TRABAJO DE GRADO
Opción Seminario-Diplomado.

Implementación de arquitectura de EC2 en AWS para la empresa Expressify S.A.S

Corporación Universitaria Remington.
Nombre de la facultad: Facultad de ingeniería
Nombre del programa académico: Ingeniería de sistemas

Carlos Mario Pareja Castaño, Karen Dayana Benítez Vásquez y Luis Alfredo Nerio Monsalve.
Juan Pablo Berrio López
Seminario Amazon Web Services(AWS).
2025.

Tabla de Contenidos

Tabla de ilustraciones

Resumen

Se hizo la implementación de una arquitectura de EC2 en AWS para la empresa Expressify S.A.S. Hicimos un diseño para la arquitectura y luego lo implementamos en Amazon Web Services. Esta es escalable, puede manejar el tráfico sin perder eficiencia y es dedicada a alta demanda y disponibilidad.

Los servicios usados para el diseño y construcción de nuestros servicios realizados están dentro de la capa gratuita, esta pasara a generar cobro después de un año de uso de la capa gratuita, esto obligara a la empresa a realizar una inversión para poder seguir haciendo uso de estos servicios, la inversión es necesaria para que la empresa pueda seguir escalando y manteniendo un crecimiento estable en su arquitectura interna de servidores.

Esta implementación, nos deja claro que los servicios en la nube también pueden ser una opción para migrar la operación interna tecnológica en las empresas que están en constante crecimiento.

Palabras clave

Instancias, EC2, Servidores, Arquitectura, Inversión, Capa Gratuita, Construcción, Operación.

Marco conceptual y contextual

Nuestro trabajo se dio en base a nuestra opción de grado o seminario de Amazon Web Services(AWS), donde aprendimos sobre cómo construir e implementar arquitecturas en la nube. Nos queda resaltar que aunque el trabajo pedido fue algo tedioso, ya que nunca habíamos usado este tipo de servicios o herramientas, el curso fue muy Teórico-Practico y esto permitió que nuestro aprendizaje desbloqueara talentos que no creíamos tener la capacidad de poner en práctica. Dentro del curso vimos herramientas que se pueden usar para implementar en empresas que tienen como meta optimizar sus procesos tecnológicos como: servicios de computo para redes, almacenamiento, balance de cargas, auto escalado.

Entendimos que AWS es un prestador de servicios en la nube, en el cual sus usuarios pueden adquirir de manera rentada equipos físicos que están ubicados en datacenters alrededor del mundo y así no tener que preocuparse por la compra de equipos físicos.

En este trabajo usamos las siguientes herramientas o servicios:

1. VIRTUAL PRIVATE CLOUD(VPC) para configurar nuestra red privada
2. ELASTIC CLOUD COMPUTER(EC2) para crear nuestras instancias o servidores
3. ELASTIC LOAD BALANCING(ELB) para la distribución y balanceo de la carga de trafico entre instancias
4. AUTOESCALADO(grupos de autoscaling) para automatizar la cantidad de recursos de computo en función.
5. AMAZON MACHINE IMAGES(AMI) para crear copias a nuestras imágenes de OS ya configuradas.
6. INSTANTANEAS para crear copias de nuestros servidores.

Este trabajo se realizo para el caso de la empresa Expressify S.A.S, la cual solicitaba esta infraestructura en la nube para poder mejorar sus operaciones logística y gestión de pedidos de usuarios, tener mejoras en seguridad de la información y también permitir a la empresa dar apertura a soluciones como análisis de rutas. Esta empresa al estar ubicada en un entorno de posible alta demanda, solicito una arquitectura capaz de soportar cambios de escalabilidad.

Primera entrega

Implementación de una red en AWS con servidores Windows y Linux accesibles desde Internet .

1. Creación de la VPC

- Esta costa de 4 subredes y le pusimos como nombre parcial vpc

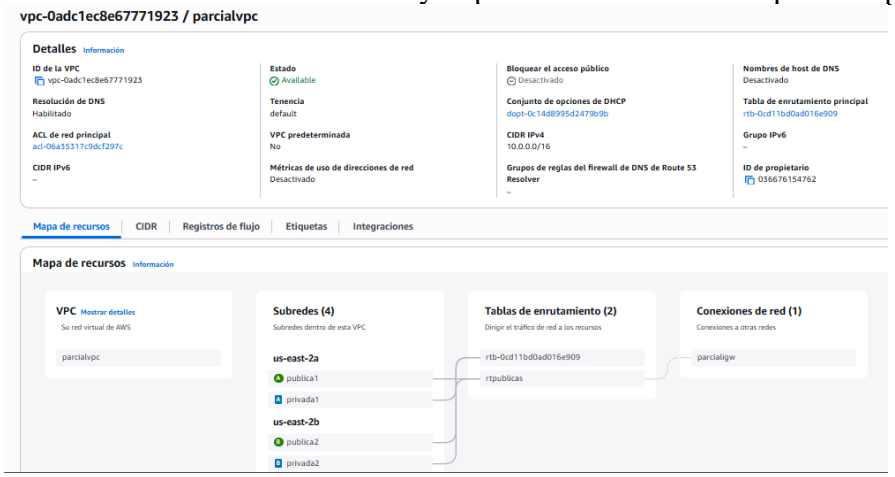


Figure 1.VPC

2. Tabla de enrutamiento para nuestra vpc

- Hicimos una asociación explicita para las subredes publicas y privadas.

