Implementation of a centralized log monitoring system using grafana, loki, and alloy

Danur Wijayanto[1*], Sharfina Mutia Syarifah[2]

[1]Information Technology Program Study, Faculty of Science and Technology, Universitas Aisyiyah Yoygakarta, Indonesia
[2]BioTechnology Program Study, Faculty of Science and Technology, Universitas Aisyiyah Yoygakarta, Indonesia
Email: danurwijayanto@unisayogya.ac.id, sharfinamutiasyarifah@unisayogya.ac.id

Abstract
Badan Pengembangan Teknologi dan Sistem Informasi (BPTSI) at Universitas Aisyiyah Yogyakarta manages a large number of websites, making it difficult for administrators to monitor web server logs manually. This manual process is inefficient for centrally observing server performance and security. This research aims to implement a centralized system capable of real-time monitoring of web server access and error logs, as well as performance metrics. The method used consists of three stages: preparation, configuration and integration, and testing. The system was built on the Grafana Cloud platform, which integrates Prometheus for metrics collection, Grafana Loki for log aggregation, and Grafana Alloy as the data collector on the client server. The results show a successful integration, producing a centralized dashboard that visualizes real-time access and error logs from 187 log files, along with Apache web server statistics such as CPU load, response time, and uptime. This system allows administrators to monitor the health and security of all web services efficiently from a single interface, eliminating the need for manual log file inspection.

Keywords: alloy; grafana; log monitoring; loki; prometheus

1. Introduction
As technology advances, the need for network system monitoring becomes increasingly important (Nurjanah et al., 2024). Monitoring systems are systems that collect data from various resources in real-time format (Husna & Rosyani, 2021). Network monitoring functions to observe, record, and evaluate every activity occurring on network devices to ensure their performance remains optimal (Pradana et al., 2022). Monitoring systems are required by various organizations, one example being companies that provide system and network services to ensure that the performance of applications and network infrastructure operates optimally.

Universitas Aisyiyah Yogyakarta is an educational institution that has a web-based application service used by the academic community, managed by Badan Pengembangan Teknologi dan Sistem Informasi (BPTSI) (“BPTSI Unisa Yogyakarta,” n.d.). The large number of websites managed complicates the server administrators' ability to monitor the logs of each website, which has still been done manually, thus requiring a platform that can centrally monitor web server logs.

Log monitoring is one way to monitor the security of systems, which is an important aspect in various sectors of organizations that require computer network assistance to perform their functions (Sheeraz et al., 2023). Continuous monitoring of logs allows for early detection of suspicious activities or cyber attacks, enabling incident responses to be executed as quickly as possible and ensuring optimal performance (Pradana et al., 2022; Sheeraz et al., 2023). Through monitoring tools, organizations can identify unauthorized access attempts, malware, and other anomalies in real-time, thereby helping security personnel to be aware of the overall condition of the organization's network. Some popular platforms used for network monitoring and data analytics are Prometheus and Grafana (Taiwo Joseph Akinbolaji et al., 2024).

Prometheus is an open source monitoring system that was originally developed by SoundCloud in 2012 and is currently an independent project managed by the Cloud Native Computing Foundation (CNCF), after Kubernetes. Prometheus stores metric data in the form of time series (Prometheus, n.d.) which is then integrated with Grafana to display metrics in the form of visual dashboards (Chakraborty, 2021). However, to address the issues faced by BPTSI, Prometheus and Grafana alone are not sufficient; a platform that can store log data is needed, namely Grafana Loki.

Loki is a horizontally scalable, highly available, multi-tenant log aggregation system inspired by Prometheus (“Grafana Loki,” n.d.; Njoera et al., 2024). It is designed to be very cost effective and easy to operate. It does not index the contents of the logs, but rather a set of labels for each log stream. To support loki Grafana Alloy needed for OpenTelemetry Collector that is compatible with Promotheus (*Grafana Alloy*, n.d.). In this paper, the researchers will utilize Grafana Alloy and Loki to assist in real-time log monitoring, which is divided into access.log and error.log across various websites.

2. Method

The method used in this research is shown in Figure 1, which consist of 3 steps : (1) Preparation; (2) Configuration and Integration; and (3) Testing.

