GRADO EN INGENIERÍA EN TECNOLOGÍAS DE TELECOMUNICACIÓN

TRABAJO FIN DE GRADO

MONITORIZACION DE UNA PLATAFORMA EDUCATIVA UTILIZANDO ELASTIC STACK

Autor: Ignacio Ortega Falces

Director: Fernando García Jiménez

Madrid

Declaro, bajo mi responsabilidad, que el Proyecto presentado con el título

Monitorización de una plataforma educativa utilizando Elastic Stack

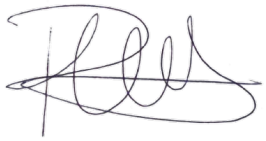
en la ETS de Ingeniería - ICAI de la Universidad Pontificia Comillas en el

curso académico 2023/24 es de mi autoría, original e inédito y

no ha sido presentado con anterioridad a otros efectos.

El Proyecto no es plagio de otro, ni total ni parcialmente y la información que ha sido

tomada de otros documentos está debidamente referenciada.



Fdo.: Ignacio Ortega Falces Fecha: 03/ 07/ 2024

Autorizada la entrega del proyecto

EL DIRECTOR DEL PROYECTO

GARCIA JIMENEZ FERNANDO - 50181172W

Firmado digitalmente por GARCIA JIMENEZ FERNANDO - 50181172W
Fecha: 2024.07.03 20:45:28 +02'00'

Fdo.: Fernando García Jiménez Fecha: 03/ 07/ 2024

GRADO EN INGENIERÍA EN TECNOLOGÍAS DE TELECOMUNICACIÓN

TRABAJO FIN DE GRADO

MONITORIZACION DE UNA PLATAFORMA EDUCATIVA UTILIZANDO ELASTIC STACK

Autor: Ignacio Ortega Falces

Director: Fernando García Jiménez

Madrid

Agradecimientos

A mi familia, por el constante apoyo y confianza en mí.

MONITORIZACION DE UNA PLATAFORMA EDUCATIVA UTILIZANDO ELASTIC STACK

Autor: Ortega Falces, Ignacio.
Director: García Jiménez, Fernando.
Entidad Colaboradora: ICAI – Universidad Pontificia Comillas

RESUMEN DEL PROYECTO

Elastic Stack es una herramienta de código abierto que se utiliza para buscar, analizar y visualizar grandes conjuntos de datos. En este informe se explicará a detalle en que consiste el software, capacidades y limitaciones, así como el desarrollo de una aplicación para poner en práctica los conocimientos.

1. Introducción

La tecnología de la información y los grandes datos han experimentado una rápida evolución en la última década. Por ello poco a poco han ido surgido diferentes herramientas y sistemas para poder manejar los enormes volúmenes de información que se procesan a diario. En este contexto de evolución tecnológica, aparece Elastic Stack como una solución integral y totalmente modular para el manejo y análisis de grandes datos.

Elastic Stack es también conocida como ELK en honor a las herramientas que lo componen: Elasticsearch, Logstash y Kibana. Tres potentes plataformas que trabajan entre si con el mismo fin: manejar volúmenes masivos de datos.

Hoy en día se presenta como una de las herramientas más potentes del mercado y las empresas lo saben. Infinidad de negocios hacen uso de ella, y entre ellos algunos de los nombres más importantes como pueden ser Amazon, Netflix o Accenture. Ya sea por el potente motor de búsqueda, uso de código abierto o capacidad de escalabilidad, Elastic Stack se posiciona un escalón por encima de una competencia de lo más ambiciosa.

2. Definición del proyecto

El proyecto constará de tres fases claramente definidas, cada una con un enfoque específico que permitirá una comprensión integral y práctica de la plataforma ELK Stack.

En la primera fase, se llevará a cabo un estudio detallado sobre la plataforma Elastic Stack. Este análisis incluirá una explicación pormenorizada de cada una de las herramientas que conforman el software, sus características, funcionalidades y cómo se interrelacionan entre sí. Se profundizará en aspectos como el motor de búsqueda de Elasticsearch, la capacidad de Logstash para recopilar y transformar datos, y las potentes herramientas de visualización que ofrece Kibana.

La segunda fase del proyecto se centrará en la implementación práctica de los conocimientos adquiridos durante la fase de estudio. Para ello, se desarrollará un programa que pondrá a prueba estos conocimientos, demostrando cómo se utiliza ELK Stack en un entorno real. Este programa incluirá la configuración de un sistema de recolección de logs, el procesamiento de dichos logs a través de Logstash, su

almacenamiento y búsqueda en Elasticsearch, y finalmente, la visualización de los datos en Kibana. El objetivo de esta fase es no solo demostrar la aplicabilidad de ELK Stack, sino también proporcionar una experiencia práctica que refuerce el aprendizaje teórico.

Por último, en la tercera fase del proyecto, se realizará un análisis orientado a la asignatura de Estadística. En esta etapa, el director del proyecto me proporcionará una aplicación en relación con la asignatura y el objetivo será poner en práctica todos los conocimientos aprendidos acerca de Elastic Stack, mediante el análisis tanto de la aplicación como de la visualización de logs registrados a través de ella.

3. Descripción del modelo/sistema/herramienta

Como se ha mencionado en el apartado anterior, la plataforma seguirá el siguiente guión para el análisis y procesamiento de datos: Beats se encargará de la recopilación de todos los datos. Según la interacción del cliente, Logstash recopilará los logs registrados por Beats y los procesará. Una vez recopilados y enviados los logs, Elasticsearch será nuestra base de datos. A través de Elasticsearch podremos almacenar y gestionar todos los logs según su naturaleza. Por último, aparece Kibana, que nos permitirá visualizar todos estos datos en diferentes tipos de dashboards. Kibana será ideal para visualizar la información de diversas maneras, permitiéndonos utilizar una gran variedad de gráficos para el estudio de la misma y adaptándose a nuestras necesidades específicas.

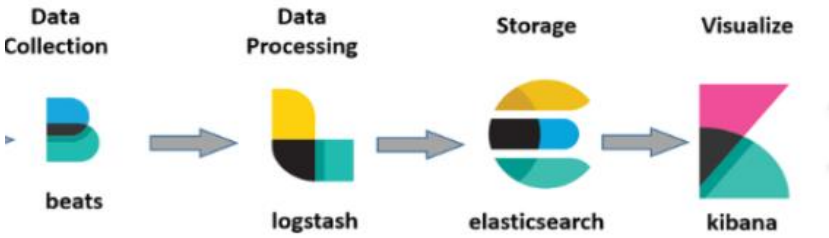


Ilustración 1 - Esquema de arcquitectura de Elastic Stack

Una vez asimilados los conceptos y entendido el funcionamiento de la herramienta, se pondrán en práctica mediante un pequeño programa para entenderlo mejor. Se ha desarrollado una aplicación para la recopilación de diferentes tipos de logs y poder visualizarlos posteriormente a través de potentes dashboards que nos generará Kibana.

El ejemplo consiste en una aplicación web desarrollada en Python utilizando el framework Flask. La aplicación Flask se despliega y ejecuta mediante contenerización en Docker. La estructura del programa seguirá la estándar: mediante la instalación de las dependencias correspondientes, correcto orden en los servicios a levantar y un archivo donde registrar los logs, somos capaces de obtener un magnífico ejemplo de la funcionalidad de la herramienta en cuestión.

La aplicación web la encontramos en el puerto 5000. Realizando diferentes interacciones con la aplicación registramos diferentes tipos de logs donde podremos visualizarlos posteriormente. Estudiaremos los resultados tanto de la aplicación de prueba como de la aplicación final de proyecto.

4. Resultados

Finalmente observamos los resultados de la aplicación web creada. Una vez registrados diferentes logs nos dirigimos al puerto 5601 donde encontramos la página web de Elasticsearch. En ella podemos crear diferentes dashboards y visualizar los logs ordenados según su patrón de índice.

Tras hacer login en la app podremos visualizar diferentes tipos de logs que se registran como registros o errores en accesos

```
logstash  | {
logstash  |        "message" => "User ricardo logged in",
logstash  |          "event" => "login",
logstash  |       "@version" => "1",
logstash  |           "path" => "/logs/flask_app.log",
logstash  |     "@timestamp" => 2024-06-20T10:14:22.307Z,
logstash  |           "host" => "5f4c0ff8b0d5"
logstash  | }
```

Ilustración 2 - registro de logs de autenticación

Estos mismos logs serán almacenados y procesados por Elasticsearch como se mencionó anteriormente. El siguiente paso será visualizarlos mediante diferentes dashboards que nos proporciona Kibana. En este caso podemos ver cómo nos aparecen todos los logs registrados por la aplicación final de proyecto.

Ilustración 3 - Dashboard de kibana para visualizacion de logs

5. Conclusiones

Gracias a su capacidad para gestionar y analizar grandes volúmenes de datos en tiempo real, su flexibilidad gracias al código abierto, y sus potentes herramientas de búsqueda y visualización, Elastic Stack se convierte en una plataforma excepcional. A través de una simple aplicación podemos comprobar cómo podemos recopilar, almacenar y visualizar infinidad de datos de cualquier tipo con una enorme versatilidad y efectividad.

6. Referencias

[1] Educative. “Elasticsearch Fundamentals”. What Is Elastic Stack? - Elasticsearch Fundamentals: Indexing and Querying Data (educative.io)

[2] Historias de éxito de ELK” Casos de uso · Historias de éxito del Elastic Stack | Elastic Customers

[3] Clustering Cuál es su uso en big data.” Clustering: qué es y cuál es su aplicación en Big Data | UNIR´“

[4] T S.: “¿Qué es Elastic Stack y cómo funciona?”,. ¿Qué Es Elastic Stack Y Cómo Funciona? – Todo Servidores (todo-servidores.com)

MONITORING AN EDUCATIONAL PLATFORM USING ELASTIC STACK

Author: Ortega Falces, Ignacio.
Supervisor: García Jiménez, Fernando.
Collaborating Entity: ICAI – Universidad Pontificia Comillas

ABSTRACT

Elastic Stack is an open-source tool used for searching, analyzing, and visualizing large datasets. This report will explain in detail what the software consists of, its capabilities and limitations, as well as the development of an application to put the acquired knowledge into practice.

1. Introduction

Information technology and big data have undergone rapid evolution in the last decade. As a result, various tools and systems have gradually emerged to handle the enormous volumes of information processed daily. In this context of technological evolution, Elastic Stack appears as an integral and fully modular solution for the management and analysis of big data.

Elastic Stack is also known as ELK in honor of the tools that comprise it: Elasticsearch, Logstash, and Kibana. These three powerful platforms work together with the same goal: to handle massive volumes of data.

Today, it stands as one of the most powerful tools on the market, and companies are well aware of this. Countless businesses use it, including some of the most important names such as Amazon, Netflix, and Accenture. Whether it's for its powerful search engine, open-source nature, or scalability capabilities, Elastic Stack positions itself a step above a very ambitious competition.

2. Project definition

The project will consist of three clearly defined phases, each with a specific focus that will allow for a comprehensive and practical understanding of the ELK Stack platform.

In the first phase, a detailed study of the Elastic Stack platform will be conducted. This analysis will include a thorough explanation of each of the tools that make up the software, their features, functionalities, and how they interrelate. It will delve into aspects such as the Elasticsearch search engine, Logstash's ability to collect and transform data, and the powerful visualization tools offered by Kibana.

The second phase of the project will focus on the practical implementation of the knowledge acquired during the study phase. For this, a program will be developed to test this knowledge, demonstrating how ELK Stack is used in a real environment. This program will include the configuration of a log collection system, the processing of these logs through Logstash, their storage and search in Elasticsearch, and finally, the visualization of the data in Kibana. The objective of this phase is not only to demonstrate the applicability of ELK Stack but also to provide a practical experience that reinforces theoretical learning.

Finally, in the third phase of the project, an analysis oriented towards the Statistics course will be conducted. In this stage, the project director will provide me with an application related to the course, and the objective will be to put into practice all the knowledge learned about Elastic Stack, through the analysis of both the application and the visualization of logs recorded through it.

3. Description of the model/system/tool

As mentioned in the previous section, the platform will follow the following sequence for data analysis and processing: Beats will be responsible for collecting all the data. According to client interactions, Logstash will collect the logs registered by Beats and process them. Once the logs are collected and sent, Elasticsearch will serve as our database. Through Elasticsearch, we will be able to store and manage all the logs according to their nature. Finally, Kibana will allow us to visualize all these data in different types of dashboards. Kibana will be ideal for visualizing information in various ways, enabling us to use a wide variety of charts for data analysis and adapting to our specific needs.

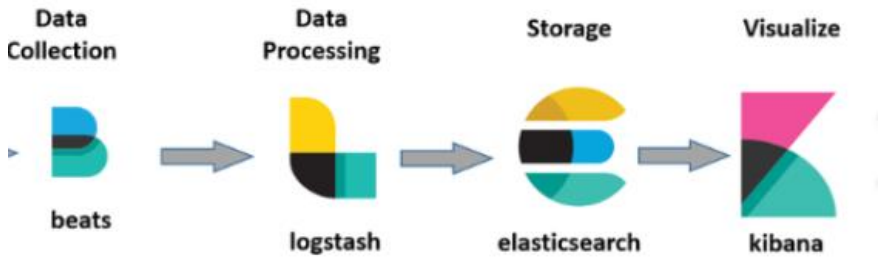


Ilustración 4 - Elastic Stack architecture diagram

Once the concepts are assimilated and the functioning of the tool is understood, they will be put into practice through a small program to better grasp the concepts. An application has been developed to collect different types of logs and visualize them later through powerful dashboards generated by Kibana.

The example consists of a web application developed in Python using the Flask framework. The Flask application is deployed and run through containerization in Docker. The program's structure follows the standard: by installing the corresponding dependencies, correctly ordering the services to be launched, and having a file to log the data, we can obtain an excellent example of the tool's functionality.

The web application is accessible on port 5000. By interacting with the application, we record different types of logs that can be visualized later. We will study the results of both the test application and the final project application

4. Results

Finally, we observe the results of the created web application. After registering different logs, we navigate to port 5601 where we find the Elasticsearch page. Here, we can create different dashboards and visualize the logs sorted according to their index pattern.

After logging into the app, we can visualize different types of logs that are registered, such as access records or errors.

```
logstash        | {
logstash        |         "message" => "User ricardo logged in",
logstash        |           "event" => "login",
logstash        |        "@version" => "1",
logstash        |            "path" => "/logs/flask_app.log",
logstash        |      "@timestamp" => 2024-06-20T10:14:22.307Z,
logstash        |            "host" => "5f4c0ff8b0d5"
logstash        | }
```

Ilustración 5 – User registration Log

These logs will be stored and processed by Elasticsearch as mentioned earlier. The next step will be to visualize them through the various dashboards provided by Kibana. In this case, we can see how all the logs recorded by the final project application appear.

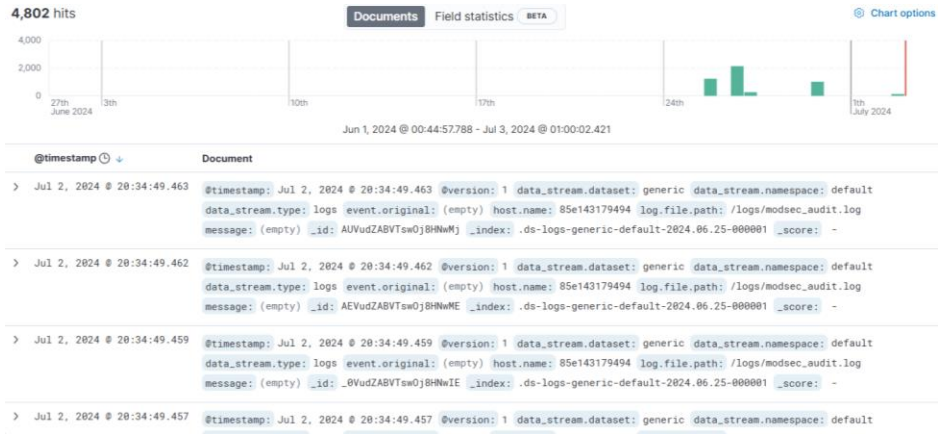


Ilustración 6 – Kibana dashboard with authentication Logs

5. Conclusion

Thanks to its ability to manage and analyze large volumes of data in real-time, its flexibility due to being open-source, and its powerful search and visualization tools, Elastic Stack becomes an exceptional platform. Through a simple application, we can see how we can collect, store, and visualize an endless amount of data of any kind with enormous versatility and effectiveness.

6. References

[1] Educative. “Elasticsearch Fundamentals”. What Is Elastic Stack? - Elasticsearch Fundamentals: Indexing and Querying Data (educative.io)

[2] Historias de éxito de ELK” Casos de uso · Historias de éxito del Elastic Stack | Elastic Customers

[3] Clustering Cuál es su uso en big data.” Clustering: qué es y cuál es su aplicación en Big Data | UNIR´“

[4] T S.: “¿Qué es Elastic Stack y cómo funciona?”,. ¿Qué Es Elastic Stack Y Cómo Funciona? – Todo Servidores (todo-servidores.com)

Índice de la memoria

Capítulo 1. Introducción 6

Capítulo 2. Descripción de las Tecnologías 8

2.1 Flujo 8
2.2 Origen 10
2.3 Componentes 11
2.3.1 Elasticsearch 11
2.3.2 Logstash 12
2.3.3 Kibana 12
2.3.4 Beats 13
2.4 Archivos de configuración necesarios 14
2.5 Clusterización 15
2.5.1 Elementos 16
2.5.2 Beneficios 17
2.5.3 Uso 18
2.6 Herramientas para la automatización de ELK 19
2.6.1 Docker 19
2.6.2 Kubernetes 20
2.7 Comparativa con otras herramientas 21
2.7.1 Principales diferencias 22
2.7.2 Opinión personal 26

Capítulo 3. Estado de la Cuestión 28

3.1 Elastic Stack en diferentes sectores 28
3.1.1 Tecnologias de la informacion y DevOps 28
3.1.2 Salud 29
3.1.3 Seguridad y analisis de amenazas 29
3.1.4 Finanzas y banca 29
3.1.5 Telecomunicaciones 29
3.2 Uber 30
3.2.1 Especificaciones 30

3.2.2 Arquitectura del sistema *31*
3.2.3 Aprendizaje *32*

Capítulo 4. Definición del Trabajo ***34***
4.1 Justificación 34
4.1.1 Búsqueda y Análisis en tiempo real *35*
4.1.2 Versatilidad en los Datos *36*
4.1.3 Visualización intuitiva y accesible *37*
4.1.4 Ecosistema y comunidad activa *37*
4.1.5 costos y licenciamiento *38*
4.2 Objetivos 39
4.3 Metodología 40

Capítulo 5. Sistema/Modelo Desarrollado ***42***
5.1 Análisis del Sistema 43
5.2 Diseño 45
5.2.1 Aplicaciones empleadas *45*
5.2.2 Estructura del proyecto *46*
5.2.3 Flujo de datos *49*
5.3 Implementación 51
5.4 Salidas del sistema 56

Capítulo 6. Análisis de Resultados ***60***
6.1 Interfaz 60
6.2 Estructura del proyecto 63
6.2.1 Fundamentos *63*
6.2.2 Servicios *64*
6.3 Análisis de Logs 71

Capítulo 7. Conclusiones y Trabajos Futuros ***75***
7.1 Análisis integral de Elastic Stack 75
7.2 Aplicación practica 76
7.3 Estadistica 76
7.4 Trabajos futuros 77

Capítulo 8. Bibliografía ***78***

ANEXO I: ALINEACIÓN DEL PROYECTO CON LOS ODS..... 80

Índice de figuras

Ilustración 1 - Esquema de arcquitectura de Elastic Stack 10
Ilustración 2 - registro de logs de autenticación 11
Ilustración 3 - Dashboard de kibana para visualizacion de logs 11
Ilustración 4 - Elastic Stack architecture diagram 13
Ilustración 5 – User registration Log 14
Ilustración 6 – Kibana dashboard with authentication Logs 14
Ilustración 7- Flujo de datos a través de los componentes de ELK Stack 9
Ilustración 8 - Estructura de un clúster en Elasticsearch 17
Ilustración 9 - Contenerización de Docker 20
Ilustración 10 - Gráfico comparativo de herramienta elegida por empresas 21
Ilustración 11 - Indexación en el Data Pipeline de Splunk 24
Ilustración 12 - Interfaz de usuario de Splunk 26
Ilustración 13- Arquitertura de real-time prediction system de Uber 31
Ilustración 14 - Pipelines de procesamiento de Logstash 37
Ilustración 15- Estructura del proyecto de prueba 47
Ilustración 16 - Log generado por evento 50
Ilustración 17 - Captura de archivo “flask_app.log” 50
Ilustración 18 – Índices para distintos tipos de Logs 51
Ilustración 19 – Sistema de login en aplicación de prueba 52
Ilustración 20 - Sistema de registro en aplicación de prueba 52
Ilustración 21 - Usuario registrado correctamente 53
Ilustración 22 - Error en el login de usuario 53
Ilustración 23 - Dashboard principal de aplicación de prueba 54
Ilustración 24 - Creación de patrones de índice en Kibana 57
Ilustración 25 - Registro de logs en Elasticsearch 57
Ilustración 26 - Visualización en Kibana I 58

Ilustración 27 - Visualización en Kibana II 59
Ilustración 28 - Sistema de login 61
Ilustración 29 - Sistema de registro 62
Ilustración 30 - Dashboard de profesor 62
Ilustración 31 - Interfaz web de base de datos 65
Ilustración 32 - Estructura interna de aplicación 67
Ilustración 33 - Flujo de datos por la app 70
Ilustración 34 - Servicios de la aplicación en Docker 71
Ilustración 35 - Espacios de trabajos en Elasticsearch 72
Ilustración 36 - Visualización final de logs 73
Ilustración 37 - Reparto de logs almacenados en proyecto 74

Capítulo 1. INTRODUCCIÓN

En un mundo donde la información fluye a una velocidad sin precedentes y la cantidad de datos generados al segundo supera la capacidad humana para procesarlos manualmente, surge la necesidad de herramientas que no solo nos manejen este torrente de información, sino que también nos aporten de él significado y valor. Elastic Stack se nos presenta como el conjunto de herramientas capaz de iluminar una salida en un laberinto interminable de datos, permitiéndonos una compresión más profunda y acción más detallada sobre ellos.