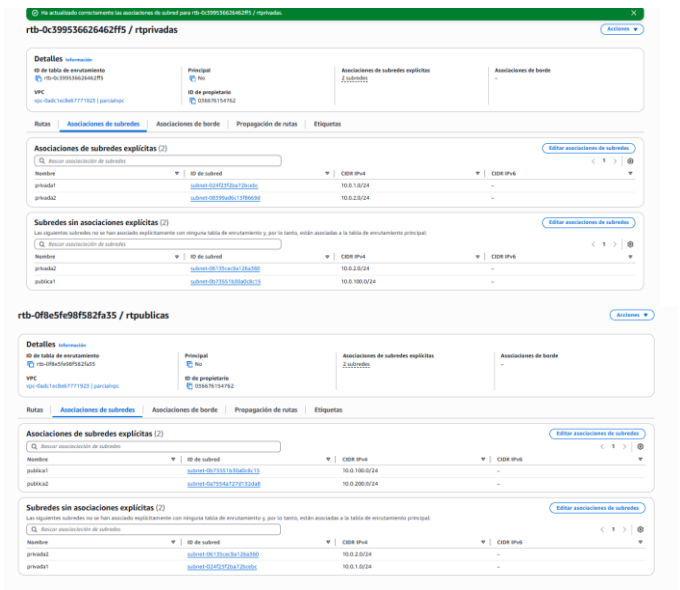


Figure 2Asociaciones explicitas

3. Puerta de enlace de nuestra VPC

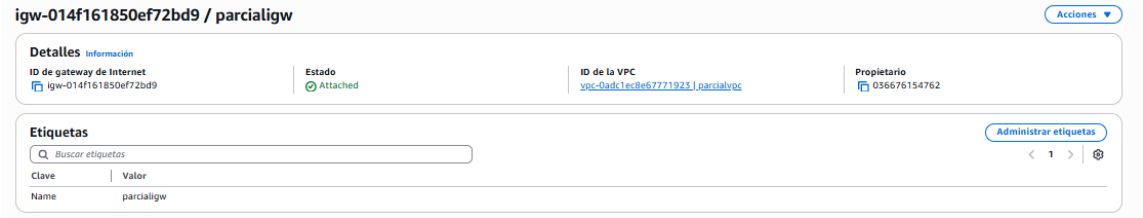


Figure 3.Puerta de enlace

4. Nat gateways

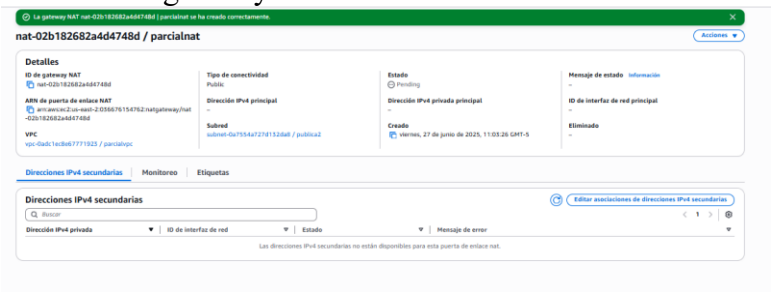


Figure 4.Natgateway

5. Reglas de entrada para el grupo de seguridad de la base de datos

- Declaramos las reglas de entrada de nuestra base de datos.

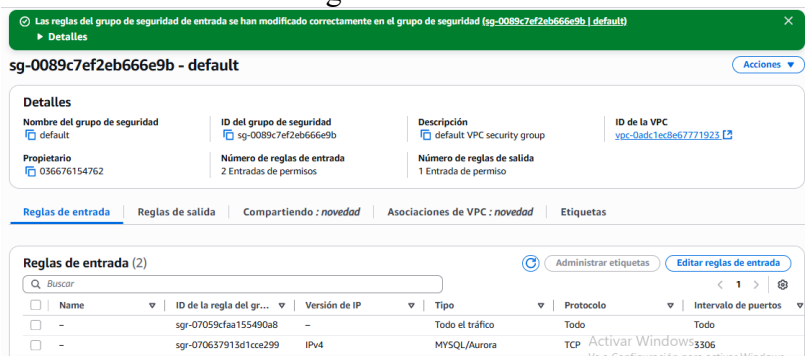


Figure 5.Reglas de entrada Bd

6. Administrador de ciclo de vida

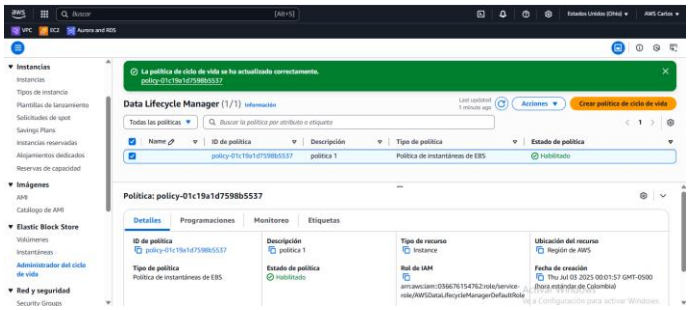


Figure 6.Administrador de ciclo de vida

7. Creación y activación del apache para Linux

- Aquí declaramos la instalación de apache para Linux para acceder desde internet

