Universidad Internacional de La Rioja

Escuela Superior de Ingeniería y Tecnología

Máster Universitario en Desarrollo y Operaciones (DevOps)

Sistema centralizado de monitorización y observabilidad de aplicaciones desplegadas en EKS de AWS con Elastic.

Trabajo fin de estudio presentado por:	Víctor Creus González
Tipo de trabajo:	Práctico
Director/a:	Rafael Merlo
Fecha:	20/09/2023

Resumen

Este trabajo de final de máster se centra en la observabilidad y monitorización de aplicaciones desplegadas en Amazon Web Services (AWS). Específicamente, la aplicación se desplegará en Amazon Elastic Kubernetes Service (EKS) y se recopilará información de diferentes fuentes, como la infraestructura, el clúster de Kubernetes (K8s), el APM (Application Performance Monitoring) y los servicios de AWS. Para llevar a cabo esta tarea, se utilizará la herramienta Elastic Observability.

El objetivo principal de este trabajo es proporcionar una comprensión completa de las prácticas de observabilidad y monitorización en el entorno de AWS. Se exploran los beneficios y desafíos de migrar aplicaciones a la nube y se enfatiza la importancia de una observabilidad y monitorizaciones efectivas en la computación en la nube moderna.

La investigación profundiza en las capacidades de Elastic Observability y su integración con los servicios de AWS, como AWS CloudWatch. Se discuten las ventajas de utilizar Elastic Observability para recopilar, analizar y visualizar datos de diferentes fuentes, incluyendo métricas de infraestructura, métricas del clúster de Kubernetes, datos de APM y registros de servicios de AWS.

Se describe la implementación de la solución propuesta, destacando la arquitectura y consideraciones de diseño para lograr una observabilidad y monitorizaciones efectivas en un entorno de AWS EKS. El trabajo también aborda los desafíos encontrados durante el proceso de implementación y proporciona información sobre las mejores prácticas para optimizar el rendimiento y la confiabilidad de las aplicaciones desplegadas en AWS.

La evaluación de la solución incluye pruebas de rendimiento, análisis de los datos recopilados y comparación con métricas y umbrales predefinidos. Los resultados demuestran la efectividad de Elastic Observability al proporcionar información en tiempo real sobre el rendimiento de la aplicación, identificar cuellos de botella y facilitar la resolución eficiente de problemas.

En resumen, este trabajo de final de máster contribuye al campo de la observabilidad y monitorización en entornos nativos de la nube, centrándose específicamente en aplicaciones desplegadas en AWS EKS. El uso de Elastic Observability como herramienta para recopilar y

analizar datos de diferentes fuentes permite a las organizaciones obtener información valiosa sobre el rendimiento de sus aplicaciones y optimizar su infraestructura en AWS.

Palabras clave: Observabilidad, Kubernetes, AWS, Elastic.

Abstract

This Master's thesis focuses on the observability and monitoring of applications deployed on Amazon Web Services (AWS). Specifically, the application will be deployed on Amazon Elastic Kubernetes Service (EKS), and information will be collected from various sources, including infrastructure, the Kubernetes cluster, Application Performance Monitoring (APM), and AWS services. The chosen tool for this purpose is Elastic Observability.

The thesis aims to provide a comprehensive understanding of observability and monitoring practices in the AWS environment. It explores the benefits and challenges of migrating applications to the cloud and emphasizes the importance of effective observability and monitoring in modern cloud computing.

The research delves into the capabilities of Elastic Observability and its integration with AWS services, such as AWS CloudWatch. It discusses the advantages of using Elastic Observability for collecting, analyzing, and visualizing data from different sources, including infrastructure metrics, Kubernetes cluster metrics, APM data, and AWS service logs.

The implementation of the proposed solution is described, highlighting the architecture and design considerations for achieving effective observability and monitoring in an AWS EKS environment. The thesis also addresses the challenges encountered during the implementation process and provides insights into best practices for optimizing the performance and reliability of applications deployed on AWS.

The evaluation of the solution includes performance testing, analysis of collected data, and comparison with predefined metrics and thresholds. The results demonstrate the effectiveness of Elastic Observability in providing real-time insights into application performance, identifying bottlenecks, and facilitating efficient troubleshooting.

Overall, this Master's thesis contributes to the field of observability and monitoring in cloud-native environments, specifically focusing on applications deployed on AWS EKS. The use of Elastic Observability as a tool for collecting and analyzing data from various sources enables organizations to gain valuable insights into their application's performance and optimize their infrastructure on AWS.

Keywords: Observability, Kubernetes, AWS, Elastic.

Índice de contenidos

1. Introducción 8
1.1. Justificación del trabajo 9
1.2. Planteamiento del problema 10
1.3. Estructura de la memoria 11
2. Contexto y estado del arte 12
2.1. Contextualización y antecedentes 13
2.2. Trabajos relacionados 18
2.3. Conclusiones del estado del arte 23
3. Objetivos y metodología de trabajo 26
3.1. Objetivo general 26
3.2. Objetivos específicos 27
3.3. Metodología del trabajo 29
4. Desarrollo específico de la contribución 31
4.1. Planificación / Análisis / Requisitos 31
4.2. Descripción del sistema desarrollado / Implementación 35
4.3. Evaluación 41
5. Conclusiones y trabajo futuro 45
5.1. Conclusiones 45
5.1.1. Conclusiones técnicas y de solución 45
5.1.2. Conclusiones viabilidad económica 47
5.2. Líneas de trabajo futuro 48
5.2.1. Mejora de la observabilidad en EKS de AWS mediante las funcionalidades de machine learning de Elastic 48

5.2.2. Mejora de la seguridad en entornos DevSecOps en AWS mediante Elastic Security 50

Referencias bibliográficas 51

Anexo A. [APM] Overview 53

Anexo B. [APM] Transactions 55

Anexo C. [APM] Dependencies 56

Anexo D. [APM] Errors 56

Anexo E. [APM] Service Map 57

Anexo F. Monitors 57

Anexo G. [Kubernetes] Overview 58

Anexo H. [Kubernetes] Nodes 58

Anexo I. [Kubernetes] Pods 59

Anexo J. [Kubernetes] Deployments 60

Anexo K. [Kubernetes] Services 61

Anexo L. [Metrics AWS] Usage Overview 61

Anexo M. [Metrics AWS] ELB Overview 62

Anexo N. [Metrics AWS] EC2 Overview 62

Anexo O. Ejemplos alertas 63

Índice de figuras

Figura 1. Arquitectura del sistema completo.....40

Figura 2. Ejemplo de mensaje de alerta en Microsoft Teams.....48

Índice de tablas

Tabla 1. Comparativa herramientas observabilidad.....18

Tabla 2. Requisitos según prioridad.....37

Tabla 3. Claves de éxito (KPIs).....42

Tabla 4. Paneles de control.....43

Tabla 5. Alertas y condiciones.....46

1. Introducción

1.1. Justificación del trabajo

En este trabajo de fin de máster, se busca implementar un sistema de supervisión y visibilidad de aplicaciones e infraestructura desplegadas en AWS mediante el SaaS de Elastic Stack (Elastic Cloud). La monitorización y observabilidad de aplicaciones y su infraestructura se han vuelto cada vez más críticas en la industria debido a que permiten detectar y solucionar problemas de manera proactiva, mejorar la experiencia del usuario y reducir el tiempo de inactividad.

En la actualidad, las aplicaciones son un componente clave de muchas empresas y organizaciones, lo que las hace críticas para la continuidad del negocio. Las aplicaciones pueden verse afectadas por problemas como tiempos de respuesta lentos, errores de software, problemas de red o fallas de hardware. La detección y solución de estos problemas de manera proactiva es esencial para garantizar la continuidad del negocio y brindar una buena experiencia al usuario final.

El control y monitorización de aplicaciones son técnicas que permiten recopilar datos en tiempo real de una aplicación y su infraestructura, lo que permite detectar problemas de manera temprana. La monitorización se enfoca en medir y supervisar la disponibilidad, el rendimiento y la utilización de los recursos, mientras que la observabilidad se enfoca en recopilar y analizar datos de la aplicación para comprender su comportamiento interno.

A pesar de la importancia del seguimiento y registro de aplicaciones, muchas empresas enfrentan desafíos al intentar implementar estas soluciones. Uno de los principales desafíos es la complejidad de la infraestructura de la nube, que puede dificultar la recopilación y análisis de datos. Además, las soluciones actuales a menudo requieren una gran cantidad de configuración y mantenimiento, lo que puede ser costoso en términos de tiempo y recursos. Uno de los puntos más críticos es el poder centralizar toda esta información en una misma herramienta de referencia para todos los equipos.

Por estas razones, se ha decidido implementar un sistema de monitorización y observabilidad de aplicaciones e infraestructura desplegadas en AWS con el SaaS de Elastic Stack. Esta solución combina la escalabilidad y la flexibilidad de AWS con la capacidad de análisis,

visualización de datos y centralización de logs de Elastic Stack, lo que la hace una opción ideal para empresas de cualquier tamaño.

La implementación de un sistema de monitorización y observabilidad de aplicaciones desplegadas en AWS con el SaaS de Elastic Stack permitirá a las empresas monitorear y analizar su infraestructura de aplicaciones de manera eficiente, lo que les permitirá detectar y solucionar problemas de manera proactiva. Además, la solución es escalable y fácil de implementar, lo que reducirá el tiempo y los recursos necesarios para implementar y mantener una solución de rastreo y observaciónde aplicaciones. En resumen, este proyecto busca brindar una solución efectiva, escalable y fácil de implementar para la inspección y seguimiento de aplicaciones en AWS.

1.2. Planteamiento del problema

La implementación de un sistema de monitorización y observabilidad de aplicaciones en AWS con Elastic Stack es una solución efectiva para detectar y solucionar problemas de manera proactiva. Sin embargo, antes de poder implementar la solución, es necesario abordar varios desafíos que pueden surgir durante el proceso. A continuación, se presentan algunos de los principales desafíos que se han identificado:

- **Complejidad de la infraestructura de AWS**: AWS es una plataforma de nube compleja que puede incluir múltiples servicios, instancias y regiones. La complejidad de la infraestructura de AWS puede dificultar la recopilación y análisis de datos.
- **Configuración de Elastic Stack**: Elastic Stack es una plataforma de análisis y visualización de datos que consiste en varios componentes, incluyendo Elasticsearch, Kibana y Logstash. La configuración de Elastic Stack puede ser compleja y requiere conocimientos técnicos. En este caso, al utilizar SaaS, la configuración es relativamente sencilla, pero hay que tener conocimientos de la herramienta para desplegarlo y controlar su rendimiento a la par que se regula su coste.
- **Implementación de políticas de seguridad**: Es fundamental implementar políticas de seguridad sólidas para proteger los datos de la empresa. La implementación de políticas de seguridad adecuadas puede ser un desafío en un entorno de nube,

especialmente en una plataforma como AWS, que tiene múltiples servicios y opciones de configuración.

- **Identificación de métricas y alertas**: Para que el control y el seguimiento sean efectivos, es necesario identificar las métricas adecuadas y configurar alertas para los problemas críticos. La identificación de métricas adecuadas y la configuración de alertas puede ser un desafío si no se tienen los conocimientos técnicos necesarios. Además de conseguir un circuito de alertas común para los equipos y que a la vez sea específico para que la información le llegue al conjunto de personas correctas.
- **Integración con herramientas existentes**: Las empresas pueden tener herramientas de rastreo y seguimientos existentes que deben integrarse con la solución de Elastic Stack. La integración con herramientas existentes puede requerir conocimientos técnicos adicionales y puede ser un desafío.

En resumen, la implementación de un sistema de monitorización y observabilidad de aplicaciones en AWS con Elastic Stack implica varios desafíos técnicos y de gestión. Para ello es necesario abordar estos desafíos para garantizar que la solución sea efectiva, escalable y fácil de implementar.

1.3. Estructura de la memoria

Tras la utilización de la plantilla proporcionada y basándose en las guidelines, la memoria y su contenido se estructurará de la siguiente forma:

- **Introducción**: En este capítulo, se presentará el contexto y la motivación del trabajo. Se explicará por qué es importante implementar un sistema de monitorización y observabilidad de aplicaciones en AWS con Elastic Stack, así como los desafíos que pueden surgir durante el proceso de implementación.
- **Estado del arte y contexto**: En este capítulo, se realizará una revisión de la literatura existente sobre los sistemas de inspección y vigilancia de aplicaciones en la nube. Se explorarán los diferentes enfoques y herramientas utilizadas para la monitorización y registro en la nube, así como las mejores prácticas para implementar estos sistemas de manera efectiva.

- **Objetivos y metodología de trabajo**: En este capítulo, se describirá la metodología empleada y los objetivos que se pretenden abordar con la implementación de la solución de observabilidad.
- **Desarrollo especifico de la solución**: Que constará de tres partes:
    - Análisis y requisitos donde, se describirá la solución diseñada para implementar el sistema de monitorización y observabilidad de aplicaciones en AWS con Elastic Stack. Se explicará la arquitectura de la solución, los componentes utilizados y las decisiones de diseño clave.
    - Implementación de la solución que describirá el proceso de implementación de la solución. Se explicará cómo se configuraron los diferentes componentes de Elastic Stack, cómo se integraron con la infraestructura de AWS y cómo se implementaron las políticas de seguridad necesarias.
    - Evaluación de la solución que evaluará la efectividad de la solución implementada. Se analizarán los resultados de la monitorización y observabilidad y se compararán con los objetivos de la solución. Además, se discutirán las lecciones aprendidas y las mejoras potenciales para futuras implementaciones.
- **Conclusiones y trabajos futuros**: En este capítulo, se presentarán las conclusiones del trabajo y se discutirán los trabajos futuros. Se resumirán los resultados obtenidos y se explicarán las contribuciones del trabajo. Además, se identificarán las oportunidades de investigación futura y las mejoras potenciales para la solución implementada.

En resumen, la memoria estará estructurada en seis capítulos que abordarán desde la introducción y justificación del trabajo, pasando por la revisión del estado del arte y contexto y el diseño e implementación de la solución, hasta la evaluación de la solución y las conclusiones y trabajos futuros. Se espera que la memoria sea una guía completa para aquellos que deseen implementar un sistema de monitorización y observabilidad de aplicaciones en AWS con Elastic Stack, abordando los desafíos técnicos y de gestión identificados y logrando una implementación exitosa.

2. Contexto y estado del arte

En esta sección se presentarán tres apartados clave para comprender el estado del arte basándonos en una contextualización y antecedentes, trabajos relacionados y conclusiones y con una conclusión global obteniendo una visión completa y actualizada del estado del arte en el campo.

2.1. Contextualización y antecedentes

La monitorización y la observabilidad son elementos básicos en los equipos de alto rendimiento, ya que permiten observar y comprender el estado y el comportamiento de los sistemas, desde las aplicaciones hasta la infraestructura (Martínez, 2023). La monitorización se refiere a la supervisión y recopilación de datos de un conjunto predefinido de métricas y registros, mientras que la observabilidad se refiere a la capacidad de inferir el estado interno de un sistema complejo a partir de sus salidas externas (Atlassian, n.d.). La observabilidad es una evolución de las prácticas de monitorización que se desarrollaron para ayudar a los equipos a innovar más rápido, satisfacer las demandas del mercado y respaldar mejor los objetivos comerciales (Atlassian, n.d.).

Relacionado con este aspecto, en DevOps implican supervisar todo el proceso de desarrollo, desde la planificación, el desarrollo, la integración y las pruebas hasta la implementación y las operaciones. Implican una vista completa y en tiempo real del estado de las aplicaciones, los servicios y la infraestructura en el entorno de producción (Martínez, 2023). Algunos de los beneficios que ofrecen son: mejorar la calidad del software, reducir el tiempo de resolución de problemas, aumentar la satisfacción del cliente, optimizar el rendimiento del sistema y facilitar la colaboración entre los equipos.

Existen diversas herramientas de observabilidad y monitorización que pueden ayudar a los equipos de DevOps a lograr estos beneficios. Algunas de las más populares son: Elastic Stack, Datadog, Dynatrace, Sentry, Jira, Opsgenie, Snyk, LaunchDarkly, Split, etc. Estas herramientas permiten recopilar, analizar y visualizar datos de diferentes fuentes, como logs, trazas, métricas, eventos, etc. También permiten integrarse con otras herramientas de DevOps para automatizar procesos, alertar incidentes, implementar soluciones y generar informes (Martínez, 2023).