Desde tiempos inmemoriales la humanidad ha tratado de comprender su entorno, descifrar patrones en la naturaleza y tomar decisiones basadas en observaciones empíricas. En la era digital, toda esta búsqueda de conocimiento perseguida por tantos años se transforma en un concepto: los datos. Cada interacción, cada transacción, cada petición y cada movimiento genera una información que, correctamente procesada y almacenada, se transformaran en el tan ansiado conocimiento y ventajas.

Elastic Stack nos permitirá encarnar la filosofía y dar un paso adelante en la era de los datos. Seremos capaces de recolectar, procesar y analizar los datos de manera integral y en tiempo real, permitiéndonos estar por delante en el mundo digital transformando información cruda en conocimientos aplicables.

La verdadera innovación nace de la capacidad para ver lo que otros no tienen a su alcance y tomar las decisiones en base a información basada en datos precisos. Elastic Stack ofrece una plataforma para la experimentación y el descubrimiento. Al permitir que los equipos exploren datos de forma libre y visualicen resultados al instante, promueve una cultura de innovación continua y mejora constante.

Finalmente, Elastic Stack se nos plantea como un ecosistema dinámico en constante evolución, respaldado por una comunidad activa y entusiasta de desarrolladores y usuarios. Esta comunidad contribuye continuamente con mejoras, nuevas funcionalidades y mejores

prácticas, garantizando que Elastic Stack no solo se mantenga relevante, sino que también esté a la vanguardia de las tecnologías de análisis de datos constantemente.

A lo largo de este proyecto se descubrirá un océano de oportunidades a nuestro alcance simplemente con los conocimientos básicos de una plataforma que está en constante crecimiento y con un potencial de lo más sorprendente.

Capítulo 2. DESCRIPCIÓN DE LAS TECNOLOGÍAS

Elastic Stack es un conjunto de herramientas de código abierto diseñado para buscar, analizar y visualizar datos en tiempo real. Ofrece una solución integral para la gestión de datos, desde la recolección y procesamiento hasta el almacenamiento y análisis.

En el siguiente epígrafe se explicará las tecnologías requeridas para el correcto funcionamiento de Elastic Stack. Hare especial hincapié en todo el software que participa en la gestión y análisis de los datos, así como otras aplicaciones que automaticen estos procesos. Finalmente explicare otras plataformas similares en el mercado y una comparativa con la líder actual.

2.1 FLUJO

Elastic Stack, también conocido como ELK Stack, está formado principalmente por 3 componentes: Elasticsearch, Logstash y Kibana (de ahí ELK); y un cuarto componente llamado Beats. Este último no es vital pero si decisivo para la total funcionalidad y flexibilidad de la aplicación en cuanto a recopilación de datos. A continuación, explicare brevemente cómo funciona el flujo de datos a lo largo de cada componente de la plataforma:

Beats es un conjunto de agentes ligeros diseñados para recolectar datos de diversas fuentes y enviarlos a una instancia central de Logstash o Elasticsearch. Cada tipo de Beat se especializa en una forma particular de datos; por ejemplo, Filebeat se encarga de los archivos de log, Metricbeat recoge métricas del sistema y Packetbeat monitorea el tráfico de red. Estos agentes son fáciles de desplegar y configurar, permitiendo una recopilación eficiente y escalable de datos en tiempo real desde múltiples orígenes.

Logstash, por su parte, es una herramienta robusta para el procesamiento y la transformación de datos. Una vez que Logstash recibe los datos enviados por los Beats, puede realizar una variedad de transformaciones y enriquecimientos en ellos. Utilizando una configuración

flexible basada en plugins, Logstash puede filtrar, modificar y estructurar los datos según sea necesario antes de enviarlos a Elasticsearch. Esto permite normalizar los datos y extraer información valiosa, como agregar etiquetas, modificar campos, o incluso combinar datos de múltiples fuentes.

Elasticsearch es el motor de búsqueda y análisis central de Elastic Stack. Una vez que Elasticsearch recibe los datos de Logstash, los indexa y los almacena en una estructura optimizada para consultas rápidas y eficientes. Elasticsearch está diseñado para manejar grandes volúmenes de datos y proporcionar respuestas en tiempo casi real, lo que permite a los usuarios realizar búsquedas complejas y análisis detallados de los datos. Su capacidad de escalabilidad horizontal garantiza que pueda crecer junto con las necesidades de la infraestructura de datos.

Finalmente, Kibana actúa como la interfaz visual para los datos almacenados en Elasticsearch. Kibana permite a los usuarios explorar y visualizar los datos de manera intuitiva a través de gráficos, tablas y mapas interactivos. Con Kibana, es posible crear dashboards personalizados que reflejen el estado de los sistemas y aplicaciones en tiempo real.

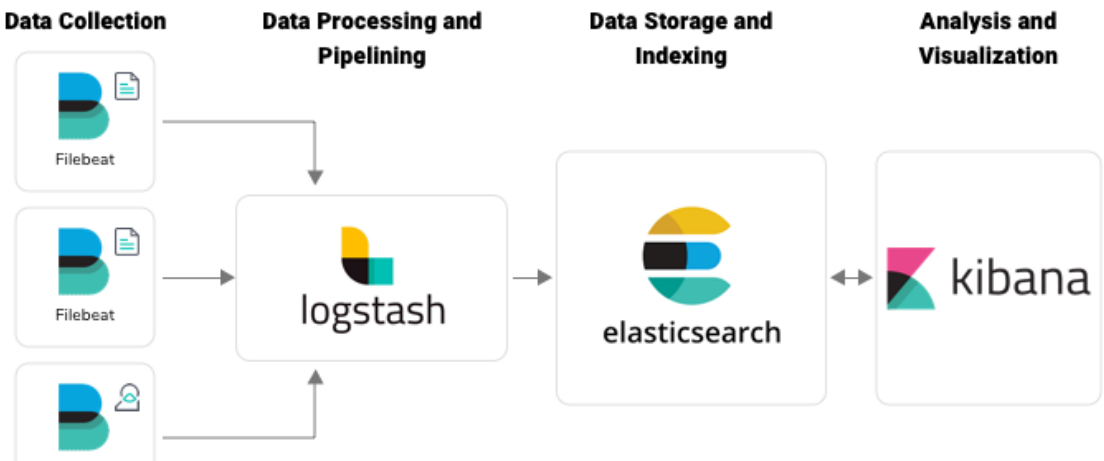


Ilustración 7- Flujo de datos a través de los componentes de ELK Stack

En conjunto, Beats, Logstash, Elasticsearch y Kibana forman un flujo de trabajo cohesivo para la recolección, procesamiento, almacenamiento y visualización de datos, proporcionando una solución completa para la gestión y análisis de grandes volúmenes de información en tiempo real.

2.2 ORIGEN

Todo comenzó con el ELK Stack, un acrónimo de tres proyectos de código abierto: Elasticsearch, Logstash y Kibana. Shay Banon creó el precursor de Elasticsearch, llamado Compass, en 2004. Buscando mejorar la funcionalidad y escalabilidad de Compass, Banon lanzó la primera versión de Elasticsearch en febrero de 2010. Elasticsearch fue diseñado como un motor de búsqueda distribuido y escalable basado en Apache Lucene, permitiendo búsquedas rápidas y eficientes a través de grandes volúmenes de datos. Logstash, creado por Jordan Sissel en 2011, se integró naturalmente con Elasticsearch para proporcionar una solución de procesamiento de logs, y Kibana, desarrollado por Rashid Khan en 2013, añadió una potente interfaz de visualización para los datos almacenados en Elasticsearch.

Elastic NV se fundó en 2012 para proporcionar servicios y productos comerciales en torno a Elasticsearch y software relacionado, consolidando así el desarrollo continuo de estas herramientas. En marzo de 2015, la compañía Elasticsearch cambió su nombre a Elastic, reflejando su expansión más allá del motor de búsqueda original. Con el tiempo, Elastic Stack evolucionó para incluir nuevos proyectos como Beats, una colección de agentes ligeros para la recolección de datos, y X-Pack, un conjunto de complementos que proporcionan características avanzadas como seguridad, monitoreo y alertas. En junio de 2018 Elastic comenzó a cotizar en la Bolsa de Nueva York y ha continuado su evolución hasta consolidarse como Elastic Stack: una plataforma integral y robusta para la gestión y análisis de datos.

2.3 COMPONENTES

2.3.1 ELASTICSEARCH

Elasticsearch es un motor de búsqueda y análisis distribuido, altamente escalable, basado en el motor de búsqueda Apache Lucene. Se utiliza principalmente para buscar, almacenar y analizar grandes volúmenes de datos en tiempo real. Una de sus principales características es su capacidad para distribuir datos y operaciones a través de múltiples nodos, lo que garantiza alta disponibilidad y redundancia, permitiendo un manejo eficiente de fallos.

Entre sus funciones, Elasticsearch nos permite realizar búsquedas de texto de manera extremadamente rápida y precisa, gracias a su potente capacidad de indexación. Además, proporciona capacidades avanzadas de análisis mediante agregaciones, las cuales permiten ejecutar cálculos y estadísticas sobre grandes conjuntos de datos de forma dinámica y eficiente. Esto lo convierte en una herramienta invaluable para convertir grandes volúmenes de datos no estructurados en índices estructurados y de consulta rápida.

Podemos utilizar Elasticsearch para una variedad de aplicaciones, incluyendo:

- **Búsqueda en sitios web y aplicaciones:** Mejora la experiencia del usuario proporcionando resultados de búsqueda rápidos y relevantes.
- **Análisis de registros:** Procesa y analiza logs para detectar patrones, anomalías y tendencias, facilitando el monitoreo y la resolución de problemas.
- **Monitoreo y seguridad en tiempo real:** Implementa soluciones de monitoreo que pueden detectar y alertar sobre eventos críticos en tiempo real, mejorando la seguridad y la operatividad de los sistemas.

Elasticsearch también soporta un esquema flexible basado en JSON, lo que facilita la indexación de datos complejos y variados. Su interfaz RESTful permite integrarse fácilmente con una amplia gama de aplicaciones y servicios

2.3.2 LOGSTASH

Logstash es una herramienta poderosa para el procesamiento de datos y recolección de logs, capaz de ingerir datos de una variedad de fuentes, transformarlos y enviarlos a un destino, generalmente Elasticsearch. Funciona como una tubería de datos, permitiendo el flujo continuo y eficiente de información desde la fuente hasta el destino final.

Logstash no solo se encarga de la ingestión de datos, sino que también aplica filtros para limpiarlos, transformarlos y enriquecerlos. Por ejemplo, puede desglosar logs complejos en campos individuales, aplicar transformaciones condicionales y añadir metadatos relevantes. Esta capacidad para transformar datos es crucial para asegurar que los datos sean indexados en un formato que maximice la eficiencia de las búsquedas y análisis posteriores.

Además, Logstash puede recibir datos de diversas fuentes, como archivos de log, bases de datos, sistemas de mensajería y otros servicios. Una vez que los datos son procesados, Logstash los envía a uno o más destinos configurados, como Elasticsearch, bases de datos o sistemas de almacenamiento en la nube. Esta flexibilidad lo hace ideal para crear flujos de procesamiento de datos configurables (pipelines), que permiten una manipulación detallada de los datos antes de su almacenamiento.

2.3.3 KIBANA

Kibana es una herramienta de visualización y exploración de datos que se integra con Elasticsearch. Permite a los usuarios analizar y visualizar datos indexados en Elasticsearch a través de gráficos y paneles interactivos.

Las principales funciones de Kibana incluyen:

- **Visualización de datos:** Kibana permite crear gráficos, tablas, mapas y otros tipos de visualizaciones para representar los datos almacenados en Elasticsearch. Estas visualizaciones pueden ser personalizadas para mostrar diferentes métricas y tendencias.

- **Paneles de control (Dashboards):** Kibana permite la construcción de dashboards interactivos que pueden actualizarse en tiempo real. Los dashboards combinan múltiples visualizaciones en una única vista, proporcionando una visión completa y dinámica de los datos.
- **Exploración de datos:** Kibana facilita la exploración detallada de los datos mediante herramientas como Discover, que permite filtrar y buscar datos específicos, y Canvas, que permite crear presentaciones visuales avanzadas

Además, Kibana incluye herramientas específicas para el análisis de logs, permitiendo identificar problemas y optimizar el rendimiento de los sistemas. La capacidad de generar reportes basados en los datos visualizados facilita la comunicación de insights a otros miembros del equipo. Kibana también ofrece opciones avanzadas de personalización y configuración, lo que permite adaptarlo a las necesidades específicas de cada usuario o equipo.

2.3.4 BEATS

Beats es una plataforma de agentes ligeros diseñados para enviar datos desde máquinas remotas a Logstash o Elasticsearch. Cada tipo de Beat está especializado en una tarea específica, lo que permite una recolección eficiente y dirigida de datos desde diversas fuentes.

Por ejemplo, Filebeat se encarga de recolectar y enviar logs de archivos, mientras que Metricbeat recoge métricas del sistema y servicios. Packetbeat captura y analiza tráfico de red, Heartbeat monitorea la disponibilidad de servicios, Auditbeat realiza auditorías de seguridad del sistema y Functionbeat recolecta datos de funciones serverless en entornos de nube. Esta especialización hace que cada Beat sea altamente eficiente en su tarea específica, asegurando que los datos recolectados sean precisos y relevantes.

Beats facilita la recolección de datos de manera eficiente y liviana, sin impactar significativamente en el rendimiento de los sistemas monitorizados. Además, su integración con Logstash y Elasticsearch permite un procesamiento y análisis fluido de los datos

recolectados, optimizando la visibilidad y el control sobre las operaciones y la infraestructura tecnológica. Esta capacidad para enviar datos desde múltiples fuentes distribuidas a una ubicación central es esencial para el monitoreo y la configuración de alertas basadas en datos, permitiendo una gestión proactiva y efectiva de los sistemas y servicios.

2.4 ARCHIVOS DE CONFIGURACIÓN NECESARIOS

Para el correcto funcionamiento de los servicios de Elastic Stack, algunos de ellos requieren archivos de configuración específicos dentro del proyecto. Estos archivos permiten personalizar y adaptar cada servicio a las necesidades particulares de la implementación.

Filebeat: Este servicio necesita un archivo de configuración llamado filebeat.yml. En este archivo se especifican las rutas de los logs que Filebeat debe monitorear y enviar a Logstash o directamente a Elasticsearch. El archivo también contiene configuraciones de módulos, procesadores, y salidas, permitiendo una gran flexibilidad en cómo se recolectan y procesan los logs. Por ejemplo, se puede definir un input para leer los logs de un directorio específico y configurarlo para que Filebeat envíe estos logs a Logstash para un procesamiento adicional.

Logstash: Logstash requiere varios archivos de configuración, típicamente almacenados en un directorio dedicado a los pipelines, como logstash/pipeline. Estos archivos definen cómo Logstash debe procesar los datos de entrada, con detalles sobre inputs, filtros y outputs. Por ejemplo, un archivo de configuración puede especificar que los datos de entrada provienen de Filebeat, aplicar una serie de filtros para transformar los datos y luego enviarlos a Elasticsearch. Esta configuración permite a Logstash manejar complejas transformaciones de datos antes de su indexación.

Elasticsearch: Aunque Elasticsearch puede funcionar con configuraciones predeterminadas, a menudo se configura mediante un archivo elasticsearch.yml. Este archivo permite ajustar parámetros importantes como la configuración de la memoria, el almacenamiento de datos, y la red. También se pueden especificar configuraciones de

seguridad, autenticación, y autorización. Estos ajustes aseguran que Elasticsearch esté optimizado para el entorno específico en el que se despliega, proporcionando un rendimiento y seguridad adecuados.

Kibana: Kibana también se configura mediante un archivo, típicamente llamado kibana.yml. En este archivo, se especifica la URL del servicio de Elasticsearch al que Kibana se conectará, así como otros parámetros de configuración como la personalización del interfaz de usuario y ajustes de seguridad. Este archivo es crucial para que Kibana pueda conectarse correctamente a Elasticsearch y proporcionar una interfaz interactiva y funcional para visualizar y analizar los datos.

2.5 CLUSTERIZACIÓN

Dentro de las técnicas descriptivas de Machine Learning basadas en análisis estadístico, encontramos el clustering. El clustering es una técnica de aprendizaje automático que tiene como objetivo agrupar conjuntos de datos en categorías o clústeres. Consiste básicamente en formar grupos cerrados y homogéneos a partir de un conjunto de elementos que tienen diferentes características o propiedades, pero que comparten ciertas similitudes. Esta técnica es utilizada para determinar patrones, agrupar ítems por temas o para segmentación, facilitando el análisis y la comprensión de grandes volúmenes de datos.

En informática, el término "clúster" es muy utilizado, aunque adquiere diferentes significados o aplicaciones dependiendo del contexto. En Elastic Stack, hacemos uso de la clusterización continuamente para la correcta realización del almacenamiento y búsqueda de datos de manera distribuida, asegurando así un rendimiento óptimo y alta disponibilidad. En Elastic Stack, un clúster consiste en un conjunto de uno o más nodos que trabajan juntos con el fin de almacenar y buscar datos de manera distribuida. Un nodo es una instancia de Elasticsearch que forma parte del clúster y maneja una parte de los datos y operaciones.

2.5.1 ELEMENTOS

Cada clúster tiene componentes esenciales que aseguran su funcionamiento eficiente y robusto. Los nodos, las unidades básicas del clúster, se dividen en varios tipos con roles específicos. El Nodo Maestro gestiona el clúster, incluyendo la creación y eliminación de índices, el seguimiento de nodos y la asignación de shards. Aunque puede haber múltiples nodos maestros, solo uno actúa como maestro activo en un momento dado.

El Nodo de Datos almacena los datos y realiza operaciones como búsquedas y agregaciones. La mayoría de los nodos en un clúster son nodos de datos debido a su rol crucial en la carga de trabajo principal. El Nodo Coordinador distribuye las solicitudes de búsqueda a los nodos de datos apropiados y combina los resultados. Cualquier nodo que no sea maestro o de datos puede actuar como nodo coordinador.

El Nodo Ingestor maneja el preprocesamiento de documentos antes de indexarlos, aplicando pipelines de ingestión. Esto transforma y enriquece los datos antes de su almacenamiento. Además, los shards y réplicas son componentes fundamentales del clúster. Los shards son las piezas básicas de los índices en Elasticsearch. Un índice se divide en varias shards, y cada shard es un índice autónomo que puede hospedarse en cualquier nodo del clúster. Las réplicas, copias redundantes de los shards, proporcionan alta disponibilidad y mejoran la tolerancia a fallos, asegurando que los datos estén disponibles si un nodo falla.

Para entender mejor cómo funcionan estos elementos, expondré un pequeño ejemplo. Supongamos que necesitamos organizar los libros en una biblioteca grande. Dividimos la biblioteca en secciones y asignamos bibliotecarios a cada sección. En este caso, el clúster es la colección completa de todos los nodos que trabajan juntos para llevar a cabo la gestión interna. Los shards son fragmentos del índice que se gestionan de manera independiente y se distribuyen entre nodos. Finalmente, cada nodo es el encargado de una sección específica de todo el conjunto.

En este ejemplo, la biblioteca representa el clúster de Elasticsearch, las secciones representan los shards de Elasticsearch y los bibliotecarios representan los nodos en un clúster, encargados de la gestión interna.

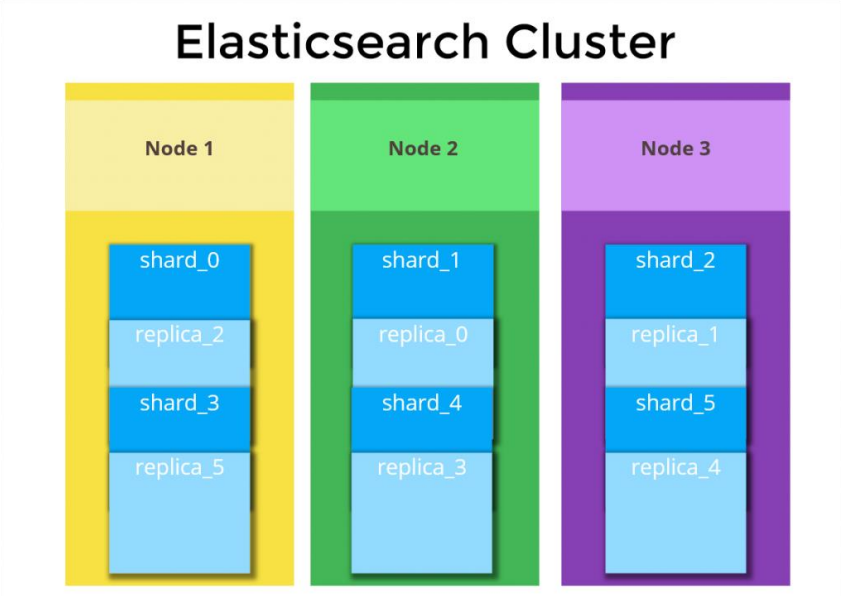


Ilustración 8 - Estructura de un clúster en Elasticsearch

2.5.2 BENEFICIOS

La clusterización es una parte vital de Elastic Stack y ofrece varias ventajas importantes que garantizan la eficiencia y robustez del sistema. Una de las ventajas más destacadas es la escalabilidad horizontal. La capacidad de añadir más nodos al clúster permite manejar mayores volúmenes de datos y cargas de trabajo. A medida que se agregan nodos, los datos y las solicitudes se distribuyen entre ellos, mejorando el rendimiento general del sistema. Esta característica es crucial para aplicaciones que deben escalar con el crecimiento de los datos y las demandas del usuario.