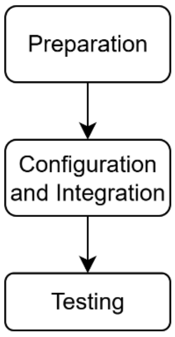


Figure 1. Method

2.1. Preparation

Researchers prepared the infrastructure by planning and analyzing the specifications of system requirements in terms of hardware and software. Table 1 shows the hardware requirements, while Table 2 shows the software requirements.

Table 1. Hardware Requirements

Component	**Specification**
	Web Server
CPU	Intel(R) Xeon(R) CPU E3-1220 V2 @ 3.10GHz
RAM	32 GB DDR3 RAM
Disk	512 GB HDD

The study used four computer devices divided into two computers for *the server*, one computer as *a client*, and one computer as an attacker. Table 4 shows the main components of the software to be integrated which consist of: (1) *Prometheus*; (2) Grafana Cloud; (3) Logstash; (4) *Alloy*;

Table 2. Software Requirements

Software	**Description**
Prometheus	Collection and storage of metrics from the system (CPU Usage, Network Traffic, and others)
Grafana Cloud	Dashboard visualitation
Loki	log aggregation system
Alloy	OpenTelemetry Collector

2.2. Configuration and Integration

The next step is to configure and integrate Grafana, Prometheus Logstash, and Alloy. The configuration is done on Grafana cloud and client server. The system architecture is shown in Figure 2.

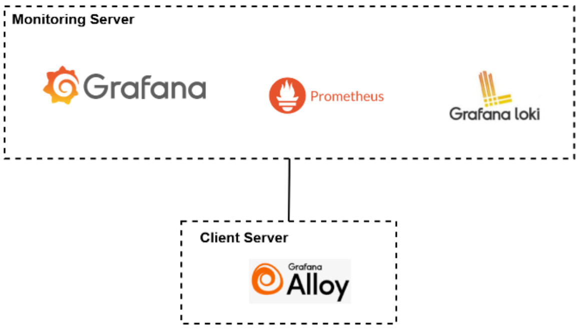


Figure 2. System Architecture

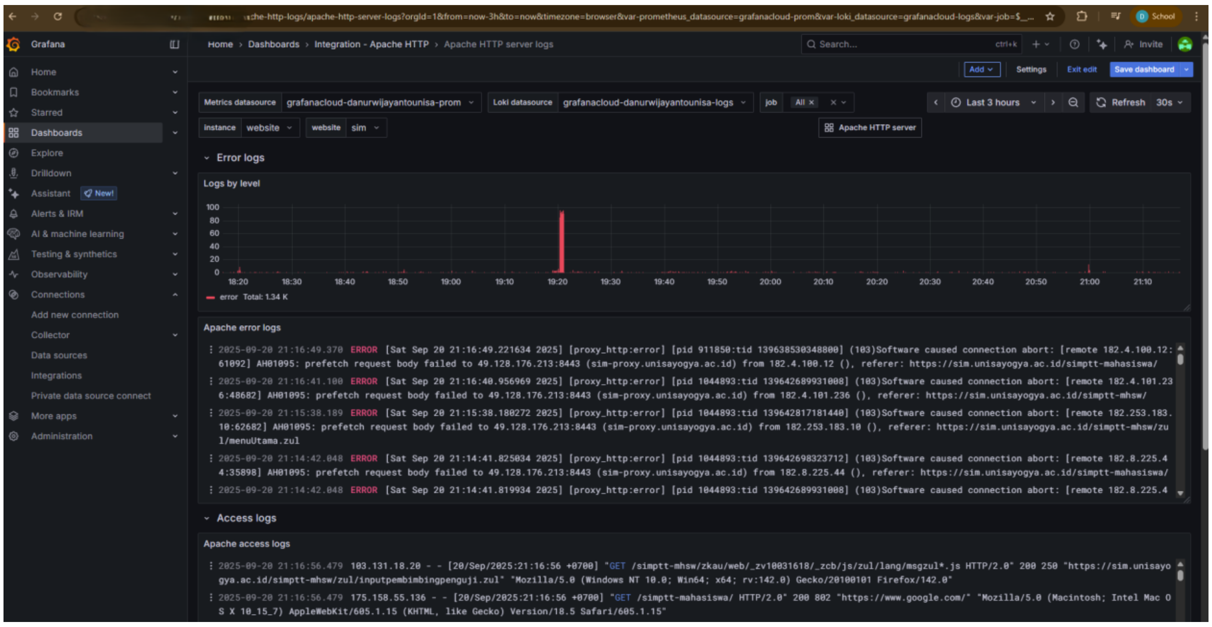
2.3. Testing

The testing was conducted by checking whether the delivery of logs and Apache web server statistics can be sent and displayed on the Grafana platform.

