engages and recommends in passive monitoring by analyzing real-time data generated by actual production activity. By examining API logs, metrics, and traces, Twilio gains insights

into the exact performance and behavior of the APIs under real-world conditions. This data-driven approach enables Twilio to detect patterns, identify errors, and refine its APIs based on authentic user interaction patterns. Twilio recommends same thing to their customers for better monitoring practices whoever want to use Twilio APIs.

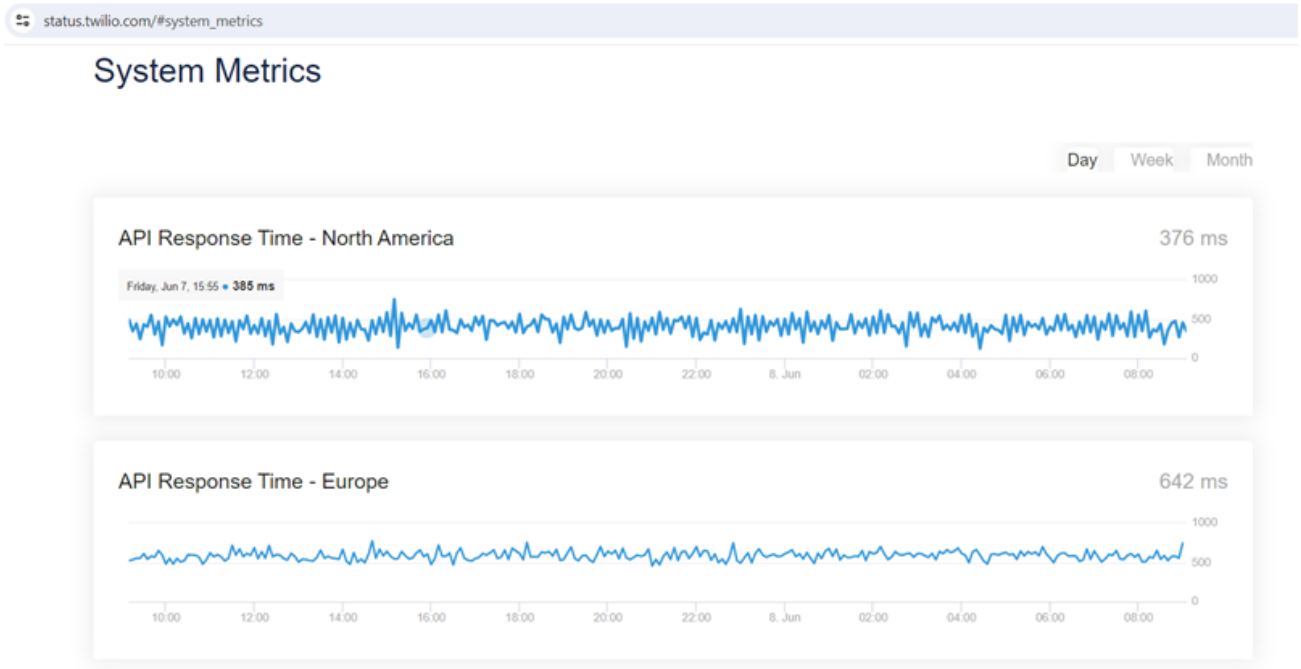

Figure 2: API response times in different regions [17].