Otra ventaja significativa es la alta disponibilidad. Con shards y réplicas, los datos se distribuyen de tal manera que, incluso si uno o más nodos fallan, los datos siguen siendo accesibles a través de las réplicas en otros nodos. Esta redundancia asegura que los datos

estén siempre disponibles, minimizando el riesgo de pérdida de datos y mejorando la resiliencia del sistema.

La tolerancia a fallos es otra característica esencial proporcionada por la clusterización. Los nodos pueden fallar sin afectar significativamente la disponibilidad de los datos o el rendimiento del clúster. Elasticsearch maneja automáticamente la reasignación de shards y réplicas en caso de fallos, asegurando que el sistema continúe funcionando sin interrupciones. Esto es particularmente importante en entornos de producción donde la continuidad del servicio es crítica.

La distribución de la carga es otro beneficio clave. Las solicitudes de búsqueda y escritura se distribuyen entre todos los nodos del clúster, lo que mejora el rendimiento al evitar que un solo nodo se convierta en un cuello de botella. Al distribuir la carga de trabajo, Elasticsearch puede manejar un mayor número de solicitudes simultáneas, mejorando la capacidad de respuesta y la eficiencia del sistema

2.5.3 USO

La clusterización en Elastic Stack se utiliza ampliamente para garantizar la gestión eficiente de grandes volúmenes de datos, proporcionar alta disponibilidad y asegurar la tolerancia a fallos. En un entorno de producción, Elastic Stack utiliza la clusterización para distribuir los datos y las operaciones de búsqueda entre múltiples nodos. Esto no solo mejora el rendimiento, sino que también asegura que la aplicación siga funcionando incluso si uno o más nodos fallan.

Por ejemplo, en una aplicación de monitoreo de infraestructura, Elastic Stack puede recibir millones de eventos y logs por segundo. La clusterización permite que estos datos se distribuyan entre varios nodos, cada uno manejando una parte del total. Esto asegura que las búsquedas y las operaciones de análisis se realicen de manera eficiente y rápida. Además, en caso de una falla en uno de los nodos, las réplicas de los shards aseguran que los datos permanezcan accesibles y que el sistema siga funcionando sin interrupciones.

Además, la clusterización es esencial para el escalado horizontal. A medida que crecen las necesidades de almacenamiento y procesamiento de datos, se pueden añadir más nodos al clúster para aumentar su capacidad y rendimiento. Esto permite a las organizaciones manejar cargas de trabajo crecientes sin sacrificar el rendimiento o la disponibilidad.

En el ámbito de la seguridad de la información, la clusterización permite una vigilancia continua y detallada. Los logs y eventos de seguridad pueden ser recolectados y analizados en tiempo real, permitiendo a las organizaciones detectar y responder rápidamente a posibles amenazas. La capacidad de Elastic Stack para manejar grandes volúmenes de datos de manera eficiente es crucial para el monitoreo de seguridad en tiempo real.

2.6 HERRAMIENTAS PARA LA AUTOMATIZACIÓN DE ELK

En la gestión y operación de Elastic Stack, la automatización juega un papel crucial para garantizar la eficiencia, escalabilidad y facilidad de despliegue. Dos de las herramientas más destacadas para lograr estos objetivos son Kubernetes y Docker. Estas tecnologías permiten la contenerización y orquestación de aplicaciones, facilitando la implementación, el escalado y la administración de los componentes de Elastic Stack.

2.6.1 DOCKER

Docker es una plataforma de código abierto que permite la automatización del despliegue de aplicaciones en contenedores. Un contenedor es una unidad estándar de software que empaqueta el código y todas sus dependencias, permitiendo que la aplicación se ejecute de manera rápida y confiable en diferentes entornos informáticos.

A diferencia de las máquinas virtuales, los contenedores comparten el mismo sistema operativo del host, lo que los hace más ligeros y rápidos de iniciar. Este enfoque garantiza que las aplicaciones funcionen de manera uniforme en diferentes entornos, eliminando los problemas de "funciona en mi máquina" que a menudo surgen en el desarrollo y despliegue de software. Los contenedores proporcionan un entorno aislado, lo que permite ejecutar

múltiples aplicaciones en el mismo sistema sin interferencias, optimizando así el uso de recursos y mejorando la seguridad.

Docker ofrece diversas ventajas en el contexto de Elastic Stack. Proporciona un entorno aislado para cada componente de ELK, evitando conflictos entre diferentes versiones de software y dependencias. Además, los contenedores Docker pueden ejecutarse de manera consistente en cualquier entorno, ya sea en una máquina local, en un servidor o en la nube, garantizando así su portabilidad. Los contenedores son ligeros y consumen menos recursos en comparación con las máquinas virtuales, lo que permite una utilización más eficiente del hardware. Con Docker Compose, se pueden definir y ejecutar aplicaciones multicontenedor, lo que simplifica el despliegue de Elasticsearch, Logstash, Kibana y Beats.

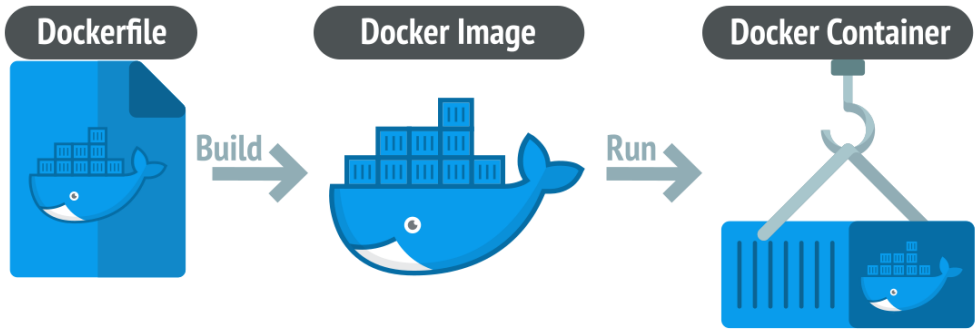


Ilustración 9 - Contenerización de Docker

2.6.2 KUBERNETES

Kubernetes es una plataforma de código abierto para la orquestación de contenedores, similar a Docker. Gestiona la automatización del despliegue, el escalado y la operación de aplicaciones en contenedores, proporcionando un entorno flexible para ejecutar aplicaciones contenerizadas de manera confiable y segura. Kubernetes ofrece numerosas ventajas para el desarrollo y la automatización de aplicaciones en Elastic Stack.

Primero, gestiona el ciclo de vida de los contenedores, asegurando su ejecución eficiente y escalado según la demanda. Además, facilita el despliegue y la actualización de aplicaciones mediante archivos de configuración YAML, simplificando estos procesos. También

proporciona alta disponibilidad mediante la replicación de contenedores y la recuperación automática en caso de fallos. Finalmente, puede escalar automáticamente los contenedores

Al igual que en la aplicación de prueba, la aplicación final del proyecto se probará mediante la contenedorización con Docker. Sin embargo, en el caso de la aplicación final, también se puede implementar eficazmente utilizando Kubernetes. Ambas opciones son viables y ofrecen buenos resultados. La aplicación es muy flexible y puede adaptarse fácilmente a Kubernetes, permitiendo una mayor escalabilidad y gestión de recursos.

2.7 COMPARATIVA CON OTRAS HERRAMIENTAS

En el mercado de análisis de datos y gestión de logs existen numerosas herramientas y todas validas como pueden ser Greylog, Loggly, Humio o Sumo Logic. Haciendo un estudio de mercado y analizando las más utilizadas por las empresas destacamos dos aplicaciones: Datadog y Splunk; ideales para empresas que buscan soluciones fáciles de usar con capacidades avanzadas de análisis y monitoreo. cada herramienta presenta sus ventajas y desventajas naturalmente, pero en este capítulo hare especial hincapié en la última mencionada, dado que se presenta como la mayor competencia a ELK.

Ilustración 10 - Gráfico comparativo de herramienta elegida por empresas

Splunk es una plataforma propietaria de seguridad y observabilidad. Está diseñada para indexar grandes cantidades de datos de máquina de una variedad de fuentes y proporcionar una gama de características para buscar, analizar y visualizar esos datos, con el fin de ofrecer valiosos conocimientos de observabilidad.

A menudo, las organizaciones usan Splunk para gestionar logs, análisis de seguridad, monitoreo de cumplimiento, análisis de negocios y mucho más para toda su infraestructura. Su lenguaje de búsqueda propietario, llamado Search Processing Language (SPL), se utiliza para consultar los conjuntos de datos recopilados, lo que facilita la creación de visualizaciones, alertas y dashboards a partir de datos no estructurados, un proceso que normalmente es tedioso. Cuenta con una amplia gama de características y funcionalidades, incluido el soporte para aprendizaje automático y otras capacidades avanzadas de análisis. Esto lo convierte en una opción popular para organizaciones de todos los tamaños e industrias.

A pesar de ser herramientas muy similares, hay algunas diferencias clave a tener en cuenta. Splunk es una herramienta propietaria que generalmente se considera más amigable para el usuario y más fácil de comenzar a usar que Elastic Stack. Está orientada hacia clientes empresariales, por lo que tiene una gama más amplia de características y funcionalidades, incluido el soporte para aprendizaje automático y otras capacidades avanzadas de análisis.

2.7.1 PRINCIPALES DIFERENCIAS

Para conocer ambas plataformas en profundidad y poder determinar si una u otra se adecua más a nuestras exigencias, realizare un a análisis exhaustivo de las principales características de cada una y compararé como se desenvuelven ambas plataformas según la prestación.

2.7.1.1 Ingesta de Datos

Antes de que ambas plataformas puedan trabajar con tus datos, primero debes enviarlos a ellas. Los datos de máquina suelen tener diferentes formatos y tipos, por lo que es crucial saber qué soporta cada plataforma y cómo llevar los datos desde tu entorno de aplicación hasta sus respectivos pipelines.

Splunk puede ingerir datos de máquina en varios formatos, incluidos XML, JSON, CSV, TSV y otros formatos estructurados o no estructurados. Además, enviar datos a Splunk es relativamente sencillo, ya que ofrece una variedad de opciones según la fuente de datos con la que estés trabajando. Entre ellas se incluyen: Servicio de Ingesta para recopilar objetos , Servicio de Reenviadores para recopilar datos a través de reenviadores, y Conectores de Streaming para recibir continuamente los datos emitidos por fuentes como Apache Kafka, Amazon Kinesis Data Stream, Google Cloud Pub/Sub, entre otros.

En la pila Elastic/ELK, Logstash es responsable de enviar los datos de máquina desde la fuente hasta el destino. Se utiliza típicamente para procesar grandes volúmenes de datos de logs, como logs de aplicaciones, logs de servidores web y logs de sistemas operativos. Trabaja con datos estructurados y no estructurados, y puede analizar y extraer detalles relevantes de los datos, que luego pueden transformarse y enriquecerse utilizando una variedad de filtros y plugins integrados. Además, ELK cuenta con otra forma más ligera de enviar datos conocida como Beats.

2.7.1.2 Indexación

Los índices se utilizan para organizar y buscar los datos ingeridos y se pueden comparar con una base de datos en un esquema de base de datos relacional. Esto significa que los documentos en un índice están típicamente relacionados entre sí. Por ejemplo, puedes tener un índice para usuarios y otro para productos.

Cuando se ingieren datos, se indexan automáticamente, lo que significa que se procesan y se añaden a uno o más índices. La indexación es una parte importante del pipeline de ingesta de datos, ya que permite que ambas plataformas busquen y recuperen datos de manera rápida y eficiente.

En Elasticsearch, cada índice se identifica a través de un nombre único mientras se realizan operaciones de indexación, búsqueda, actualización y eliminación contra los documentos en él. Los índices en Elasticsearch utilizan la estructura de datos de índice invertido, que almacena un mapeo del contenido (como cadenas o números) a uno o más documentos,

permitiendo encontrar las mejores coincidencias para búsquedas de texto completo incluso en grandes conjuntos de datos.

Splunk utiliza su componente indexador para procesar y organizar los logs enviados, extrayendo valores predeterminados como el host, la fuente del evento y el tipo de fuente, y configurando la codificación de caracteres. Los datos se descomponen en líneas y se les asignan marcas de tiempo para ordenar los eventos, permitiendo también el enmascaramiento de datos sensibles en esta etapa. Tras el análisis, los datos se segmentan en buckets que afectan la velocidad y la capacidad de búsqueda, se escriben en disco y se comprimen. Un beneficio clave del indexador de Splunk es que almacena múltiples copias de los datos para minimizar el riesgo de pérdida de información.

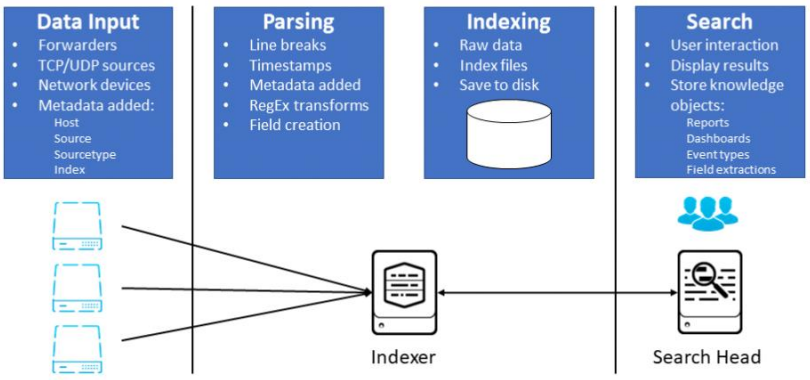


Ilustración 11 - Indexación en el Data Pipeline de Splunk

2.7.1.3 Visualización de Datos

La visualización de datos en la pila Elastic/ELK se maneja como ya sabemos con Kibana. Una vez que tienes tus datos indexados en Elasticsearch, puedes usar Kibana para crear una variedad de visualizaciones, como gráficos de líneas, gráficos de barras y gráficos de pastel, para obtener información sobre los datos. Cuando inicias sesión en Kibana, puedes consultar tus datos indexados y reducir los resultados a los datos específicos que deseas visualizar. Después, puedes elegir tu tipo de visualización preferido y usar las opciones en el editor para personalizar su apariencia.

La interfaz de Dashboards de Splunk también permite a los usuarios visualizar datos agregados. Los dashboards están compuestos por paneles que contienen varios módulos, como gráficos, gráficos de barras, cuadros de entrada, entre otros. Puedes conectarlos a una búsqueda guardada y ver los resultados en tiempo real a medida que se actualizan en segundo plano.

2.7.1.4 Configuración y Mantenimiento

Tanto Splunk como Elasticsearch son herramientas poderosas para el análisis y la visualización de datos. Sin embargo, el proceso de configuración de estas herramientas puede ser bastante diferente.

Elasticsearch como motor de búsqueda está diseñado para ejecutarse en múltiples servidores en un clúster. Para configurarlo, necesitas instalar el software en cada servidor del clúster y luego configurar los servidores para que se comuniquen entre sí. Esto generalmente implica configurar la red y los ajustes de seguridad, así como definir los roles y responsabilidades de cada servidor en el clúster.

Configurar Splunk es un poco más sencillo. Necesitas instalar el software de Splunk en un servidor y seguir las instrucciones del asistente de instalación. Esto generalmente implica especificar un directorio para los datos y logs de Splunk, así como configurar una cuenta de usuario y contraseña. Una vez completada la instalación, puedes iniciar sesión en la interfaz web de Splunk y comenzar a ingerir y analizar datos.

En resumen, configurar Splunk es más fácil y requiere menos experiencia técnica, mientras que Elasticsearch exige un entendimiento más profundo de los sistemas distribuidos.

2.7.1.5 Interfaz de Usuario

Splunk y Elasticsearch tienen interfaces de usuario basadas en la web, pero encontramos multitud de diferencias en términos de diseño y funcionalidad.

La interfaz de usuario de Splunk se centra en la analítica basada en búsquedas e incluye una barra de búsqueda en la parte superior de la pantalla. Esto te permite introducir consultas de

búsqueda y ver los resultados en tiempo real. La interfaz de usuario también incluye una serie de dashboards y visualizaciones preconstruidas que te permiten obtener rápidamente información sobre los datos ingeridos.

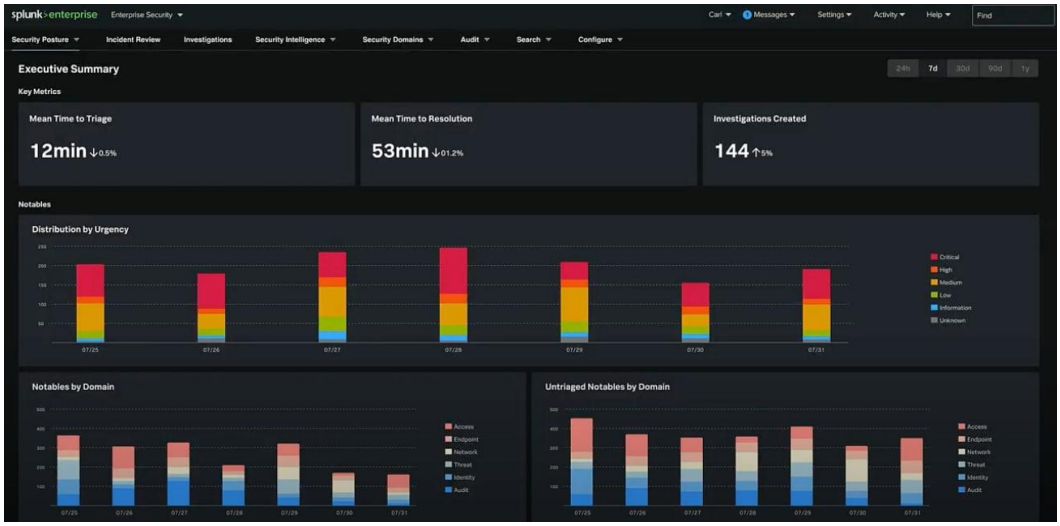


Ilustración 12 - Interfaz de usuario de Splunk

Por otro lado, la interfaz de usuario de Kibana se centra en el descubrimiento y la exploración de datos. Incluye una serie de herramientas de visualización y análisis de datos preconstruidas, como las aplicaciones Discover, Visualize y Dashboard. Estas herramientas te permiten explorar y analizar rápidamente tus datos sin necesidad de escribir consultas de búsqueda complejas.

2.7.2 OPINIÓN PERSONAL

En general, la elección entre Splunk y Elastic depende de las necesidades específicas de tu organización y los recursos disponibles para ti. Si necesitas una solución robusta y fácil de usar que pueda manejar una amplia gama de tareas de gestión y análisis de logs, Splunk puede ser la mejor opción. Si deseas una herramienta más personalizable y escalable que pueda adaptarse a tus necesidades específicas, ELK será la que mejor se adapte a tus necesidades.

Una vez realizado el proyecto y testeado cada funcionalidad de Elastic Stack al completo, puedo decir que ELK es la mejor opción debido a su gran flexibilidad y escalabilidad. La capacidad para manejar grandes volúmenes de datos y la profundidad de personalización que ofrece lo convierten en una herramienta extremadamente poderosa para la gestión y análisis de logs, superando a Splunk en términos de adaptabilidad y costo.

Capítulo 3. ESTADO DE LA CUESTIÓN

En la actualidad, la gestión y el análisis de grandes volúmenes de datos son esenciales para diversas industrias, desde el comercio electrónico hasta la salud, pasando por el transporte y la logística. Elastic Stack se ha posicionado como una de las soluciones más versátiles y eficientes en el mercado para abordar estos desafíos. Su capacidad para indexar, buscar y visualizar datos en tiempo real ha sido aprovechada en múltiples sectores, mejorando la toma de decisiones y la eficiencia operativa.

A continuación, se presentan algunas de estas soluciones disponibles en el mercado, destacando sus características y aplicaciones principales, así como un caso detallado en el sector del transporte.

3.1 ELASTIC STACK EN DIFERENTES SECTORES

Elastic Stack se utiliza en una amplia variedad de sectores debido a su flexibilidad, escalabilidad y capacidad para proporcionar análisis en tiempo real. A continuación, se detallan algunos de los sectores donde Elastic Stack resulta mas útil en la actualidad.

3.1.1 TECNOLOGIAS DE LA INFORMACION Y DEVOPS

En el sector de TI y DevOps, Elastic Stack es una herramienta crucial para el monitoreo y la gestión de logs. Permite a los equipos de TI recopilar, analizar y visualizar logs de servidores, aplicaciones y redes en tiempo real. Esto facilita la detección y resolución de problemas, mejora el rendimiento del sistema y asegura el cumplimiento de las políticas de seguridad. Kibana, con sus dashboards personalizables, permite a los equipos de DevOps monitorear el estado y el rendimiento de sus infraestructuras de manera eficiente.

3.1.2 SALUD

En el sector de la salud, Elastic Stack se utiliza para el análisis de datos clínicos y administrativos. Permite a las organizaciones de salud almacenar y buscar grandes volúmenes de datos médicos, incluidos registros de pacientes, resultados de pruebas y datos de monitoreo de pacientes en tiempo real. La capacidad de Elasticsearch para realizar búsquedas rápidas y detalladas ayuda a los profesionales de la salud a acceder rápidamente a la información necesaria para la toma de decisiones clínicas.

3.1.3 SEGURIDAD Y ANALISIS DE AMENAZAS

Elastic Stack es ampliamente utilizado en el sector de la seguridad para el análisis de amenazas y la gestión de incidentes. Elastic Security, una extensión de Elastic Stack, permite la recolección y análisis de datos de seguridad en tiempo real, ayudando a detectar actividades sospechosas y responder rápidamente a incidentes de seguridad. Los analistas de seguridad pueden utilizar Elasticsearch para buscar patrones y correlacionar eventos de seguridad, mientras que Kibana proporciona visualizaciones que facilitan la interpretación de estos datos.