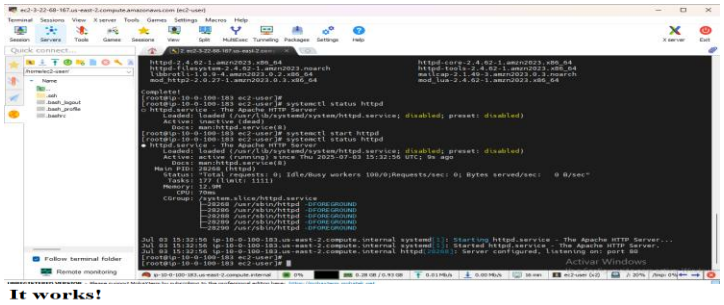
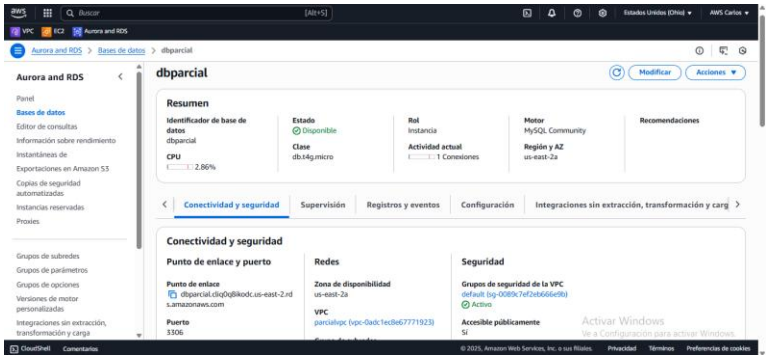


Figure 7.Apache para Linux

8. Base de datos y su conexión a Linux

- Creamos la base de datos y la conectamos a linux



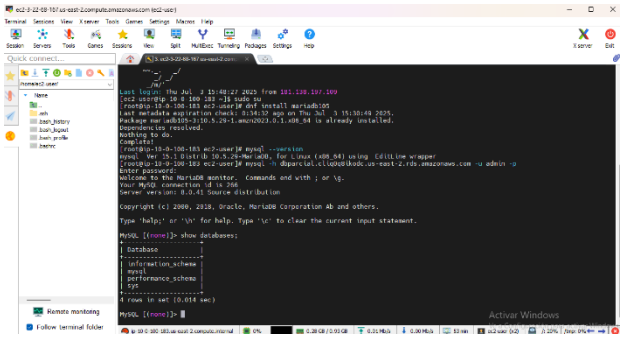


Figure 8.Base de datos y conexion

9. Carpetas o buckets

- Creamos los buckets necesarios para la entrega.

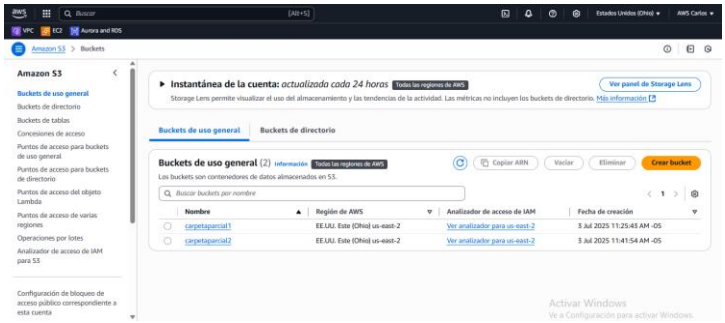


Figure 9.Carpetas o buckets

Nuestro Becket numero 1 contiene un archivo .html de prueba en correcto funcionamiento

9.1 Archivo del bucket

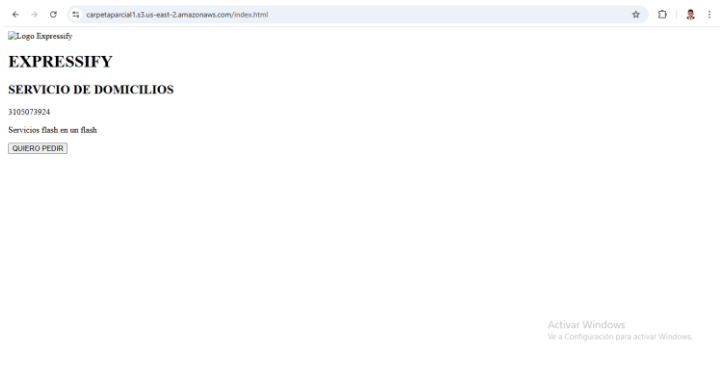


Figure 10.Archivo Bucket

10. Finalizacion de todas nuestras instancias

- Aquí creamos nuestras instancias.