Desde un punto de vista cloud, los equipos de DevOps que trasladan sus aplicaciones a la nube o que operan en entornos híbridos o multicloud encuentran esenciales las herramientas de monitorización y observabilidad. El cloud ofrece flexibilidad, escalabilidad y agilidad, pero también implica una mayor complejidad y dinamismo en los sistemas. Por lo tanto, los equipos de DevOps necesitan recopilar, correlacionar y analizar datos de diferentes fuentes y servicios cloud, como AWS, Azure, Google Cloud, etc. Esto les permite tener una visión completa y en tiempo real del estado y el comportamiento de las aplicaciones e infraestructuras cloud, así como de las interacciones entre ellas. Con esta observabilidad en el cloud, los equipos de DevOps pueden detectar e investigar problemas rápidamente, optimizar el rendimiento y la eficiencia de los recursos cloud, mejorar la seguridad y la resiliencia de los sistemas, y ofrecer una mejor experiencia al usuario final [Dynatrace, 2022][Monitoreo Y Observabilidad En AWS - Operaciones De La Nube De AWS - Amazon Web Services, n.d.].

Como punto de partida las herramientas de monitorización y observabilidad que pueden ayudar a los equipos de DevOps a lograr una mayor observabilidad en el cloud son: AWS CloudWatch, AWS X-Ray, Azure Monitor, Google Cloud Operations Suite, Elastic Observability, Datadog Cloud Monitoring, Dynatrace Cloud Application Monitoring, Sentry Cloud Error Monitoring, etc. Estas herramientas ofrecen diferentes funcionalidades y capacidades para recopilar, visualizar y analizar datos de telemetría como logs, trazas, métricas y eventos desde diferentes servicios y plataformas cloud *(Elastic, n.d*.).

Para lograr una observabilidad efectiva, los equipos de DevOps necesitan aplicaciones o softwares que les permitan recopilar, almacenar, analizar y visualizar datos de telemetría desde diferentes fuentes y sistemas. Estos datos pueden incluir logs, métricas, trazas, eventos, alertas, etc. Como punto de partida de aplicaciones o softwares que pueden ayudar a alcanzar los objetivos de la observabilidad son: Datadog, Dynatrace, Sentry, Jira, Opsgenie, Snyk, LaunchDarkly, Split, etc. Estas aplicaciones o softwares ofrecen diferentes funcionalidades y capacidades para monitorear y observar el estado y el comportamiento de las aplicaciones e infraestructuras en el cloud o en entornos híbridos o multicloud.

Elastic stack es una muy buena opción para la observabilidad debido a que es una solución abierta, extensible y de la pila completa desarrollada a partir de la búsqueda (*Introducción Al Logging Y La Observabilidad Con El ELK Stack*, 2021). Elastic Stack está compuesto por cuatro componentes principales: Elasticsearch, Logstash, Kibana y Beats. Elasticsearch es un motor

de búsqueda y análisis distribuido que permite almacenar y consultar grandes volúmenes de datos estructurados y no estructurados. Logstash es una herramienta de procesamiento de datos que permite recopilar, transformar y enviar datos a Elasticsearch. Kibana es una interfaz de usuario que permite visualizar y explorar los datos en Elasticsearch. Beats son agentes ligeros que se instalan en los servidores o dispositivos para enviar datos a Logstash o Elasticsearch.

Elastic Stack ofrece varias ventajas para la observabilidad, como:

- Permite recopilar y analizar datos de telemetría desde diferentes fuentes y servicios cloud, como AWS, Azure, Google Cloud, Kubernetes, etc (Bragin & Bragin, 2020).
- Permite correlacionar y mapear dependencias entre la aplicación e infraestructura, e investigar los problemas con un solo clic (Bragin & Bragin, 2020(.
- Permite integrarse con otras herramientas de DevOps para automatizar procesos, alertar incidentes, implementar soluciones y generar informes (Bragin & Bragin, 2020).
- Permite escalar fácilmente para manejar grandes volúmenes de datos con alta velocidad y flexibilidad (Bragin & Bragin, 2020).
- Permite aprovechar el poder de la búsqueda para obtener información procesable y personalizada sobre el sistema.

Aquí tienes una tabla que resume algunas de las características y diferencias entre Elastic Observability, Splunk, Dynatrace y Datadog:

Tabla 1. *Comparativa de herramientas de Obervabilidad.*

Característica	Elastic Observability	Splunk	Dynatrace	Datadog
Fuente de datos	Abierta y extensible, basada en el ELK Stack y compatible con estándares abiertos como OpenTelemetry, Jaeger y Prometheus	Propietaria, basada en Splunk Enterprise y compatible con estándares abiertos como OpenTelemetry	Propietaria, basada en la plataforma Dynatrace OneAgent y compatible con estándares abiertos como OpenTelemetry	Propietaria, basada en la plataforma Datadog Agent y compatible con estándares abiertos como OpenTelemetry
Tipo de datos	Métricas, logs, trazas, experiencia de usuario, sintéticos y perfiles universales	Métricas, logs, trazas y experiencia de usuario	Métricas, logs, trazas, experiencia de usuario y sintéticos	Métricas, logs, trazas, experiencia de usuario y sintéticos
Entornos soportados	Híbridos y multicloud, cloud-native y microservicios	Híbridos y multicloud, cloud-native y microservicios	Híbridos y multicloud, cloud-native y microservicios	Híbridos y multicloud, cloud-native y microservicios
Ciclo de vida del software	DevOps y observabilidad CI/CD	DevOps e integración con herramientas de CI/CD	DevOps e integración con herramientas de CI/CD	DevOps e integración con herramientas de CI/CD

Insights accionables	Machine learning y automatización	Machine learning y automatización	Machine learning y automatización	Machine learning y automatización
Precio	Versión open-source. Asequible y transparente según el volumen de datos ingesta dos, el nivel de soporte y las características utilizadas. Prueba gratis de 14 días.	Alto y complejo según el volumen de datos indexados, el nivel de soporte y las características utilizadas. Prueba gratis de 15 días.	Alto y complejo según el número de hosts monitoreados, el nivel de soporte y las características utilizadas. Prueba gratis de 15 días.	Alto y complejo según el número de hosts monitoreados, el nivel de soporte y las características utilizadas. Prueba gratis de 15 días.

Fuente: Elaboración propia.

Como podemos apreciar, Elastic Observability se destaca por ser una solución abierta y extensible que se basa en el ELK Stack, la plataforma de búsqueda más utilizada del mundo. También ofrece una visibilidad completa y unificada de tus entornos híbridos y multicloud, así como de tus arquitecturas cloud-native y microservicios. Además, te ayuda a acelerar la entrega de software al proporcionarte una visibilidad total del ciclo de vida del desarrollo. Por último, te ofrece insights accionables basados en el machine learning y la automatización a un precio asequible y transparente.

Además, como se menciona en el artículo (*Monitoreo De AWS Con Elastic Observability*, n.d.). Los puntos principales que hacen de Elastic Observability una herramienta muy potente a la hora de monitorizar y observar la plataforma AWS:

- Permite el monitoreo del stack completo, desde aplicaciones hasta contenedores y sin servidor, reuniendo logs, métricas y rastreos en una plataforma de búsqueda y analíticas única.
- Ofrece información procesable, permitiendo detectar problemas antes que los usuarios, buscar en el almacén de objetos como Amazon S3 sin afectar el rendimiento, y convertir los datos en acción con dashboards, alertas y machine learning.
- Se integra con facilidad con CloudWatch y CloudTrail para obtener telemetría nativa de AWS, y con servicios populares como Amazon EC2, Amazon ECS, Amazon EKS, Amazon Lambda, Amazon Fargate, Amazon S3, Amazon RDS y Amazon DynamoDB.
- Proporciona rastreo distribuido integral para comprender las arquitecturas de microservicios complejas y mapear las dependencias de servicios de aplicaciones que abarcan varios entornos híbridos o cloud.

2.2. Trabajos relacionados

En el marco de este trabajo de fin de máster, se llevó a cabo una exhaustiva búsqueda en fuentes académicas y en la biblioteca de trabajos de la UNIR, así como en Google Scholar, con el objetivo de encontrar investigaciones y trabajos relacionados específicamente con la observabilidad y la monitorización de aplicaciones. Sin embargo, lamentablemente no se encontraron trabajos que abordaran directamente este tema en las fuentes mencionadas.

Ante esta situación, se ha decidido abordar el estudio de casos de éxito reales en los que se ha utilizado Elastic como herramienta de observabilidad y monitorización. Estos casos de éxito proporcionarán una visión práctica y aplicada de cómo se ha implementado y utilizado Elastic para lograr una observabilidad efectiva en diferentes entornos y aplicaciones.

A través de este enfoque, se espera obtener un conocimiento profundo sobre las capacidades y beneficios de Elastic como herramienta de observabilidad y monitorización, así como comprender los desafíos y las mejores prácticas asociadas a su implementación en entornos reales. Esto permitirá obtener conclusiones relevantes y recomendaciones prácticas que puedan ser aplicadas en futuros proyectos relacionados con la observabilidad y la monitorización de aplicaciones.

Entel

Entel es una empresa de telecomunicaciones que opera en Chile y Perú, y que ofrece servicios de telefonía móvil, fija, internet y televisión a más de 20 millones de clientes. Para garantizar la calidad y la continuidad de sus servicios, Entel necesita monitorear y analizar una gran cantidad de datos generados por sus sistemas y aplicaciones, tales como logs, métricas, trazas y eventos.

Entel utilizaba Splunk como plataforma de análisis de datos, pero se enfrentaba a varios problemas. Splunk era una solución propietaria y costosa, que limitaba la cantidad de datos que Entel podía indexar y analizar por día. Además, Splunk era difícil de escalar y de administrar, lo que requería una gran inversión en infraestructura y personal. Estos problemas afectaban la visibilidad y la capacidad de respuesta de Entel ante las incidencias y las demandas de sus servicios.

En consecuencia, decidió migrar a Elastic Stack, una solución que integra búsqueda, observabilidad y seguridad en una sola plataforma. Elastic Stack es una solución abierta y flexible, que permite a Entel indexar y analizar una mayor cantidad de datos sin restricciones ni costos adicionales. Además, Elastic Stack es fácil de escalar y de administrar, lo que reduce la complejidad y el esfuerzo operativo. Estas ventajas permiten a Entel tener una visión más completa y detallada de sus servicios críticos, mejorar su disponibilidad y su rendimiento, y facilitar la colaboración entre los equipos de TI y de negocios.

Además, también se benefició de las integraciones de Elastic Stack con otras herramientas que utilizaba, como Kafka, Grafana, Prometheus y Azure. Kafka es un sistema distribuido que permite gestionar flujos de datos en tiempo real. Grafana es una herramienta de visualización que permite crear paneles interactivos. Prometheus es un sistema de monitoreo que recoge y almacenas métricas. Azure es un proveedor de servicios en la nube que ofrece recursos computacionales, de almacenamiento y de red. Estas integraciones permiten a Entel aprovechar al máximo sus datos y sus recursos, y ofrecer una mejor experiencia a sus clientes.

Aquí dejo dos reflexiones que deja Helder Branco en la presentación de este caso en un artículo junto a Elastic. (*Al Migrar Desde Splunk, Entel Permite Una Mayor Capacidad De Ingesta De Datos Y Ahora Puede Proporcionar Visibilidad Las 24 Horas Del Día, Los 7 Días De La Semana, a Los Servicios Críticos En Los Equipos De TI Y De Negocios En Chile Y Perú, Todo Impulsado Por Elastic Stack*, n.d.)

“Entel ahora tiene visibilidad en más del 95 % de todos los servicios de operaciones. Ahora solo tenemos que iniciar sesión en Kibana y podemos ver lo que sucede en cada plataforma. Podemos extraer datos de pedidos, recargas, evaluaciones comerciales, plataformas de tiendas, reclamos e incluso problemas técnicos. Integramos datos de máquinas, tiendas, bases de datos, todos los problemas de indicadores de infraestructura, y hemos creado un marco único para Perú y Chile" - Helder Branco, gerente de operaciones de TI de Entel

“Estoy agradecido por tener a Elastic porque no tengo restricciones de rendimiento, problemas de tráfico o espacio. Elastic ha sido increíblemente útil para crear paneles e informes.” - Helder Branco, gerente de operaciones de TI de Entel

Energisa

Como podemos ver en un artículo de casos de éxito de Elastic (*Energisa Monitorea Sistemas Críticos Del Negocio Con Elastic Observability*, n.d.) Energisa es la empresa privada más grande del sector eléctrico en Brasil, con más de 8 millones de clientes y una presencia en 18 estados. Su misión es generar y distribuir energía de forma sustentable y eficiente, contribuyendo al desarrollo económico y social del país. Para cumplir con esta misión, necesita monitorear el rendimiento de sus sistemas críticos para el negocio, que ofrecen información y servicios esenciales a los clientes y empleados, tales como facturación, cobranza, atención al cliente, gestión de redes y medidores inteligentes. Mantener la disponibilidad permanente de estos sistemas mediante el logging y la resolución de problemas técnicos lo más rápido posible es primordial para garantizar la calidad del servicio y la satisfacción del cliente.

Sin embargo, hasta hace poco, los analistas de Energisa no tenían la capacidad de observar con facilidad el rendimiento de estos sistemas críticos para el negocio. Los logs de aplicaciones y servicios se generaban en formato de texto, lo que dificultaba encontrarlos en un sistema descentralizado. La mera cantidad de información y el tamaño de los archivos, que se almacenaban en una base de datos relacional tradicional, complicaban el desafío. Los analistas dedicaban mucho tiempo a examinar las bases de datos o archivos de texto; con frecuencia recurrían a hojas de cálculo complejas para observar el rendimiento del sistema. Rómulo Maini Pinto, líder del equipo de ingeniería de software de Energisa, dice: “Nuestro proceso anterior hacía que fuera desafiante responder preguntas simples y complicaba la detección

de la causa raíz del problema o incluso tomar medidas preventivas" (*Energisa Monitorea Sistemas Críticos Del Negocio Con Elastic Observability*, n.d.).

Para mejorar su observabilidad, Energisa eligió Elastic Observability, una solución que permite centralizar, indexar y visualizar los datos de logs, métricas y trazas en una sola plataforma. Con Elastic Observability, los analistas de Energisa pueden recibir alertas automáticas sobre problemas genuinos, en lugar de falsas alarmas, y pueden identificar, priorizar y resolver problemas del sistema en minutos, en lugar de en horas. Además, pueden aprovechar el machine learning para detectar anomalías y tendencias en los datos, así como crear paneles personalizados para visualizar los indicadores clave de rendimiento. Con Elastic Observability, Energisa ha logrado reducir significativamente el tiempo de resolución de problemas, mejorar la disponibilidad de sus sistemas y aumentar la satisfacción y confianza de sus clientes. Según Maini: "Con Elastic Observability hemos logrado un nivel muy alto de estabilidad en nuestros sistemas críticos para el negocio. Ahora podemos anticiparnos a los problemas antes de que afecten a nuestros clientes" (*Energisa Monitorea Sistemas Críticos Del Negocio Con Elastic Observability*, n.d.). Además, Energisa ha establecido un roadmap tecnológico más estable y rentable para el monitoreo de infraestructura y mantenimiento, con planes para expandir el uso de Elastic a otros casos de uso como seguridad y búsqueda empresarial.

Auchan

En este otro artículo (*Auchan Francia: Creación De Experiencias Minoristas "Figitales" Con Tecnología De Daos De Próxima Generación*, n.d.) podemos ver como Auchan Francia, que es una empresa minorista con más de 600 ubicaciones y 6000 empleados, que busca ofrecer a sus clientes experiencias de compra excelentes tanto en las tiendas como en línea. Para ello, necesita crear experiencias minoristas "figitales", que unan lo físico y lo digital, y optimizar el acceso a la inteligencia comercial en todos sus canales favoritos. Esto implica reemplazar los sistemas heredados por servicios basados en el cloud más avanzados para la infraestructura, la analítica de datos, el desarrollo de aplicaciones y la gestión de la relación con el cliente.

Auchan Francia tiene un sistema de gestión de datos masivo que se encarga de los datos de 500 aplicaciones que se usan en todo el negocio, desde la cadena de suministro hasta las finanzas. Antes, la empresa dependía de la infraestructura en las instalaciones para gestionar

los datos, pero tenía problemas con la velocidad, el monitoreo y la escalabilidad. Por ejemplo, la empresa procesa alrededor de cinco millones de flujos de datos al día que se mueven por su base de datos y una gran cantidad de aplicaciones de inteligencia comercial (BI). Los analistas dedicaban mucho tiempo a examinar las bases de datos o archivos de texto; con frecuencia recurrían a hojas de cálculo complejas para observar el rendimiento del sistema.