3.1.4 FINANZAS Y BANCA

Las instituciones financieras utilizan Elastic Stack para monitorear transacciones, detectar fraudes y asegurar el cumplimiento normativo. La capacidad de Elastic Stack para manejar grandes volúmenes de datos transaccionales y realizar análisis en tiempo real es crucial para la detección de actividades fraudulentas y la gestión de riesgos. Además, las capacidades de búsqueda y visualización de Kibana permiten a los analistas financieros monitorear y analizar las tendencias del mercado y el comportamiento de los clientes.

3.1.5 TELECOMUNICACIONES

Finalmente, el sector más esperado: las telecomunicaciones. Elastic Stack se utiliza para monitorear redes, analizar datos de uso y mejorar la calidad del servicio. Permite a las empresas de telecomunicaciones recopilar y analizar datos de rendimiento de la red en

tiempo real, identificar y resolver problemas rápidamente, y optimizar la infraestructura de la red. Los dashboards de Kibana proporcionan una visión clara y detallada del estado de la red y del comportamiento del usuario.

3.2 UBER

Los servicios de Uber dependen de la precisión de sus herramientas de predicción y pronóstico de eventos. Desde estimar la demanda de pasajeros en una fecha específica hasta predecir cuándo llegará un pedido de UberEATS, Uber utiliza algoritmos de pronóstico para mejorar la experiencia del usuario en toda su cartera de productos.

Para llevar a cabo dicha experiencia de pronóstico precisa, Uber construye un sistema de predicción personalizado mediante un motor de búsqueda distribuido RESTful: ELK. El motor de consulta Elasticsearch, el pipeline de indexación de datos Logstash y la herramienta de visualización Kibana, se convierten en la arquitectura resultante del sistema de predicción de una empresa de lo más exitosa.

3.2.1 ESPECIFICACIONES

La calidad de las predicciones de Uber se mide por qué tan estrechamente coinciden con los resultados finales de las características del viaje. Para ello necesitaban una arquitectura altamente personalizable para lograr sincronizar el sistema de predicción con los productos existentes. Esto es así dado que Uber cuenta con ciertas complicaciones en el registro de servicios como puede ser tomar una ruta diferente en el camino sugerido o las características del viaje aplicables a según qué tipo de vehículo.

En Uber usan una combinación de datos históricos y en tiempo real de viajes para entrenar los algoritmos de predicción de viajes. Aunque pueden predecir algunos patrones como la densidad de viajes y la tasa de coincidencia según la ubicación, estos patrones se ajustan a medida que los sistemas avanzan y las operaciones se expanden a nuevos mercados. Como resultado, capturar la dinámica del sistema en tiempo real o casi en tiempo real es crítico para la precisión de la predicción.

Mientras evaluaban otras bases de datos que soportan consultas geoespaciales como MySQL, las demandas de flexibilidad en el esquema y requisitos de búsqueda hicieron que fuera una decisión fácil usar la herramienta Elastic Stack como solución para impulsar su almacén de datos y motor de búsqueda.

3.2.2 ARQUITECTURA DEL SISTEMA

A un nivel básico, la arquitectura de predicción de Uber se construyó utilizando Kafka para la transmisión de datos, Hive como un almacén de datos para gestionar consultas y análisis, y cuatro servicios internos separados que emplean el ELK Stack para crear un sistema de predicción robusto. A continuación, se describen estos cuatro servicios y se delinea la arquitectura general:

- Servicio de predicción: Proporciona predicciones en tiempo real a los usuarios.
- Servicio de entrenamiento: Entrena parámetros y tuberías de datos fuera de línea.
- Servicio de viajes: Gestiona el estado de los viajes a lo largo de su ciclo de vida, desde la solicitud hasta la finalización.
- Servicio de configuración: Almacena y sirve parámetros entrenados para otros servicios.

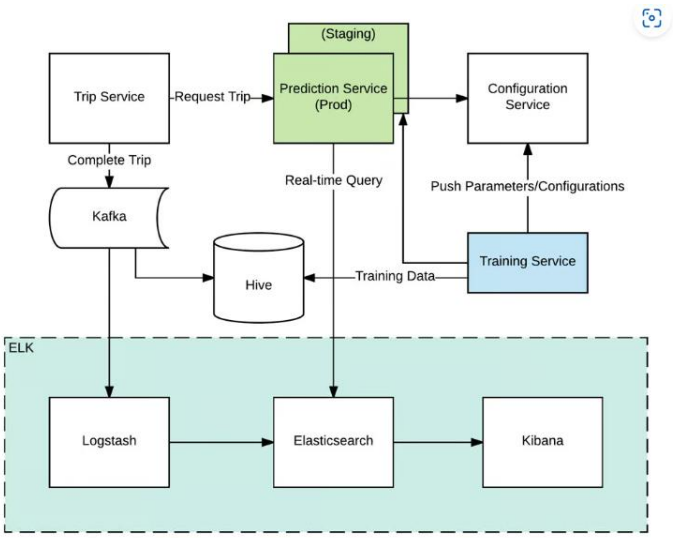


Ilustración 13- Arquitertura de real-time prediction system de Uber

3.2.3 APRENDIZAJE

Uber utiliza Elasticsearch como un almacén secundario, enviando datos a él de forma asincrónica, dado que no está diseñado para la consistencia. Para utilizar eficazmente sus valiosas funciones de búsqueda, es necesario operar Elasticsearch con cuidado. A continuación, se presentan las conclusiones que Uber extrajo para trabajar correctamente con la plataforma:

En primer lugar, es fundamental proveer suficientes recursos. Elasticsearch funciona de maravilla cuando se alimenta con suficientes recursos. Está diseñado para la velocidad y funciones potentes, asumiendo que el hardware y el personal son abundantes. Sin embargo, falla cuando los recursos son limitados. Para aprovechar al máximo Elasticsearch, es esencial comprender la escala con la que se está lidiando y proveer suficientes recursos. Esto incluye no solo el hardware adecuado sino también el personal capacitado para gestionar y optimizar el sistema de manera efectiva.

En segundo lugar, es crucial organizar los datos según la lógica de negocio. Cuando el tamaño de los datos es demasiado grande para caber en un solo nodo, es necesario organizar los datos inteligentemente. Esto implica dividir los datos de manera lógica y estructurada, lo que facilita el acceso y la consulta. Una buena organización de los datos no solo mejora el rendimiento de Elasticsearch, sino que también simplifica el mantenimiento y la escalabilidad del sistema.

Por último, es importante ordenar las consultas de Elasticsearch para ganar eficiencia. Elasticsearch soporta muchos filtros en sus consultas, y su orden afecta enormemente el rendimiento. Los filtros más específicos deben priorizarse y colocarse antes que los menos específicos para filtrar la mayor cantidad de datos lo antes posible en el proceso de consulta. Los filtros menos intensivos en recursos deben implementarse antes que los más intensivos en recursos para que tengan menos datos que filtrar. De manera similar, los filtros almacenables en caché deben colocarse antes que los no almacenables en caché para aprovechar al máximo las cachés. Esta optimización del orden de los filtros permite que las

consultas se ejecuten más rápidamente y con mayor eficiencia, mejorando la experiencia del usuario y reduciendo la carga en los servidores.

El nuevo sistema de predicción en tiempo real de Uber se ha implementado en 400 ciudades en todo el mundo y continúa creciendo, lo que demuestra la efectividad de Elastic Stack en cuanto a desarrollo y tecnológico para escalar y mejorar servicios globalmente.

Capítulo 4. DEFINICIÓN DEL TRABAJO

En este epígrafe, se explorarán las razones por las cuales Elastic Stack ha ganado una gran popularidad en el mercado, destacando su adopción por empresas de renombre y las ventajas que la hacen una herramienta indispensable en la gestión y análisis de datos. Se analizarán las justificaciones para elegir Elastic Stack como la plataforma central de este proyecto, así como los objetivos que se pretenden alcanzar mediante su implementación. Finalmente, se describirá la metodología que se seguirá para llevar a cabo el proyecto, así como los objetivos marcados para el mismo, asegurando una comprensión completa del proceso y los resultados esperados.

4.1 JUSTIFICACIÓN

Un estudio reciente de Black Duck Software encuestó a más de 1.300 individuos, desde analistas de negocios hasta ingenieros de software, acerca del tipo de programación empleado y sobre el futuro del código abierto. De los encuestados el 78% afirmo que ya sea parte, o la totalidad de todas las operaciones llevadas a cabo en la empresa funcionan con software de código abierto. Además, el 66% declaró que sus empresas emplean código abierto en el desarrollo de software para clientes.

ELK Stack se posiciona como un gran ejemplo de la tendencia hacia código abierto que viene sonando tan fuerte en los últimos años. Fue creada en 2010 y ya es empleada por numerosas empresas tecnológicas de gran prestigio como pueden ser Netflix, Stack Overflow, Linkedin o Accenture.

A pesar de la coexistencia de múltiples softwares con altas capacidades para llevar a cabo los objetivos de las empresas, ELK Stack se posiciona como una de las mejores y más seleccionadas por los grandes. A continuación, mencionare alguna de las funcionalidades relevantes de la plataforma que hizo que empresas lideres en el mercado se decantaran por ella y no por otras opciones.

En el caso de Netflix, La empresa eligió Elasticsearch por su capacidad de fragmentación y replicación automática, su esquema flexible, su excelente modelo de extensión y su ecosistema con muchos complementos. El uso de Elasticsearch por parte de Netflix para almacenar, indexar y buscar documentos ha crecido de un par de despliegues aislados a más de quince clústeres compuestos por casi 800 nodos que son gestionados centralmente por un equipo de ingeniería de bases de datos en la nube.

Para el caso de LinkedIn cuenta con una conocida historia de adopción de ELK. La red social enfocada en el mundo empresarial utiliza ELK para monitorear el rendimiento y la seguridad. El equipo de TI integra ELK con Kafka para gestionar su carga en tiempo real. Sus operaciones con ELK incluyen más de 100 clústeres distribuidos entre más de veinte equipos y seis centros de datos.

Por último, la empresa con más prestigio en hacer uso de los servicios seria Accenture. Accenture es una de las empresas de servicios de consultoría de TI más grandes del mundo, y en ella se lideran numerosos proyectos de implementación de ELK. En la presentación vinculada, el empleado de la empresa, Alexander Szalonnas, afirma que la empresa prefiere ELK Stack a Splunk porque es de código abierto, tiene una interfaz web simple y puede usar complementos para extender su funcionalidad.

A continuación, se analizarán las distintas funcionalidades y ventajas que nos presenta Elastic Stack que la hace ser tan cotizada para el desarrollo y mejora de las empresas.

4.1.1 BÚSQUEDA Y ANÁLISIS EN TIEMPO REAL

Elasticsearch, el núcleo de Elastic Stack, está diseñado para manejar volúmenes de datos masivos y a su vez realizar búsquedas rápidas. La capacidad para escalar horizontalmente permite que grandes empresas gestionen altas cantidades de datos sin comprometer el rendimiento. Por ello Elastic Stack se define como una de las herramientas de búsqueda más rápidas y escalables.

Como funcionalidad adicional, Elastic Stack cuenta con la capacidad de analizar datos en tiempo real. esto les proporciona una ventaja competitiva de gran valor, dado que permite a las organizaciones tomar decisiones informadas a detalle de manera inmediata

4.1.2 VERSATILIDAD EN LOS DATOS

4.1.2.1 Ingesta Flexible de Datos

Logstash y Beats permiten la ingesta de datos desde múltiples fuentes, como logs de servidores, aplicaciones, bases de datos, y más. Esta versatilidad facilita la integración de datos de diferentes sistemas y tecnologías, crucial para las organizaciones que manejan diversas fuentes de información. La combinación de Logstash y Beats no solo optimiza la ingesta de datos, sino que también asegura una gestión robusta y escalable, adaptándose rápidamente a los cambios en la infraestructura tecnológica y proporcionando una solución flexible para el análisis de datos en tiempo real.

4.1.2.2 Procesamiento y Transformación

Debido a la capacidad para aceptar entradas de una amplia variedad de fuentes, aplicar filtros personalizados y ejecutar salidas hacia múltiples destinos, Logstash nos permitirá una total versatilidad en el procesamiento de datos. Su arquitectura basada en plugins permite configurar pipelines de procesamiento que pueden filtrar, agregar, y transformar datos en tiempo real, adaptándose a las necesidades específicas de cada organización. Además, su capacidad para manejar formatos de datos complejos y realizar transformaciones avanzadas, como la normalización de datos, el enriquecimiento con información adicional y la eliminación de duplicados, facilita la preparación de datos para un análisis más preciso y eficiente. Esto lo convierte en una herramienta flexible y poderosa para gestionar y optimizar el flujo de datos en entornos diversos y cambiantes.

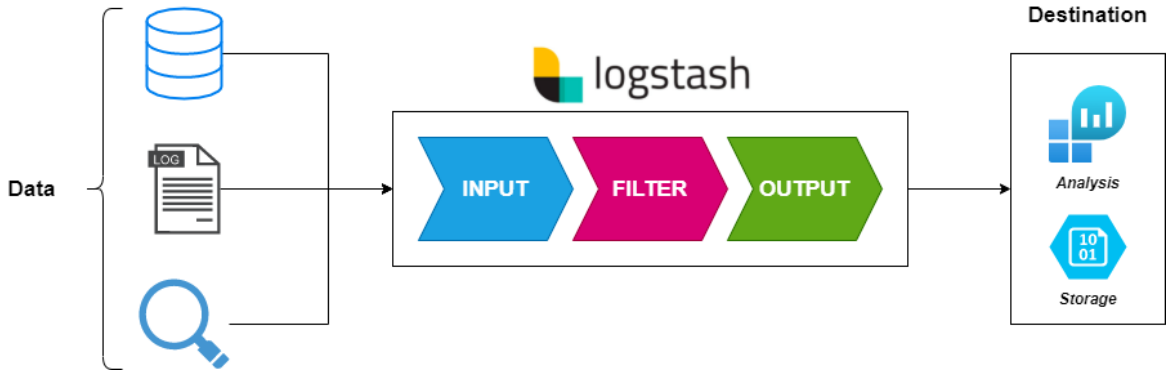


Ilustración 14 - Pipelines de procesamiento de Logstash

4.1.3 VISUALIZACIÓN INTUITIVA Y ACCESIBLE

Una vez contamos con toda la información procesada y almacenada, Elastic Stack nos brinda con una potente solución para la correcta visualización de todos los datos. Kibana se presenta como una herramienta avanzada capa de crear dashboards interactivos y gráficos detallados, facilitando el análisis visual de los datos y haciéndolo accesible tanto a usuarios técnicos como no técnicos.

Kibana también nos ofrece multitud de funciones de monitoreo y configuración personalizada de alertas, crucial para la gestión proactiva de sistemas y la identificación temprana de problemas.

Finalmente como funcionalidades adicionales, destacar que ofrece una interfaz de usuario intuitiva que permite a los usuarios crear visualizaciones complejas sin necesidad de programación, así como la integración de funciones de machine learning de Elastic, permitiendo detectar patrones y anomalías en los datos automáticamente.

4.1.4 ECOSISTEMA Y COMUNIDAD ACTIVA

Elastic Stack cuenta con un gran ecosistema de plugins y extensiones que nos permiten personalizar y ampliar sus funcionalidades con el fin de satisfacer las necesidades específicas para nuestro negocio. Gracias a este ecosistema ampliable logramos integrar herramientas

adicionales que mejoran la recopilación, el análisis y la visualización de datos. Además, podemos adaptar la plataforma para casos de uso específicos, como la seguridad de la información mediante el uso de Elastic Security, la observabilidad con Elastic Observability y la gestión de infraestructura a través de Elastic Infrastructure. Esta flexibilidad nos permite optimizar nuestros procesos y obtener insights más profundos y accionables, asegurando que la solución se mantenga alineada con nuestros objetivos empresariales y tecnológicos.

En adición a esto, Elastic Stack cuenta con una amplia comunidad de desarrolladores y usuarios que contribuyen continuamente al desarrollo y mejora de la plataforma. Esto garantiza acceso a recursos, soporte y mejoras prácticas.

4.1.5 COSTOS Y LICENCIAMIENTO

La mayoría de los componentes de Elastic Stack están disponibles bajo licencias de código abierto, lo que reduce significativamente los costos iniciales y permite una mayor flexibilidad en su implementación; esta naturaleza de código abierto no solo facilita el acceso y la experimentación con las herramientas, sino que también fomenta una comunidad activa que contribuye al desarrollo continuo de la plataforma.

Por otro lado, para las empresas que requieren soporte adicional y características avanzadas Elastic ofrece opciones de licenciamiento comercial, proporcionando un balance entre costo y funcionalidad. Estas opciones comerciales incluyen soporte técnico especializado, actualizaciones de seguridad prioritarias y acceso a características exclusivas que no están disponibles en las versiones de código abierto, lo que garantiza que las organizaciones puedan maximizar el valor de su inversión en Elastic Stack mientras mantienen un alto nivel de rendimiento y seguridad en sus operaciones diarias.

Además, Elastic proporciona modelos de suscripción flexibles que permiten a las empresas elegir el plan que mejor se adapte a sus necesidades específicas, ya sea para pequeñas implementaciones o grandes despliegues a escala empresarial, lo que asegura una escalabilidad y adaptabilidad adecuadas para cualquier tipo de proyecto.

4.2 OBJETIVOS

El objetivo principal del proyecto es aplicar los conocimientos adquiridos sobre Elastic Stack a través de un pequeño proyecto práctico. La meta es desarrollar una aplicación que permita recopilar diversos tipos de logs y analizarlos posteriormente.

Una de las principales ventajas de ELK Stack es su capacidad para el análisis de datos en tiempo real, lo que proporciona tanto una potente monitorización de datos como una mayor seguridad en los sistemas. Este proyecto busca poner en práctica estas ventajas mediante la creación de una página web con múltiples interacciones. A través de un sistema de login y varios botones de interacción, seremos capaces de recopilar una variedad de logs, incluyendo eventos de usuario y registros de errores, y visualizarlos en Kibana.

La finalidad de la aplicación es principalmente poner a prueba los conocimientos aprendidos de la plataforma Elastic Stack. El proyecto no solo se enfoca en la recolección de logs, sino también en demostrar la capacidad de Elastic Stack para proporcionar insights valiosos a través del análisis de estos datos. La implementación del sistema de login y los botones de interacción permitirá generar datos reales que reflejen el comportamiento del usuario y los posibles problemas del sistema, ofreciendo una visión práctica y detallada de cómo ELK Stack puede ser utilizado en un entorno real.

Una vez puesta a prueba la aplicación finalizaremos el informe con la aplicación propuesta por el director del proyecto. Una aplicación desarrollada por otro compañero y refinada por el coordinador se pondrá a mi disposición para el estudio de la misma, posibles mejoras o cambios que quiera realizar y finalmente el análisis de los diferentes tipos de logs que genere la propia aplicación.

Este proyecto tiene como objetivo demostrar la efectividad de Elastic Stack en la monitorización y análisis de logs, así como su capacidad para mejorar la seguridad y la eficiencia operativa en sistemas complejos. Se espera que, al finalizar, la aplicación sirva como un ejemplo tangible de cómo Elastic Stack puede ser integrado en proyectos reales para obtener una mejor comprensión y gestión de los datos.

4.3 METODOLOGÍA

El proyecto se desarrollará en varias etapas, cada una diseñada para garantizar una comprensión profunda y una implementación efectiva de Elastic Stack. En primer lugar, realizaremos un análisis exhaustivo de la plataforma, comprendiendo sus límites y funcionalidades al detalle. Esto incluirá un estudio de los fundamentos de Elasticsearch, el funcionamiento del registro de logs, la comunicación entre las diferentes clases del sistema y la visualización de los datos recopilados.

Toda la información recopilada ira plasmada a lo largo de este informe. Tratare de explicar de manera detallada todas las tecnologías empleadas, el funcionamiento de las mismas, formas de empleo o casos de uso y por último hasta dónde puede llegar ELK Stack en un futuro muy próximo.

Para la elaboración de la aplicación, será crucial establecer claramente las fases del proyecto y entender en todo momento el objetivo de la recopilación de datos. Comenzaremos por definir una estructura de proyecto y cómo distribuiremos las tareas. Una vez establecida la estructura, procederemos a la fase de codificación. La aplicación se desarrollará en Python, utilizando la contenerización con Docker. Esta metodología permitirá un desarrollo modular y escalable, facilitando la gestión y el despliegue de la aplicación.

El siguiente paso será desplegar la aplicación flask mediante Docker Desktop, lo que nos permitirá acceder a la página web en el puerto 5000. Este entorno de desarrollo contendrá todas las herramientas necesarias para la recolección y análisis de logs. Una vez que la aplicación esté en funcionamiento, todos los datos recopilados serán accesibles en el puerto 5601 a través de Elasticsearch. Para aprovechar al máximo las capacidades de la aplicación, será fundamental dominar Kibana, creando dashboards efectivos que permitan visualizar y analizar los datos de manera óptima.

La fase final del proyecto implicará pruebas exhaustivas para asegurar que la aplicación funciona correctamente y que los logs se recopilan y visualizan como se espera. Evaluaremos la seguridad y la monitorización del sistema, implementando medidas que garanticen la

integridad y la eficiencia operativa del entorno desarrollado. Esta etapa permitirá identificar posibles mejoras y ajustes necesarios, asegurando que la aplicación no solo cumple con los objetivos planteados, sino que también ofrece un rendimiento robusto y fiable en un entorno real.

