Spectrum of Engineering Sciences
ISSN (e) 3007-3138 (p) 3007-312X Volume 3, Issue 3, 2025

ZERO-DOWNTIME MASTERY: ADVANCED DEBUGGING AND MONITORING SOLUTIONS FOR SERVERLESS APPLICATIONS

Ahmad Mehmood[1], Muhammad Zulkifl Hasan[2], Muhammad Zunnurain Hussain[*3]

[1,*3]*Department of Computer Science, Bahria University Lahore, Pakistan.*
[2]*Department of Computer Science, University of Central Punjab Lahore, Pakistan.*

[1]ahmedmehmood832@gmail.com, [2]zulkifl.hasan@ucp.edu.pk,
[*3]zunnurain.bulc@bahria.edu.pk

DOI: https://doi.org/10.5281/zenodo.15542643

Keywords
Applications, Cloud Computing, Debugging, Distributed Systems, Monitoring, Serverless Computing.

Article History
Received on 20 January 2025
Accepted on 20 February 2025
Published on 27 February 2025

Corresponding Author: *

Abstract
Serverless computing is a popular concept that contributes heavily to the construction of highly cost-efficient applications while having the qualities of being massively scalable in parallel. When it comes to server less computing there are countless advantages that affirm server less computing. These include the ecosystem and system that serverless computing is built on which can be further described as a distributed system. The system portrays unique challenges in regard to the debugging and monitoring of these systems. As far as debugging is concerned, it is a task with extreme complexity as a result of the distributed nature of serverless computing as a system and the applications developed on this system with the addition of the processes being conducted at the back end of the system being hidden from the developers. This research paper entails the details regarding the debugging and monitoring of server-less systems as well as applications and studies the existing technology, methods, and tools which facilitate the debugging and monitoring of server-less applications. It also focuses on the features, strengths, and limitations of these systems and arrives at an effective method for the monitoring and debugging of the server less systems in cloud computing.
The debugging of serverless applications is essential for server less computing for a plethora of reasons. The topmost priority goes to the optimization and performance of server less applications. By doing so, not only does it prevent potential bottlenecks to present themselves but also increases the efficiency of the applications by optimizing the code which increases performance in terms of execution and resource usage. Following the debugging of the application, the monitoring of the serverless systems also played a huge role in the execution of the applications. It analyzes the mechanisms regarding the logging system which in turn gains insight into the behavior of the serverless architecture in terms of the functions, their outputs, and their intermediate stages.

INTRODUCTION

The current technological era sees the huge role played by server-less architectures, in the deployment of its applications and their development. In order for the developers to focus on writing efficient code and designing impeccable architecture which is not only cost-efficient but also scalable we need to separate the fundamental infrastructure of serverless

systems. By doing so it will also reduce the operational overhead. However, as time progress, newer cloud applications are built and the complexity linked to these projects and embedded deep since them increases massively all due to the distributed nature of the system which immensely hardens the debugging and monitoring processes. Upon extensive research, we conclude that the debugging and monitoring of these systems could be achieved efficiently with the help of tools that not only attain insights into the system but also help in other areas such as performance, behavior, and availability of the serverless ecosystem. Moreover, the monitoring of this system done in the most effective methods promises results in various aspects such as in the execution time of functions. the percentage of error rates, event triggers as well as resource utilization, and app performance and reliability.

Debugging refers to the process of isolating problems and identifying them as well as the methods that need to be adopted in order to resolve them and achieve an application free from bugs and errors. Due to the complex distributed computing nature of this system, debugging can prove to be a challenge due to dedicated infrastructure being absent. Another issue also arises from the complexity presented inside the event-driven systems. In order to tackle this multitude of problems it is absolutely pivotal to create an effective serverless debugging mechanism that can identify and correct errors which increase the performance substantially and maintain a healthy architecture.

The need for the monitoring of serverless applications increases as time goes on and if executed in an effective and efficient manner, monitoring could prove itself a crucial element of the system. It attributes to various advantages which can reduce the number of bottlenecks in the system which in turn allows the developers to identify these problems faster and allow them to optimize the application in order to provide the users of the applications which a smooth experience in terms of task performance and applications execution. Monitoring also provides means for the underutilized resources to be adjusted accordingly and provides crucial information regarding the user's behavior and usage patterns as well accelerates data-driven decisions. [19]

Each domain whether it is monitoring or debugging requires unique tools. These tools provide the developers with an accelerated method to hasten the monitoring and debugging of server-less systems. These tools include but are not limited to, monitoring dashboards in real-time, analysis and aggregation of logs, scaling alerts automatically, distributed tracing, and error tracking. In order to gain visibility into the applications and troubleshoot issues followed by the optimization of its code and performance of the system is the primary reason for their usage.