Para mejorar su observabilidad y su capacidad de análisis, Auchan Francia eligió Elastic Observability y Google Cloud como sus socios tecnológicos. Con Elastic Observability, los analistas pueden centralizar, indexar y visualizar los datos de logs, métricas y trazas en una sola plataforma. Con Google Cloud, pueden aprovechar el poder de procesamiento de soluciones como BigQuery para desarrollar su nueva infraestructura de datos. Con esta combinación, Auchan Francia puede recibir alertas automáticas sobre problemas genuinos, identificar y resolver problemas del sistema con rapidez, crear paneles personalizados para visualizar los indicadores clave de rendimiento y aprovechar el machine learning para detectar anomalías y tendencias en los datos.

Con Elastic Observability y Google Cloud, Auchan Francia ha logrado mejorar la disponibilidad y el rendimiento de sus sistemas críticos para el negocio, optimizar el acceso a la inteligencia comercial en todos los canales y ofrecer a sus clientes experiencias minoristas "figitales" más rápidas y relevantes.

Zurich Insurance

Y por último en otro artículo (*Zurich Insurance Aumenta La Confianza De Los Clientes Y Respalda El Futuro Con Elastic Cloud*, n.d.), podemos ver como Zurich Insurance, que es una empresa de seguros con más de 150 años de historia, que tiene como misión proteger a sus clientes y las cosas que aman, dándoles la tranquilidad y seguridad para construir un futuro. Para ello, necesita ofrecer a sus clientes productos y servicios de seguro innovadores y adaptados a sus estilos de vida cada vez más digitales. Esto implica retirar las aplicaciones heredadas y migrar muchas cargas de trabajo a Elastic Cloud, una solución basada en el cloud que proporciona observabilidad, seguridad y búsqueda empresarial.

Zurich Insurance tiene un sistema de gestión de datos que se encarga de los datos de 500 aplicaciones que se usan en todo el negocio, desde la evaluación de riesgos hasta el reporte

de reclamos. Antes, la empresa dependía de la infraestructura en las instalaciones para gestionar los datos, pero tenía problemas con la velocidad, el monitoreo y la escalabilidad. Los analistas dedicaban mucho tiempo a examinar las bases de datos o archivos de texto; con frecuencia recurrían a hojas de cálculo complejas para observar el rendimiento del sistema.

Para mejorar su observabilidad y su capacidad de análisis, Zurich Insurance eligió Elastic Cloud en Kubernetes como su socio tecnológico. Con Elastic Cloud en Kubernetes, los analistas pueden centralizar, indexar y visualizar los datos de logs, métricas y trazas en una sola plataforma. Con Elastic Cloud en Kubernetes, Zurich Insurance puede recibir alertas automáticas sobre problemas genuinos, identificar y resolver problemas del sistema con rapidez, crear paneles personalizados para visualizar los indicadores clave de rendimiento y aprovechar el machine learning para detectar anomalías y tendencias en los datos.

Con Elastic Cloud en Kubernetes, Zurich Insurance ha logrado mejorar la disponibilidad y el rendimiento de sus sistemas críticos para el negocio, optimizar el acceso a la inteligencia comercial en todos los canales y ofrecer a sus clientes productos y servicios nuevos mejorados, como Zurich Klinc, que proporciona cobertura para dispositivos electrónicos personales. Según Sandra Hauser, jefa de transformación y tecnología de Zurich Insurance: “Con Elastic Cloud en Kubernetes hemos logrado un nivel muy alto de estabilidad en nuestros sistemas críticos para el negocio. Ahora podemos anticiparnos a los problemas antes de que afecten a nuestros clientes” (*Zurich Insurance Aumenta La Confianza De Los Clientes Y Respalda El Futuro Con Elastic Cloud*, n.d.).

2.3. Conclusiones del estado del arte

En conclusión, el estado del arte de la monitorización y la observabilidad en DevOps ha evolucionado para abordar las demandas de los entornos tecnológicos modernos. La monitorización se ha convertido en una práctica fundamental para supervisar y recopilar datos predefinidos sobre el estado y el rendimiento de los sistemas. Por otro lado, la observabilidad se ha desarrollado como una técnica más avanzada que permite inferir el estado interno de un sistema complejo a partir de sus salidas externas.

La implementación de prácticas de monitorización y observabilidad en DevOps ofrece beneficios significativos, como mejorar la calidad del software, reducir el tiempo de resolución

de problemas, aumentar la satisfacción del cliente y optimizar el rendimiento del sistema. Estas prácticas también fomentan la colaboración entre equipos al proporcionar una visión compartida y en tiempo real del estado y el comportamiento de las aplicaciones e infraestructuras.

Existen numerosas herramientas de monitorización y observabilidad disponibles en el mercado, que permiten recopilar, analizar y visualizar datos de diferentes fuentes. Estas herramientas se integran con otras herramientas de DevOps para automatizar procesos, alertar incidentes, implementar soluciones y generar informes. Algunas de las herramientas destacadas incluyen Elastic Stack, Datadog, Dynatrace, Splunk, entre otras.

En el contexto del cloud, la monitorización y la observabilidad desempeñan un papel crucial debido a la flexibilidad y escalabilidad que ofrece esta tecnología. Las herramientas de monitorización y observabilidad en el cloud permiten recopilar, correlacionar y analizar datos de diferentes servicios y plataformas cloud, como AWS, Azure y Google Cloud. Esto facilita a los equipos de DevOps obtener una visión completa y en tiempo real del estado y el comportamiento de las aplicaciones e infraestructuras cloud, mejorando la detección y resolución de problemas, la optimización del rendimiento y la experiencia del usuario final.

Algunas de las herramientas destacadas de monitorización y observabilidad en el cloud son AWS CloudWatch, AWS X-Ray, Azure Monitor, Google Cloud Operations Suite, Elastic Observability, Datadog Cloud Monitoring y Dynatrace Cloud Application Monitoring. Estas herramientas ofrecen diversas capacidades para recopilar, visualizar y analizar datos de telemetría, como logs, métricas, trazas y eventos, desde diferentes servicios y plataformas cloud.

En resumen, la monitorización y la observabilidad son prácticas esenciales en el contexto de DevOps, y su adopción ha evolucionado para abordar las demandas actuales. Las herramientas disponibles en el mercado ofrecen soluciones potentes y flexibles para recopilar y analizar datos de telemetría, brindando una visión completa y en tiempo real del estado y el comportamiento de los sistemas. Esto ayuda a los equipos de DevOps a optimizar el rendimiento, mejorar la calidad del software y brindar una experiencia excepcional a los usuarios finales.

Por otra parte hemos visto que Elastic Observability se destaca como una de las mejores opciones en el ámbito de la monitorización y observabilidad debido a su amplia gama de capacidades y su enfoque integral. Elastic, la compañía detrás de Elastic Observability, es reconocida por su popularidad y éxito en el campo del motor de búsqueda y análisis de datos, lo que le ha permitido aplicar su experiencia en el dominio de la inspección y seguimiento.

Elastic Observability ofrece una plataforma unificada que combina tres pilares fundamentales: Elastic Logs, Elastic Metrics y Elastic APM (Application Performance Monitoring). Esta combinación permite a los equipos de DevOps obtener una visión completa y detallada de los sistemas, desde registros de aplicaciones y métricas de infraestructura hasta la identificación de cuellos de botella y el seguimiento del rendimiento de las aplicaciones.

Una de las ventajas clave de Elastic Observability es su escalabilidad y flexibilidad. Puede manejar grandes volúmenes de datos de telemetría en tiempo real y almacenarlos de forma eficiente en Elasticsearch, su motor de búsqueda y análisis distribuido. Esto permite a las organizaciones recopilar y analizar datos de telemetría de diversas fuentes, como logs, métricas, trazas y eventos, a una escala masiva.

Además, Elastic Observability ofrece una interfaz intuitiva y fácil de usar, que permite a los usuarios visualizar y explorar los datos de telemetría de manera interactiva. Las poderosas capacidades de búsqueda y filtrado facilitan la identificación y solución de problemas, mientras que los tableros personalizables y los gráficos en tiempo real brindan una visión instantánea del estado y el rendimiento de los sistemas.

Otra ventaja significativa de Elastic Observability es su capacidad de integración con otras herramientas y sistemas. Ofrece conectividad con una amplia gama de fuentes de datos, como servicios cloud, sistemas de registro y métricas, y frameworks de desarrollo populares. Esto permite a los equipos de DevOps aprovechar su infraestructura existente y obtener una visión completa de su entorno tecnológico.

En resumen, Elastic Observability destaca como una de las mejores opciones en el campo de la monitorización y observabilidad debido a su enfoque integral, su escalabilidad y flexibilidad, su interfaz intuitiva y sus capacidades de integración. Proporciona a los equipos de DevOps las herramientas necesarias para supervisar, analizar y optimizar el rendimiento de sus sistemas, lo que les permite ofrecer aplicaciones y servicios de alta calidad a sus usuarios finales.

3. Objetivos y metodología de trabajo

En este apartado se describirán los objetivos, tanto generales como específicos, del proyecto. Así como también la metodología utilizada para alcanzar dichos objetivos.

3.1. Objetivo general

El objetivo general del proyecto es implementar prácticas efectivas de observabilidad y monitorización en aplicaciones desplegadas en AWS. Esto se llevará a cabo con el fin de mejorar la calidad del software, reducir el tiempo de resolución de problemas, aumentar la satisfacción del cliente y optimizar el rendimiento del sistema.

La observabilidad y la monitorización son aspectos fundamentales en el contexto de DevOps, ya que permiten obtener una visión completa y en tiempo real del estado y el comportamiento de las aplicaciones e infraestructuras. Al implementar estas prácticas, se busca obtener los siguientes resultados:

- **Mejorar la calidad del software**: La monitorización y la observabilidad permiten identificar y solucionar problemas de manera proactiva, lo que ayuda a mejorar la calidad del software y reducir la presencia de errores o fallos en la aplicación.
- **Reducir el tiempo de resolución de problemas**: Al contar con una visión completa y detallada del estado de la aplicación, se agiliza el proceso de identificación y resolución de problemas, lo que reduce el tiempo de respuesta y minimiza el impacto en los usuarios.
- **Aumentar la satisfacción del cliente**: Al mejorar la calidad del software y reducir los tiempos de respuesta, se logra una mejor experiencia para el cliente, lo que se traduce en una mayor satisfacción y fidelidad hacia la aplicación.
- **Optimizar el rendimiento del sistema**: La monitorización y la observabilidad permiten identificar cuellos de botella, problemas de rendimiento y áreas de mejora en la infraestructura, lo que facilita la optimización y el aprovechamiento eficiente de los recursos disponibles.

En resumen, el objetivo general del proyecto es implementar prácticas de observabilidad y monitorización en aplicaciones desplegadas en AWS con el fin de mejorar la calidad del

software, reducir el tiempo de resolución de problemas, aumentar la satisfacción del cliente y optimizar el rendimiento del sistema. Esto se logrará a través de la recopilación de información de infraestructura, la monitorización del cluster de Kubernetes (EKS), la implementación de APM y la recopilación de datos de AWS, utilizando la herramienta Elastic Observability.

3.2. Objetivos específicos

Los objetivos específicos de este proyecto son los siguientes:

- Definir la infraestructura en Saas de Elastic: Configurar y establecer la infraestructura necesaria en Elastic Cloud para almacenar y analizar los datos recopilados de la monitorización y observabilidad.
- Desplegar un cluster de EKS mediante Terraform: Utilizar Terraform para automatizar el despliegue y configuración de un cluster de Amazon Elastic Kubernetes Service (EKS), que servirá como entorno de ejecución para la aplicación.
- Modificar el código fuente para recoger información de APM mediante el APM Agent de Elastic: Realizar modificaciones en el código fuente de la aplicación para integrar el Agente de Elastic APM, permitiendo la recopilación de datos de rendimiento y trazas de la aplicación.
- Construir la imagen Docker de la aplicación con los cambios para recoger información de APM: Crear una imagen Docker actualizada de la aplicación, que incluya los cambios necesarios para habilitar la recopilación de datos de APM.
- Desplegar la aplicación en EKS estableciendo réplicas y servicios necesarios: Implementar la aplicación en el cluster de EKS, configurando el número de réplicas y los servicios necesarios para garantizar su disponibilidad y escalabilidad.
- Desplegar y configurar un Elastic-Agent en el cluster de EKS como Daemonset para recoger información de K8s: Instalar y configurar un Elastic-Agent en el cluster de EKS como un Daemonset, permitiendo la recopilación de datos de monitorización y observabilidad específicos de Kubernetes.
- Desplegar y configurar una máquina EC2 para instalar el Elastic-Agent y recoger información de AWS: Configurar una instancia de Amazon EC2 y realizar la instalación y configuración del Elastic-Agent, permitiendo la recopilación de datos de seguimiento y registros específicos de los servicios de AWS.

- Identificar KPIs de la información recogida de APM: Definir los indicadores clave de rendimiento (KPIs) que se utilizarán para evaluar el rendimiento y la eficiencia de la aplicación, basados en los datos recopilados por el Agente de Elastic APM.
- Identificar KPIs de la información recogida de K8s: Establecer los indicadores clave de rendimiento (KPIs) que se utilizarán para evaluar el rendimiento y la eficiencia del cluster de EKS y las aplicaciones desplegadas en él, basados en los datos recopilados por el Elastic-Agent de K8s.
- Identificar KPIs de la información recogida de AWS: Definir los indicadores clave de rendimiento (KPIs) que se utilizarán para evaluar el rendimiento y la eficiencia de los servicios de AWS utilizados, basados en los datos recopilados por el Elastic-Agent de AWS.
- Definir alertas basadas en los KPIs de APM: Establecer reglas y umbrales para generar alertas automáticas cuando los KPIs de APM superen ciertos límites predefinidos, lo que permitirá una detección temprana de problemas y una respuesta rápida.
- Definir alertas basadas en los KPIs de K8s: Configurar reglas y umbrales para generar alertas automáticas cuando los KPIs de K8s indiquen un mal funcionamiento o una degradación del rendimiento, lo que permitirá una gestión proactiva del cluster de EKS.
- Definir alertas basadas en los KPIs de AWS: Establecer reglas y umbrales para generar alertas automáticas cuando los KPIs de AWS muestren anomalías o problemas en los servicios utilizados, lo que permitirá una respuesta rápida y eficiente.
- Establecer un canal de comunicación para las alertas de APM: Definir un sistema o plataforma de comunicación para recibir y gestionar las alertas generadas por los KPIs de APM, asegurando que sean entregadas a los responsables adecuados para su pronta atención.
- Establecer un canal de comunicación para las alertas de K8s: Configurar un sistema o plataforma de comunicación para recibir y gestionar las alertas generadas por los KPIs de K8s, garantizando que sean notificadas a los miembros del equipo de DevOps encargados de la gestión del cluster de EKS.
- Establecer un canal de comunicación para las alertas de AWS: Establecer un sistema o plataforma de comunicación para recibir y gestionar las alertas generadas por los KPIs de AWS, asegurando que sean enviadas a los responsables correspondientes para su pronta atención.

- Establecer dashboards de la información recogida de APM: Configurar paneles de control y visualización de los datos recopilados por el Agente de Elastic APM, permitiendo una visualización clara y concisa del rendimiento y la salud de la aplicación.
- Establecer dashboards de la información recogida de K8s: Configurar paneles de control y visualización de los datos recopilados por el Elastic-Agent de K8s, proporcionando una visión general del estado y el rendimiento del cluster de EKS y las aplicaciones desplegadas en él.
- Establecer dashboards de la información recogida de AWS: Configurar paneles de control y visualización de los datos recopilados por el Elastic-Agent de AWS, brindando una visión completa de la utilización y el rendimiento de los servicios de AWS utilizados.

Estos objetivos específicos guiarán el desarrollo y la implementación del proyecto, asegurando la correcta monitorización y observabilidad de las aplicaciones desplegadas en AWS, utilizando la herramienta Elastic Observability.

3.3. Metodología del trabajo

La metodología de trabajo escogida para llevar a cabo este proyecto, es la metodología ágil Kanban, que se trata de un enfoque de gestión de proyectos que se basa en la visualización y el control del flujo de trabajo. Se centra en la mejora continua, la colaboración y la entrega de valor de manera eficiente. Algunos de los puntos fuertes de la metodología Kanban son:

- **Visualización del flujo de trabajo**: Kanban utiliza tableros visuales para representar el flujo de trabajo y las tareas pendientes. Esto permite a todos los miembros del equipo tener una visión clara y compartida de las tareas en curso, las que están en espera y las que se han completado. La visualización facilita la identificación de cuellos de botella, la asignación de tareas y la toma de decisiones basadas en datos.
- **Limitación del trabajo en progreso (WIP)**: Kanban establece límites claros para la cantidad de trabajo que se puede realizar simultáneamente. Esto evita la sobrecarga del equipo y ayuda a mantener un flujo constante de trabajo. Al limitar el trabajo en progreso, se fomenta la finalización de las tareas antes de comenzar nuevas, lo que reduce la acumulación de trabajo y mejora la eficiencia.