Capítulo 5. SISTEMA/MODELO DESARROLLADO

Una vez comprendidos y asimilados todos los conceptos sobre el funcionamiento de la plataforma Elastic Stack y la infinidad de beneficios que nos puede aportar, en este capítulo plasmaremos esos conocimientos en un ejemplo práctico. En este apartado, explicaré paso a paso el desarrollo de la aplicación y los resultados que obtenemos de ella. Será crucial establecer un esquema claro para el desarrollo de la misma, asegurando que cada paso del proceso esté bien definido y documentado.

En primer lugar, se realizará un análisis exhaustivo de los diferentes tipos de logs que trataremos de registrar y gestionar. Este análisis incluirá la identificación de los eventos y datos clave que necesitamos monitorear, tales como registros de accesos, errores del sistema, eventos de usuario y cualquier otra información relevante para nuestro contexto específico. Este paso es fundamental, ya que nos permitirá establecer las bases del proyecto y definir claramente en qué consistirá.

Una vez descrito el sistema, se explicará detalladamente cómo funcionan las clases del proyecto y cómo se relacionan entre sí para llevar a cabo las tareas necesarias. Esto incluirá una descripción de la arquitectura de la aplicación, destacando cómo se estructuran los diferentes módulos y componentes, así como el flujo de datos desde la recopilación de logs hasta su almacenamiento y visualización. Cada clase y función será explicada con ejemplos prácticos, mostrando su propósito y cómo contribuyen al funcionamiento general del sistema.

Posteriormente, se presentará la aplicación web desarrollada en Python utilizando el framework flask, Se proporcionarán ejemplos detallados de cómo se registra cada tipo de log, desde la interacción del usuario hasta los errores del sistema. Se mostrarán capturas de pantalla y fragmentos de código para ilustrar cómo se configuran y envían los logs a Logstash, cómo se procesan y almacenan en Elasticsearch, y cómo se visualizan en Kibana.

Finalmente, se mostrarán los resultados obtenidos de la aplicación. Se explicará cómo se procesan y almacenan los logs en Elasticsearch, destacando la capacidad de esta herramienta para manejar grandes volúmenes de datos de manera eficiente. Además, se presentará la gran variedad de dashboards que ofrece Kibana para la visualización de estos logs. Se incluirán ejemplos de diferentes tipos de dashboards y gráficos que pueden ser utilizados para analizar los datos, identificar patrones, detectar anomalías y generar informes detallados.

Este capítulo tiene como finalidad proporcionar una visión completa y detallada del funcionamiento de Elastic Stack, así como demostrar la aplicabilidad y potencia de la herramienta en un entorno real. Ofrece una guía práctica para otros que deseen implementar soluciones similares a otros proyectos.

5.1 ANÁLISIS DEL SISTEMA

Dependiendo del contexto o necesidad de la organización, podemos encontrar una infinidad de tipos de logs para una aplicación. A lo largo de mi carrera, he trabajado con diferentes tipos de logs según los objetivos específicos del monitoreo o análisis a realizar. Estos incluyen logs de redes, logs de seguridad, logs de aplicaciones y logs de sistemas operativos, entre otros. Cada tipo de log registra eventos que proporcionan información valiosa para diversos propósitos, como mejorar la seguridad, optimizar el rendimiento y garantizar la integridad del sistema.

En este proyecto, decidí centrarme en logs que sean muy visuales y accesibles para todos los públicos. En el capítulo 4, hice especial hincapié en la gran utilidad que proporciona el monitoreo de logs para asuntos de seguridad y análisis en tiempo real. Por ello, he orientado la aplicación hacia ejemplos simples que pongan en práctica estos conceptos, demostrando cómo Elastic Stack puede ser de gran ayuda para el análisis y gestión de logs.

El caso más gráfico y práctico, a mi parecer, consiste en el monitoreo de logs de seguridad en una aplicación. Es fundamental tener control en todo momento sobre quién accede a tu plataforma, quién se registra y quién intenta acceder de manera intrusiva o por error. Por

ello, la primera parte de la aplicación consistirá en un sistema de login para la página web. Se ofrecerán dos páginas al cliente: una para que pueda introducir su usuario y contraseña, y otra para darse de alta si no está registrado. Todos los eventos generados por el cliente serán recogidos y analizados posteriormente.

A través de los logs, se organizarán los eventos según si el cliente hace login, se registra, o si hay un error de acceso debido a un usuario no existente. Todos los usuarios serán almacenados en una base de datos MySQL ubicada en la misma carpeta del proyecto. Mediante este pequeño sistema de login, logramos un ejemplo perfecto de cómo Elastic Stack nos permite gestionar la seguridad en las aplicaciones con un simple monitoreo de logs generados por el sistema.

Para completar la aplicación y manejar otros tipos de logs, he decidido implementar el monitoreo de logs de aplicaciones, como transacciones o errores. En el dashboard principal, el cliente contará con diferentes botones y enlaces a vínculos. Por cada botón pulsado, se generará un log con el contenido de este, el cual podrá ser visualizado posteriormente en gráficos que muestren cuál es el más frecuentado según las interacciones del cliente. En cuanto a los enlaces, consistirán en accesos directos a algunas páginas web, donde el sistema recogerá a través de logs el tiempo que tarda en cargarse cada página. Gracias a esto, podemos monitorear el tiempo de carga, la operatividad del sitio web y la velocidad de conexión a la red.

El registro de estos tipos de logs no solo permite una mejor gestión de la seguridad y la operatividad de la aplicación, sino que también proporciona insights valiosos para la optimización continua.

5.2 DISEÑO

A continuación, se detallará la estructura interna del proyecto y cómo funcionan cada una de las clases para lograr el flujo de datos y obtener los resultados esperados. Este capítulo cubrirá la organización del proyecto, la elección del lenguaje de programación, el uso de Docker para la contenerización, las clases que componen el proyecto, y las tareas específicas que realiza cada una de ellas.

5.2.1 APLICACIONES EMPLEADAS

Para el correcto desarrollo de la aplicación, tuve la opción de elegir entre numerosas plataformas de código y aplicaciones donde levantar el proyecto. A continuación, explicaré mis elecciones y el porqué de cada una.

5.2.1.1 Python

He elegido Python como el lenguaje de programación para este proyecto debido a varias razones. Python es un lenguaje de programación versátil y poderoso, adecuado tanto para proyectos pequeños como para aplicaciones a gran escala. Su naturaleza de alto nivel y su sintaxis limpia y legible hacen que el desarrollo sea más accesible y eficiente. Esta simplicidad facilita la lectura y escritura del código, permitiendo a los desarrolladores centrarse en resolver problemas y construir funcionalidades en lugar de luchar con la complejidad del lenguaje.

Además, Python cuenta con una extensa biblioteca de paquetes y módulos que cubren prácticamente cualquier necesidad que pueda surgir durante el desarrollo. Para este proyecto, se ha utilizado flask para el desarrollo web, así como otras bibliotecas para la integración con Elasticsearch y la gestión de logs. La comunidad activa de Python es otro factor decisivo, garantizando un soporte continuo y una amplia disponibilidad de recursos y documentación.

Finalmente, Python se integra fácilmente con Docker, permitiendo una contenerización eficiente de las aplicaciones. Esto facilita el despliegue y la escalabilidad del proyecto, asegurando que la aplicación pueda ejecutarse de manera consistente en cualquier entorno.

5.2.1.2 Docker

Docker se ha utilizado en este proyecto para contenerizar la aplicación flask, lo que ofrece varias ventajas. Docker es una plataforma de software que permite empaquetar aplicaciones y sus dependencias en contenedores, asegurando que funcionen de manera uniforme en cualquier entorno.

Docker garantiza que la aplicación se ejecute de manera consistente en cualquier entorno, ya sea en desarrollo, pruebas o producción, mediante la creación de contenedores que incluyen todas las dependencias necesarias. Los contenedores Docker proporcionan un entorno aislado para la aplicación, evitando conflictos con otras aplicaciones o servicios que puedan estar ejecutándose en el mismo sistema.

Además, Docker facilita la escalabilidad horizontal, permitiendo ejecutar múltiples instancias de la aplicación en diferentes contenedores para manejar mayores cargas de trabajo. Con Docker, el proceso de despliegue es mucho más sencillo y rápido, ya que se puede automatizar gran parte de este utilizando herramientas como Docker Compose. Esto reduce significativamente el tiempo y el esfuerzo necesarios para poner en marcha y mantener las aplicaciones en diferentes entornos.

5.2.2 ESTRUCTURA DEL PROYECTO

Una vez presentadas las herramientas de trabajo, explicare que clases serán necesarias para lograr el funcionamiento de la aplicación. En la siguiente ilustración se puede observar cómo están definidas y organizadas las diferentes clases y carpetas

Ilustración 15- Estructura del proyecto de prueba

En ella se pueden apreciar clases con las que hemos ido trabajando a lo largo del grado como son los archivos HTML y los scripts de Python, y otras nuevas, como los archivos .yml o .conf.

En primer lugar, se analizaran y explicaran las clases que resultan indispensables en todo proyecto Elastic Stack.

- **Docker-compose.yml**

El archivo Docker-compose.yml es crucial para definir y ejecutar aplicaciones Docker multicontenedor. Facilita la configuración y orquestación de múltiples servicios Docker, lo que simplifica enormemente el despliegue y la gestión de aplicaciones complejas. En nuestro caso, este archivo se utiliza para iniciar los contenedores necesarios, como la aplicación Flask, Elasticsearch, Logstash, Filebeats y Kibana. Define cómo estos contenedores interactúan entre sí y asegura que todos los componentes de la aplicación se inicien en el orden correcto, manteniendo la coherencia y funcionalidad del sistema.

- **filebeat.yml**

El archivo filebeat.yml es fundamental para la recolección y envío de logs en Elastic Stack. Filebeat es un componente ligero que se instala en el servidor donde se generan los logs y

está configurado para enviar estos logs a Logstash o directamente a Elasticsearch. Este archivo de configuración define cómo Filebeat debe recopilar y enviar los logs generados por la aplicación flask, especificando las rutas de los archivos de log y los detalles del host de Logstash.

- **Logstash.conf**

El archivo logstash.conf es esencial para el procesamiento de datos en Logstash. Logstash es una herramienta flexible de procesamiento de datos que puede recibir datos de diversas fuentes, transformarlos y luego enviarlos a Elasticsearch. Este archivo de configuración define cómo Logstash debe procesar los datos recibidos de Filebeat y enviarlos a Elasticsearch. Incluye la definición de entradas, filtros para transformar los datos, y salidas.

El resto de clases que componen el proyecto son clases con las que ya estamos familiarizados, como app.py y los templates HTML. Estas clases son las que darán forma a la aplicación web ya sea mediante una interfaz gráfica o aplicándole la lógica al programa.

El archivo app.py contiene toda la lógica del programa. Aquí se manejan las rutas de la aplicación, la creación de usuarios, la redirección a las páginas web y la generación de distintos eventos cuando se pulsan los botones. Este script es el núcleo de la aplicación flask, manejando tanto las operaciones del lado del servidor como la interacción con la base de datos y los logs.

Los templates HTML proporcionan la interfaz de usuario del sistema. Estos archivos incluyen dashboard.html, layout.html, login.html y register.html. Cada template define la estructura y el diseño de las páginas web, incluyendo las ventanas emergentes y los botones a pulsar. La interacción del usuario con estos elementos desencadena eventos que se registran y procesan en el backend.

El archivo de base de datos database.db es crucial para almacenar la información de los usuarios, incluyendo sus contraseñas encriptadas. Esta base de datos se utiliza para autenticar a los usuarios y gestionar sus perfiles dentro de la aplicación.

El archivo flask_app.log registra todos los logs generados por la aplicación Flask. Este archivo es monitoreado por Filebeat, que recoge los logs y los envía a Logstash para su procesamiento posterior. Los logs incluyen información sobre eventos como inicios de sesión, registros de usuarios y errores que son estructurados de cierta manera para tratar con ellos con mayor facilidad posteriormente.

Finalmente mencionar archivos del programa vitales para el correcto funcionamiento como son Dockerfile y requirements.txt. El archivo Dockerfile define cómo construir la imagen Docker para la aplicación flask. Especifica las instrucciones para instalar Python, flask y otras dependencias necesarias, asegurando que la aplicación pueda ejecutarse dentro de un contenedor Docker. El archivo requirements.txt lista todas las dependencias de Python necesarias para la aplicación. Esto incluye flask y cualquier otra biblioteca utilizada en el proyecto.

5.2.3 FLUJO DE DATOS

El flujo de la aplicación desde que el cliente genera un log en la página web pulsando un botón hasta su visualización en Kibana involucra varios componentes y pasos cruciales. A continuación, se detalla cada etapa del proceso, explicando las funciones y tareas que realiza cada clase y herramienta involucrada.

Cuando un cliente interactúa con la aplicación web, por ejemplo, pulsando un botón en el dashboard, se desencadena un evento en la aplicación. Este evento es capturado por la lógica de la aplicación definida en el archivo app.py. En este ejemplo, cada vez que el usuario pulsa un botón, se registra una acción en los logs. El método logger.info será el utilizado para escribir la información del evento en el archivo de logs flask_app.log

```
logstash      | {
logstash      |        "message" => "User ricardo logged in",
logstash      |          "event" => "login",
logstash      |       "@version" => "1",
logstash      |           "path" => "/logs/flask_app.log",
logstash      |     "@timestamp" => 2024-06-20T10:14:22.307Z,
logstash      |           "host" => "5f4c0ff8b0d5"
logstash      | }
```

Ilustración 16 - Log generado por evento

El log generado por la acción del usuario se almacena en el archivo logs/flask_app.log. Este archivo actúa como el repositorio inicial para todos los eventos registrados por la aplicación. Cada línea en este archivo representa un evento capturado, incluyendo la información relevante como la acción del usuario, la fecha y hora, y cualquier otro dato que se haya decidido registrar.

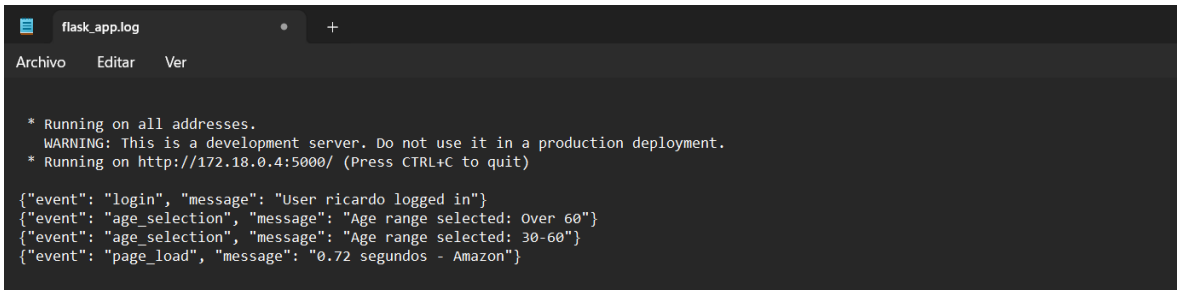


Ilustración 17 - Captura de archivo "flask_app.log"

Filebeat está configurado para leer los logs desde flask_app.log y enviarlos a Logstash. Filebeat monitorea continuamente el archivo de logs flask_app.log, por lo que cada vez que se escribe una nueva entrada en el archivo, Filebeat la detecta y la envía a Logstash a través del puerto 5044.

Logstash recibe los logs enviados por Filebeat, los procesa y los envía a Elasticsearch. La configuración de Logstash especifica cómo debe manejar y transformar los datos antes de enviarlos a Elasticsearch. Una vez Logstash recibe los datos de Filebeat, se pueden aplicar filtros para transformar los datos, como eliminar campos innecesarios, modificar el formato de los datos, o enriquecer los logs con información adicional. Finalmente, los datos procesados se envían a Elasticsearch, donde se almacenan en índices específicos.

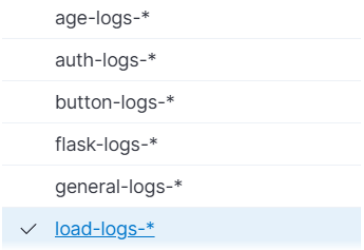


Ilustración 18 – Índices para distintos tipos de Logs

Elasticsearch actúa como la base de datos para todos los logs procesados. Una vez que Logstash envía los datos, Elasticsearch los almacena en índices, lo que permite búsquedas rápidas y eficientes. Los logs almacenados en Elasticsearch están organizados en documentos dentro de índices. Cada documento representa un evento registrado por la aplicación, y los índices permiten agrupar y gestionar estos documentos de manera eficiente.

Por último, Kibana nos proporciona una interfaz gráfica para buscar, visualizar y analizar los datos almacenados en Elasticsearch. Los usuarios pueden acceder a Kibana a través de un navegador web en el puerto 5601. El potencial de Kibana lo explicare más en detalle en el último apartado mostrando ejemplos de gráficos y dashboards con los resultados.

5.3 IMPLEMENTACIÓN

Una vez entendido el funcionamiento de la aplicación y las herramientas empleadas, mostrare la interacción con la aplicación a modo de cliente. Se mostrarán capturas de toda la aplicación, así como todas las interacciones posibles para el registro de los diferentes logs. En primer lugar, para arrancar la aplicación será necesario levantar los contenedores de Docker mediante el comando – *Docker-compose up –build* – desde la command window. Una vez se construye el proyecto podremos dirigirnos a el puerto 5000 localhost donde se situará la interfaz web.

En primera instancia como se explicó anteriormente encontraremos el sistema de login y registro

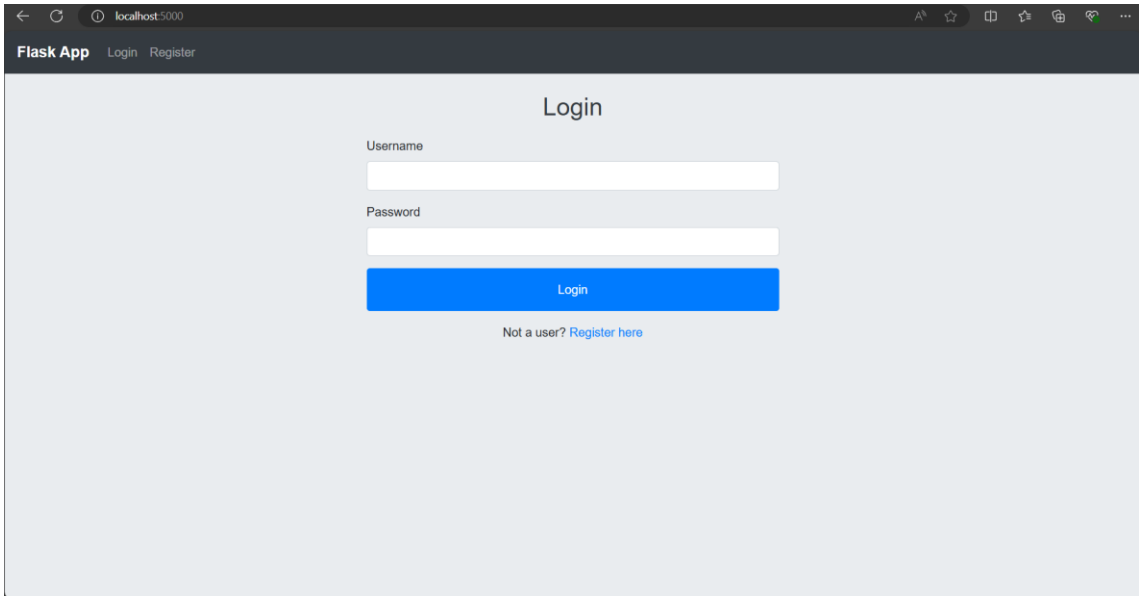


Ilustración 19 – Sistema de login en aplicación de prueba

Si aun el usuario no tiene credenciales registradas en la base de datos del programa, debe hacer click en *Register here* para poder darse de alta en la aplicación y acceder a los servicios

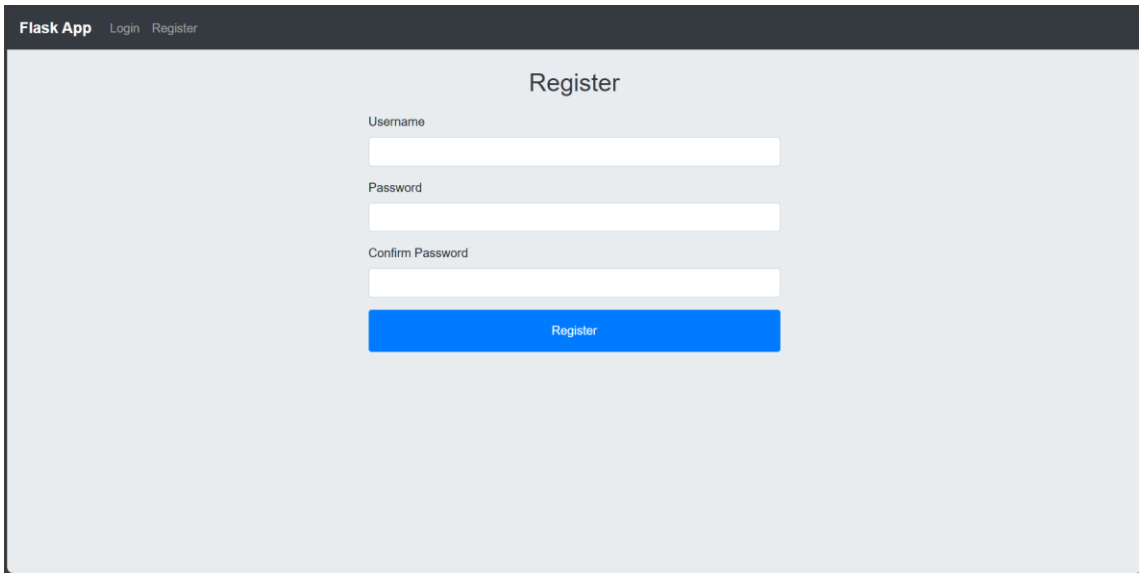


Ilustración 20 - Sistema de registro en aplicación de prueba

Una vez introduce usuario y contraseña disponibles, atendiendo a las condiciones mínimas requeridas para la contraseña, se le redirigirá al login junto con una ventana emergente de notificación de cuenta creada con éxito. El registro de un usuario en la base de datos generara un log de tipo "register".