2. LITERATURE REVIEW

The article "Distributed Monitoring and Debugging of Serverless Applications" recently presents a way to monitor and debug serverless apps. To get precise performance and debugging data, it suggests a flexible architecture that makes tracing approaches distributed. It also examines the efficiency by discovering and evaluating faults in serverless applications, with a focus on the possibility that enhancing capability for monitoring and troubleshooting.[7] This scalable architecture for data debugging and improved speed is covered in this article.

The key subjects covered by Kosti, R., Verma, D., and Ranchal (n.d.) were Serverless Performance and Monitoring. [6] In this article, the challenges of performance monitoring in serverless computing are described along with potential solutions. It discusses characteristics unique to serverless, such as autoscaling and cold starts, impact performance monitoring. The authors look into a number of performances monitoring approaches, including resource consumption tracking, throughput analysis, and latency assessment, in order to ensure optimal performance in serverless applications.

The complexity that has been increased due to the wrong execution, implicated auxiliary services, software layer loss control, and several

other functions was discussed by Manner, J., Kolb, & Wirtz [5]. Manually reviewing log data is a time-consuming yet typical method. The identification, as well as resolution of serverless function failures, can be done by a semi-automated process. Due to data offered in the stored data, The steps of the concept's process improve log quality, offer test templates, and automatically identify unsuccessful executions. Ultimate templates result in more reliable functions, improved regression testing, and more test coverage.

Castro et al.'s Monitoring and Debugging in Serverless Computing, which focuses on the present situation and potential future approaches, [4] gives an overview of existing serverless monitoring and debugging practices. It discusses issues with the sophisticated event-driven architecture and restricted observability of serverless programs. The authors also provide potential improvements and future research directions to enhance the monitoring and debugging capabilities of serverless computing.

Performance evaluation and app monitoring are crucial tasks for developers of serverless applications. Application developers frequently ignore the performance constraints of the serverless functionality in favor of constructing their logic. There are several tools for the performance evaluation for monitoring and debugging explored by Benedict, S. [3] in this article. The benefits and drawbacks of the available performance analysis tools and performance measures are examined. In addition, several challenges are examined, highlighting the need for developing thorough performance metrics utilizing technologies from the Internet of Things in serverless applications. This document will be helpful to developers of serverless cloud apps or tools for performance analysis.

Dodd, P. S., and Ravishankar, C. V. done the creation and execution an accepted monitor and debug platform was presented by [2]. Because of software assistance, the monitor provides features for transparent observation and ongoing facilities across a real-time system with little to no unanticipated disturbance. Due to the flexibility of the monitor, it examines both superior operating systems- and definite events as well as subordinate ones. It offers a fresh strategy that delivers transparent observing with minimal cost for managing shared variable references. The monitor is made to assist with duties like tracking system performance, helping with scheduling tasks in real time, and troubleshooting applications in real-time.

Marinescu et al. [1] explore the issue along with problems that result from providing checking help tools for software monitoring and also analyze the parallelly distributed architecture. The model of architecture is layered so it incorporates a process-level formal event-action model. Additionally, it describes how this system was used to create the architecture of a layered model, and this study was motivated by the need to understand potential interactions between a monitored system and a monitoring system. To create effective software development tools and debugging tools, it is essential to understand the crucial concepts underlying the interaction across monitoring as well as monitored architecture.

3 Methodology

The effort put into this research article expands across different scholar articles which include and are not limited to articles from Google Scholar as well as other relevant platforms. Moreover, research articles from the ACM Digital Library and the IEE Explore were read thoroughly to understand the requirements and methods order to increase the efficiency of the debugging and monitoring systems. Furthermore, this research paper also shows the importance of debugging and monitoring in server less computing and how it could improve computational time and application performance.

4. Analysis

After a thorough analysis of the issue regarding the productivity of the debugging and monitoring of serverless computing more specifically on serverless applications, we proposed related solutions to the presented challenges by studying various researchers and scholars and the work they have done on this issue. By doing so we arrived at a comprehensive

understanding of the issue. As the stated problem is divided into two parts, i.e., debugging followed by monitoring, we first examined debugging followed by monitoring. After that, we moved toward the challenges that we face in serverless computing regarding debugging and monitoring.