Decidimos usar Windows Server 2016 y Linux 2023, ambas dentro de la capa gratuita con tipo de instancia t2.micro.

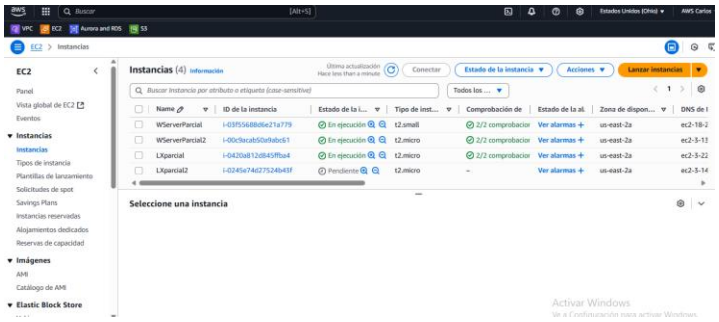


Figure 11.Instancias

11. Grupo de seguridad del equilibrador de carga

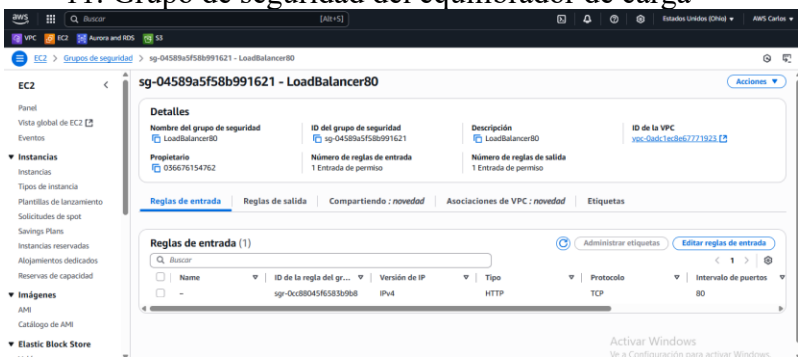


Figure 12.Grupo de seguridad del LoadBalancer

12. Grupo de destino para nuestro equilibrador de carga

- En este punto creamos el grupo de destino para nuestro LoadBalancer y agregamos dos instancias de prueba.

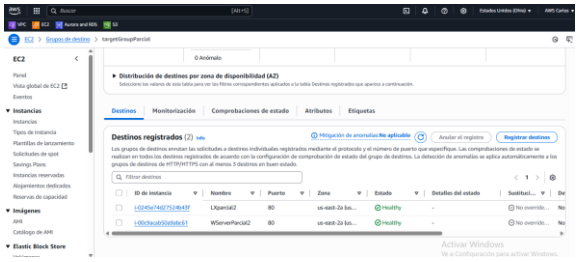


Figure 13.TargetGroup

13. Balanceador de carga

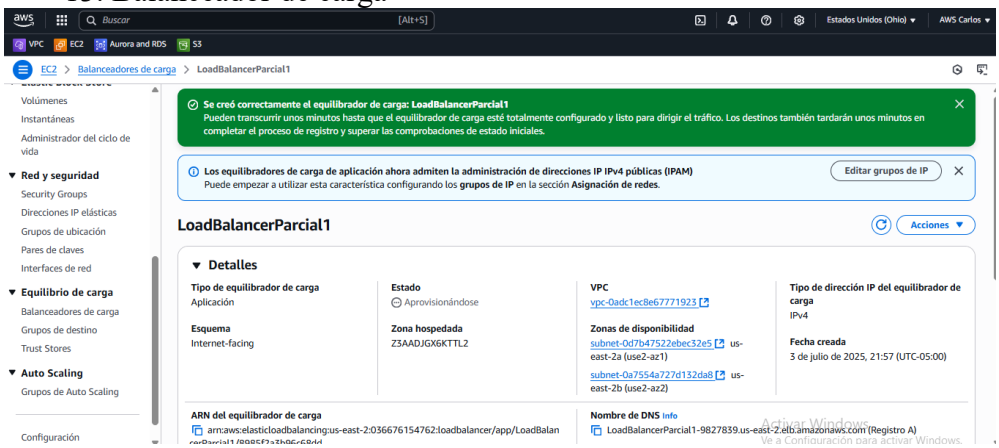


Figure 14.LoadBalancer

14. Plantilla de lanzamiento

- Aquí creamos la plantilla con la cual se lanzaran las instancias que demandara el auto escalado cuando detecte un daño en alguna instancia.

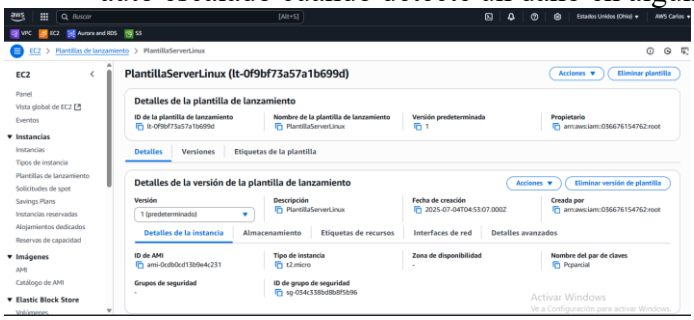


Figure 15.Plantilla de lanzamiento

15. Grupo de autoescalin

- Aquí, ya creamos el grupo de autoescalado que se encargara las nuevas instancias que se crearan.