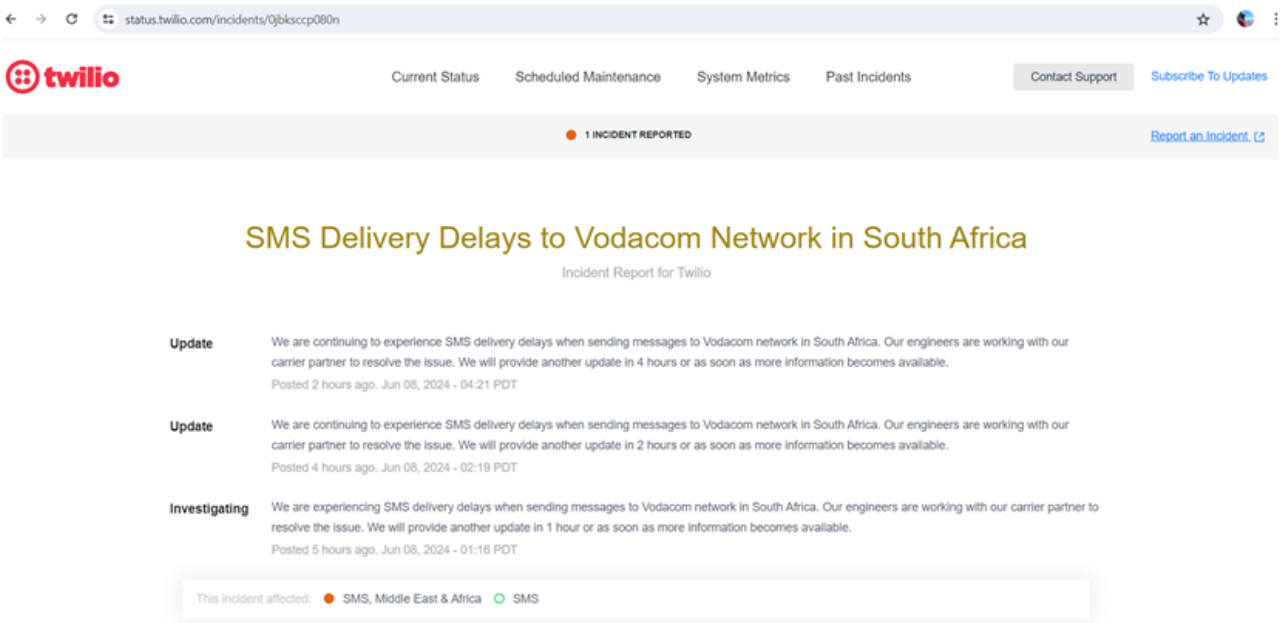

Figure 3: incidents reported by customers from different parts of the world [18].

Clear and transparent communication with customers is a key focus for Twilio. Through open channels of communication, Twilio ensures that any investigation-related issues are quickly addressed, clear investigation procedures are in place, rapid remediation procedures are implemented in the event of a service outage and information is

provided regular updates to customers, reinforcing trust, and reducing any potential impact on it their experience. Twilio maintains a real-time API status page [16] that provides complete insight into the status and health of all their services. To ensure transparency for users, this page is constantly updated to reflect any bugs or performance issues. The public availability of the Twilio API Status page underscores the company's commitment to better familiarize customers with the Twilio APIs they use. This sophisticated monitoring tool shows incidents reported by other customers, as well as timestamps, regions, and API response times. Figure 2 represents API response times in regions around the world [17] and Figure 3 shows incidents reported by customers [18]. Such information gives customers greater transparency. Also, the Twilio status page has a dedicated section that notifies users if a service is experiencing a bottleneck or going down. Improving customer transparency increases customer loyalty.

The key takeaway from Twilio's case study is the importance of closely monitoring the most critical aspects of communication services. By enhancing both active and passive monitoring capabilities and maintaining robust API monitoring practices, Twilio has significantly improved its service reliability. Other organizations can learn from Twilio's experience and adopt similar best practices to optimize the performance of their APIs. Ensuring effective incident management and fostering open communication with customers are essential strategies for building trust and mitigating the effects of potential service issues.

Case Study: Zillow Group's Real Estate Domain

Zillow Group is a leading real estate and rental marketplace that aims to empower consumers with comprehensive data, inspiration, and knowledge about their homes. Zillow connects users with real estate professionals and services, offering an extensive database of homes for sale and rent, along with detailed property information and market analytics. Zillow uses APIs to aggregate and disseminate vast amounts of real estate data from multiple sources, ensuring users have access to the latest information [19]. These APIs enable developers to integrate Zillow's extensive property listings and data into their own applications, facilitating seamless user experiences and extending Zillow's reach across various platforms and services.