Those challenges include the distributed nature of the system, cold start latency, limited visibility and control, debugging, and logging tools, lack of infrastructure control, scalability, and performance issues followed by the function composition and orchestration. These challenges were met by solutions such as defining relevant metrics and KPIs, Proactive Monitoring and Alerting Strategies, distributed Tracing, and performance and load testing and simulations. [21]

Furthermore, the best tool for the analysis of monitoring and debugging are Amazon CloudWatch, Datadog, New Relic, Lumigo, Dashbird and Thundra. [8]. These evaluations are based on functionality, usability, scalability, integration potential, customer service, and general user happiness.

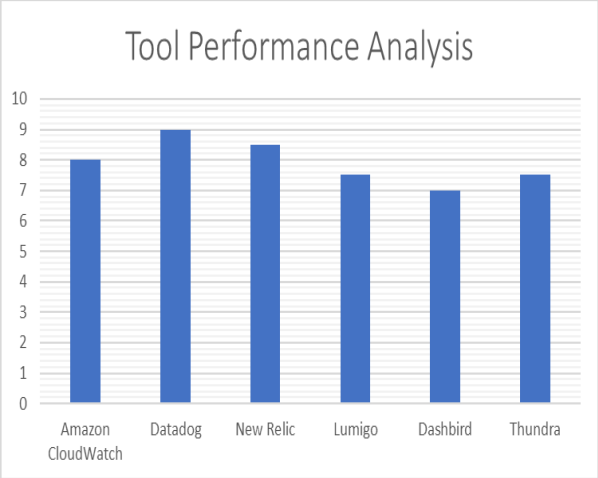


Fig 1: Debugging & Monitoring Tools Analysis

A. Amazon CloudWatch

The accuracy of Amazon CloudWatch's monitoring of AWS resources and services is well established. It offers inthe-moment monitoring and records thorough metrics, logs, and traces. It enables dependable and accurate monitoring within the AWS environment and connects seamlessly with other AWS services because it is a native AWS service. [9]

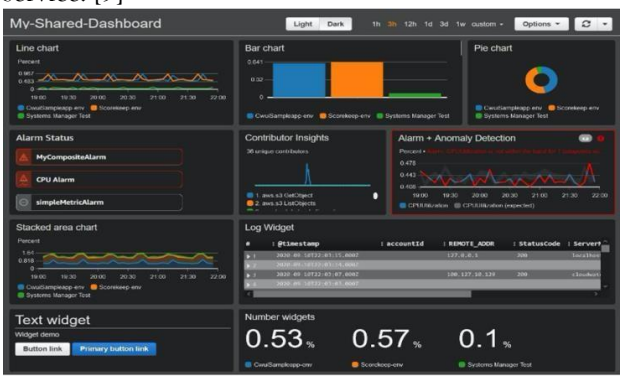


Fig 2: AWS CloudWatch Dashboard [9]

B. DataDog

The precision and dependability of Datadog's monitoring and observability are well known. It gathers metrics, logs, and traces from a variety of sources, including serverless

Fig 3: Dashboard

platforms, and analyses them. A key factor in Datadog's accuracy in delivering insights on the performance of apps and infrastructure is its robust integration capabilities. [10] Fig 3: Data Dog Dashboard [15]

C. New Relic

The precise monitoring and observability capabilities of New Relic are well-known. Metrics, logs, and distributed tracing provide it comprehensive insights into application performance and user experience. New Relic's ability to accurately identify performance issues and provide actionable insights is facilitated by its sophisticated analytics and AIdriven capabilities. [11]

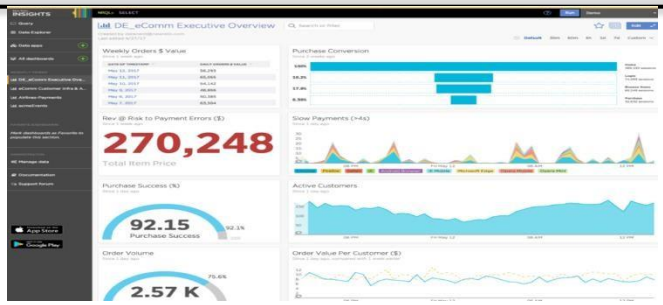


Fig 4: New Relic Dashboard [16]