- **Enfoque en la entrega continua**: Kanban se centra en la entrega continua de valor al cliente. En lugar de trabajar en grandes lotes de trabajo, se priorizan y se entregan pequeñas mejoras de manera constante. Esto permite obtener retroalimentación temprana y realizar ajustes rápidos, lo que resulta en una mayor satisfacción del cliente y una mayor adaptabilidad a los cambios.
- **Mejora continua**: Kanban promueve la mejora continua a través de la retroalimentación y la revisión regular del proceso. Se alienta a los equipos a identificar áreas de mejora, experimentar con cambios y medir los resultados. Esto permite optimizar el flujo de trabajo, eliminar desperdicios y maximizar la eficiencia.
- **Colaboración y autonomía del equipo**: Kanban fomenta la colaboración y la autonomía del equipo. Los miembros del equipo tienen la libertad de tomar decisiones sobre cómo organizar y ejecutar su trabajo. Esto promueve la responsabilidad individual y el compromiso con los resultados. Además, Kanban facilita la colaboración entre diferentes roles y departamentos, lo que mejora la comunicación y el trabajo en equipo.

En resumen, la metodología Kanban es una forma efectiva de gestionar proyectos al proporcionar una visualización clara del flujo de trabajo, limitar el trabajo en progreso, enfocarse en la entrega continua, fomentar la mejora continua y promover la colaboración y la autonomía del equipo. Estos puntos fuertes permiten a los equipos ser más eficientes, adaptarse rápidamente a los cambios y entregar valor de manera constante.

Los pasos necesarios para adoptar esta metodología en el proyecto son:

Creación del tablero: Iniciar creando un tablero en Trello con las siguientes columnas: "To do", "In progress" y "Done". Estas columnas representarán las diferentes etapas de trabajo.

Creación de tarjetas: Para cada objetivo específico definido, crear una tarjeta en la columna "To do". Cada tarjeta debe tener una descripción clara del objetivo específico y se puede asignar una etiqueta para identificar la categoría a la que pertenece (por ejemplo, "Infraestructura", "Despliegue", "Alertas", etc.).

Priorización de tarjetas: Priorizar las tarjetas en función de su importancia y dependencias. Esto se puede hacer moviendo las tarjetas en el tablero para reflejar su prioridad.

Asignación de tarjetas: Tomar una tarjeta de la columna "To do" y moverla a la columna "In progress". Esto indica que se está trabajando en esa tarea específica.

Trabajo en tarjetas: Realizar el trabajo necesario para completar la tarea asignada. Esto puede incluir la investigación, el desarrollo de código, la configuración de herramientas, entre otros.

Actualización del estado: Una vez que se haya completado una tarea, mover la tarjeta a la columna "Done". Esto indica que la tarea se ha completado con éxito.

Revisión y seguimiento: Realizar revisiones periódicas del tablero de Trello para evaluar el progreso y realizar ajustes si es necesario. Esto puede incluir la reasignación de tarjetas, la actualización de prioridades o la adición de nuevas tarjetas si surgen nuevos objetivos o requisitos.

Comunicación y colaboración: Utilizar los comentarios de las tarjetas de Trello para comunicarse con otros miembros del equipo o para hacer seguimiento de cualquier problema o pregunta relacionada con una tarea específica.

4. Desarrollo específico de la contribución

En este apartado se detallarán los aspectos relacionados con el desarrollo propio del sistema a implementar, así como la planificación, el desarrollo y la evaluación del mismo.

4.1. Planificación / Análisis / Requisitos

El punto de partido de nuestro proyecto se basa en el análisis de nuestros objetivos tanto generales como específicos para realizar una correcta definición de requisitos en forma de historias de ususario:

- Como administrador del proyecto, quiero definir la infraestructura en Saas de Elastic para poder utilizar sus servicios de monitorización y observabilidad de manera eficiente.
- Como administrador del proyecto, quiero realizar el despliegue de un cluster de EKS utilizando Terraform para asegurar una infraestructura escalable y gestionable.
- Como desarrollador, quiero realizar cambios en el código fuente de la aplicación para integrar el APM Agent de Elastic y así recopilar información de rendimiento y estado de la aplicación.

- Como desarrollador, quiero construir una imagen Docker de la aplicación con los cambios necesarios para recoger información de APM, para facilitar su despliegue y gestión.
- Como administrador del proyecto, quiero desplegar la aplicación en EKS, estableciendo las réplicas y servicios necesarios, para asegurar su disponibilidad y escalabilidad.
- Como administrador del proyecto, quiero desplegar y configurar un Elastic-Agent en el cluster de EKS como Daemonset, para recoger información de K8s y tener una visión completa del estado del cluster.
- Como administrador del proyecto, quiero desplegar y configurar una máquina EC2 para instalar el Elastic-Agent y recoger información de AWS, para tener una visión completa del estado de los recursos de AWS utilizados.
- Como analista de datos, quiero identificar los KPIs relevantes de la información recogida por APM, para poder evaluar el rendimiento y la calidad de la aplicación.
- Como analista de datos, quiero identificar los KPIs relevantes de la información recogida de K8s, para evaluar el rendimiento y la eficiencia del cluster de EKS.
- Como analista de datos, quiero identificar los KPIs relevantes de la información recogida de AWS, para evaluar el rendimiento y la utilización de los recursos de AWS.
- Como administrador del proyecto, quiero definir alertas basadas en los KPIs de APM, para ser notificado de posibles problemas o degradaciones en el rendimiento de la aplicación.
- Como administrador del proyecto, quiero definir alertas basadas en los KPIs de K8s, para ser notificado de posibles problemas o degradaciones en el cluster de EKS.
- Como administrador del proyecto, quiero definir alertas basadas en los KPIs de AWS, para ser notificado de posibles problemas o degradaciones en los recursos de AWS utilizados.
- Como administrador del proyecto, quiero definir un canal de comunicación para recibir las alertas generadas por APM, para poder actuar rápidamente en caso de problemas.
- Como administrador del proyecto, quiero definir un canal de comunicación para recibir las alertas generadas por K8s, para poder actuar rápidamente en caso de problemas en el cluster de EKS.

- Como administrador del proyecto, quiero definir un canal de comunicación para recibir las alertas generadas por AWS, para poder actuar rápidamente en caso de problemas en los recursos de AWS utilizados.
- Como analista de datos, quiero establecer dashboards que muestren la información recogida por APM, para tener una visión visual y en tiempo real del rendimiento de la aplicación.
- Como analista de datos, quiero establecer dashboards que muestren la información recogida de K8s, para tener una visión visual y en tiempo real del estado y rendimiento del cluster de EKS.
- Como analista de datos, quiero establecer dashboards que muestren la información recogida de AWS, para tener una visión visual y en tiempo real del estado y rendimiento de los recursos de AWS utilizados.

Una vez establecidas todos los requisitos en forma de historias de usuario hay que priorizarlas, para ello utilizaremos el método MoSCoW, en el cual las historias de usuario se priorizan en cuatro categorías: Must (debe hacerse), Should (debería hacerse), Could (podría hacerse) y Won't (no se hará). Las historias de usuario que se consideran esenciales y fundamentales para el proyecto se han clasificado como "Must". Las historias de usuario que son importantes, pero no esenciales se han clasificado como "Should". Las historias de usuario que son opcionales y podrían agregarse si hay tiempo y recursos se han clasificado como "Could".

Tabla 2. Tabla de requisitos según prioridad.

Prioridad	Requisito
Must	Definir infraestructura en SaaS de Elastic
	Despliegue de cluster de EKS mediante terraform
	Cambiar el código fuente para recoger información de APM mediante el APM Agent de Elastic
	Construcción de la imagen docker de la aplicación con los cambios para recoger información de APM
	Desplegar la aplicación en EKS estableciendo replicas y servicios necesarios
	Despliegue y configuración de un Elastic-Agent en el cluster de EKS como Daemonset para recoger información de K8s
	Despliegue y configuración de una máquina ec2 para instalar el Elastic-Agent para recoger información de AWS
Should	Identificar KPIs de la información recogida de APM
	Identificar KPIs de la información recogida de K8s
	Identificar KPIs de la información recogida de AWS
	Definir alertas en base a las KPIs de APM
	Definir alertas en base a las KPIs de K8s
	Definir alertas en base a las KPIs de AWS
	Definir un canal de comunicación para las alertas de APM

	Definir un canal de comunicación para las alertas de K8s
	Definir un canal de comunicación para las alertas de AWS
Could	Establecer dashboards de la información recogida de APM
	Establecer dashboards de la información recogida de K8s
	Establecer dashboards de la información recogida de AWS

Fuente: Elaboración propia.

En resumen, el análisis y los requisitos para este proyecto se centran en implementar una infraestructura adecuada, realizar despliegues eficientes, recopilar información relevante y establecer mecanismos de alerta y visualización. Estas acciones permitirán mejorar la monitorización y observabilidad de las aplicaciones desplegadas en AWS, optimizando su rendimiento y garantizando una experiencia de usuario excepcional.

4.2. Descripción del sistema desarrollado / Implementación

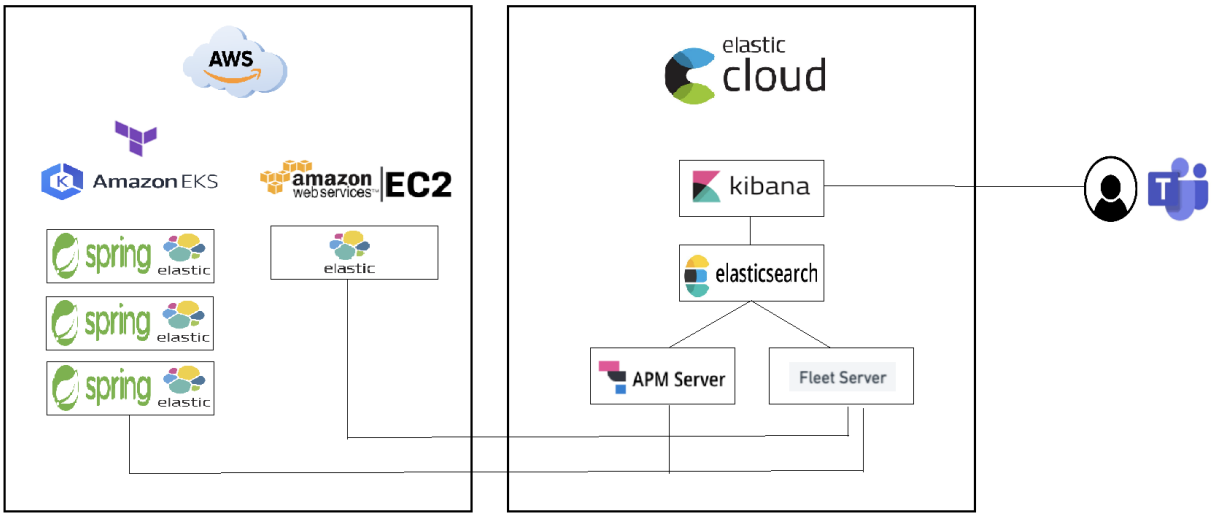
El inicio de este proceso de desarrollo integral del sistema de observación y supervisión requiere la atención primordial hacia un requisito fundamental: la creación de una aplicación web que servirá como piedra angular y sobre la cual se construirá el resto del sistema. Esta aplicación web será posteriormente implementada en un entorno Kubernetes (EKS) alojado en AWS, el cual proporciona un sólido marco de infraestructura para la ejecución y gestión de aplicaciones en la nube.

El punto de partida consistió en llevar a cabo un meticuloso análisis y evaluación de diversas alternativas de aplicaciones disponibles en el mercado. Tras un exhaustivo examen de las opciones disponibles, se llegó a la determinación de que la elección más apropiada sería utilizar como cimiento la aplicación Petclinic. Esta elección se respalda en el hecho de que dicha aplicación se encuentra accesible en el repositorio oficial de Github de Spring Boot, una fuente de confianza y reconocimiento en la comunidad de desarrollo (Spring-Projects, s. f.).

Con el siguiente paso, se emprende la creación de una imagen Docker, la cual se convierte en el contenedor para nuestra aplicación web. Esta imagen se genera a partir de un archivo Dockerfile especialmente configurado para este propósito. Dentro del proceso de construcción de la imagen, se incorpora el código de la aplicación, y se configuran elementos cruciales, tales como la integración del agente de APM (Application Performance Monitoring) y la adecuada configuración de las variables de entorno en Java para establecer una conexión efectiva con el servidor de APM. Esta meticulosa preparación garantiza que la aplicación esté equipada para proporcionar un monitoreo preciso y completo de su rendimiento y operatividad.

Una vez que se ha completado la creación y configuración de la imagen de la aplicación, se abre la puerta para definir la arquitectura del sistema en su totalidad. Esta etapa representa un momento crucial en el desarrollo, donde se toman decisiones estratégicas acerca de cómo los diferentes componentes y servicios interactuarán y se integrarán para alcanzar los objetivos de observación y supervisión establecidos. La arquitectura diseñada será el cimiento sólido sobre el cual se construirá y desplegará el sistema completo, permitiendo un control efectivo y una supervisión integral de todas las facetas de la aplicación y la infraestructura subyacente. Esta planificación arquitectónica es esencial para garantizar la eficacia y la confiabilidad del sistema de observación y supervisión en su conjunto.

Figura 1. *Arquitectura del sistema completo.*



Fuente: Elaboración propia.

Como se puede apreciar en el diagrama anterior, se distinguen dos componentes fundamentales: la arquitectura desplegada en el entorno de AWS y la arquitectura implementada en Elastic Cloud.

En lo que respecta a la porción relacionada con AWS, consta de los siguientes elementos:

- Clúster de Kubernetes (EKS) desplegado mediante la utilización de Terraform. La implementación de este despliegue se llevó a cabo siguiendo las directrices oficiales proporcionadas por HashiCorp, y el código de Terraform correspondiente se encuentra disponible en su repositorio oficial en GitHub (Hashicorp, s. f.). Una vez completado el despliegue del clúster, se han creado los siguientes elementos:
    - Despliegue de Petclinic: Este despliegue se realiza utilizando una imagen Docker que ha sido generada a partir del código fuente de Spring, e incluye la configuración del agente de APM que recoge las métricas y logs de aplicación.
    - Elastic-Agent Daemonset: Este componente se encarga de recopilar las métricas y registros relativos al estado del clúster de Kubernetes.
- Instancia EC2 con Elastic-Agent instalado: Esta instancia está configurada para recopilar métricas y supervisar el estado de la infraestructura en la nube de AWS.

En lo que respecta a la parte correspondiente a Elastic Cloud, se compone de los siguientes elementos:

- Kibana: Se trata de una interfaz gráfica que permite a los desarrolladores visualizar las métricas y registros relacionados tanto con el rendimiento de la aplicación como con el estado del clúster de Kubernetes y la infraestructura en la nube de AWS.
- Elasticsearch: En este sistema, se emplea exclusivamente un nodo que alberga el motor de búsqueda.
- APM Server: Este servidor tiene la función de recibir la información proporcionada por el APM Agent, que está instalado en la imagen de Docker utilizada.
- Fleet Server: Actúa como receptor de la información proporcionada por los agentes de Elastic desplegados tanto en AWS como en los nodos de Kubernetes. De esta manera, recopila los datos pertinentes para monitorizar el estado de ambos entornos de manera efectiva.

Una vez que ambas arquitecturas han sido implementadas de manera adecuada y han establecido una comunicación efectiva, con la aplicación funcionando en el servicio Amazon Elastic Kubernetes Service (EKS) de Amazon Web Services (AWS) y el componente Elastic encargado de recopilar la información, procedemos a definir Indicadores Clave de Desempeño (KPIs) con el propósito de crear alertas o presentar visualmente en paneles de control de forma expedita.

Con ese fin, procederemos a segmentar los Indicadores Clave de Desempeño (KPIs) en las siguientes categorías:

- APM (Gestión del Rendimiento de Aplicaciones): Esta sección se centrará en KPIs directamente vinculados al rendimiento de la aplicación.
- Arquitectura: En este apartado, se abordarán los KPIs que estén relacionados con la salud y el funcionamiento del clúster Kubernetes desplegado en AWS.

Esta organización permitirá una gestión más eficaz de los indicadores clave y facilitará la creación de alertas o la visualización de datos en los paneles de control correspondientes.

En la tabla que sigue, se presentarán los Indicadores Clave de Desempeño (KPIs) previamente definidos:

Tabla 3. *Claves de desempeño (KPIs).*

Componente	Clave de Desempeño (KPIs)	Descripción
APM	Tiempo de respuesta promedio.	Medición del tiempo de respuesta de la aplicación.
	Tasa de errores de la aplicación.	Registro de errores o fallos en la aplicación.
	Tiempo de respuesta promedio de dependencias.	Medición del tiempo de respuesta de las dependencias de la aplicación.