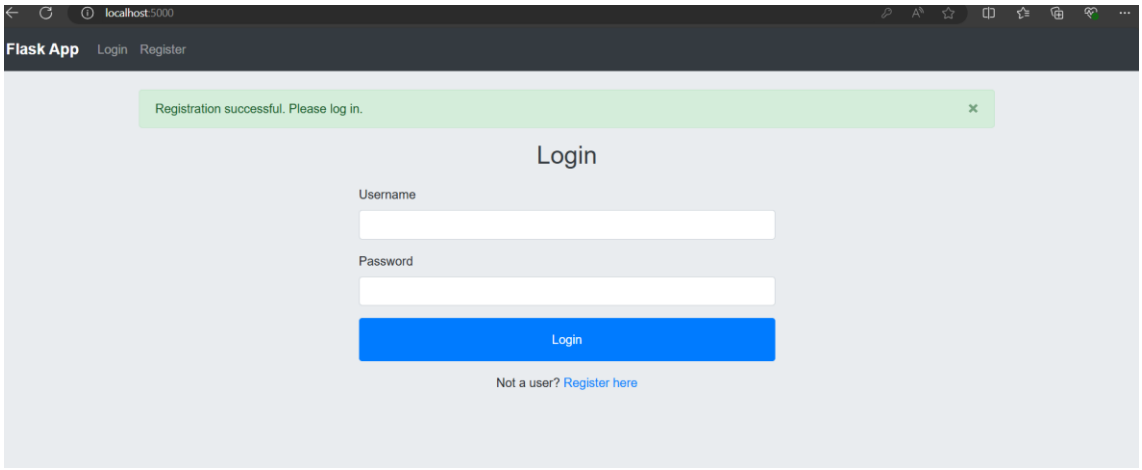


Ilustración 21 - Usuario registrado correctamente

Si un usuario introduce mal sus credenciales o no existe en la base de datos saltara una ventana emergente notificando del error.

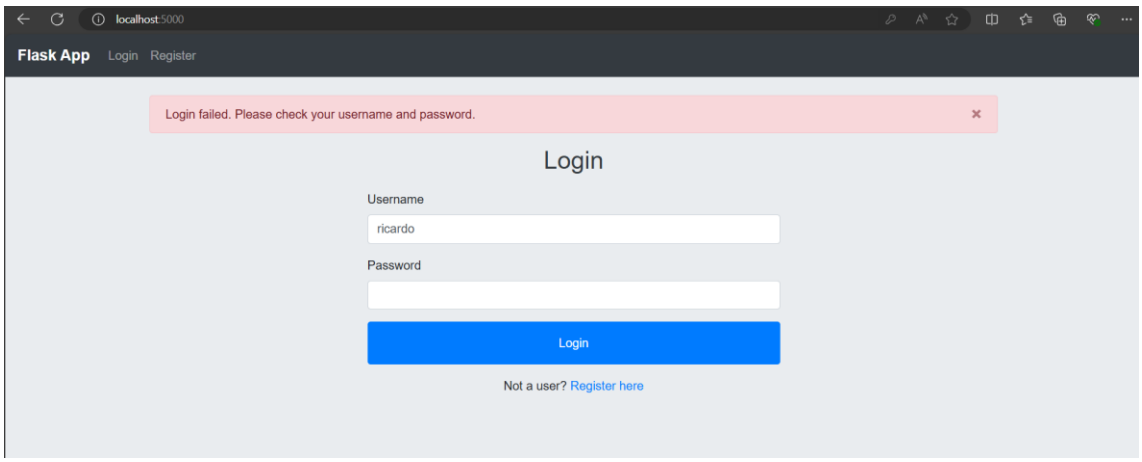


Ilustración 22 - Error en el login de usuario

De nuevo un error de login generará un log de tipo "login_error" que será registrado en Elasticsearch.

Una vez accedido correctamente al sistema aparecerá la siguiente interfaz con diferentes tipos de botones.

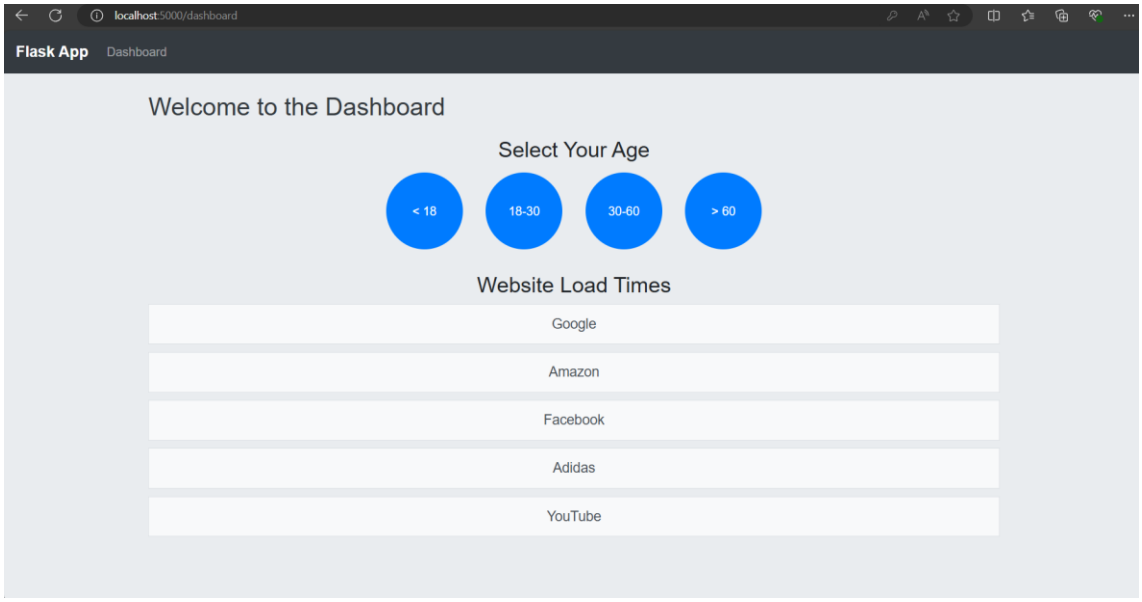


Ilustración 23 - Dashboard principal de aplicación de prueba

La aplicación cuenta con un único dashboard en el que registraremos diferentes eventos según los botones pulsados. En primer lugar, encontramos en la parte superior cuatro botones que hacen referencia a la selección de edad. Una vez pulsamos un rango de edad, se queda marcada la selección y se registrara un log del tipo "age_selection".

Por último, se pueden observar cinco botones con diferentes nombres de empresas. Estos botones consisten en accesos directos a las páginas web de dichas empresas. Una vez pulsado uno, se te redirigirá automáticamente allí quedando registrado el tiempo que tarda en cargar la misma página web. El tiempo de carga de una página resulta un evento de lo más practico dado que nos puede indicar mucha información en cuanto a latencia, conexión o disponibilidad. Ese tiempo de carga quedara registrado como un log del tipo "page_load".

Para entender más a fondo el funcionamiento de la aplicación se mostrará un ejemplo práctico mostrando todos los logs que quedan registrados. Todo evento que se registra se

puede ir monitoreando a su vez desde los logs que va procesado el contenedor de Logstash. Para el ejemplo seguiremos el guión explicado hasta ahora en la interfaz.

Un usuario no cuenta con credenciales para el acceso de la app por lo que se registra

```
logstash        | {
logstash        |         "message" => "User ricardo registered",
logstash        |           "event" => "register",
logstash        |        "@version" => "1",
logstash        |            "path" => "/logs/flask_app.log",
logstash        |      "@timestamp" => 2024-06-20T09:58:10.593Z,
logstash        |            "host" => "5f4c0ff8b0d5"
logstash        | }
```

De la captura podemos extraer 2 índices: “message” y “event”. El campo “message” nos dará la información acerca del log, en este caso un registro y el autor del mismo. El campo “event” nos indicara el tipo de log registrado.

```
logstash        | {
logstash        |         "message" => "Failed login attempt for user ricardo"
,
logstash        |           "event" => "login_error",
logstash        |        "@version" => "1",
logstash        |            "path" => "/logs/flask_app.log",
logstash        |      "@timestamp" => 2024-06-20T10:13:11.812Z,
logstash        |            "host" => "5f4c0ff8b0d5"
logstash        | }
```

```
logstash        | {
logstash        |         "message" => "User ricardo logged in",
logstash        |           "event" => "login",
logstash        |        "@version" => "1",
logstash        |            "path" => "/logs/flask_app.log",
logstash        |      "@timestamp" => 2024-06-20T10:14:22.307Z,
logstash        |            "host" => "5f4c0ff8b0d5"
logstash        | }
```

Aquí podemos ver dos tipos de eventos nuevos registrados. Estructurar los logs según el tipo de evento será crucial para poder visualizar después correctamente los dashboards de Kibana.

```
logstash    | {
logstash    |         "message" => "Age range selected: 30-60",
logstash    |           "event" => "age_selection",
logstash    |        "@version" => "1",
logstash    |            "path" => "/logs/flask_app.log",
logstash    |      "@timestamp" => 2024-06-20T10:16:39.055Z,
logstash    |            "host" => "5f4c0ff8b0d5"
logstash    | }
```

```
logstash    | {
logstash    |         "message" => "0.72 segundos - Amazon",
logstash    |           "event" => "page_load",
logstash    |        "@version" => "1",
logstash    |            "path" => "/logs/flask_app.log",
logstash    |      "@timestamp" => 2024-06-20T10:17:14.263Z,
logstash    |            "host" => "5f4c0ff8b0d5"
logstash    | }
```

Finalmente los botones de la página principal, donde podemos diferenciar el evento de registro de edad, y el tiempo de carga de la aplicación, con el tiempo de carga especificado en el campo del mensaje.

5.4 SALIDAS DEL SISTEMA

Por último, la visualización de dichos eventos en su conjunto. Una vez registrados suficientes logs para darle funcionalidad a la aplicación, nos dirigimos al puerto 5601 (Elasticsearch) donde estarán almacenados todos estos eventos y donde podremos visualizarlos a través de gráficos o paneles.

Una vez nos situamos en la página web de Elastic, nos dirigimos a stack monitoring- kibana – index patterns donde podremos visualizar los índices en los que organizamos los distintos tipos de logs, así como crear nuevos.

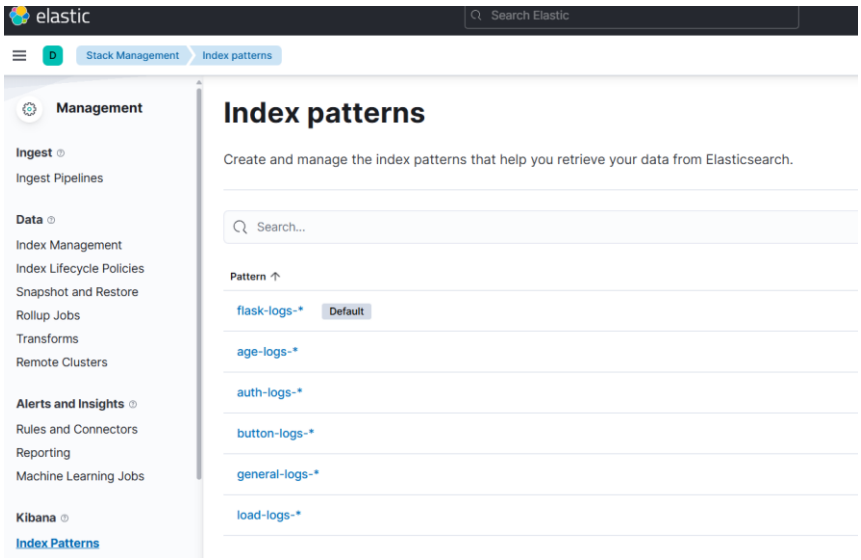


Ilustración 24 - Creación de patrones de índice en Kibana

En este caso podemos ver 6 diferentes tipos de índices en los que estructuramos los logs. Los eventos explicados hasta ahora se organizarán entre age-logs (rango de edad), auth-logs (login, error de login y registro) y load-logs (tiempo de carga). General-logs recogerá el resto de eventos que no he explicado hasta ahora dado que en una interacción de usuario se registran miles de logs como puede ser un 200ok o interacción con la página de Elasticsearch. Por ùltimo, flask-logs y button-logs que fueron utilizados para pruebas unitarias.

Una vez conocidos los distintos índices, nos dirigimos a la página analytics-discover dentro de elasticsearch donde podremos visualizar todos los logs registrados.

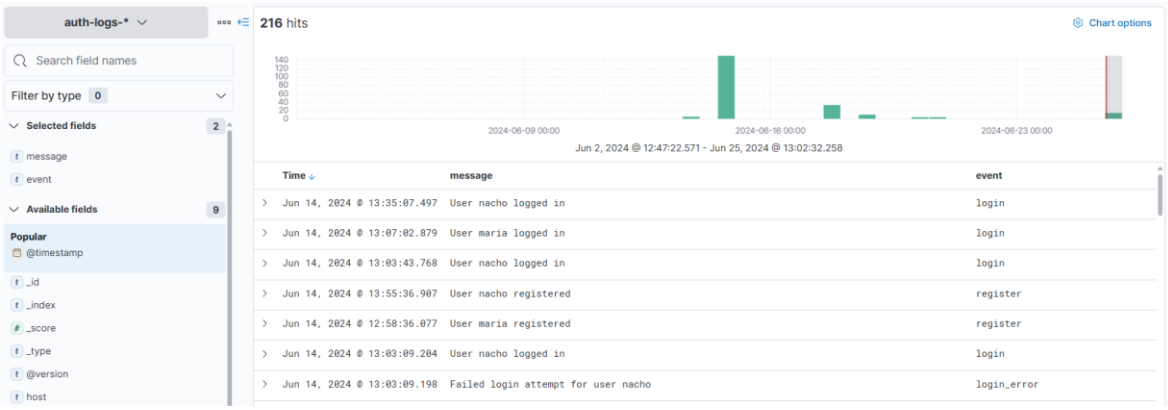


Ilustración 25 - Registro de logs en Elasticsearch

En este caso vemos todos los logs de tipo auth. Podemos aplicar filtros para visualizar diferentes campos del log registrado, así como marcar un intervalo de tiempo para visualizar únicamente lo registrado en ese marco.

>	Jun 25, 2024 @ 12:48:07.065 User ricardo logged in	login
>	Jun 25, 2024 @ 12:48:07.059 Failed login attempt for user ricardo	login_error
>	Jun 25, 2024 @ 12:48:07.045 User ricardo registered	register

Y efectivamente vemos registrado los 3 logs que generamos con el ejemplo anteriormente.

Para exprimir a fondo la aplicación podremos visualizar todos estos datos con gráficos y demás como se ha ido explicando a lo largo del proyecto. Dirigiéndonos al apartado de Kibana se nos ofrecerán distintas formas de proyectar la información.

Podemos visualizar los distintos eventos de autenticación en un rango de tiempo con un gráfico de barras, distinguiendo el evento por colores

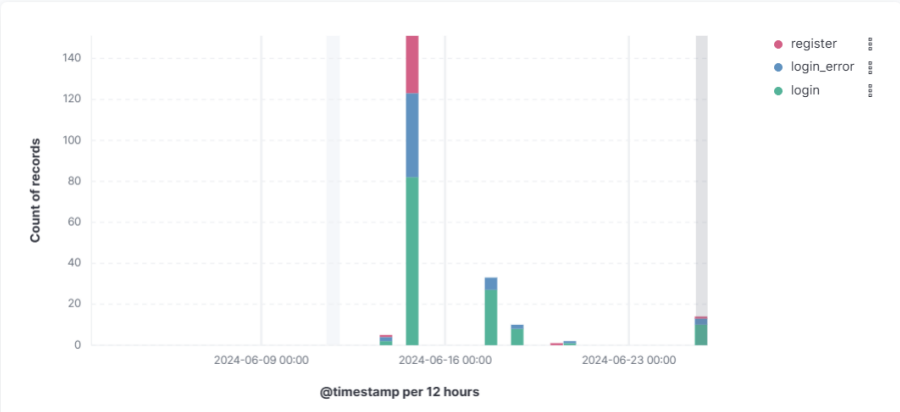


Ilustración 26 - Visualización en Kibana I

Podemos visualizar como se reparten los rangos de edad de nuestros clientes por porcentajes generare dos tipos diferentes de gráficos para conocer mas a fondo las capacidades de kibana. De esta manera vemos claramente que rango de edad abunda mas de dos formas.

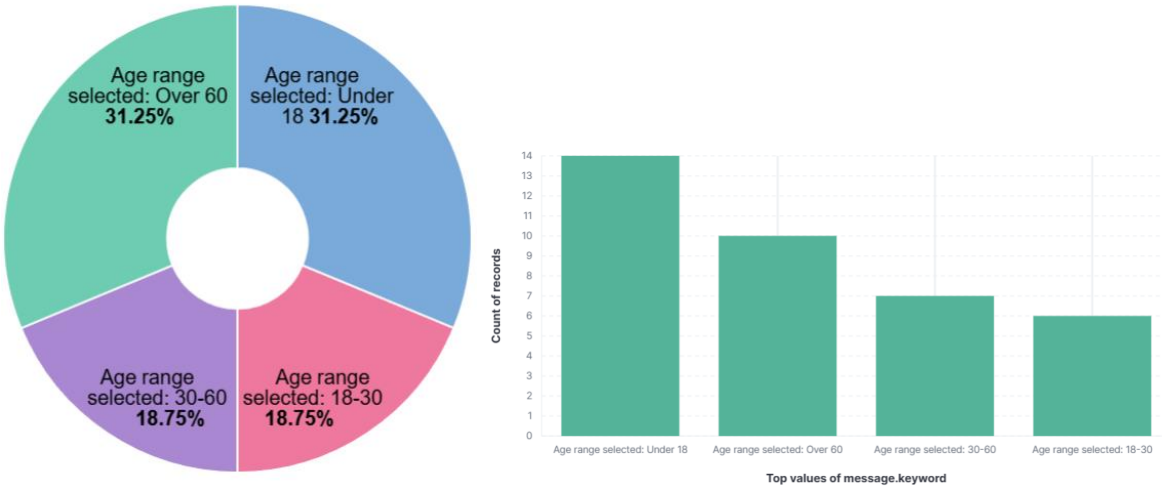


Ilustración 27 - Visualización en Kibana II

Finalmente visualizamos el tiempo de carga de las diferentes páginas web. Gracias a Kibana podemos visualizar los tiempos medios de carga y extraer nuestras conclusiones en cuanto a velocidad, servicio y demás.

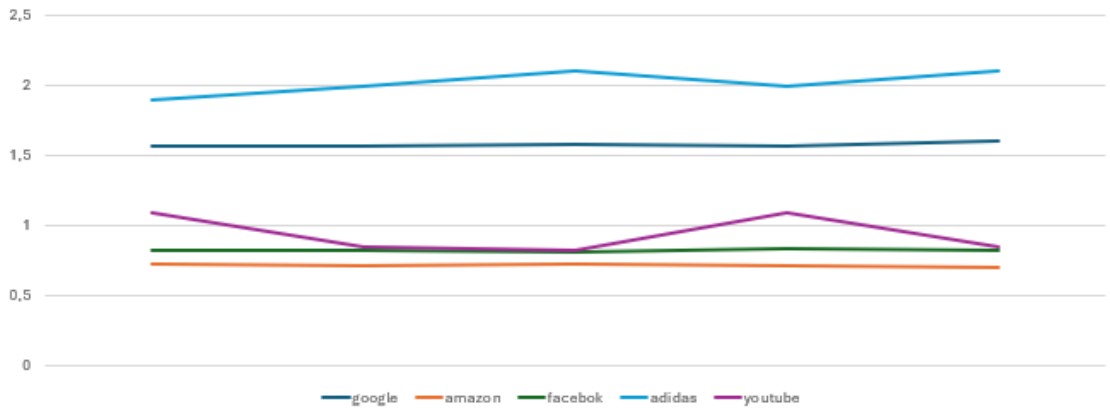


Ilustración 28 - Visualizacion en Kibana III

Capítulo 6. ANÁLISIS DE RESULTADOS

En el siguiente capitulo se realizará un análisis de resultados con la que se concluirán las 3 fases del proyecto. El análisis consistirá en revisar el funcionamiento de una aplicación, generar una serie de logs, entender cómo funciona el sistema y por último añadir alguna funcionalidad nueva. El director del TFG me ha proporcionado una aplicación en desarrollo, que hace uso de la tecnología de Elastic Stack para el estudio y monitorización de los logs generados por la misma. Se me proporciona una carpeta con todas las clases, dependencias y configuraciones donde a través de Docker podremos levantar la aplicación.

La aplicación proporcionada va ligada a la asignatura de Estadística. El resultado final del proyecto por lo tanto consistirá en aplicar todos los conocimientos estudiados a lo largo de las fases del estudio a través de esta pequeña aplicación. Mostraré en primer lugar la interfaz de la aplicación, así como su arquitectura y funcionamiento en el flujo de datos. Se mostrarán los distintos logs generados y su visualización a través de Kibana.

6.1 INTERFAZ

Una vez construimos el proyecto a través de Docker, podremos acceder a la aplicación a través del puerto 8443. Introduciré brevemente la interfaz gráfica de la aplicación para poder entender correctamente los logs en el análisis de eventos posterior.

Accediendo a la aplicación nos encontramos con la siguiente pantalla

Ilustración 29- Interfaz principal

En ella encontramos una serie de botones donde generaremos diferentes tipos de logs, como son los de acceso en este caso. Ya estamos familiarizados con esta primera parte de la aplicación, ya que la aplicación de prueba explicada previamente recogia misma gestión de eventos.