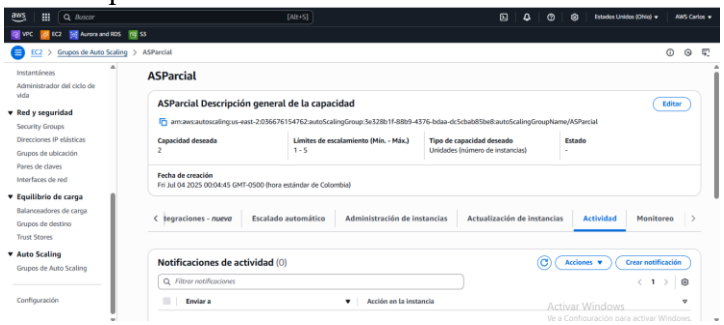


Figure 16.Grupo de autoescalado

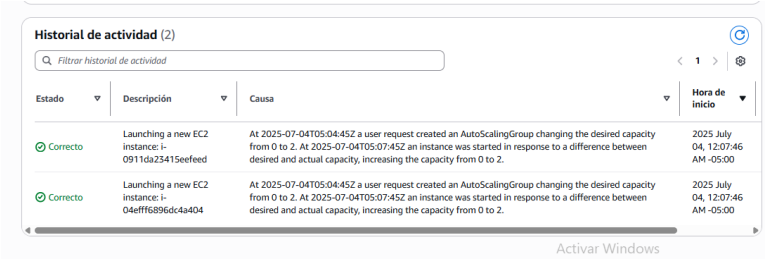


Figure 17.Actividad de autoescalado

16. Nueva instancia

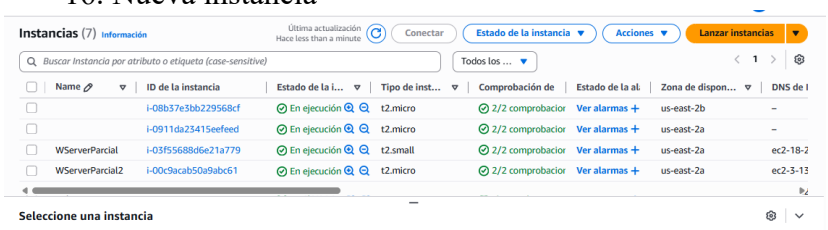


Figure 18.Instancias después de autoescalado

17. Instancias agregadas al grupo de seguridad funcionando correctamente desde la ip publica.

- WindowsServer2

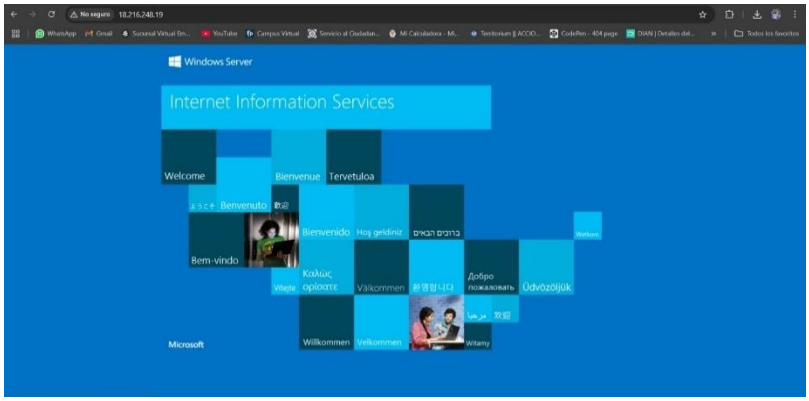


Figure 19.Windows IP

- LXparcial2

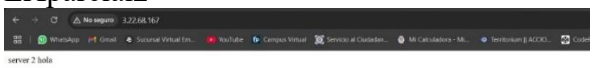


Figure 20.Linux IP

Segunda entrega

Implementación de Arquitectura en AWS con Balanceador de Carga y Contenedores.

Para este trabajo cada uno de los integrantes asumió el papel de arquitecto en la nube, esto para permitirnos lograr que la empresa pueda mejorar sus tiempos en peticiones de usuarios al realizar pedidos, operaciones logísticas, optimización de rutas y además mejoras en la seguridad de la información.

Expressify S.A.S es una empresa que ha generado gran impacto en la zona y se ha visto envuelta en un crecimiento estable, esto permite que la empresa pueda interactuar con nuevos usuarios.

Nosotros como arquitectos en la nube decidimos brindarle soluciones a Expressify mediante una infraestructura en la nube capaz de soportar estos cambios y hacer de Expressify una empresa escalable con capacidad de brindar sus servicios a muchas más personas.