	Disponibilidad de servicio de aplicación.	Consulta constante para establecer la disponibilidad (up/down) de la aplicación.
Arquitectura	Uso de recursos de los nodos del clúster (CPU y memoria)	Monitoreo del consumo de recursos en los nodos.
	Uso de recursos de los podd del clúster (CPU y memoria)	Monitoreo del consumo de recursos en los pods.
	Rendimiento de los balanceadores.	Monitoreo del rendimiento de los balanceadores

Fuente: Elaboración propia.

Y por otro lado como dashboards visuales se establecen los siguientes:

Tabla 4. *Paneles de control.*

Componente	Dashboard	Descripción	Anexo
APM	Overview	Proporciona una vista general del rendimiento de la aplicación, incluyendo transacciones, dependencias, errores y un mapa de servicios.	Anexo A
	Transacciones	Ofrece información detallada sobre las transacciones realizadas en la aplicación.	Anexo B
	Dependecias	Muestra las dependencias entre los componentes de la aplicación para identificar cuellos de botella o puntos críticos.	Anexo C

	Errores	Registra y presenta los errores que se producen en la aplicación, permitiendo un seguimiento y análisis detallados.	Anexo D
	Service Map	Ofrece una representación gráfica de la arquitectura de servicios, ayudando a visualizar la estructura de la aplicación.	Anexo E
	Monitors	Proporciona una visión clara de la disponobilidad de la aplicació desde diferentes regiones.	Anexo F
Arquitectura	Kubernetes Overview	Proporciona una visión general del estado y el rendimiento del clúster Kubernetes.	Anexo G
	Kubernetes Nodes	Detalla información sobre los nodos individuales en el clúster Kubernetes, incluyendo su estado y recursos.	Anexo H
	Kubernetes Pods	Ofrece datos específicos sobre los pods desplegados en el clúster, como su estado y consumo de recursos.	Anexo I
	Kubernetes Deployments	Ofrece datos específicos sobre los deployments desplegados en el clúster, como el estado de las réplicas de cada uno.	Anexo J
	Kubernetes Services	Ofrece datos específicos sobre los servicios desplegados en el clúster.	Anexo K
	[Metrics AWS] Usage Overview	Brinda una visión general del uso de recursos de AWS en la arquitectura.	Anexo M

	[Metric AWS] ELB Overview	Muestra métricas relacionadas con los balanceadores de carga (ELB) en AWS.	Anexo N
	[Metrics AWS] EC2 Overview	Proporciona una visión general de las instancias de EC2 en AWS, incluyendo su estado y recursos utilizados.	Anexo O

Fuente: Elaboración propia.

Estas métricas clave de desempeño (KPIs) y paneles de control desempeñan un papel fundamental al delimitar y presentar una perspectiva integral del rendimiento de la aplicación, abarcando desde sus etapas iniciales de desarrollo hasta la compleja infraestructura arquitectónica en el entorno de AWS. Dicha amalgama de indicadores no solo permite la evaluación detallada de aspectos críticos en la gestión de aplicaciones, sino que también se extiende hacia la esfera de la arquitectura, ofreciendo una comprensión completa y precisa del estado operativo, la eficiencia y la confiabilidad de todo el ecosistema aplicativo. Este enfoque holístico resulta esencial en la toma de decisiones estratégicas y la garantía de un rendimiento óptimo a lo largo de todo el ciclo de vida de la aplicación.

4.3. Evaluación

En el presente apartado, se llevará a cabo una evaluación exhaustiva del sistema de observabilidad y monitorización implementado en el entorno de Amazon Elastic Kubernetes Service (EKS) de AWS. Dada la importancia crítica de la observabilidad en la gestión efectiva de aplicaciones desplegadas en contenedores Kubernetes, se busca analizar a fondo la eficacia y la capacidad de este sistema para proporcionar una visión completa y precisa del rendimiento, la disponibilidad y la salud de las aplicaciones en el clúster EKS. Este análisis se centrará en la identificación de las métricas clave vistas anteriormente y la capacidad de generación de alertas, con el objetivo de evaluar en su totalidad la robustez y la utilidad de la solución de observabilidad en el contexto de la infraestructura desplegada en EKS.

Asimismo, como parte integral de la evaluación del sistema de observabilidad y monitorización en el entorno de Amazon Elastic Kubernetes Service (EKS) de AWS, se ha habilitado un canal de comunicación esencial para la generación de alertas críticas. Esto se ha

logrado mediante la implementación de un webhook y un conector de Elastic que permite la integración efectiva con la plataforma de colaboración Teams de Microsoft. La configuración de este canal proporciona una vía de comunicación inmediata y eficaz para notificar a los equipos operativos y de desarrollo sobre situaciones excepcionales o eventos importantes, permitiendo una respuesta ágil y coordinada ante desafíos que puedan surgir en la operación del clúster EKS y las aplicaciones desplegadas en él.

Con el propósito de evaluar de manera precisa la capacidad de respuesta y el rendimiento de la aplicación desplegada en Amazon Elastic Kubernetes Service (EKS) de AWS, se ha empleado una metodología rigurosa de simulación de tráfico. Para llevar a cabo esta simulación, se ha utilizado el lenguaje de programación Python en conjunto con su biblioteca especializada, Locust. Esta herramienta ha permitido la creación de escenarios realistas de carga y ha facilitado la generación de tráfico controlado y escalable en el entorno de la aplicación. De esta manera, se ha logrado una evaluación exhaustiva de la capacidad de la aplicación para manejar situaciones de alta demanda, lo que resulta esencial para garantizar un funcionamiento óptimo y una experiencia del usuario sin contratiempos.

Locust es una herramienta que crea un conjunto de funciones de prueba que simulan un gran número de usuarios. Esto determinará el punto principal de ruptura en cuanto a rendimiento, seguridad y gestión de carga de la aplicación (Locust.io, s. f.).

Las alertas que se han definido son las siguientes:

Tabla 5. *Alertas y condiciones.*

Nombre	Condición	Every
[CPU Usage] K8s Nodes	CPU Usage is above 90% for the last 1 minutes.	1 min
[CPU Usage] K8s Pods	CPU Usage is above 50% for the last 1 minutes.	1 min

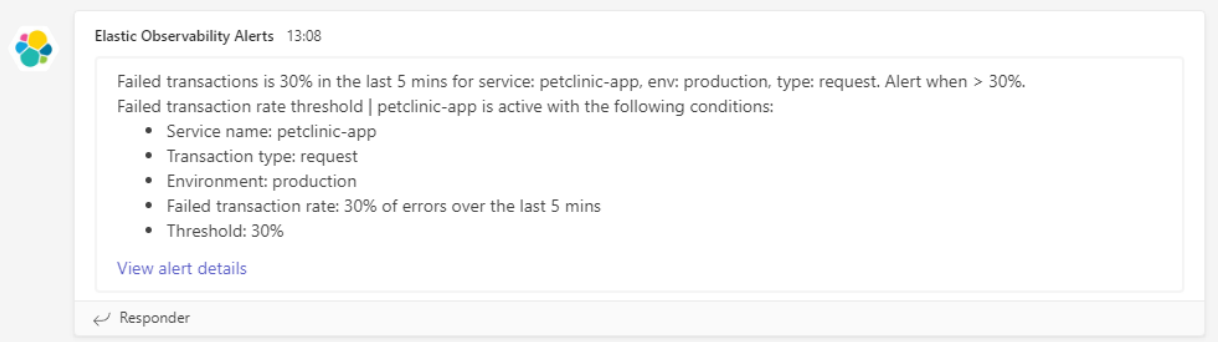
[Memory Usage] K8s Nodes	Memory Usage is above 80% for the last 1 minutes.	1 min
[Memory Usage] K8s Pods	Memory Usage is above 80% for the last 1 minutes.	1 min
[Uptime] Petclinic Front-End	Petclinic Front-End monitor is down or Petclinic Front-End monitor is recovered.	1 min
Latency threshold | petclinic-app	Service petclinic: average latency is above 500ms for the last 5 minutes.	5 min
Failed transaction rate threshold | petclinic-app	Service petclinic: average failed transaction rate is above 30% for the last 5 minutes	5 mins
Error count threshold | petclinic-app	Service petclinic: error count is above 100 errors for the last 5 minutes	5 mins

Fuente: Elaboración propia.

Es importante resaltar que estas condiciones se encuentran diseñadas para un sistema que se halla en una fase de estudio, y como tal, muchas de las reglas han sido ajustadas con el propósito de verificar su funcionamiento, no necesariamente para representar un escenario real.

En la siguiente ilustración, se presenta un ejemplo del mensaje que se envía al canal de Microsoft Teams:

Figura 2. *Ejemplo de mensaje de alerta en Microoft Teams.*



Fuente: Elaboración propia.

En cada mensaje, se incluyen consistentemente los valores que han activado la alerta, información pertinente sobre el servicio o recurso afectado, así como un enlace directo que permite acceder rápidamente a la regla correspondiente en Elastic. En el Anexo O, hay más ejemplos de los resultados de estas alarmas.

Estas reglas han desempeñado un papel esencial en la evaluación de la viabilidad y eficacia del sistema en cuestión, desde una perspectiva funcional a nivel empresarial. Su implementación y aplicación rigurosa han proporcionado una evaluación crítica y detallada de la funcionalidad del sistema, permitiendo una evaluación precisa de su capacidad para satisfacer las necesidades y expectativas de la empresa.

5. Conclusiones y trabajo futuro

En este bloque se comentan las conclusiones extraídas tanto durante el desarrollo del sistema como al finalizar-lo. Además, se establecen futuras líneas de trabajo para este proyecto.

5.1. Conclusiones

Este apartado de conclusiones vamos a dividirlo en dos áreas, las conclusiones referentes a la viabilidad técnica y la solución aportada por el sistema, y las conclusiones de viabilidad económica.

5.1.1. Conclusiones técnicas y de solución

Este trabajo de investigación de nivel de maestría se ha centrado en la observabilidad y monitorización de aplicaciones implementadas en la plataforma de Amazon Web Services (AWS), con un enfoque particular en la utilización de la herramienta Elastic Observability en el contexto de Amazon Elastic Kubernetes Service (EKS). A lo largo de esta indagación, se han derivado diversas conclusiones de relevancia:

- La relevancia estratégica de la observabilidad en AWS: La migración de aplicaciones hacia la nube, en particular dentro del ecosistema de AWS, plantea imperativos estratégicos y técnicos. Se ha constatado la necesidad imperativa de la observabilidad como una disciplina crucial para la comprensión y optimización del rendimiento de las aplicaciones en el paradigma de la nube contemporánea.
- Elastic Observability como solución Eficaz: Elastic Observability se ha revelado como una solución altamente efectiva para la recopilación, análisis y visualización de datos provenientes de múltiples fuentes en el contexto de AWS EKS. Su capacidad de integración con los servicios nativos de AWS, como AWS CloudWatch, habilita una comprensión integral de la infraestructura subyacente y de las aplicaciones implementadas.
- Identificación de cuellos de botella y optimización: Elastic Observability ha probado ser una herramienta valiosa para la identificación oportuna de cuellos de botella y problemas de rendimiento en tiempo real. Esta funcionalidad concede a las organizaciones la capacidad de tomar acciones proactivas para optimizar tanto sus aplicaciones como la infraestructura subyacente.
- Resolución efectiva de problemas: La capacidad de correlacionar métricas de infraestructura con datos de monitoreo de rendimiento de aplicaciones (APM) y registros de servicios simplifica el proceso de resolución de problemas. Esta capacidad

no solo reduce los períodos de inactividad, sino que también eleva la calidad de la experiencia del usuario final.

- Mejores prácticas y consideraciones de diseño: En este trabajo se han abordado los obstáculos comunes encontrados durante la implementación de Elastic Observability en el entorno de AWS EKS, además de proporcionar directrices y mejores prácticas para garantizar un monitoreo eficaz y eficiente.
- Validación de eficacia mediante pruebas empíricas: La efectividad de Elastic Observability se ha validado mediante pruebas de rendimiento y comparaciones con métricas y umbrales predefinidos. Estos procedimientos han ratificado la capacidad de Elastic Observability para suministrar información crucial sobre el rendimiento de la aplicación en tiempo real.

En resumen, este trabajo ha realizado una contribución significativa al campo de la observabilidad y monitorización en entornos nativos de la nube, especialmente en AWS. La implementación exitosa de Elastic Observability ha demostrado ser una estrategia efectiva para mejorar la visibilidad y el rendimiento de las aplicaciones en el entorno de la nube, con posibles repercusiones positivas en la eficiencia operativa y en la satisfacción del cliente. La observabilidad y la monitorización apropiadas son elementos fundamentales en el camino hacia la excelencia en la computación en la nube, y este estudio brinda valiosas perspectivas y orientación para aquellas organizaciones que buscan optimizar sus operaciones en AWS.

5.1.2. Conclusiones viabilidad económica

Si bien la implementación de Elastic Observability ha demostrado ser altamente efectiva en la mejora de la observabilidad y la monitorización de las aplicaciones en el entorno de AWS EKS, es fundamental abordar las consideraciones financieras que acompañan a esta solución. Varias áreas clave de coste merecen una atención cuidadosa:

- Suscripción a Elastic Cloud: La adopción de Elastic Cloud suele requerir una suscripción, y el coste asociado depende de diversos factores, como la escala de implementación y las características específicas necesarias. Las organizaciones deben evaluar cuidadosamente sus requisitos y presupuestos para seleccionar la suscripción más adecuada.
- Uso de funcionalidades premium: Elastic ofrece características avanzadas, como el sistema de conectores para las alertas a diferentes canales, que pueden incurrir en

costes adicionales. Las organizaciones deben considerar si estas funcionalidades son necesarias para sus casos de uso y si el beneficio justifica el gasto adicional.

- Costos de almacenamiento de datos: La recopilación y retención de datos de monitoreo pueden generar costos de almacenamiento a medida que se acumulan grandes volúmenes de datos. Es importante definir políticas de retención de datos adecuadas para evitar costos innecesarios.
- Costes de acceso a la API de AWS: Si Elastic Observability se integra estrechamente con los servicios de AWS, es relevante tener en cuenta los costes asociados con las llamadas a la API de AWS. Dependiendo de la frecuencia y volumen de estas llamadas, los costos pueden ser significativos y deben ser gestionados y optimizados con prudencia.
- Optimización de recursos: La eficiencia en el uso de recursos es esencial para controlar los costes. Las organizaciones deben monitorear de cerca el rendimiento y la capacidad de Elastic Observability y ajustar sus recursos en consecuencia para evitar el exceso de aprovisionamiento.
- Evaluación continua de costes: La evaluación periódica de los costes asociados con Elastic Observability es crucial. Esto implica revisar y ajustar las suscripciones, políticas de retención de datos y estrategias de integración con servicios de AWS a medida que evolucionan las necesidades y la carga de trabajo.

Es importante que las organizaciones adopten un enfoque proactivo para gestionar y optimizar los costes relacionados con la observabilidad y monitorización en AWS. Esto garantizará que los beneficios obtenidos de Elastic Observability superen con creces los costes incurridos y que la inversión en esta herramienta sea financieramente justificada. La eficiente gestión de gastos es un componente esencial para garantizar el éxito continuo de la implementación de Elastic Observability en un entorno de AWS EKS.

5.2. Líneas de trabajo futuro

En este apartado se establecen futuras líneas de trabajo para la continuación de este sistema desarrollado.

5.2.1. Mejora de la observabilidad en EKS de AWS mediante las funcionalidades de machine learning de Elastic

El objetivo principal de esta nueva línea de trabajo es utilizar técnicas de Machine Learning para mejorar el registro y la monitorización de aplicaciones desplegadas en la plataforma de Amazon Web Services (AWS), con un enfoque específico en la detección proactiva de anomalías en tiempo real. Como hemos visto anteriormente, estos trabajos de monitoreo y analisis de métricas son cruciales en entornos de AWS para garantizar un rendimiento óptimo y la disponibilidad de las aplicaciones. Sin embargo, las infraestructuras modernas en la nube generan enormes cantidades de datos de monitoreo, lo que hace que la detección temprana de problemas y anomalías sea un desafío. La adopción de técnicas de Machine Learning puede permitir una detección proactiva y precisa de anomalías, lo que mejora significativamente la capacidad de respuesta y la eficiencia operativa.