Pulsando los botones de login o registro se nos dirige a las dos siguientes pantallas

Ilustración 28 - Sistema de login

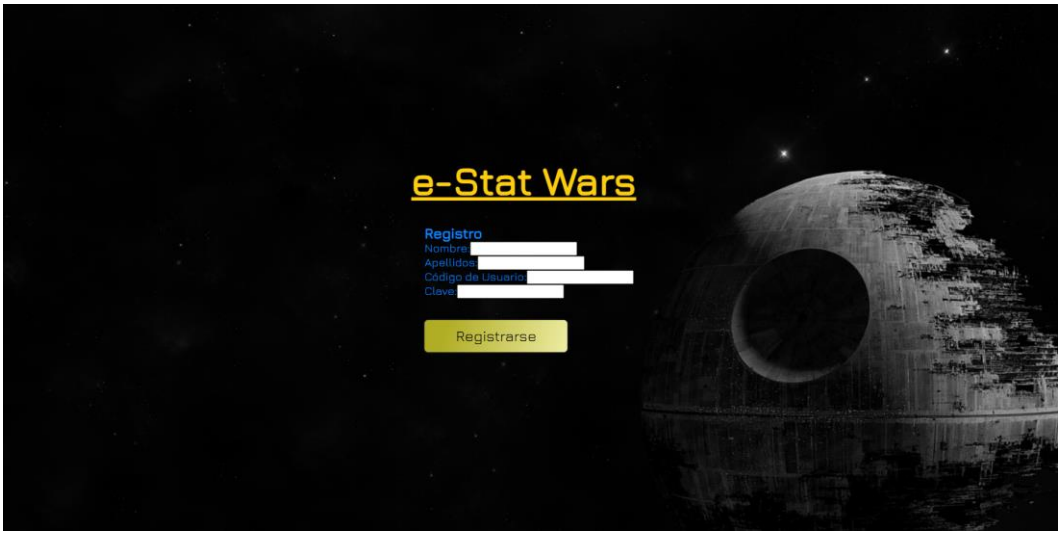


Ilustración 29 - Sistema de registro

Al igual que en la aplicación de prueba, encontramos 2 formularios a rellenar con nuestros datos. El sistema de login va ligado a una base de datos donde deberemos estar registrados para poder acceder a la aplicación.

Finalmente, Si accedemos a modo alumno encontraremos los diferentes concursos habilitados. Por otro lado si accedemos con permisos de profesor nos encontramos con la siguiente pantalla

Ilustración 30 - Dashboard de profesor

En ella encontramos multitud de funcionalidades como puede ser crear un quiz nuevo o añadir preguntas a uno ya creado.

La idea de aplicación será principalmente que el profesor de la asignatura pueda elaborar distintos concursos con multitud de preguntas, donde únicamente los alumnos registrados en la BBDD podrán acceder a dichos concursos y participar.

6.2 ESTRUCTURA DEL PROYECTO

En el siguiente apartado se analizará en profundidad la estructura interna de la aplicación, abarcando diversos aspectos técnicos y arquitectónicos que contribuyen a su funcionamiento y eficiencia. Se examinará el uso de Node.js como entorno de ejecución, destacando sus beneficios en el contexto del desarrollo de la aplicación. Se describirá la arquitectura cliente-servidor empleada, explicando cómo se distribuyen las responsabilidades entre el cliente y el servidor para optimizar el procesamiento y la gestión de datos. Se analizarán los servicios levantados en docker, evaluando cada contenedor y su función dentro del ecosistema de la aplicación. Finalmente se discutirán otros elementos relevantes que influyen en el rendimiento y la escalabilidad de la aplicación, ofreciendo una visión integral de su diseño y capacidades técnicas.

6.2.1 FUNDAMENTOS

La aplicación seguirá una estructura cliente-servidor: modelo de arquitectura de red que divide las tareas entre los proveedores de servicios (servidores) y los solicitantes de servicios (clientes). En esta arquitectura, el cliente es la interfaz del usuario final que interactúa con el servidor para solicitar servicios, mientras que el servidor es responsable de proporcionar dichos servicios. El cliente puede ser una interfaz web, una aplicación móvil o cualquier otro sistema que genere y envíe logs. El servidor recibe estos logs, los procesa y los almacena para su posterior análisis, por lo que esta estructura juega un papel fundamental para el correcto procesamiento de información.

En cuanto a entorno de ejecución se hará uso de Node.js (ODJS). Este entorno nos permitirá ejecutar JavaScript en el servidor. Node.js se utiliza comúnmente en el desarrollo de aplicaciones web en tiempo real, APIs y servicios de backend debido a su capacidad para procesar grandes volúmenes de datos rápidamente. Su amplio ecosistema de bibliotecas y módulos facilita la integración con otros componentes del Elastic Stack, como Elasticsearch y Logstash, optimizando el desarrollo y la gestión de logs. Gracias al modelo de E/S no bloqueante y basado en eventos, lo hace ideal para aplicaciones que requieren manejar múltiples conexiones concurrentes con alta eficiencia, como los sistemas de registro de logs

Para entender mejor la estructura y funcionamiento de cada servicio en el proyecto, expondré los detalles en el siguiente apartado

6.2.2 SERVICIOS

Para el correcto entendimiento de la aplicación analizaré el documento Docker-compose.yml. En el evaluaré los contenedores más relevantes y su función dentro del ecosistema de la aplicación. Ya sabemos que el documento Docker-compose contiene la configuración de servicios, redes y volúmenes para la ejecución de contenedores. Este análisis permitirá comprender cómo se orquestan y administran los servicios necesarios para el correcto funcionamiento de la aplicación en un entorno de contenedores.

6.2.2.1 Mongo DB

La aplicación utiliza MongoDB como base de datos. MongoDB es una base de datos NoSQL orientada a documentos que almacena datos en formato BSON; una excelente elección para el proyecto para almacenar todos los concursos, preguntas y usuarios debido a su capacidad para manejar grandes volúmenes de datos no estructurados o semiestructurados con alta eficiencia y flexibilidad.

```
db:
  container_name: mongodb
  image: mongo
  restart: unless-stopped
  env_file: .env
  volumes:
    - ./mongodb:/data/db
  environment:
    - MONGO_INITDB_ROOT_USERNAME=root
    - MONGO_INITDB_ROOT_PASSWORD=${db_password}
    - MONGO_INITDB_DATABASE=stat-wars
```

MongoDB permite un almacenamiento y consulta rápidos de logs, lo que facilita el análisis en tiempo real. Además, su arquitectura escalable y su soporte para índices avanzados mejoran significativamente el rendimiento y la capacidad de respuesta de la aplicación. Finalemnte, MongoDB se integra fácilmente con Node.js y otros componentes del Elastic Stack, como Elasticsearch, permitiendo una solución robusta y eficiente para la gestión y análisis de logs.

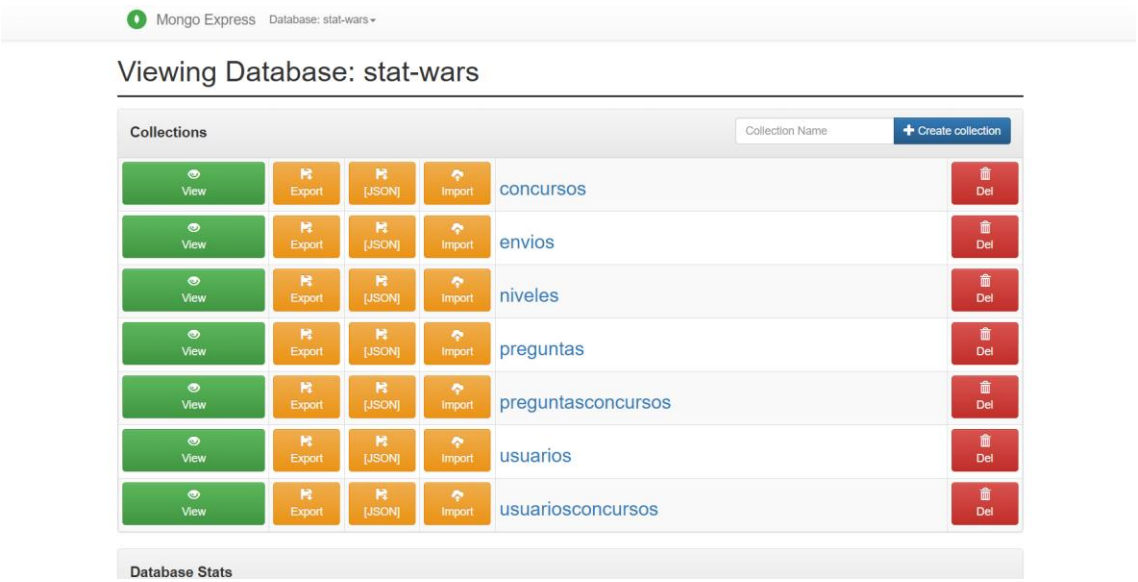


Ilustración 31 - Interfaz web de base de datos

Aquí se puede ver la interfaz de usuario web Mongo Express donde podemos visualizar todos los datos almacenados en cualquier momento, siempre y cuando tengamos permisos para ello. Es una base de datos muy potente con una interfaz muy simple y accesible.

6.2.2.2 WAF

Un Web Application Firewall (WAF) es una solución de seguridad diseñada para proteger aplicaciones web al filtrar y monitorear el tráfico HTTP entre la aplicación web y la internet. Actúa como una barrera que detiene ataques cibernéticos comunes como inyecciones SQL, cross-site scripting y otras amenazas que pueden explotar vulnerabilidades en las aplicaciones web. Utilizar un WAF durante el desarrollo de la aplicación permitirá monitorear cualquier tipo de error en el acceso al concurso, como intentos de acceso sin credenciales o incumplimientos de las normativas del concurso, además de mitigar posibles ataques.

Un WAF proporciona múltiples ventajas significativas para la seguridad de aplicaciones web: protección mejorada contra una amplia gama de ataques a nivel de aplicación, visibilidad detallada del tráfico y los ataques para una identificación rápida de incidentes, y fácil implementación para una protección inmediata sin necesidad de alterar el código de la aplicación.

Se hará uso de un WAF para el desarrollo de la aplicación con el fin de monitorizar cualquier tipo de error durante concurso. Principalmente se trata de evitar inyecciones SQL en acceso o respuesta, así como scripting entre sitios.

```
apache-waf:
  build: ./waf/
  restart: unless-stopped
  ports:
    - 8443:443
    - 8080:80
  depends_on:
    - api
  links:
    - api
  volumes:
    - ./logwaf:/var/log/apache2
```

Aquí vemos la estructura del servicio en Docker-compose. Podemos observar el puerto en el que se levanta o la carpeta donde almacenaremos todos los logs recogidos en cuanto a seguridad.

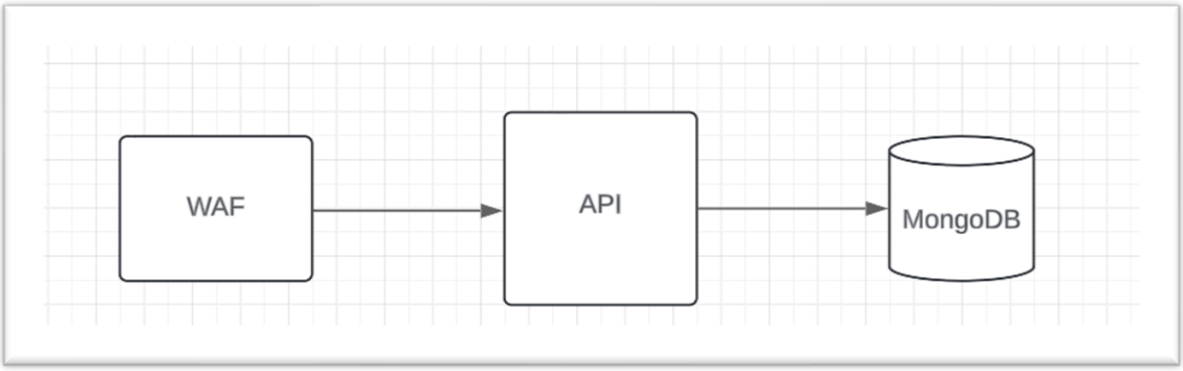


Ilustración 32 - Estructura interna de aplicación

El WAF estará directamente ligado a la API, donde esta tendrá conexión directa con la base de datos para la autenticación y autorización de la información.

6.2.2.3 Fluentd

Fluentd es un colector de datos de código abierto diseñado para unificar la capa de registro. Fue creado para facilitar la recopilación, transformación y almacenamiento de datos de registro de diversas fuentes, ideal para volúmenes de datos de gran tamaño.

En el contexto del Elastic Stack, Fluentd actúa como un intermediario que facilita la ingesta y el procesamiento de datos antes de que lleguen a Elasticsearch. Puede recolectar logs de múltiples fuentes, transformarlos según sea necesario y luego enviarlos a Logstash o directamente a Elasticsearch para su indexación y análisis. Funciona a modo de Filebeat pero ligeramente mas complejo.

```
fluentd:
  build: ./fluentd
  restart: unless-stopped
  volumes:
    - ./fluentd/conf:/fluentd/etc
  links:
    - "elasticsearch"
  ports:
    - "24224:24224"
    - "24224:24224/udp"
```

Fluentd cuenta con la capacidad de monitoreo en tiempo real, así como la transformación de datos recogidos. A través de Fluentd podremos aplicar filtros y formatear los logs con el fin de adaptarlos a los requisitos específicos de análisis.

Para el desarrollo de la aplicación se han usado las 3 diferentes formas de recopilación de logs. Filebeat y logstash ya estábamos familiarizados con sus funcionalidades y grandes capacidades, pero el uso de las 3 puede resultar muy positivo, no solo por el aprender diferentes caminos, sino por la alta eficiencia que nos otorga cada mecanismo. Mediante el uso de 3 caminos podremos distribuir la carga de trabajo en cuanto a la recopilación de logs y procesamiento de logs.

```
logstash:
  image: docker.elastic.co/logstash/logstash:8.1.2
  container_name: logstash
  volumes:
    - ./configuracionelk/logstash.conf:/usr/share/logstash/pipeline/logstash.conf
    - ./logwaf:/logs
  depends_on:
    - elasticsearch
```

```
filebeat:
  image: docker.elastic.co/beats/filebeat:8.1.2
  container_name: filebeat
  volumes:
    - ./configuracionelk/filebeat.yml:/usr/share/filebeat/filebeat.yml:ro
    - ./logwaf:/logs
```

6.2.2.4 Elasticsearch

La aplicación centraliza la recolección de estos tres tipos de logs utilizando 3 fuentes diferentes como se explicó anteriormente. Gracias a Filebeat, Logstash y Fluentd, se recogerán y procesaran todos los logs generados por el programa, y posteriormente enviados a Elasticsearch para ser almacenados.

```
elasticsearch:
  image: docker.elastic.co/elasticsearch/elasticsearch:8.1.2
  container_name: elasticsearch
  restart: unless-stopped
  environment:
    - discovery.type=single-node
    - xpack.security.enabled=false
    - "ES_JAVA_OPTS=-Xms1g -Xmx1g"
  expose:
    - "9200"
    - "9300"
  ports:
    - "9200:9200"
  volumes:
    - ./esdata:/usr/share/elasticsearch/data
```

Elasticsearch seguirá en la línea de lo que venimos estudiando. Nos proporcionara una base de datos para todos los logs, donde los ordenara según la naturaleza del evento y representara a través de la herramienta Kibana.

En la recolección de logs dispondremos de 3 maneras diferentes de procesamiento, donde cada una ira orientada a un tipo de log diferente. En la aplicación se trabajarán con logs de api, logs de apache y por ultimo logs de WAF. Brevemente comentare en que consiste cada uno y que campos abarcan:

- **Logs de API**

Los logs de API son registros detallados de todas las solicitudes y respuestas que pasan a través de una interfaz de programación de aplicaciones (API). Estos logs capturan información crítica como las direcciones IP de origen, los endpoints solicitados, los métodos HTTP utilizados (GET, POST, PUT, DELETE), los códigos de respuesta, los tiempos de respuesta y cualquier mensaje de error generado

- **Logs de Apache**

Los logs de Apache son registros generados por el servidor web Apache. Apache genera principalmente dos tipos de logs: el log de acceso y el log de errores. El log de acceso registra todas las solicitudes que recibe el servidor, incluyendo detalles como la dirección IP del cliente, la fecha y hora de la solicitud, el método HTTP utilizado, el recurso solicitado y el código de estado de la respuesta. El log de errores, por otro lado, registra cualquier error que

ocurra en el servidor, ya sea en la configuración, la ejecución de scripts o problemas de conectividad.

- **Logs de WAF**

Los logs de WAF registran el tráfico inspeccionado y las acciones tomadas por el firewall de aplicaciones web. Estos logs incluyen información sobre solicitudes bloqueadas, la naturaleza de los ataques detectados (como inyecciones SQL, XSS, etc.), direcciones IP de los atacantes y las reglas de seguridad que se activaron. Los logs de WAF son cruciales para comprender y mitigar amenazas a la seguridad de las aplicaciones web.

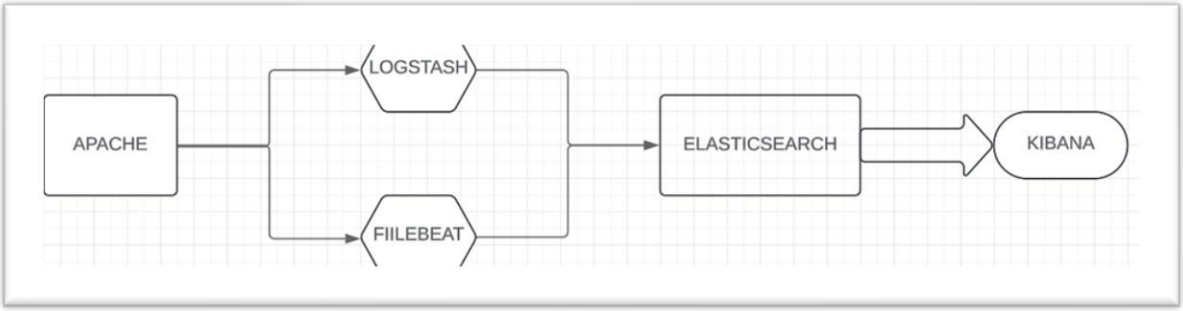


Ilustración 33 - Flujo de datos por la app

En el diagrama podemos visualizar claramente el funcionamiento del flujo de datos a través de la aplicación. Observamos como un evento desde que es recogido en la máquina, es procesado y almacenado hasta finalmente visualizarlo con Kibana.

Una vez levantada la aplicación a través de Docker, podremos empezar a monitorizar el funcionamiento de cada servicio. Estos son todos los contenedores en uso:

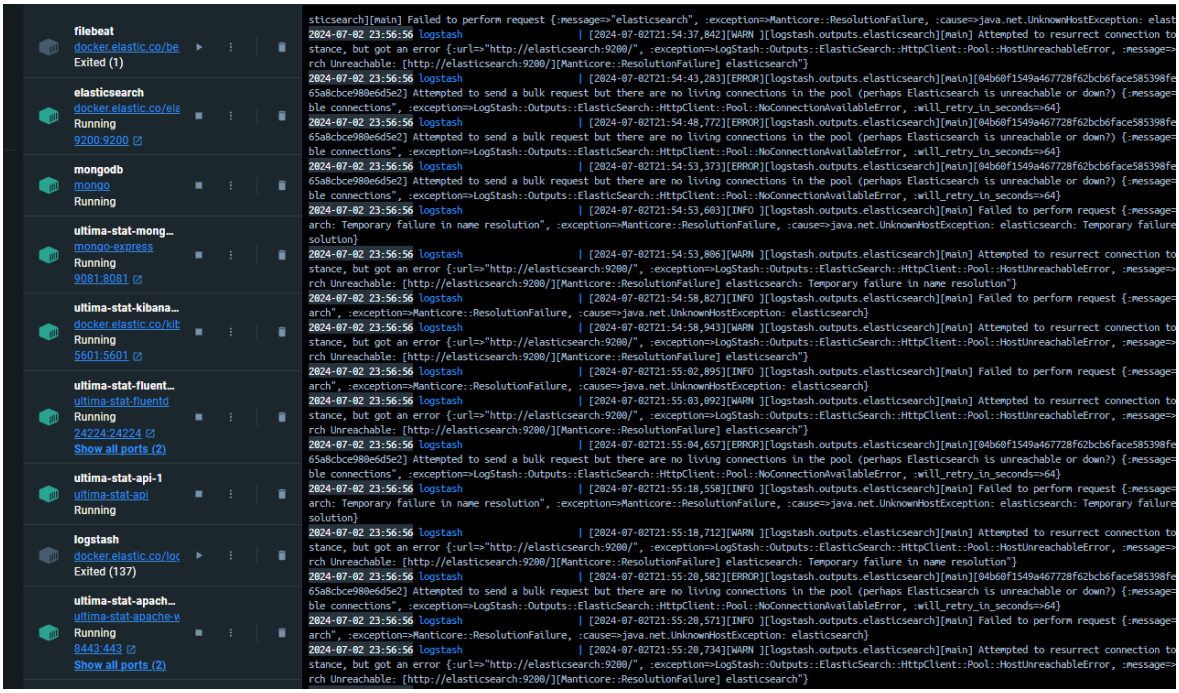


Ilustración 34 - Servicios de la aplicación en Docker

A la izquierda podemos ver el nombre de los 9 contenedores en ejecución, mientras a la derecha cada servicio va generando logs acerca del funcionamiento y estado.

6.3 ANÁLISIS DE LOGS

Finamente, se estudiarán los resultados obtenidos de la aplicación. A través de la plataforma de kibana tendremos acceso a todo el registro de logs de elasticsearch. A continuación se mostrará el recorrido por la aplicación hasta la visualización de datos.