3. Results and Discussion

3.1. Result

The results after successfully performing the integration process are shown in Figure 3. Figure 3 displays the statistics of the error logs, details of the error logs, and details of the access logs. We separated our log files according to the names of the log files from the existing website, with a total of 187 files for each error logs and access logs.



Apache error logs

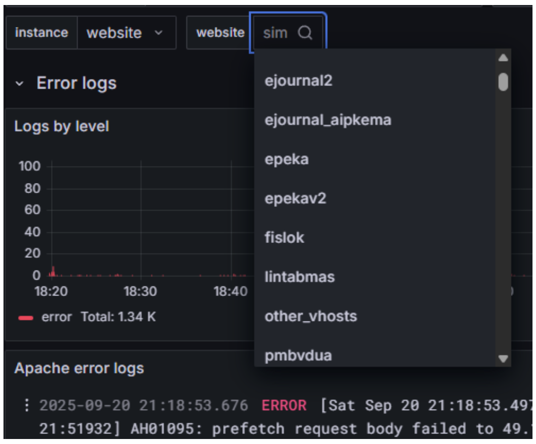
```
2025-09-20 21:16:49.370 ERROR [Sat Sep 20 21:16:49.221634 2025] [proxy_http:error] [pid 911850:tid 139638530348800] (103)Software caused connection abort: [remote 182.4.100.12:61092] AH01095: prefetch request body failed to 49.128.176.213:8443 (sim-proxy.unisayogya.ac.id) from 182.4.100.12 (), referer: https://sim.unisayogya.ac.id/simptt-mahasiswa/
2025-09-20 21:16:41.100 ERROR [Sat Sep 20 21:16:40.956969 2025] [proxy_http:error] [pid 1044893:tid 139642689931008] (103)Software caused connection abort: [remote 182.4.101.236:48682] AH01095: prefetch request body failed to 49.128.176.213:8443 (sim-proxy.unisayogya.ac.id) from 182.4.101.236 (), referer: https://sim.unisayogya.ac.id/simptt-mhsw/
2025-09-20 21:15:38.189 ERROR [Sat Sep 20 21:15:38.180272 2025] [proxy_http:error] [pid 1044893:tid 139642817181440] (103)Software caused connection abort: [remote 182.253.183.10:62682] AH01095: prefetch request body failed to 49.128.176.213:8443 (sim-proxy.unisayogya.ac.id) from 182.253.183.10 (), referer: https://sim.unisayogya.ac.id/simptt-mhsw/zul/menuUtama.zul
2025-09-20 21:14:42.048 ERROR [Sat Sep 20 21:14:41.825034 2025] [proxy_http:error] [pid 1044893:tid 139642698323712] (103)Software caused connection abort: [remote 182.8.225.44:35898] AH01095: prefetch request body failed to 49.128.176.213:8443 (sim-proxy.unisayogya.ac.id) from 182.8.225.44 (), referer: https://sim.unisayogya.ac.id/simptt-mahasiswa/
2025-09-20 21:14:42.048 ERROR [Sat Sep 20 21:14:41.819934 2025] [proxy_http:error] [pid 1044893:tid 139642689931008] (103)Software caused connection abort: [remote 182.8.225.4
```

Apache access logs

```
2025-09-20 21:16:56.479 103.131.18.20 - - [20/Sep/2025:21:16:56 +0700] "GET /simptt-mhsw/zkau/web/_zv10031618/_zcb/js/zul/lang/msgzul*.js HTTP/2.0" 200 250 "https://sim.unisayogya.ac.id/simptt-mhsw/zul/inputpembimbingpenguji.zul" "Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:142.0) Gecko/20100101 Firefox/142.0"
2025-09-20 21:16:56.479 175.158.55.136 - - [20/Sep/2025:21:16:56 +0700] "GET /simptt-mahasiswa/ HTTP/2.0" 200 802 "https://www.google.com/" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/605.1.15 (KHTML, like Gecko) Version/18.5 Safari/605.1.15"
```