Como metodología para esta nueva línea de trabajo se propone lo siguiente:

- Entrenamiento de modelos de machine learning: Se desarrollarán y entrenarán modelos de Machine Learning, como algoritmos de detección de anomalías y modelos de series temporales, utilizando conjuntos de datos históricos. Estos modelos aprenderán los patrones normales de comportamiento del sistema y las aplicaciones.
- Implementación en tiempo real: Los modelos entrenados se implementarán en un entorno en tiempo real dentro de la infraestructura de Elastic para monitorear continuamente los datos de rendimiento. Se utilizarán técnicas de procesamiento en tiempo real para habilitar la detección inmediata de anomalías.
- Alertas y acciones automatizadas: Cuando se detecten anomalías, se generarán alertas automáticas y, en algunos casos, se podrían desencadenar acciones automatizadas para abordar problemas potenciales antes de que afecten a las aplicaciones.
- Evaluación y aprendizaje continuo: Los modelos se evaluarán continuamente en función de su precisión en la detección de anomalías y se ajustarán conforme cambien las condiciones y los patrones de comportamiento.

Aplicando estas directrices se pretende obtener los siguientes beneficios:

- Detección proactiva de anomalías: La integración de Machine Learning permitirá detectar anomalías de manera proactiva y, en muchos casos, incluso antes de que

impacten negativamente en las aplicaciones, lo que mejora la confiabilidad y la disponibilidad.

- Optimización de recursos: La detección temprana de problemas permite una asignación más eficiente de recursos y una reducción de costes al evitar tiempos de inactividad no planificados.
- Mejora de la experiencia del usuario: La resolución rápida de problemas y la reducción de interrupciones mejoran la experiencia del usuario final.
- Aprendizaje automatizado: Los modelos de Machine Learning pueden aprender y adaptarse a patrones cambiantes de manera automática, lo que reduce la carga operativa.

En resumen, esta nueva línea de trabajo representa una evolución natural en el campo de la observabilidad y monitorización en la nube, aprovechando las capacidades de Machine Learning que proporciona Elastic para mejorar la eficacia y la eficiencia de las operaciones en entornos de AWS.

5.2.2. Mejora de la seguridad en entornos DevSecOps en AWS mediante Elastic Security

El objetivo principal de esta nueva línea de trabajo es utilizar Elastic Security para fortalecer la seguridad en el contexto de prácticas de DevSecOps en aplicaciones desplegadas en Amazon Web Services (AWS). Se busca la integración de Elastic Security en todo el ciclo de vida de desarrollo y despliegue de aplicaciones.

DevSecOps es una metodología que enfatiza la colaboración continua entre equipos de desarrollo, operaciones y seguridad para integrar la seguridad desde el principio en el ciclo de vida de desarrollo de aplicaciones. La seguridad en la nube es fundamental en un entorno como AWS. Elastic Security ofrece un conjunto de herramientas y capacidades que pueden utilizarse para la detección, prevención y respuesta a amenazas en tiempo real, lo que lo convierte en un candidato valioso para fortalecer la seguridad en entornos DevSecOps.

Como metodología para esta nueva línea de trabajo se propone lo siguiente:

- Integración de Elastic Security: Se implementará Elastic Security en la infraestructura de Elastic Cloud y se integrará en los procesos de desarrollo y despliegue de

aplicaciones. Esto incluye la configuración de detección de amenazas, políticas de seguridad y análisis de registros de seguridad.

- Automatización de pruebas de seguridad: Se diseñarán y automatizarán pruebas de seguridad en todo el ciclo de vida de desarrollo, desde la fase de desarrollo hasta la implementación. Estas pruebas deben abordar vulnerabilidades conocidas y escenarios de amenazas potenciales.
- Análisis continuo de amenazas: Elastic Security proporcionará un monitoreo continuo para la detección de amenazas en tiempo real. Se configurarán alertas para notificar sobre actividades sospechosas o incidentes de seguridad.
- Gestión de incidentes: Se establecerá un proceso de gestión de incidentes que permita responder de manera efectiva a las amenazas detectadas. Esto incluye la respuesta a incidentes y la implementación de medidas correctivas.

Aplicando estas directrices se pretende obtener los siguientes beneficios:

- Mayor seguridad en el desarrollo y despliegue: La integración de Elastic Security garantiza que la seguridad sea una preocupación constante en todo el proceso de desarrollo y despliegue, lo que reduce el riesgo de vulnerabilidades y amenazas.
- Respuesta rápida a amenazas: La detección en tiempo real y las alertas permiten una respuesta más rápida a las amenazas, lo que minimiza el impacto de los incidentes de seguridad.
- Cumplimiento de normativas: La adopción de buenas prácticas de seguridad contribuye al cumplimiento de normativas y estándares de seguridad.
- Mayor concienciación en seguridad: La capacitación y la concienciación en seguridad mejoran la cultura de seguridad en la organización y reducen los errores humanos.
- Mejora de la imagen de la marca: Un enfoque sólido en la seguridad demuestra el compromiso de la organización con la protección de los datos y la confidencialidad de los clientes.

Resumiendo, esta línea de trabajo busca fortalecer la seguridad en entornos DevSecOps en AWS mediante la implementación de Elastic Security y la integración de prácticas de seguridad en todas las etapas del ciclo de vida de desarrollo y despliegue de aplicaciones.

Referencias bibliográficas

Atlassian. (n.d.). *Mejora la observabilidad de DevOps con tutoriales*. https://www.atlassian.com/es/devops/observability-tutorials

Bragin, T., & Bragin, T. (2020). Observabilidad con el Elastic Stack. *Elastic Blog*. https://www.elastic.co/es/blog/observability-with-the-elastic-stack

Dynatrace. (2022). Automatización y observabilidad - Digital Biz. *Digital Biz*. https://www.digitalbizmagazine.com/automatizacion-y-observabilidad/

Elastic. (n.d.). *DevOps observability*. https://www.elastic.co/. https://www.elastic.co/es/explore/devops-observability

Elastic Observability: una solución abierta y extensible para los equipos de DevOps. (n.d.). Elastic. https://www.elastic.co/es/observability

Introducción al logging y la observabilidad con el ELK Stack. (2021, April 13). Elastic. https://www.elastic.co/es/webinars/introduction-to-logging-and-observability-with-the-elk-stack

Martínez, J. (2023, March 31). Top herramientas DevOps: Del Monitoreo a la Observabilidad. *OpenWebinars.net*. https://openwebinars.net/blog/top-herramientas-devops-del-monitoreo-la-observabilidad/

Monitoreo de AWS con Elastic Observability. (n.d.). Elastic. https://www.elastic.co/es/observability/aws-monitoring

Monitoreo y observabilidad en AWS - Operaciones de la nube de AWS - Amazon Web Services. (n.d.). Amazon Web Services, Inc. https://aws.amazon.com/es/cloudops/monitoring-and-observability/

Al migrar desde Splunk, Entel permite una mayor capacidad de ingesta de datos y ahora puede proporcionar visibilidad las 24 horas del día, los 7 días de la semana, a los servicios

críticos en los equipos de TI y de negocios en Chile y Perú, todo impulsado por Elastic Stack. (n.d.). Elastic Customers. https://www.elastic.co/es/customers/entel

Auchan Francia: Creación de experiencias minoristas "figitales" con tecnología de daos de próxima generación. (n.d.). Elastic Customers. https://www.elastic.co/es/customers/auchan

Energisa monitorea sistemas críticos del negocio con Elastic Observability. (n.d.). Elastic Customers. https://www.elastic.co/es/customers/energisa

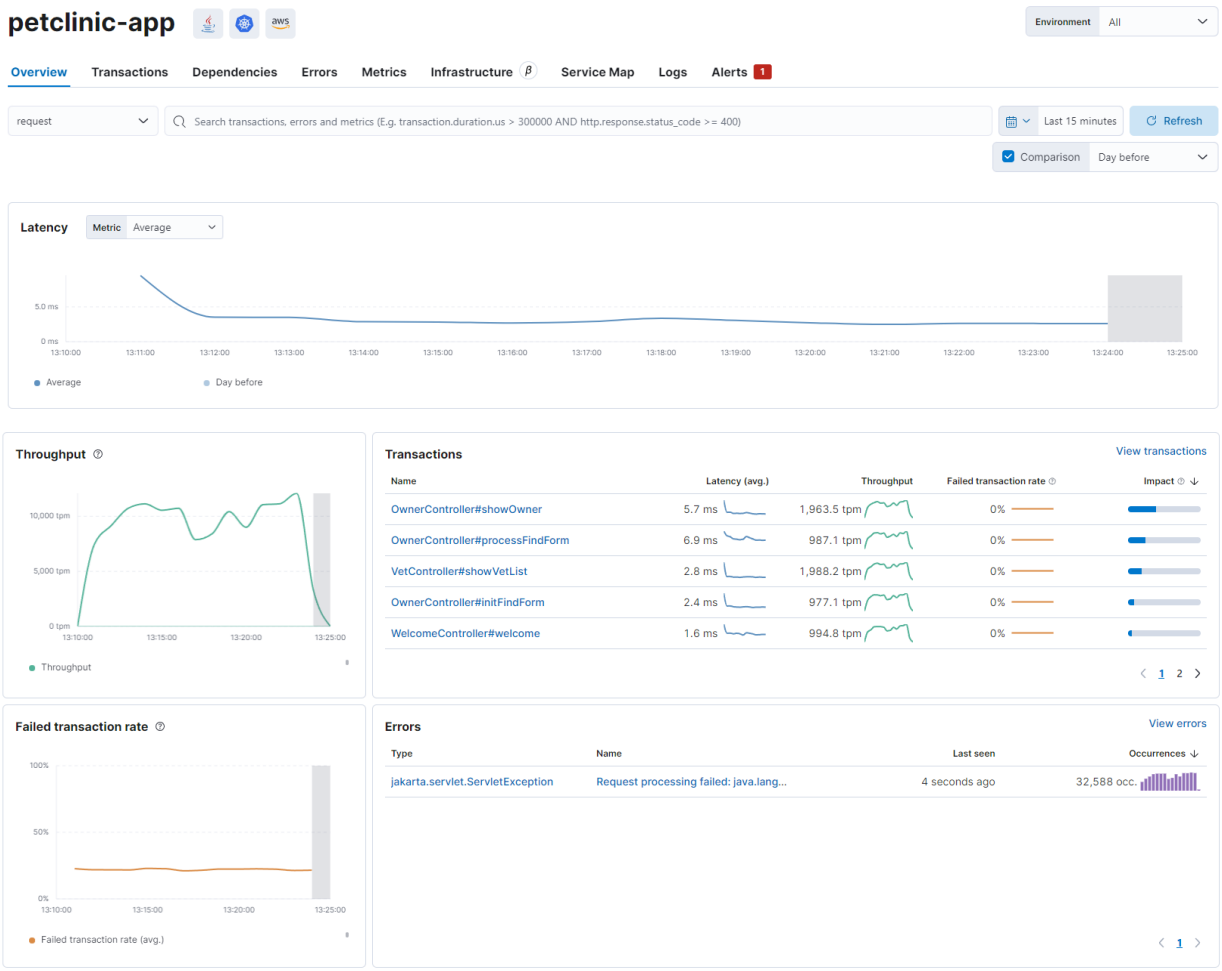
Zurich Insurance aumenta la confianza de los clientes y respalda el futuro con Elastic Cloud. (n.d.). Elastic Customers. https://www.elastic.co/es/customers/zurich-insurance-group

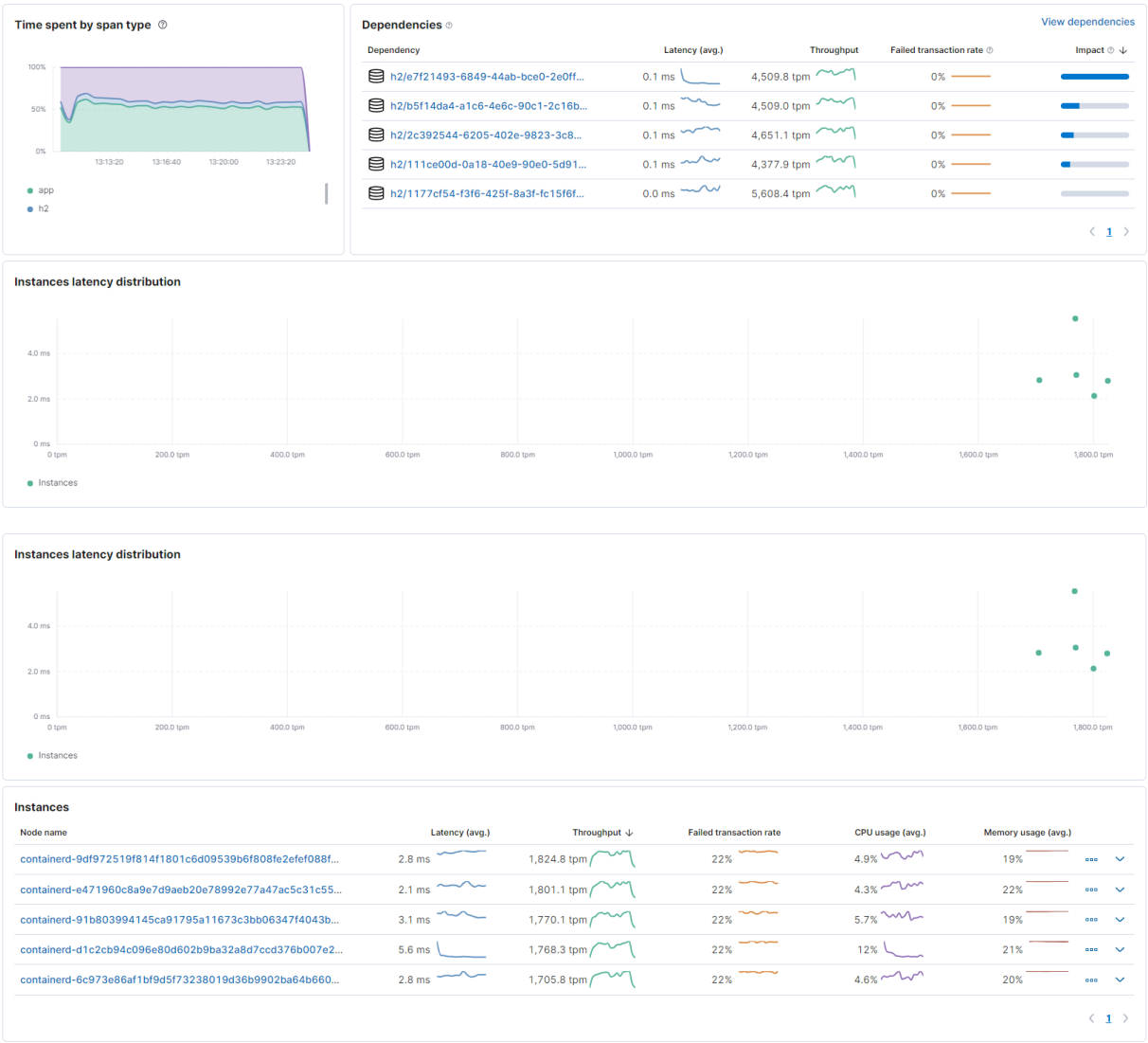
Hashicorp. (s. f.). *GitHub - hashicorp/learn-terraform-provision-eks-cluster*. GitHub. https://github.com/hashicorp/learn-terraform-provision-eks-cluster

Spring-Projects. (s. f.). *GitHub - Spring-projects/spring-Petclinic: a sample spring-based application*. GitHub. https://github.com/spring-projects/spring-petclinic