Monitoring API performance at Zillow involves tackling challenges related to data volume, real-time updates, and maintaining data accuracy. Given the high volume of data transactions and the need for up-to-date information, ensuring optimal response times and system reliability is critical. Zillow employs comprehensive monitoring strategies, including sophisticated logging and alerting systems, to track API traffic, error rates, and performance metrics. By utilizing machine learning algorithms, Zillow can predict and mitigate potential issues before they impact users, ensuring the reliability and efficiency of their APIs.

Zillow leverages a range of tools and technologies to monitor API performance. Solutions such as AWS CloudWatch, Datadog, and Prometheus are integral to their monitoring infrastructure [20]. These tools provide real-time analytics, anomaly detection, and automated alerting, enabling Zillow to maintain high standards of API performance and reliability. The use of these technologies allows Zillow to track key performance indicators, swiftly diagnose issues, and implement solutions to optimize their APIs. Additionally, cloud-based monitoring solutions enable Zillow to scale their monitoring capabilities to align with the growth of their platform and user base.

Zillow's approach to API monitoring offers several valuable insights. Integrating advanced analytics and machine learning into the monitoring process enhances the ability to predict and prevent issues, ensuring a seamless user experience. Comprehensive logging and real-time alerting systems facilitate swift detection and resolution of problems, minimizing downtime and maintaining service quality. Utilizing cloud-based monitoring tools allows Zillow to scale their operations efficiently and adapt to increasing demands. These practices underscore the importance of a proactive and scalable approach to API monitoring in maintaining the reliability and performance of digital platforms.

Best Practices for API Endpoint Monitoring

Based on the case studies and insights from leading companies, several best practices emerge for implementing effective API endpoint monitoring:

1. By identifying key metrics such as error rates, response time and latency and establishing goals for API monitoring, organizations can prioritize the most critical aspects of API performance, thus contributing to overall success [10].

2. Conduct periodic system and process audit to determine number of incidents that could have been resolved with the implementation of API monitoring tools and process. This audit can be done even if such processes are implemented to identify its effectiveness with the mindset of continuous improvement.

3. Implement monitoring solutions to gain insight into crucial factors affecting API health and performance, such as response time, latency, error rates, and usage patterns.

4. Set up alerts and notifications to promptly detect and respond to any issues or anomalies in API performance, ensuring rapid troubleshooting and resolution.

5. Regularly simulate realistic API usage scenarios to proactively identify potential issues and ensure the reliability of critical functionalities [21]. This involves creating and executing test cases that mimic actual usage patterns to catch problems before they impact real users.

6. Analyze data from the production traffic to gain insight into the performance of API. This can be achieved through identifying data patterns, detecting errors, and making data-driven optimizations to improve overall performance and reliability.

7. Provide timely and transparent customer communication during incidents or disruptions [7]. This can be achieved through dedicated channels to share updates, status reports, and time estimates to resolve issues, thereby building trust and minimizing frustration among users.

8. Evaluate and update API monitoring standards periodically. This ongoing process is crucial for improving overall performance and maintaining organizational competitiveness.

9. Define transparent processes and responsibilities for handling API-related incidents [7]. Ensure rapid investigation, diagnosis, and resolution of issues to minimize customer impact.

Conclusion

For an organization to thrive and maintain the stability of its performance, it is crucial to utilize and monitor API endpoints. The case studies of leading companies have shown the efficiency of API monitoring in practice. These practices, not only they enable the overall system monitoring but also, they significantly boost system availability, reliability and maintainability which is really a success determinant in contemporary API-driven world [22]. Therefore, it demonstrates the advantages of detecting and resolving issues through monitoring, detailed logging, and immediate analysis of actual production data. It further highlights key elements of incident management and transparent communication. This is vital since it maintains and keeps trust among customers, thereby minimizing any factor that might cause failure. Therefore, organizations must engage in mechanisms based on the lessons learned from these leading companies. This will boost their API monitoring and deliver reliable applications. To succeed in an API-driven digital environment, organizations should venture into practical solutions to develop a culture of continuous improvement.