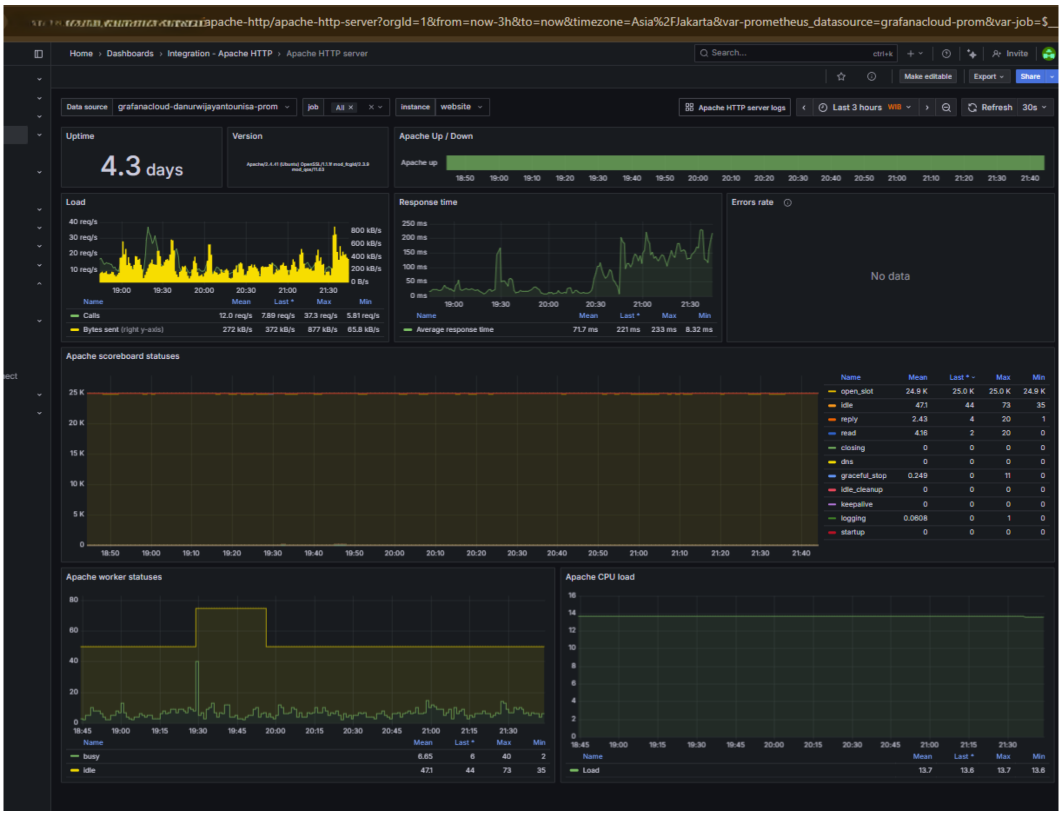
Figure 3. Apache HTTP Server Logs

To view the access logs and error logs for each website, you must select the dropdown available in the header of the dashboard as shown in Figure 4. In addition to displaying logs, the dashboard also shows statistics from the Apache web server such as uptime, response time, and CPU Load as indicated in Figure 5. This result shows that the integration of Grafana, Prometheus, Loki, and Alloy has been successfully completed.



```
2025-09-20 21:18:53.676 ERROR [Sat Sep 20 21:18:53.497
21:51932] AH01095: prefetch request body failed to 49.
```

Figure 4. List website Logs



apache-http/apache-http-server?orgId=1&from=now-3h&to=now&timezone=Asia%2FJakarta&var-prometheus_datasource=grafanacloud-prom&var-job=$_

Name	Mean	Last *	Max	Min
Calls	12.0 req/s	7.89 req/s	37.3 req/s	5.81 req/s
Bytes sent (right y-axis)	272 kB/s	372 kB/s	877 kB/s	65.8 kB/s

Name	Mean	Last *	Max	Min
Average response time	71.7 ms	221 ms	233 ms	8.32 ms

Name	Mean	Last *	Max	Min
open_slot	24.9 K	25.0 K	25.0 K	24.9 K
idle	47.1	44	73	35
reply	2.43	4	20	1
read	4.16	2	20	0
closing	0	0	0	0
dns	0	0	0	0
graceful_stop	0.249	0	11	0
idle_cleanup	0	0	0	0
keepalive	0	0	0	0
logging	0.0608	0	1	0
startup	0	0	0	0

Name	Mean	Last *	Max	Min
busy	6.65	6	40	2
idle	47.1	44	73	35

Name	Mean	Last *	Max	Min
Load	13.7	13.6	13.7	13.6

Figure 5. Apache HTTP Server metrics

3.2. Discussion

To achieve the desired results, an integration process needs to be conducted with the following flow as shown in the Figure 6.