D. Lumingo

A focused observability solution for serverless apps is called Lumigo. Even if precise accuracy data is difficult to get by, Lumigo concentrates on offering precise monitoring and debugging tools designed for serverless systems. To assist in locating performance problems and bottlenecks in serverless applications, it records and examines metrics, logs, and traces. [12]

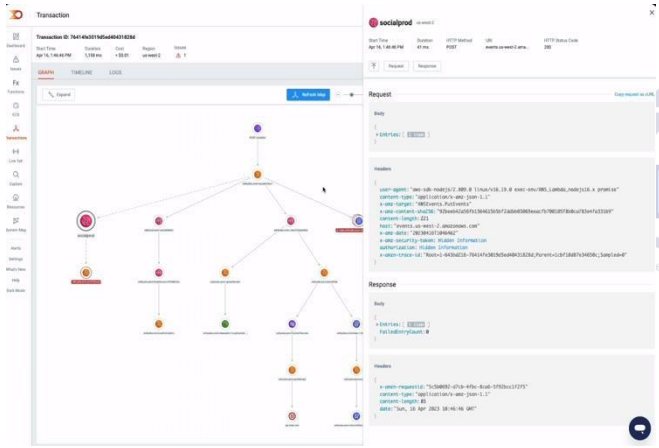


Fig 5: Lumingo Dashboard [12]

E. DashBird

In order to provide precise monitoring and observability for serverless applications, *Dashbird* was created. It especially gathers and examines metrics, logs, and traces for serverless services. Despite the lack of widely accessible precise accuracy information, *Dashbird's* emphasis on serverless monitoring implies a dedication to provide accurate insights and performance statistics. [13]

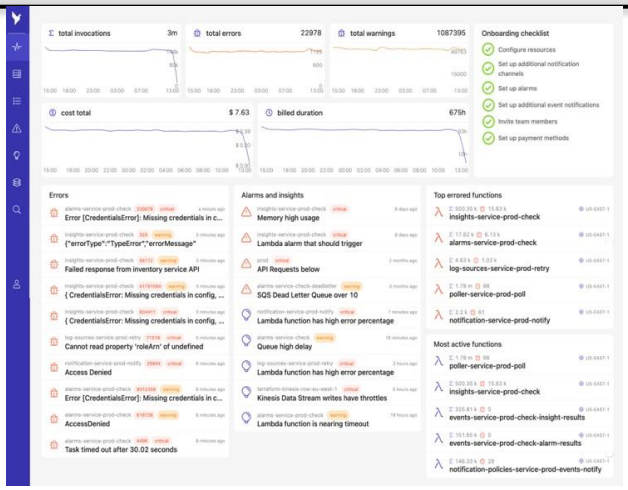


Fig 6: DashBird Dashboard [17]

F. Thundra

With an emphasis on performance insights, Thundra offers monitoring and observability for serverless apps. Thundra gathers and examines metrics, logs, and traces for serverless services even while precise accuracy data is not easily accessible. Its capabilities are designed to give precise visibility into the performance and troubleshooting of serverless applications. [14] [22]

5. Challenges

Serverless architecture creates several difficulties and problems with monitoring and debugging that must be resolved. The following are some of the main difficulties with serverless monitoring and debugging:

A. Distributed nature

Serverless architectures frequently include several modules or processes that communicate with one another and with other external services. It can be difficult to debug and monitor a distributed system since it requires recording and comparing events across several components. According to demand. A function that hasn't been called lately on its initial call may experience a cold start, which adds delay. To distinguish between normal delay and cold start.

Tool	Amazon CloudWatch	Datadog	New Relic	Lumigo	Dashbird
Metrics	Yes	Yes	Yes	Yes	Yes
Logs	Yes	Yes	Yes	Yes	Yes
Tracing	Yes	Yes	Yes	Yes	Yes
Distributed Tracing	No	Yes	Yes	Yes	Yes
Real-time Monitoring	Yes	Yes	Yes	Yes	Yes
Custom Dashboards	Yes	Yes	Yes	Yes	Yes
Integration	AWS Services	Wide range of integrations	Wide range of integrations	AWS Services	AWS Services
Pricing	Pay-per-usage	Pay-per-usage	Subscription-based	Pay-per-usage	Subscription-based
Alerting	Yes	Yes	Yes	Yes	Yes
Auto Instrumentation	Limited	Yes	Yes	Yes	Yes
Third-Party Integrations	Limited	Extensive	Extensive	Limited	Limited
Community Support	Active	Active	Active	Active	Active
Documentation	Comprehensive	Comprehensive	Comprehensive	Comprehensive	Comprehensive