References

1. Gough J, Bryant D, Auburn M (2021) Mastering API Architecture. " O'Reilly Media, Inc.".

2. Netflix Tech Blog on Atlas: https://netflixtechblog.com/introducing-atlas-netflixs-primary-telemetry-platform-bd31f4d8ed9a

3. Netflix Tech Blog on Hystrix: https://netflixtechblog.com/introducing-hystrix-for-resilience-engineering-13531c1ab362

4. Netflix Tech Blog on Open sourcing Zuul 2: https://netflixtechblog.com/open-sourcing-zuul-2-82ea476cb2b3

5. Kew A (n.d.) How to leverage chaos Engineering for API Gateway Optimization. Kong Inc. https://konghq.com/blog/engineering/optimize-your-api-gateway-with-chaos-engineering

6. Westerveld D (2021) API Testing and Development with Postman: A practical guide to creating, testing, and managing APIs for automated software testing. Packt Publishing Ltd.

7. Srivastava R (2021) Cloud Native Microservices with Spring and Kubernetes: Design and Build Modern Cloud Native Applications using Spring and Kubernetes (English Edition). BPB Publications.

8. Payment APIs: What they are & benefits for businesses | Stripe. (n.d.). Stripe.com. Retrieved April 20, 2024, from https://stripe.com/ae/resources/more/payment-application-program-interfaces-apis

9. Godefroid P, Lehmann D, Polishchuk M (2020) Differential regression testing for REST APIs. In Proceedings of the 29th ACM SIGSOFT International Symposium on Software Testing and Analysis, 312-23.

10. Koivula E (2023) Building Competitive Advantage with API Strategy–Case Study of Established Enterprises.

11. Pro-Tips to achieve a successful Cerner Community Works Implementation. (2019) https://blog.avaap.com/2019/04/18/pro-tips-for-achieving-a-successful-cerner-communityworks-implementation

12. Rhapsody: https://rhapsody.health/blog/interfacing-with-cerner-ehr/

13. Cerner Millennium Performance Monitoring. (2022) Goliath Technologies. https://goliathtechnologies.com/software/goliath-application-monitoring/cerner-millennium--monitoring/

14. Docker container monitoring (n.d.) eG Innovations. https://www.eginnovations.com/supported-technologies/docker-container-monitoring and Kubernetes monitoring tools. (2021) eG Innovations. https://www.eginnovations.com/supported-technologies/kubernetes-monitoring

15. Healthcare IT monitoring at Cerner (2021). https://www.eginnovations.com/supported-technologies/cerner

16. Monitor your application | Twilio (n.d.) https://www.twilio.com/docs/messaging/onboarding/monitor-your-application

17. Twilio status (n.d.) https://status.twilio.com/#system_metrics

18. SMS delivery delays to Vodacom network in South Africa (n.d.) https://status.twilio.com/incidents/0jbksccp080n

19. Daud F (2020) Building a Data Streaming Platform - How Zillow Sends Data to its Data Lake - Zillow Tech Hub. Zillow. https://www.zillow.com/tech/building-a-data-streaming-platform/

20. Zillow case study (n.d.) Amazon Web Services, Inc. https://aws.amazon.com/solutions/case-studies/zillow/

21. Liu Y, Li Y, Deng G, Liu Y, Wan R, et al. (2022) More: Model-based RESTful API testing with execution feedback. In Proceedings of the 44th International Conference on Software Engineering, 1406-17.

22. API Monitoring Metrics, Tips, and BestPractices (n.d.) Www.catchpoint.com. Retrieved 2024, from https://www.catchpoint.com/guide-to-synthetic-monitoring/api-monitoring