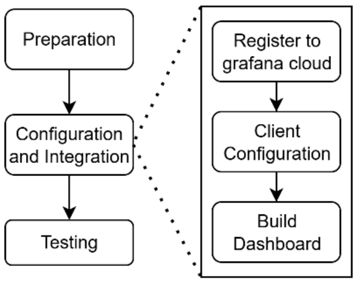


Figure 6. Integration process

3.2.1. Grafana Cloud Registration

At this stage, registration to Grafana Cloud is carried out. Then, the integration process is done using the Apache HTTP Server Integration. By using Grafana Cloud, there is no need to manually install Grafana, Loki, and Prometheus.

3.2.2. Client Configuration

The Client Configuration process is carried out by configuring the HTTP Server and Alloy.

3.2.2.1. HTTP Server Configuration

This configuration is done by adding a server-status page with the endpoint http://localhost:8080/server-status. This data will later be sent from alloy to prometheus to display the Apache web server statistics on the Grafana Dashboard as shown in Figure 5.

3.2.2.2. Alloy Configuration

The alloy configuration is carried out to obtain metric data from the Apache web server such as uptime, response time, and CPU Load, and to retrieve *access.log and *error.log data for each website located in the directory /var/log/apache2/. The process of the alloy configuration in this study is shown in Figure 7.

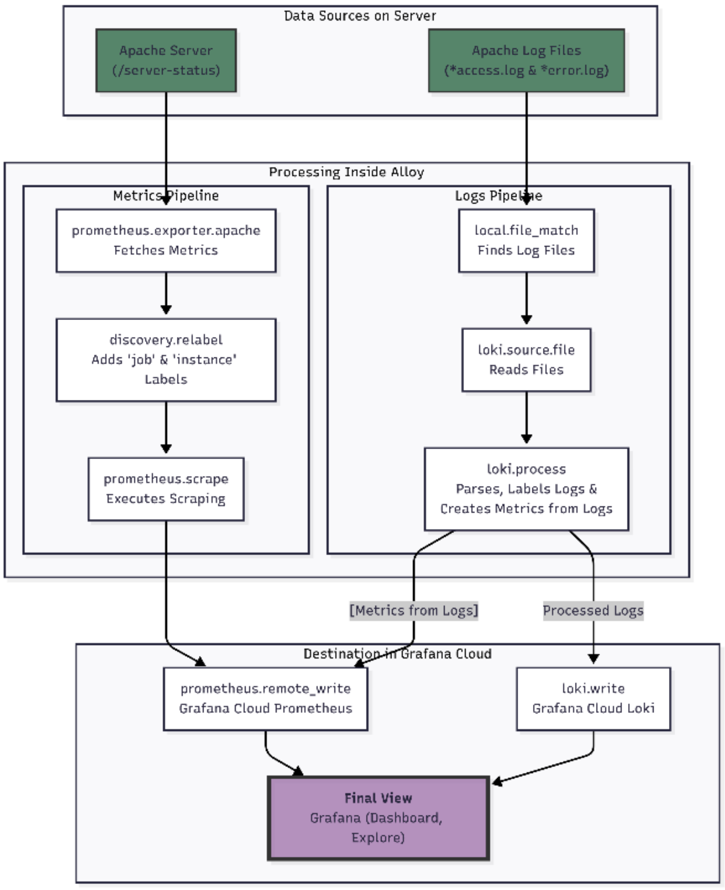

Data Sources on Server
Apache Server (/server-status)
Apache Log Files (*access.log & *error.log)
Processing Inside Alloy
Metrics Pipeline
prometheus.exporter.apache Fetches Metrics
discovery.relabel Adds 'job' & 'instance' Labels
prometheus.scrape Executes Scraping
Logs Pipeline
local.file_match Finds Log Files
loki.source.file Reads Files
loki.process Parses, Labels Logs & Creates Metrics from Logs
[Metrics from Logs]
Processed Logs
Destination in Grafana Cloud
prometheus.remote_write Grafana Cloud Prometheus
loki.write Grafana Cloud Loki
Final View Grafana (Dashboard, Explore)

Figure 7. Alloy Configuration

a. Apache Web Server Metrics Pipeline

The data source comes from the metric /server-status data which is periodically visited by prometheus.exporter.apache to collect raw data. Then the data is labeled with job and instance for easier identification on the Grafana dashboard. After that, promotheus.scrape runs the data collection process and sends it to promotheus in Grafana Cloud using prometheus.remote_write.

b. Apache Logs Pipeline

The data source from this process is from the *access.log and *error.log files located in the /var/log/apache2/ directory. To be able to recognize multiple files at once, a directory scan is required which is performed by the local.file_match function, followed by reading each new line by loki.source.file. The reading process is carried out by loki.process, which performs parsing using Regex to be sent to Prometheus and Loki via prometheus.remote_write and loki.write.