Fig 7: DashBoard Comparison

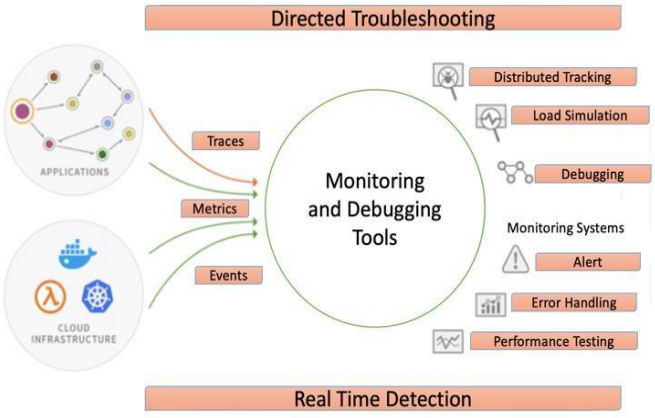
B. Cold Start Latency

Serverless solutions employ a pay-per-use business model, allowing functions to be dynamically launched and scaled latency in such cases, monitoring and debugging may call for extra care. [18]

Fig. 8 Monitoring and debugging tools

C. Limited Visibility And Control

The insight into the underlying infrastructure and runtime environment offered by serverless systems is frequently limited. Due to this lack of visibility, it may be difficult to locate and resolve platform-specific problems.



D. Debugging And Logging Tools

In serverless systems, traditional debugging methods might not be immediately relevant. Despite offering specialized logging and debugging tools, serverless platforms could have several drawbacks when compared to conventional debugging environments.

E. Lack Of Infrastructure Control

The underlying infrastructure is abstracted away with a serverless design, and developers have little influence over the execution environment. Because of this, it may be difficult to track down and troubleshoot serverless environment code.

F. Scalability And Performance Monitoring

Functions in serverless architectures may be automatically scaled to deal with changing demands. To discover possible bottlenecks or performance concerns, monitoring and debugging in such setups requires visibility into the performance and scalability of the functions.

G. Function Composition And Orchestration

Multiple functions are frequently combined or coordinated in serverless systems to manage complex processes. Such constructed functions need to be monitored and debugged in order to ensure proper execution, handling of failures, and data and event flow between the many functions.

6. Improvement Techniques

Determining pertinent KPIs and matrices, employing proactive waning and debugging tactics, developing efficient ways to handle errors, and other measures are the best approaches to decrease serverless monitoring and debugging. Utilizing tracing methods for the testing of performance, simulation of load, and issue fixing.

A. Establishing appropriate KPIs & metrices

The selection of pertinent metrics and KPIs is crucial for efficient serverless monitoring and troubleshooting. By establishing precise measurements and KPIs, you can evaluate the functionality, reliability, and performance of your serverless apps, spot issues, and then take informed decisions to improve their performance. Some suggestions for developing measurements and KPIs

in serverless settings are as follows: Create baseline metrics, provide actionable metrics, keep an eye on concurrency along with scalability, use certain metrics, collect exception stats along with errors, implement better tracing method that is distributed, and continually analyze and improve kpis.

B. Strategies for Timely Alert and Monitoring

Proactive tracking and alerting systems are essential for efficient serverless monitoring and debugging. By employing proactive strategies, you may identify and address issues before they have an impact on your application's usability or user experience. A combination of technical expertise, subject-matter comprehension, and ongoing team communication go into proactive monitoring and alerting. Setting up thorough monitoring, utilizing anomaly detection, creating relevant thresholds, using real-time alerts, and adopting intelligent alerting are some recommended practices.

C. Implementing Effective Login and Error Handling

The right logging and error handling practices must be implemented for serverless monitoring and debugging. By employing adequate logging and error handling, you can provide a great user experience, get insight into your serverless application's behavior, and fix issues. Join activities and logs from processes of serverless to use centralized logging method and a specified format. According to the importance, gravity one of the recorded data, use different log levels. Record both successful and failed activities to gain a complete picture of how your application behaves..