1. Instancia principal

Creamos una instancia de Linux para empezar a darle vida a este trabajo

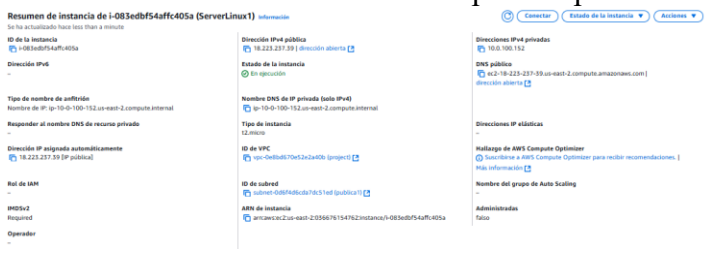


Figure 21.Instancia LINUX

2. Instantanea a partir de instancia de Linux

En este punto creamos una instantánea a nuestro almacenamiento de la instancia principal, esta nos servirá mas adelante.

Figure 22.Instantanea Linux

3. AMI a partir de la instantánea de la instancia principal

Creamos una AMI con nuestra primera instancia ya configurada para saltarnos ese paso.

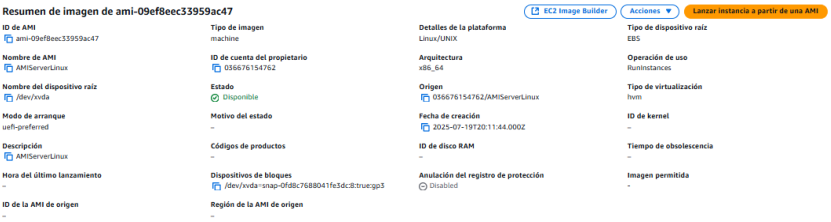


Figure 23.AMI Linux

Illustracion 21.AMI Linux

4. Creacion de segunda instancia a partir de la AMI

Esta es nuestra segunda instancia ya configurada y lanzada mediante nuestra AMI.

Figure 24.Instancia2

5. Balanceador de carga

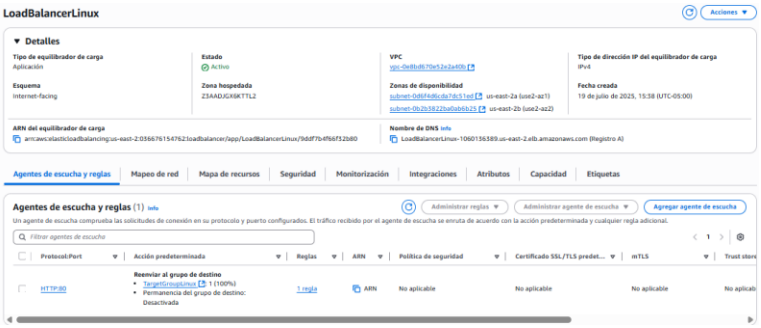


Figure 25.LoadBalancer

6. Grupo de destino para nuestro LoadBalancer

Aquí declaramos que instancias van a estar como destinos para balancear el trafico.

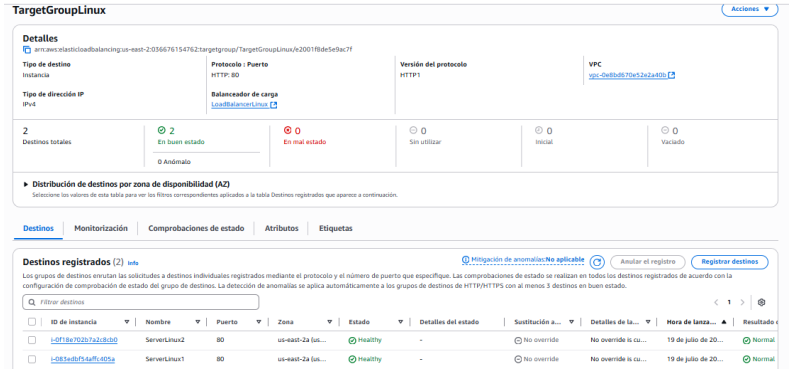


Figure 26.TargetGroup

7. Creacion de Docker con página de prueba

Creamos 3 docker y especificamos el archivo de prueba en el Docker3 para poder visualizarlo desde la IP publica por el puerto 83, para esto usamos una plantilla reutilizable de html como aplicación de prueba.