3.2.3. Build Dashboard

The dashboard used is the default dashboard from Apache Integration that allows viewing statistics from the Apache web server and statistics from error logs, detailed error logs, and access logs as shown in Figure 3.

4. Conclusion

In this paper, the integration of Loki, Alloy, and Prometheus using the Grafana Cloud platform has been successfully accomplished. The use of this platform can shorten configuration time because the services of Loki and Prometheus are already provided, allowing for instant configuration. With this integration, service administrators can monitor logs and web server statistics directly from the Grafana Cloud dashboard, eliminating the need to manually check log files by accessing the server and reading the files one by one.

5. Acknowledgements

Thank you to the LPPM Unisa Yogyakarta for funding this research and to the BPTSI Unisa Yogyakarta for providing support and resources so that this research could be completed.

References

BPTSI Unisa Yogyakarta. (n.d.). *Layanan BPTSI Unisa Yogyakarta*. Retrieved September 21, 2025, from https://bptsi.unisayogya.ac.id/layanan-badan-pengembangan-teknologi-dan-sistem-informasi/

Chakraborty, M. K., Ajit Pratap. (2021). Grafana. In *Monitoring Cloud-Native Applications* (pp. 187–200). Apress. https://doi.org/10.1007/978-1-4842-6888-9_6

Grafana Alloy. (n.d.). Retrieved September 21, 2025, from https://grafana.com/docs/alloy/latest/

Grafana Loki. (n.d.). *Grafana Loki*. Retrieved September 21, 2025, from https://grafana.com/oss/loki/

Husna, M. A., & Rosyani, P. (2021). Implementasi Sistem Monitoring Jaringan dan Server Menggunakan Zabbix yang Terintegrasi dengan Grafana dan Telegram. *JURIKOM (Jurnal Riset Komputer)*, *8*(6), 247. https://doi.org/10.30865/jurikom.v8i6.3631

Njoera, Y. A. D., Hartawan, I. N. B., & Krisna, E. D. (2024). The Analysis Of Honeypot Performance Using Grafana Loki And ELK Stack Visualization. *Jurnal Info Sains : Informatika Dan Sains*, *14*(03). https://doi.org/10.54209/infosains.v14i03

Nurjanah, S., Sembiring, F., & Ayuningsih, R. R. (2024). Integration of Telegram Bot and Uptime KUMA for Wi-Fi Network Monitoring using Mikrotik. *Jurnal Pilar Nusa Mandiri*, *20*(2), 118–126. https://doi.org/10.33480/pilar.v20i2.5535

Pradana, A., Widiasari, I. R., Efendi, R., & Informatika, T. (2022). Implementasi Sistem Monitoring Jaringan Menggunakan Zabbix Berbasis SNMP. *AITI: Jurnal Teknologi Informasi*, *19*(Agustus), 248–262.

Prometheus. (n.d.). *Overview—What is Prometheus?* Prometheus. Retrieved March 16, 2025, from https://prometheus.io/docs/introduction/overview/

Sheeraz, M., Paracha, M. A., Haque, M. U., Durad, M. H., Mohsin, S. M., Band, S. S., & Mosavi, A. (2023). Effective Security Monitoring using Efficient SIEM Architecture. *Human-Centric Computing and Information Sciences*, *13*(0), 16–30. https://doi.org/10.22967/HCIS.2023.13.023

Taiwo Joseph Akinbolaji, Godwin Nzeako, David Akokodaripon, & Akorede Victor Aderoju. (2024). Proactive monitoring and security in cloud infrastructure: Leveraging tools like Prometheus, Grafana, and HashiCorp Vault for Robust DevOps Practices. *World Journal of Advanced Engineering Technology and Sciences*, *13*(2), 074–089. https://doi.org/10.30574/wjaets.2024.13.2.0543