D. Leveraging Distributed Tracing for Debugging

To better understand how your application operates and performs, you may utilize distributed tracing to follow the flow of requests across several serverless processes and services. Set up your serverless activities and, when tracing, transcend service boundaries. Examining scattered traces, pay attention to important routes and outliers that deviate from the expected behavior. It is best to combine distributed tracing with other types of monitoring. Distributed tracing is often used by many teams, including the operations, development, and infrastructure teams.

E. Performance Testing and Load Simulation

You may use distributed tracing to trace the flow of requests across several serverless processes and services to gain a better understanding of how your application behaves and performs. Configure serverless activities and while tracking, transcend service barriers. Look for efficient methods for typical behaviour when evaluating a collection of traces. Tracking of distributing works best when combined with other monitoring techniques. Many teams, including the operations, development, and infrastructure teams, frequently employ distributed tracing.

7. CONCLUSION

This study examines the complexities of monitoring and debugging in an effort to add to the body of information already available on serverless computing. It offers insights into the difficulties encountered, examines the methodologies and resources at hand, and suggests best practises to guarantee the stability and dependability of serverless systems. Organisations may fully utilise serverless computing while upholding the appropriate degree of performance and operational perfection by comprehending and applying efficient monitoring and debugging procedures.

REFERENCES

JOURNAL OF PARALLEL AND DISTRIBUTED COMPUTING 9, 17 t - 184 (1990) Models for Monitoring and Debugging Tools for Parallel and Distributed Software * Dan C. Marinescu computer sciences department, purdue university, west lafayette, Indiana 47907 James E. lumpp, jr .,-f and Thomas L. Casavant

Dodd, P. S., & Ravishankar, C. V. (1992). "Monitoring and debugging distributed realtime programs." Software: Practice & Experience,22(10), 863–877.

Benedict, S. (2021). "Performance Issues and Monitoring Mechanisms for Serverless IoT Applications—An Exploratory Study". Smart Computing Techniques and Applications. Smart Innovation, Systems and Technologies, vol 225. Springer, Singapore.

Castro et al.'s (2019). "Rise of Serverless Computing, Overview of Current State and Future Trends in Research and Industry."

Manner, J., Kolb, S. & Wirtz, G. (2019). "Troubleshooting Serverless functions: a combined monitoring and debugging approach." SICS Softw.-Inensiv. Cyber-Phys. Syst. 34, 99–104.

Kosti, R., Verma, D., & Ranchal, R. (n.d.). (2019) "Serverless Performance Monitoring: Challenges and Solutions."

Iqbal, S., Sarker, M. R., & Wang, S. (n.d.). (2021) "Distributed Monitoring and Debugging of Serverless Applications."

Kumar, C. (2018, September 19). 10 best tools to monitor and debug serverless applications.

(N.d.). Amazon.com. Retrieved June 13, 2023, from https://aws.amazon.com/blogs/mt/communicate-monitoringinformation-by-sharing-amazon-cloudwatch-dashboards/

(N.d.-b). Datadoghq.com. Retrieved June 13, 2023, from https://www.datadoghq.com/serverless-monitoring/

Monitor, debug and improve your entire stack. (n.d.). New Relic. Retrieved June 13, 2023, from https://newrelic.com/

Lumigo - serverless monitoring and troubleshooting platform. (2023, May 28). Lumigo. https://www.lumigo.io/

Monitor serverless AWS applications at any scale. (2020, December 10). Dashbird. https://dashbird.io/

(N.d.-c). Thundra.Io. Retrieved June 13, 2023, from https://www.thundra.io/

Datadog. (2016, March 2). Amazon ELB Dashboard. Amazon ELB Dashboard. https://www.datadoghq.com/dashboards/elb-dashboard/

New Relic insights: best practices for success. (2018, December 20). New Relic. https://newrelic.com/blog/best-practices/new-relicinsights-getting-started-best-practices

Dashbird app: Improve your infrastructure. (2020, October 2). Dashbird. https://dashbird.io/docs/

Kanu, C. (2023, May 29). A comprehensive guide to serverless monitoring and debugging. Bejamas.

Debugging full-stack serverless apps. (n.d.). SST. Retrieved June 13, 2023.

Clark, A. (2019, March 28). Monitoring & Debugging Serverless applications for red nose day 2019. Comic Relief Technology.

Poojary, R. (n.d.). Managing observability on serverless application. Antstack.com. Retrieved June 13, 2023

Monitoring serverless applications. (2020, July 5). Devmio - Software Know-How.