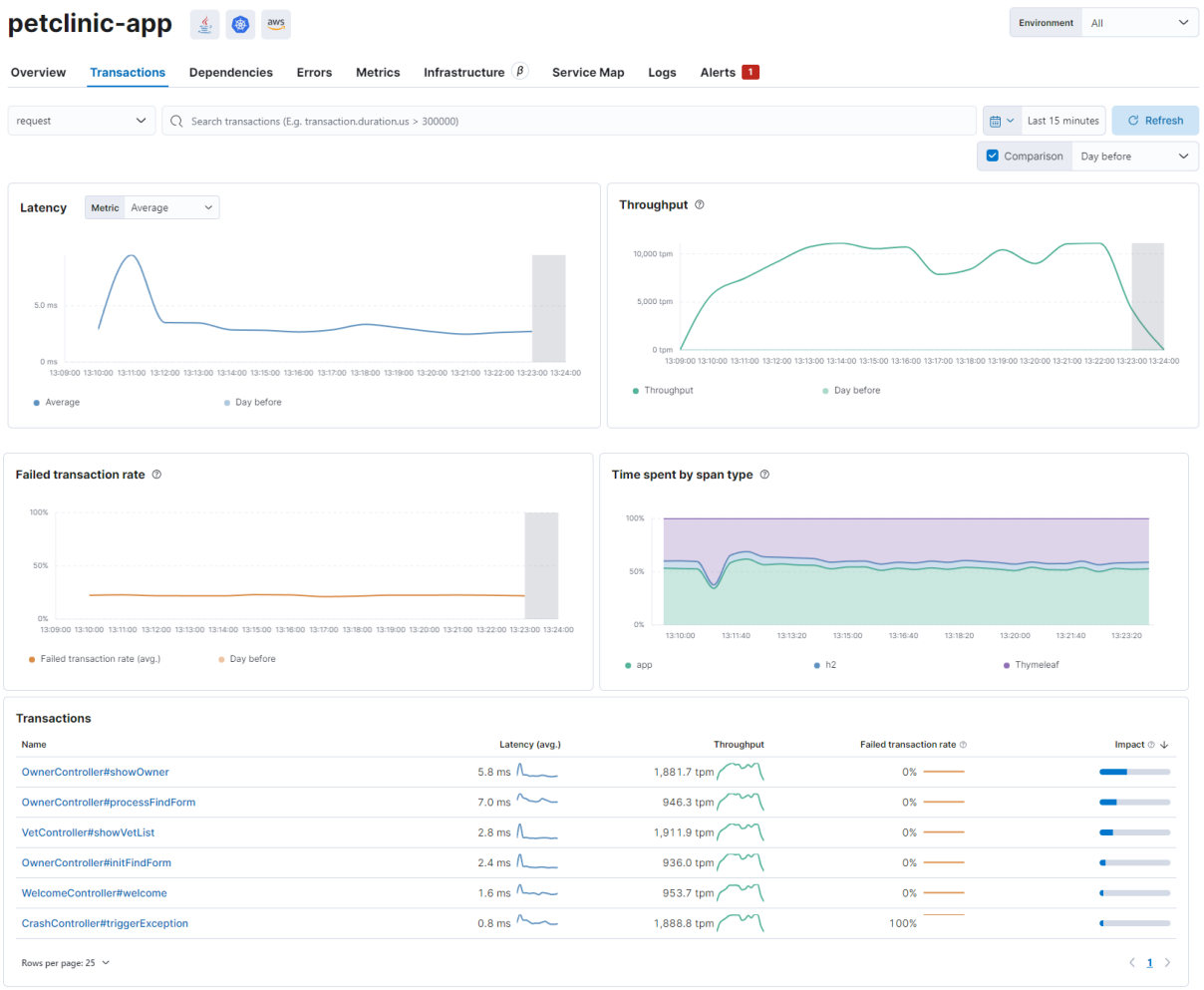
Locust.io. (s. f.). https://locust.io/

Anexo A. [APM] Overview

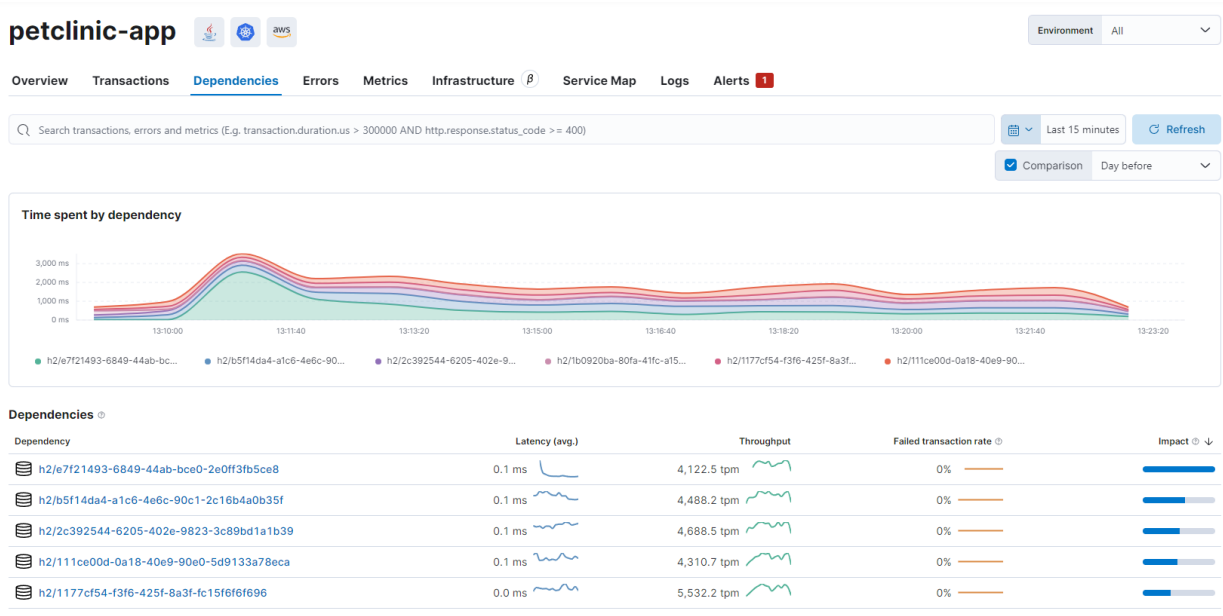



Time spent by span type
100%
50%
0%
13:13:20
13:16:40
13:20:00
13:23:20
app
h2
Dependencies
View dependencies
Dependency
Latency (avg.)
Throughput
Failed transaction rate
Impact
h2/e7f21493-6849-44ab-bce0-2e0ff...
0.1 ms
4,509.8 tpm
0%
h2/b5f14da4-a1c6-4e6c-90c1-2c16b...
0.1 ms
4,509.0 tpm
0%
h2/2c392544-6205-402e-9823-3c8...
0.1 ms
4,651.1 tpm
0%
h2/111ce00d-0a18-40e9-90e0-5d91...
0.1 ms
4,377.9 tpm
0%
h2/1177cf54-f3f6-425f-8a3f-fc15f6f...
0.0 ms
5,608.4 tpm
0%
1
Instances latency distribution
4.0 ms
2.0 ms
0 ms
0 tpm
200.0 tpm
400.0 tpm
600.0 tpm
800.0 tpm
1,000.0 tpm
1,200.0 tpm
1,400.0 tpm
1,600.0 tpm
1,800.0 tpm
Instances
Instances latency distribution
4.0 ms
2.0 ms
0 ms
0 tpm
200.0 tpm
400.0 tpm
600.0 tpm
800.0 tpm
1,000.0 tpm
1,200.0 tpm
1,400.0 tpm
1,600.0 tpm
1,800.0 tpm
Instances
Instances
Node name
Latency (avg.)
Throughput
Failed transaction rate
CPU usage (avg.)
Memory usage (avg.)
containerd-9df972519f814f1801c6d09539b6f808fe2efef088f...
2.8 ms
1,824.8 tpm
22%
4.9%
19%
containerd-e471960c8a9e7d9aeb20e78992e77a47ac5c31c55...
2.1 ms
1,801.1 tpm
22%
4.3%
22%
containerd-91b803994145ca91795a11673c3bb06347f4043b...
3.1 ms
1,770.1 tpm
22%
5.7%
19%
containerd-d1c2cb94c096e80d602b9ba32a8d7ccd376b007e2...
5.6 ms
1,768.3 tpm
22%
12%
21%
containerd-6c973e86af1bf9d5f73238019d36b9902ba64b660...
2.8 ms
1,705.8 tpm
22%
4.6%
20%
1