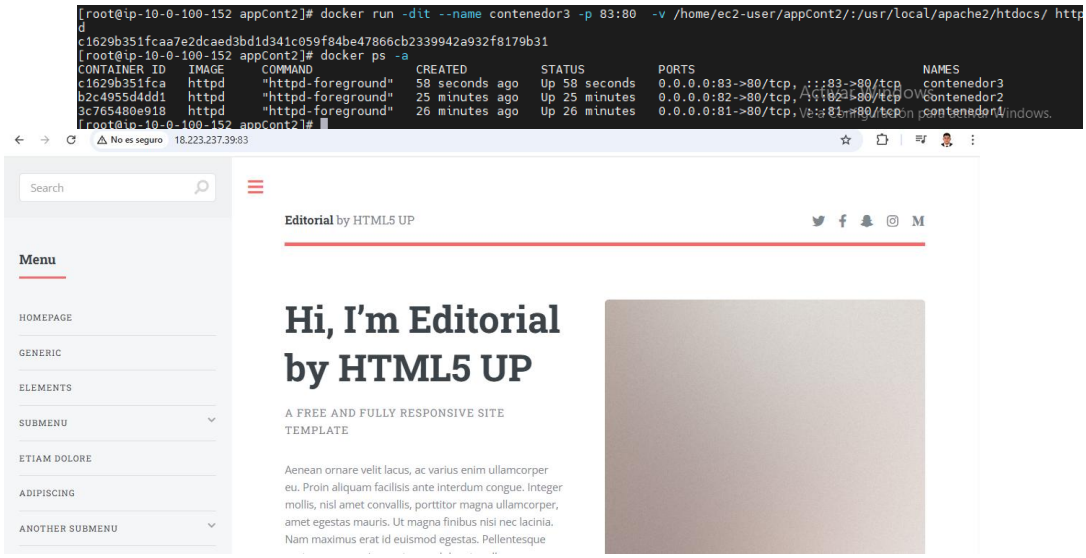


Figure 27.Docker pagina de prueba.

docker #3 “contenedor3” ese es el que contiene la pagina de prueba.

8. Instantanea de nuestra instancia configurada con dockers y Nginx

Creamos esta instantánea mediante nuestra instancia principal ya configurada con Dockers y Nginx, para así no tener que configurar esto siempre.

Figure 28.Instantanea con Docker y nginx

Illustracion 26.Instantanea con Docker y nginx

9. AMI a partir de la instantánea

   Esta AMI nos permitirá crear nuestra plantilla de lanzamiento nueva para el autoescalado con instancias ya configuradas.