En primer lugar, podremos personalizarnos el espacio de trabajo donde trabajaremos con los diferentes tipos de logs. de esta forma podemos repartir la carga de trabajo en la visualización de logs. En caso de trabajar con volúmenes de datos masivos, será optimo crearnos diferentes espacios de trabajos donde cada uno manejara un grupo de logs.

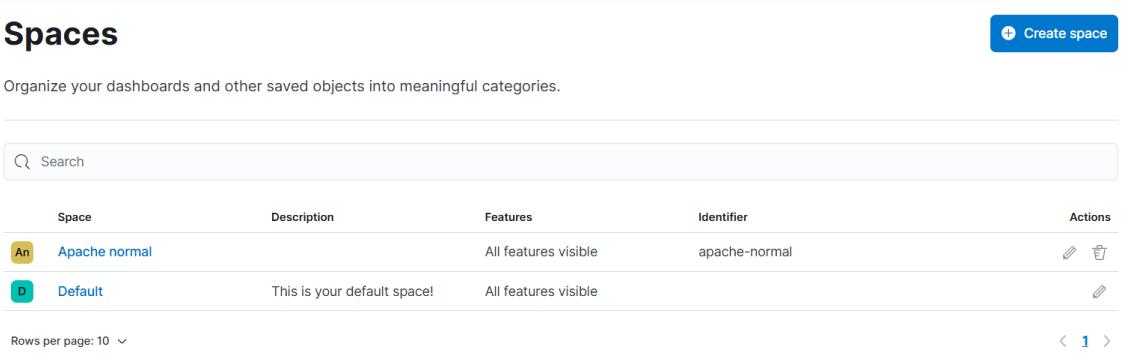


Ilustración 35 - Espacios de trabajos en Elasticsearch

En este caso vemos el "Default" que será el espacio de trabajo que nos aparecerá por defecto para empezar a trabajar, y el espacio "Apache normal" que ha sido creado con la finalidad de aislar una serie de logs de ese tipo.

El siguiente paso será dirigirnos al apartado de discover donde podremos visualizar su conjunto. Antes de ello, nos dirigiremos a los archivos de configuración del proyecto para revisar si se están registrando correctamente los logs. Archivos de texto donde se escribirán los eventos registrados, y posteriormente se leerán para el almacenaje en Elasticsearch y visualización en Kibana.

```
[Tue Jul 02 20:18:09.501441 2024] [ssl:info] [pid 10:tid 140313721587456] [client 172.18.0.1:48612] AH01998: Connection closed to child 64 with abortive shutdown
(server ICAI:443)
[Tue Jul 02 20:18:09.510221 2024] [ssl:info] [pid 9:tid 140313721587456] [client 172.18.0.1:48610] AH01998: Connection closed to child 0 with abortive shutdown
(server ICAI:443)
[Tue Jul 02 20:18:09.529180 2024] [ssl:info] [pid 10:tid 140313643038464] [client 172.18.0.1:48654] AH01998: Connection closed to child 66 with abortive shutdown
(server ICAI:443)
[Tue Jul 02 22:28:59.532417 2024] [ssl:info] [pid 9:tid 140313427441408] [client 172.18.0.1:41724] AH01964: Connection to child 3 established (server ICAI:443)
[Tue Jul 02 22:28:59.559622 2024] [ssl:info] [pid 10:tid 140313284830976] [client 172.18.0.1:41736] AH01964: Connection to child 68 established (server ICAI:443)
[Tue Jul 02 22:28:59.811653 2024] [ssl:info] [pid 10:tid 140313284830976] [client 172.18.0.1:41736] AH02008: SSL library error 1 in handshake (server ICAI:443)
[Tue Jul 02 22:28:59.816804 2024] [ssl:info] [pid 9:tid 140313427441408] [client 172.18.0.1:41724] AH02008: SSL library error 1 in handshake (server ICAI:443)
[Tue Jul 02 22:28:59.818593 2024] [ssl:info] [pid 9:tid 140313427441408] SSL Library Error: error:14094416:SSL routines:ssl3_read_bytes:sslv3 alert certificate
unknown (SSL alert number 46)
[Tue Jul 02 22:28:59.819067 2024] [ssl:info] [pid 9:tid 140313427441408] [client 172.18.0.1:41724] AH01998: Connection closed to child 3 with abortive shutdown
(server ICAI:443)
[Tue Jul 02 22:28:59.827572 2024] [ssl:info] [pid 9:tid 140313251260160] [client 172.18.0.1:41746] AH01964: Connection to child 8 established (server ICAI:443)
[Tue Jul 02 22:28:59.837004 2024] [ssl:info] [pid 10:tid 140313284830976] SSL Library Error: error:14094416:SSL routines:ssl3_read_bytes:sslv3 alert certificate
```

He elegido el archivo donde se registran los logs de tipo error en el servicio de WAF. Efectivamente se aprecia como se están registrando correctamente dichos eventos junto con el @timestamp acorde.

El siguiente paso será dirigirnos al puerto 5601 donde encontraremos la interfaz web de Elasticsearch. Navegamos al apartado de Kibana y tratamos de buscar los logs que encontrábamos en el archivo de configuración.

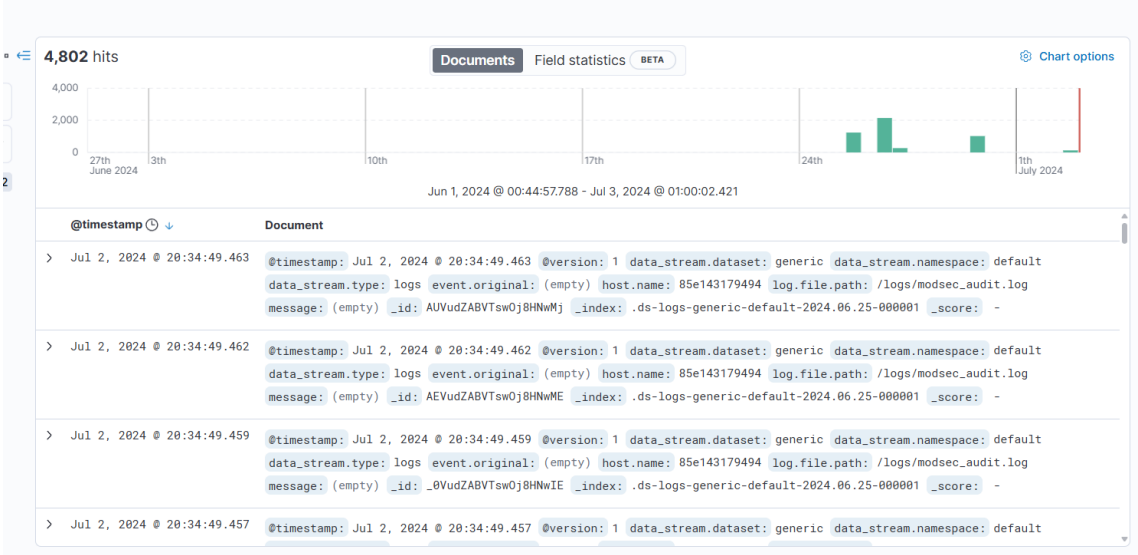


Ilustración 36 - Visualización final de logs

Si nos fijamos en la parte izquierda del marco, aparecerá la fecha de registro del log. Una vez mas vemos como coinciden con los registrados en el archivo de configuración, por lo que la aplicación funciona con éxito.

Una vez disponemos de todos los logs, podremos analizarlos con detalle fijándonos en los campos que lo componen como puede ser el path donde lo almacena, el evento o el índice donde se almacena

Kibana nos proporcionará diferentes gráficos para analizarlos. En este caso un ejemplo de grafico muy simple donde vemos como se reparten todos los logs registrados según quien los captura y como son procesados.

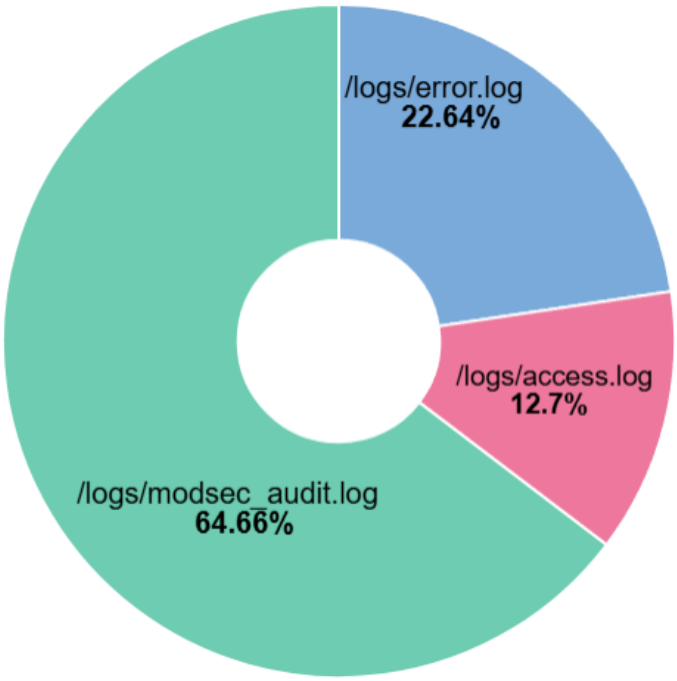


Ilustración 37 - Reparto de logs almacenados en proyecto

Capítulo 7. CONCLUSIONES Y TRABAJOS FUTUROS

El proyecto presentado como trabajo final de grado ha logrado cubrir todas las especificaciones marcadas con éxito. Ha abordado un análisis y descripción integral de la plataforma Elastic Stack, demostrando su aplicabilidad y efectividad en la gestión y análisis de registros de logs. A través de este estudio, se han puesto a prueba las diversas soluciones que nos permite el uso de esta herramienta mediante un pequeño ejemplo práctico. Finalmente, una vez asimilada la teoría y entendida la práctica, se han plasmado los conocimientos mediante el estudio y análisis de la aplicación proporcionada, logrando analizar con éxito todos los posibles logs generados por ella.

A lo largo de este epígrafe analizare y extraeré conclusiones de cada una de las fases que han compuesto todo el proyecto, así como un estudio de posibles aplicaciones o trabajos futuros.

7.1 ANÁLISIS INTEGRAL DE ELASTIC STACK

Elastic Stack, compuesto por Elasticsearch, Logstash, Kibana y Beats, se ha consolidado como una solución robusta y versátil para la gestión de grandes volúmenes de datos y el análisis en tiempo real. La capacidad de Elasticsearch para indexar y buscar rápidamente a través de grandes conjuntos de datos, Logstash con su habilidad para ingerir y transformar datos de diversas fuentes, Kibana proporcionando una interfaz de usuario para la visualización de datos, y finalmente Beats con la recolección eficiente de datos desde diversos entornos. Todo ello forma una de las herramientas más potentes del mercado actual y por muchos años en adelante.

A través de este estudio se han conocido conceptos nuevos y extrapolables a otras aplicaciones. Se ha estudiado la contenerización, tecnología que nos permite encapsular la aplicación y dependencias en un contenedor ligero. Esto que nos permitirá ejecutar de

manera consistente en cualquier entorno, así como evitar conflictos entre diferentes aplicaciones.

De igual manera, se han introducido otros métodos como es el de clusterización. Hemos visto cómo organizar las estructuras en clústeres y agrupar múltiples nodos para trabajar juntos como un sistema unificado. Tecnologías que manejan estos clústeres como es kubernetes para proporcionar orquestación automática de contenedores. Todo ello nos ha permitido la distribución de carga de trabajo, resiliencia de aplicaciones e incluso una alta escalabilidad.

Por último, hemos conocido la profundidad en la que Elastic Stack ha llegado a recalar. Empresas de alto prestigio como es Netflix, Uber o Accenture llevando a cabo importantes proyectos de la mano de ELK hacen más y más grande lo que puede denominarse como la mejor aplicación para la gestión y análisis de logs.

7.2 APLICACIÓN PRACTICA

El desarrollo de un programa de prueba para distintos registros de logs ha permitido evaluar de manera práctica el funcionamiento de Elastic Stack, así como posibles aplicaciones en distintos tipos de proyectos. Mediante un sistema de login y una serie de botones se ha demostrado cómo los componentes de Elastic Stack pueden trabajar juntos para proporcionar una solución de lo más práctica para el monitoreo y análisis de logs. Estudiando los resultados que nos proporciona Kibana mediante distintos gráficos concluimos que hay un océano de oportunidades y aplicaciones con el uso de esta plataforma.

7.3 ESTADISTICA

Gracias al estudio y entendimiento de la aplicación proporcionada por el director de proyecto, he podido ver como desde una misma base de proyecto se puede escalar e ir añadiendo nuevas y potentes funcionalidades a tu aplicación. Investigando entre las clases del proyecto he descubierto posibles mejoras a mi proyecto, así como descubrir nuevas

clases que aportan funcionalidad que desconocía como es el caso de Fluentd o el uso de ODJS. Además, he ido un paso más allá en el análisis final viendo nuevos tipos de logs o diferentes dashboards para la visualización de los mismos.

7.4 TRABAJOS FUTUROS

Elastic Stack está implementado en numerosas soluciones actuales como hemos visto anteriormente. Abarcan diversos sectores como la salud, el comercio y las telecomunicaciones, demostrando su versatilidad y eficacia en la gestión de datos. Conociendo las bases y aplicaciones puedo concluir que plataforma tiene el potencial de ir mucho más allá, cubriendo una amplia gama de necesidades en todo tipo de industrias. En un mundo donde la cantidad de información que nos rodea crece exponencialmente, una gestión eficiente de estos datos puede transformar operaciones y procesos. Elastic Stack ofrece herramientas cruciales para la toma de decisiones informada y estratégica, por lo que su potencial para innovar en áreas como la seguridad, el monitoreo de infraestructuras y el análisis predictivo, convierte a Elastic Stack en una solución integral para los desafíos del futuro.

Además de todo el potencial que contiene, otra área prometedora para el futuro de la plataforma consiste en la integración con tecnologías emergentes como son la inteligencia artificial y el aprendizaje automático. La incorporación de algoritmos de machine learning en el análisis de datos, así como la integración con tecnologías de Big Data podría ampliar aún más las numerosas capacidades que contiene Elastic Stack.

Capítulo 8. BIBLIOGRAFÍA

[1] Ayooluwa Isaiah. "Splunk vs Elastic/ELK Stack: the Key Differences to Know". Splunk vs Elastic/ELK Stack: The Key Differences to Know | Better Stack Community

[2] Guillermo Alvarado. "Introducción a Elastic Stack". Introducción a Elastic Stack | Guillermo Alvarado (galvarado.com.mx)

[3] Educative. " What Is Elastic Stack? . What Is Elastic Stack? - Elasticsearch Fundamentals: Indexing and Querying Data (educative.io)

[4] Sagar loke. "arRESTful development: How Netflix Uses Elasticsearch". arRESTful Development: How Netflix Uses Elasticsearch to Better Understand | Elastic

[5] "Splunk vs ELK Stack: the key differences to know". Splunk vs Elastic/ELK Stack: The Key Differences to Know | Better Stack Community

[6] "Engineering Uber Predictions in Real Time with ELK" Engineering Uber Predictions in Real Time with ELK | Uber Blog

[7] "Clustering Cuál es su uso en big data." Clustering: qué es y cuál es su aplicación en Big Data | UNIR´

[8] "Ventajas y desventajas de Python" Ventajas y desventajas de Python | KeepCoding Bootcamps

[9] "Historias de éxito de ELK" Casos de uso · Historias de éxito del Elastic Stack | Elastic Customers

[10] Todoservidores. "¿Qué es Elastic Stack y cómo funciona?." ¿Qué Es Elastic Stack Y Cómo Funciona? – Todo Servidores (todo-servidores.com)

[11] "Objetivos de Desarrollo Sostenible". Objetivos de Desarrollo Sostenible | Programa De Las Naciones Unidas Para El Desarrollo (undp.org)

[12] How to Collect, Process, and Ship Log Data with Fluentd: How to Collect, Process, and Ship Log Data with Fluentd | Better Stack Community

[13] Introducción a Express / Node: Introducción a Express/Node - Aprende desarrollo web | MDN (mozilla.org)

[14] "¿Qué es WAF firewall?"; ¿Qué es WAF firewall? | Cortafuegos de aplicaciones web | Cloudflare

[15] Arsys, "Arquitectura Cliente-Servidor": Arquitectura cliente servidor: qué es, tipos y ejemplos | Blog de Arsys

[16] "WAF vs Firewall": WAF vs. Firewall: Firewall de aplicaciones web y de red | Fortinet

[17] Creación de diagramas UML: Software para diagramas UML | Lucidchart

[18] "21 Mejores Herramientas en Gestion de Logs": Las 21 mejores herramientas de software de gestión de logs | Apium Academy

[19] "¿Qué es apache y como funciona?": Qué es Apache y cómo funciona - Webempresa

[20] "API: qué es y para qué sirve": API: qué es y para qué sirve (xataka.com)

ANEXO I: ALINEACIÓN DEL PROYECTO CON LOS ODS

Los Objetivos de desarrollo sostenible, comúnmente referidos como ODS, son un conjunto de 17 objetivos globales adoptados por todos los Estados Miembros de la ONU en 2015 como parte de la Agenda 2030 para el Desarrollo Sostenible. Estos objetivos abarcan diversas áreas de importancia crucial para la humanidad y el planeta, tales como la erradicación de la pobreza, la protección del planeta y la garantía de que todas las personas gocen de paz y prosperidad.

La alineación de tu proyecto con los ODS demuestra el impacto positivo que la tecnología puede tener en la sociedad y el medio ambiente. En este anexo, detallare como la plataforma Elastic Stack no solo mejora el análisis de logs y datos, sino que también contribuyen a los objetivos globales de desarrollo sostenible.

Se explorarán específicamente los ODS relacionados con la innovación tecnológica, la sostenibilidad urbana, la producción y el consumo responsables, la acción climática y la mejora de las instituciones públicas. A lo largo del capítulo, se describirá de manera general la relevancia de Elastic Stack en cada uno de estos contextos y se ofrecerán ejemplos hipotéticos de aplicaciones prácticas, destacando el potencial de la tecnología para fomentar prácticas sostenibles y responsable

ODS 9: Industria, Innovación e Infraestructura

El objetivo perseguido consiste en la construcción de infraestructuras resilientes, promover la industrialización inclusiva y sostenible, y fomentar la innovación. En relación con este objetivo, el proyecto aporta dos posibles soluciones: innovación tecnológica e infraestructura digital.

En cuanto a la innovación tecnológica, Elastic Stack es una solución avanzada para la gestión de datos y logs que puede fomentar la innovación en diversas industrias, permitiendo un análisis de datos más rápido y eficiente. Por otro lado, este proyecto contribuye directamente al desarrollo de la infraestructura digital. Al utilizar Elastic Stack, las empresas pueden mejorar su infraestructura digital, asegurando un monitoreo y análisis continuo de sus sistemas, lo cual es fundamental para la resiliencia operativa. Para entenderlo mejor expondré un ejemplo especifico.

Utilizando Elastic Stack, se pueden recopilar datos en tiempo real sobre el rendimiento de las máquinas en una planta de manufactura. Esto permite predecir fallos y realizar mantenimiento preventivo, aumentando la eficiencia y reduciendo el tiempo de inactividad. Por ejemplo, si Elastic Stack detecta un aumento gradual en las vibraciones de una máquina, puede señalar un posible desgaste en una pieza, permitiendo que se realicen reparaciones antes de que ocurra una falla grave. Además, este análisis continuo ayuda a identificar patrones de uso y optimizar los procesos de producción, mejorando la eficiencia energética y reduciendo los costos operativos.

ODS 11: Ciudades y Comunidades Sostenibles

Hacer que las ciudades y los asentamientos humanos sean inclusivos, seguros, resilientes y sostenibles conforman los principales objetivos del ODS número 11. En relación con este objetivo, el proyecto aporta dos posibles soluciones: gestión inteligente de la ciudad y resiliencia.

En cuanto a la gestión inteligente de la ciudad, Elastic Stack puede ser utilizado para el monitoreo de servicios urbanos como el tráfico, la gestión de residuos y la eficiencia energética, contribuyendo a la creación de ciudades inteligentes. Por otro lado, la resiliencia se fortalece mediante la capacidad de monitorear y analizar datos en tiempo real, ayudando a las ciudades a responder rápidamente a emergencias y a gestionar los recursos de manera más efectiva.

Utilizando Elastic Stack para recopilar y analizar datos de tráfico en tiempo real, las ciudades pueden optimizar los flujos de tráfico, reducir la congestión y minimizar las emisiones de carbono. Por ejemplo, los sensores de tráfico pueden enviar datos a Elastic Stack, donde se analizan para ajustar los tiempos de los semáforos y desviar el tráfico en caso de accidentes, mejorando la fluidez y reduciendo el impacto ambiental.

ODS 13: Acción por el Clima

El Objetivo global número 13 se define como Adoptar medidas urgentes para combatir el cambio climático y sus efectos. En relación con este objetivo, el proyecto aporta dos posibles soluciones: monitorización ambiental y respuesta rápida.

En cuanto a la monitorización ambiental, Elastic Stack puede ser utilizado para recopilar y analizar datos ambientales, como la calidad del aire, niveles de CO2 y patrones climáticos, facilitando la toma de decisiones informadas para mitigar el impacto del cambio climático. Por otro lado, la respuesta rápida se mejora mediante la capacidad de analizar datos en tiempo real, permitiendo una respuesta rápida a eventos climáticos extremos y mejorando la resiliencia ante desastres naturales.

Implementar Elastic Stack para rastrear los niveles de contaminación en tiempo real puede ayudar a las autoridades a identificar áreas problemáticas y tomar medidas inmediatas para mejorar la calidad del aire. Por ejemplo, sensores de calidad del aire pueden enviar datos a Elastic Stack, donde se analizan para alertar a las autoridades y al público sobre niveles peligrosos de contaminación, permitiendo tomar acciones preventivas y correctivas.