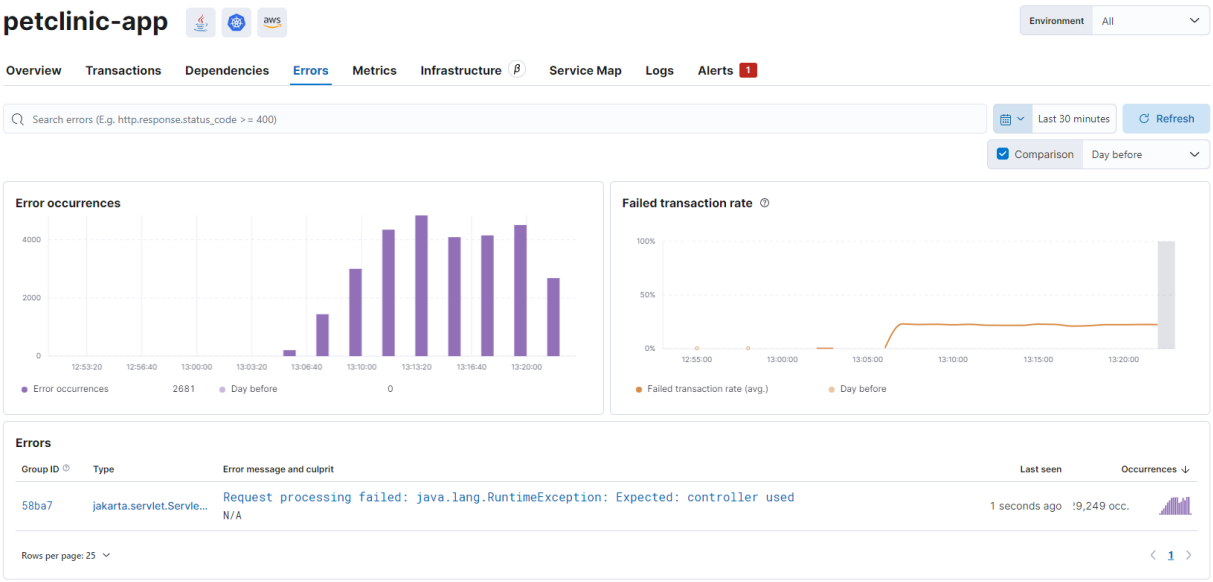
Anexo B. [APM] Transactions



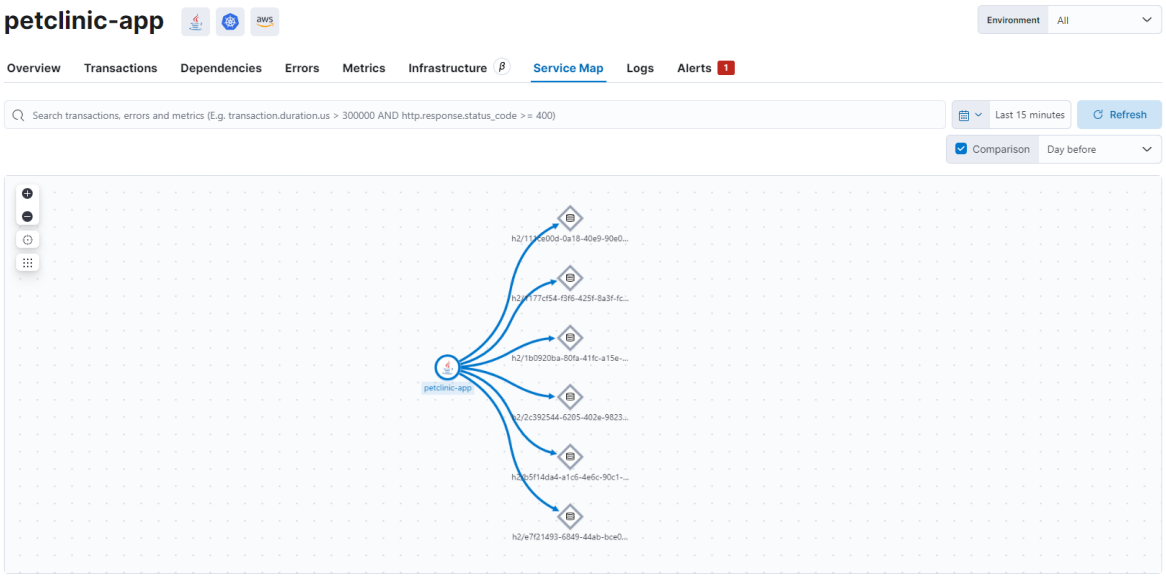
Anexo C. [APM] Dependencies



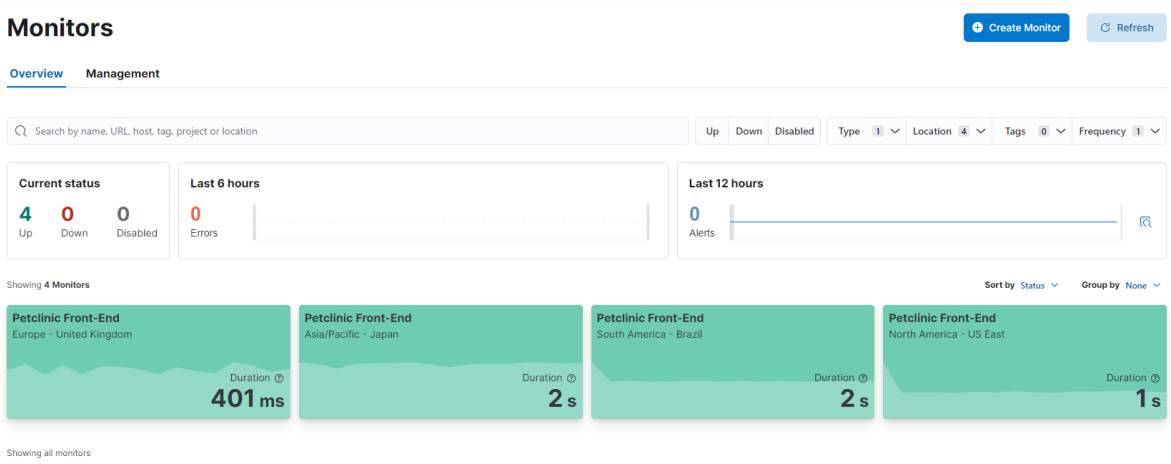
Anexo D. [APM] Errors



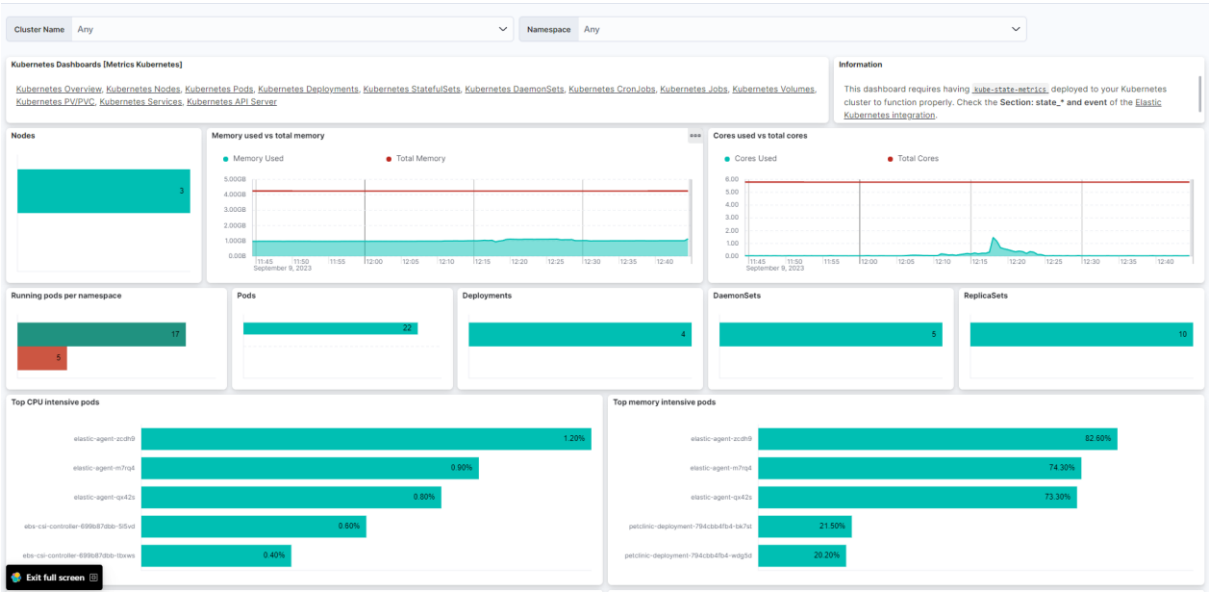
Anexo E. [APM] Service Map



Anexo F. Monitors

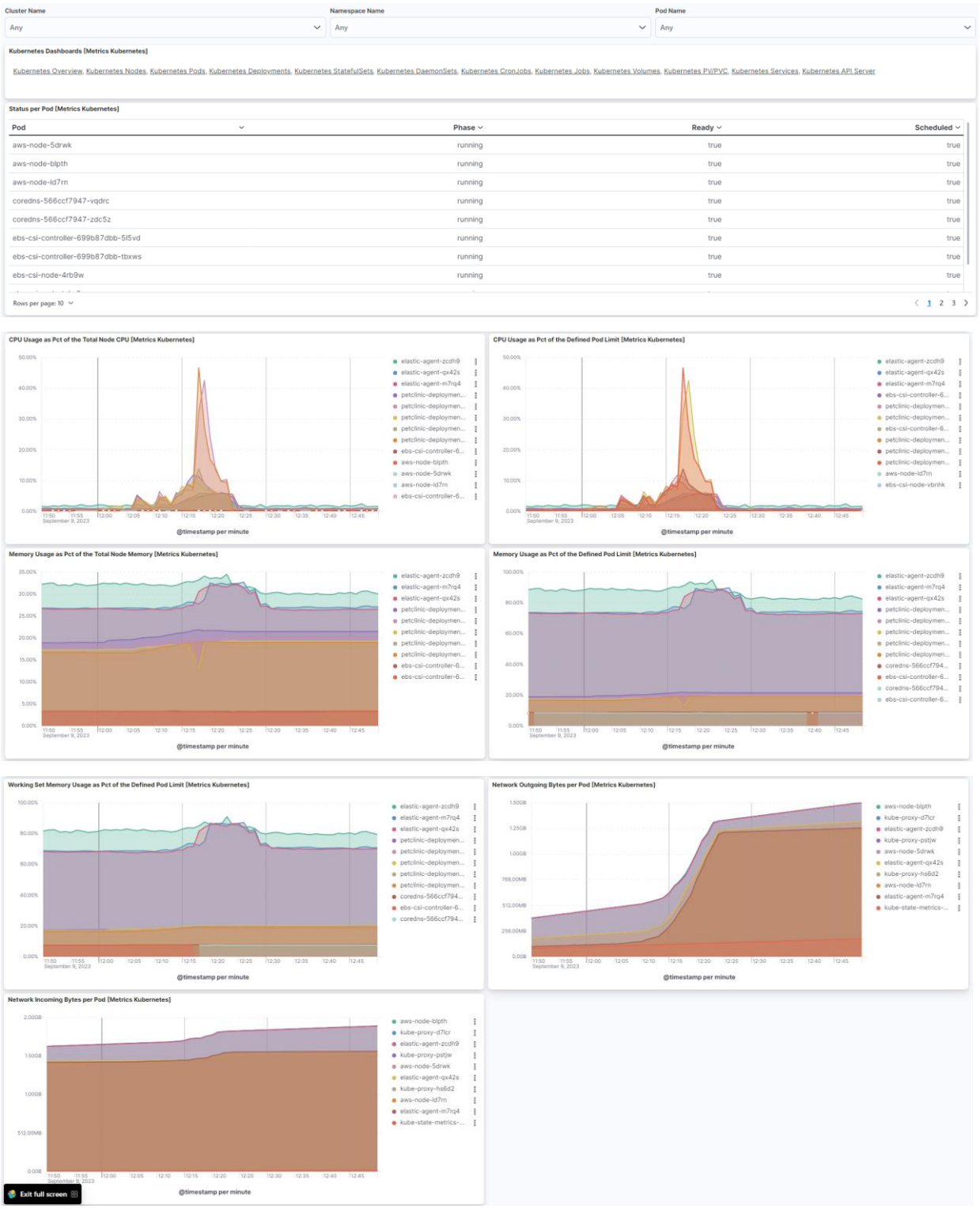


Anexo G. [Kubernetes] Overview

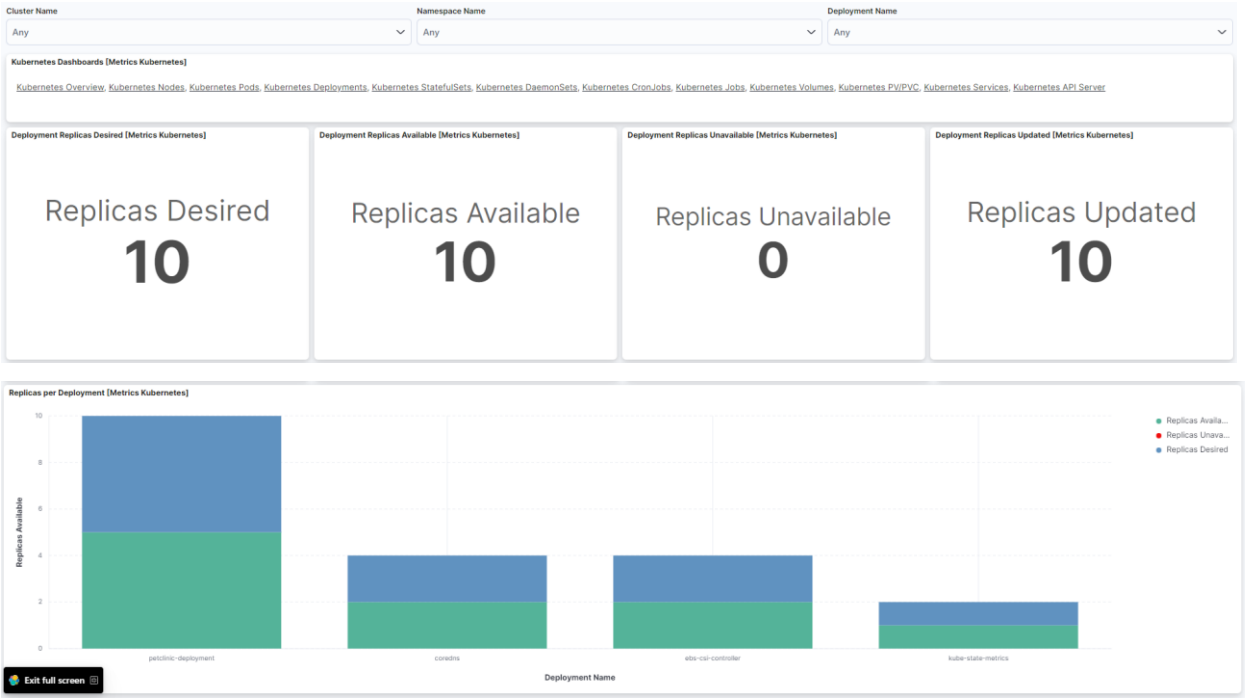


Anexo H. [Kubernetes] Nodes

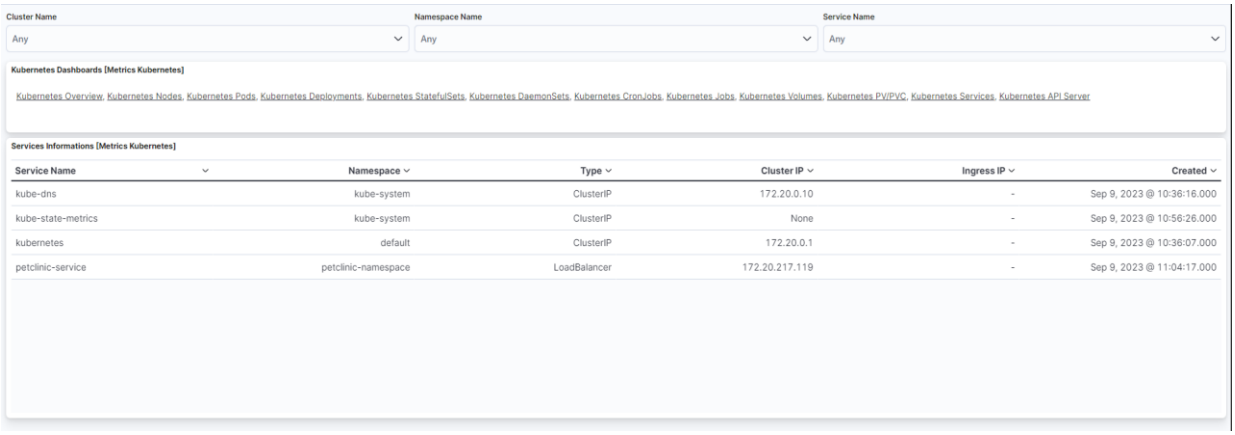
Anexo I. [Kubernetes] Pods



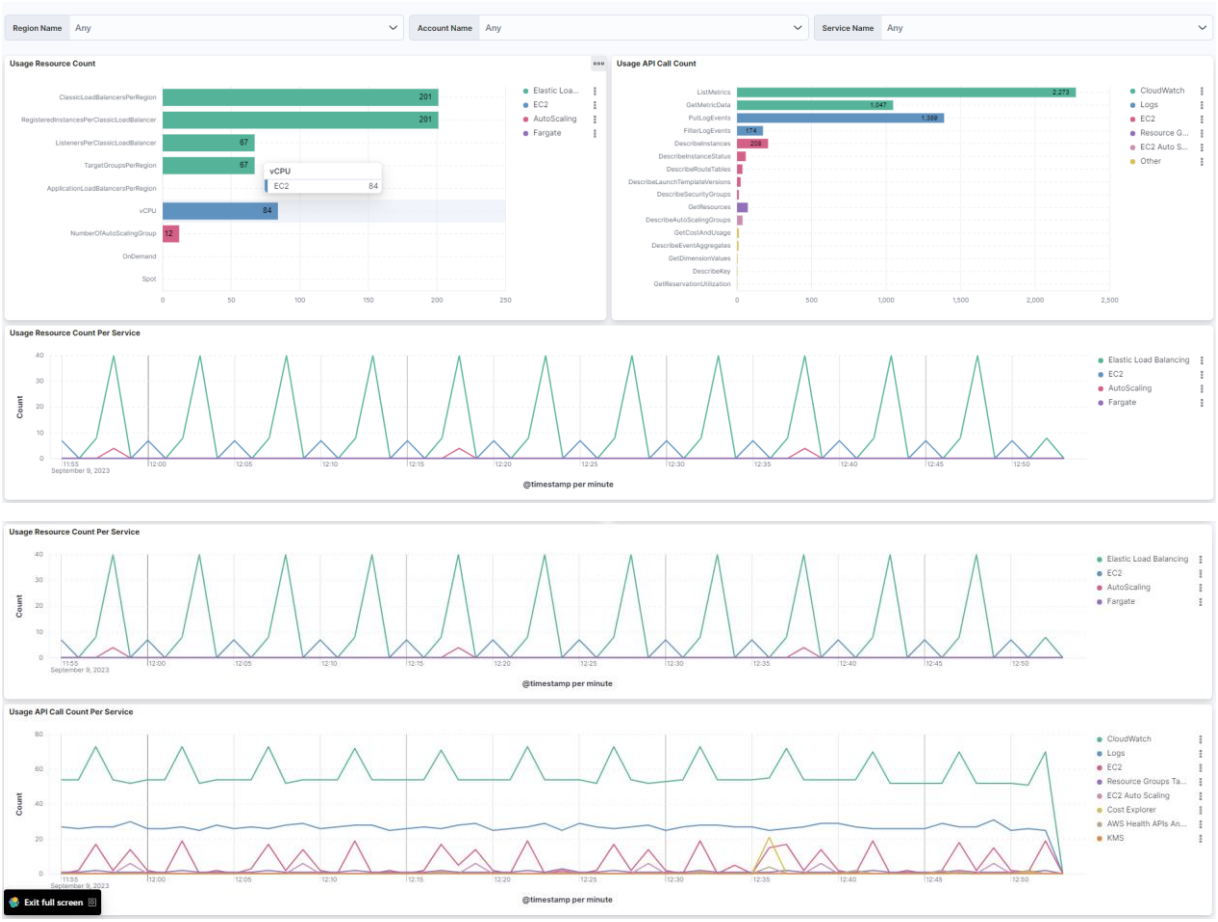
Anexo J. [Kubernetes] Deployments



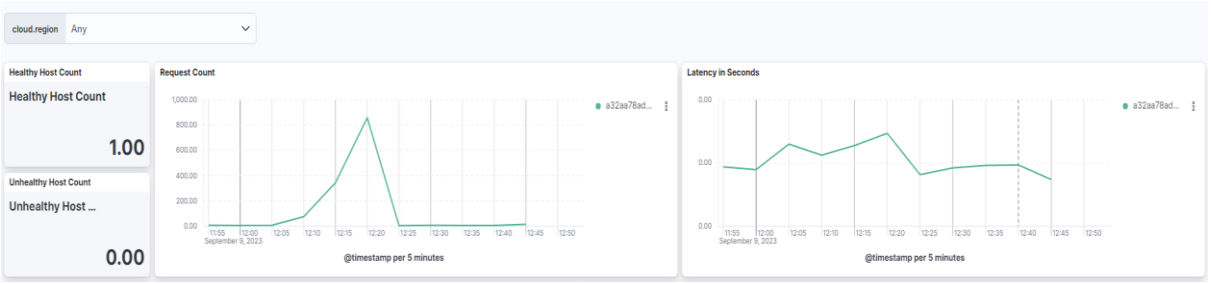
Anexo K. [Kubernetes] Services



Anexo L. [Metrics AWS] Usage Overview

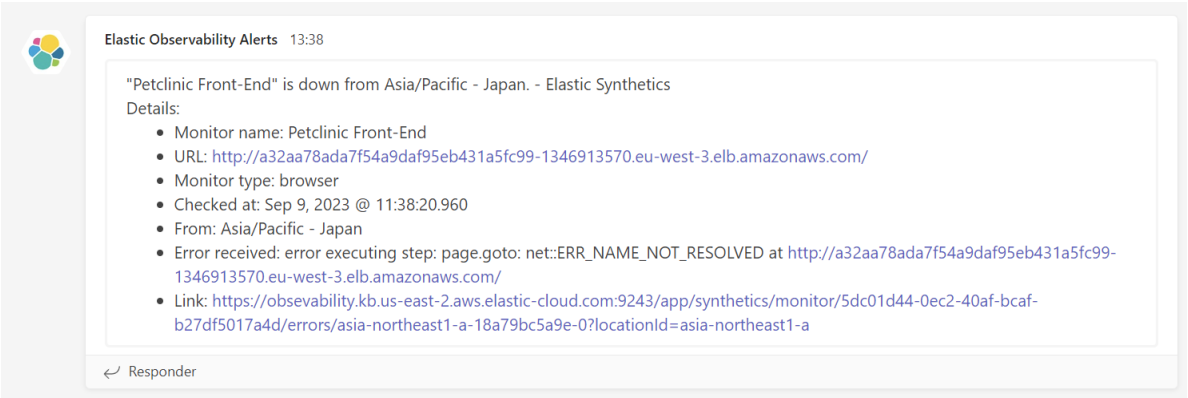


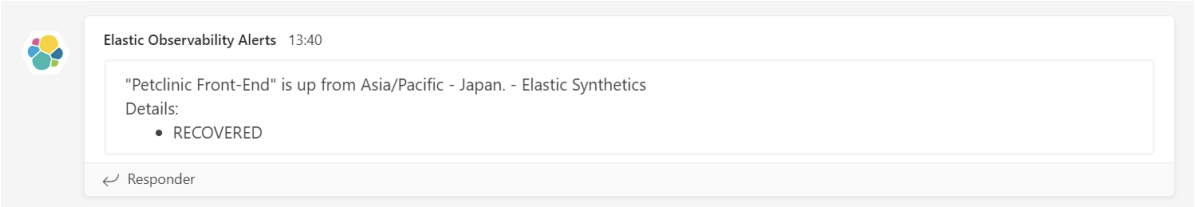
Anexo M. [Metrics AWS] ELB Overview

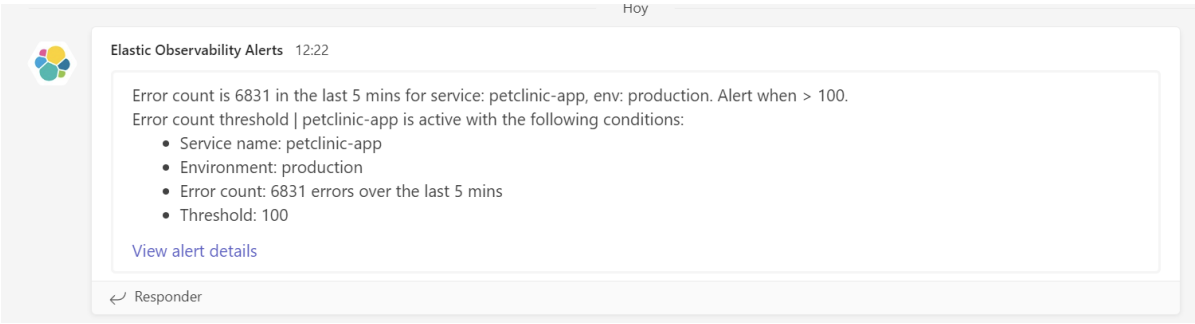


Anexo N. [Metrics AWS] EC2 Overview

Anexo O. Ejemplos alertas




Elastic Observability Alerts 13:40
"Petclinic Front-End" is up from Asia/Pacific - Japan. - Elastic Synthetics
Details:
RECOVERED
Responder


Hoy
Elastic Observability Alerts 12:22
Error count is 6831 in the last 5 mins for service: petclinic-app, env: production. Alert when > 100.
Error count threshold | petclinic-app is active with the following conditions:
Service name: petclinic-app
Environment: production
Error count: 6831 errors over the last 5 mins
Threshold: 100
View alert details
Responder