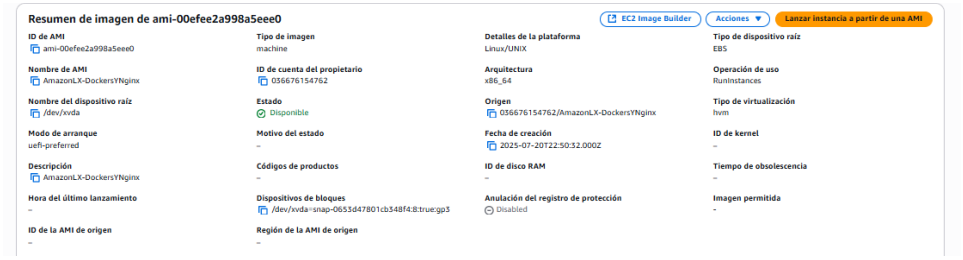


Figure 29. AMI configurada NginxDockers

10. Nuestro nuevo balanceador con instancias configuradas con Docker y nginx

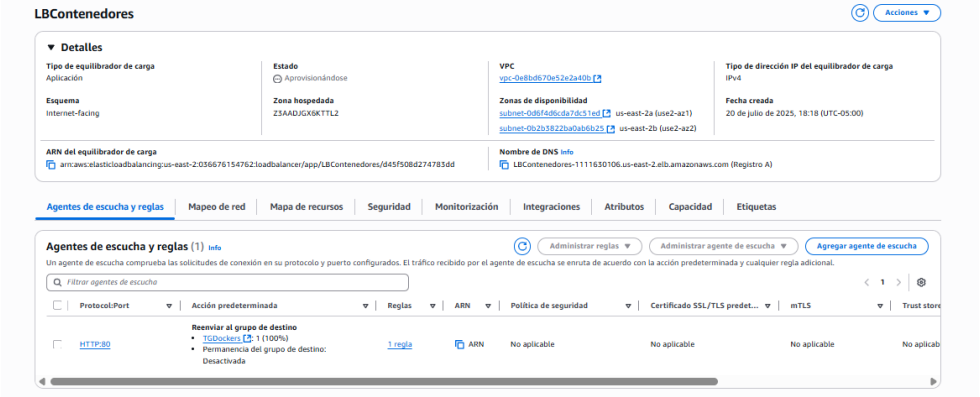


Figure 30. LoadBalancer DockersNginx

11. Nuevo grupo de destinos con instancias configuradas con Docker y nginx

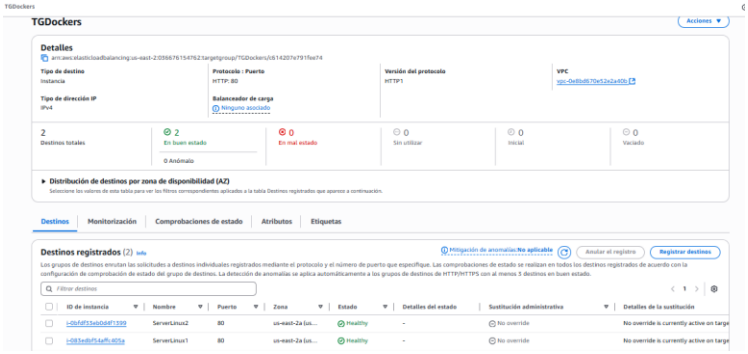


Figure 31.TargetGroup DockerNginx

12. Grupo de autoescalado

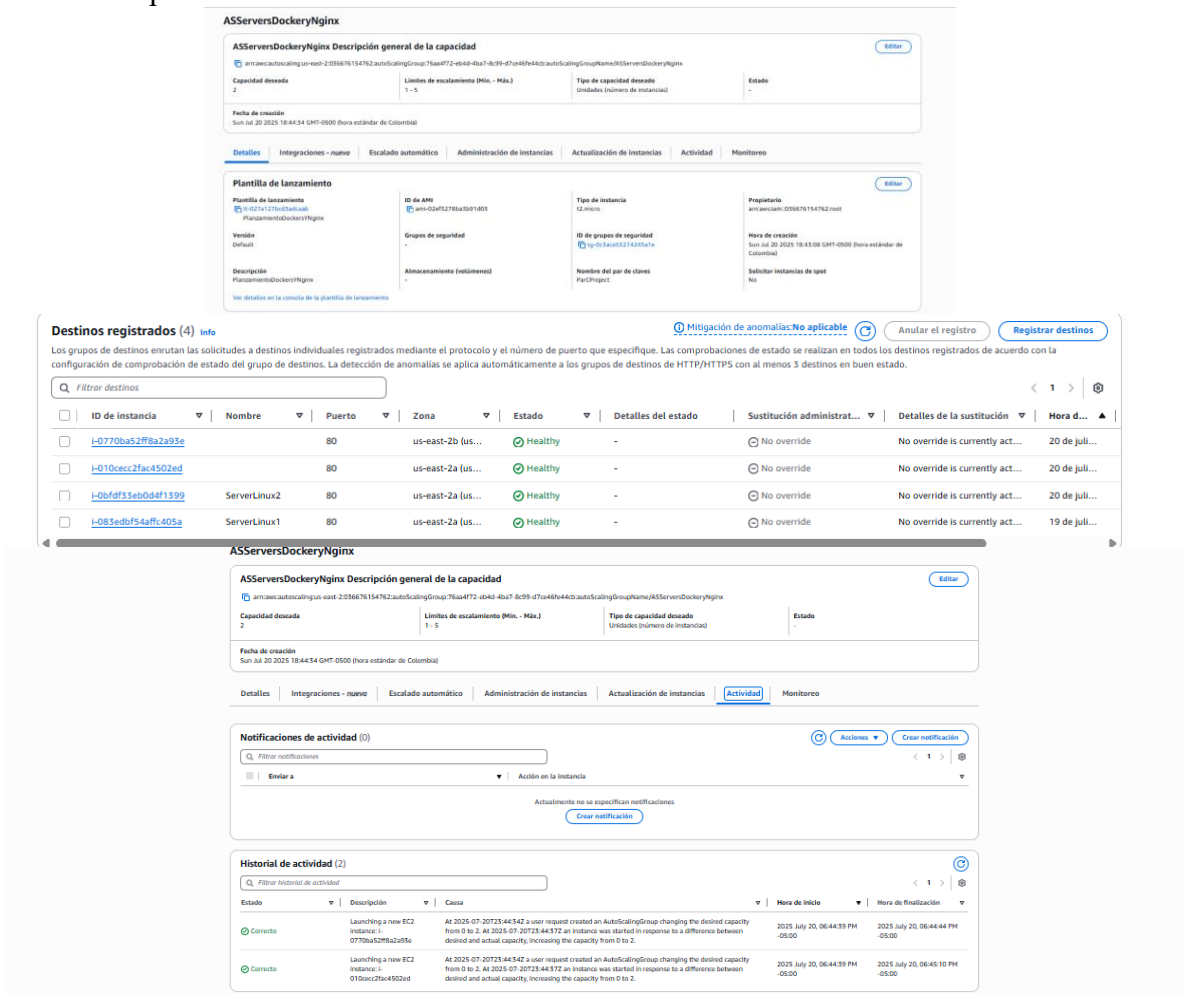


Figure 32Grupo de autoescalado

Conclusiones

Podemos concluir que AWS puede ser una opción tecnológica mas completa para aquellas empresas que como Expressify quieren escalar en el entorno productivo que es los servicios en la nube. En el transcurso de la elaboración del trabajo pusimos en practica las habilidades de cada uno de los integrantes, asumimos roles y encargamos tareas específicas, para así poder finalizar la solución mas optima para hacer de Expressify una empresa eficiente.

En AWS se pueden usar servicios de cualquier categoría para cualquier tipo de problema, tales como: Bases de datos, Redes privadas, Servidores, entre muchos otros servicios. Los servicios de amazon Web Services suelen ser de mucha importancia ya que cada servicio puede conectarse a otros y así poder tener un entorno de trabajo seguro, aparte de todo usamos una herramienta dentro del servicio ec2 llamada AMI, esta facilita crear imágenes de SERVIDORES mediante copias de seguridad para así poder lanzar instancias ya configuradas.

Con este trabajo queda claro que el futuro del entorno en la nube está en constante crecimiento y que gracias a lo aprendido y a lo que nos brinda AWS, podemos lograr dar soluciones a muchas problemáticas que nacen en el día a día y pudimos sobre todo brindar solución a Expressify S.A.S