



TRABAJO DE GRADO
Opción Seminario-Diplomado.

Amazon web services (AWS)

Corporación Universitaria Remington.
Facultad de ingeniería.
ingeniería de sistemas.

Camilo Andres Berrocal Ruiz.
Juan Pablo Berrio.
Opción de Trabajo de grado Seminario.
2025

Dedicatoria

Le dedico este trabajo a mis padres que siempre han estado ahí para mí, apoyándome en todo momento y sirviendo de guía, mi madre que nunca dejo de ayudarme sin importar que, y mi padre que siempre fue un apoyo incondicional, siempre estuvieron para mí, gracias por todo lo que han hecho por mí.

Tabla de Contenidos

Resumen.....	4
Marco conceptual y contextual	5
Marco conceptual.....	5
Marco contextual	6
Desarrollo e implementación del aprendizaje.....	8
¿Qué es la virtualización?	8
Inicios de la virtualización	8
¿Qué es la computación en la nube?	8
Inicio de la computación en la nube	8
Línea de tiempo.....	9
DIFERENCIAS ENTRE AWS, AZURE Y GCP.....	9
Evidencia de realización de prácticas del seminario.....	11
Conclusiones	23
Referencias.....	24

Resumen

En el siguiente trabajo se va a hablar de algunos servicios que trae Amazon web services (AWS), cómo funcionan esos servicios de AWS, cual es el funcionamiento de esos servicios, se mostrara algunas pruebas que se hicieron con algunos de los distintos servicios, por ejemplo, el servicio de EC2, se usaran recursos como el auto scaling, se podrá evidenciar en ocasiones el uso de esos recursos y como usarlos, se mostrara el uso de la herramienta de balanceador de carga. Se hará un breve recuento de que es la computación en la nube, sus orígenes, se mostrara una línea de tiempo donde se explicara algunos de los momentos clave para la computación en la nube, en que año fueron y cual fue ese hito, también se hablara brevemente sobre algunos otros servicios en la nube, como lo son Azure y GCP, donde se hará una comparación de esos servicios de computación en la nube.

Palabras clave

- Computación en la nube.
- Instancias.
- Vpc (virtual private cloud).
- Balanceador de carga.
- Auto scaling.
- Docker.
- Comandos de Linux.

Marco conceptual y contextual

Marco conceptual

Computación en la nube: la computación en la nube es el uso de recursos como almacenamiento, procesamiento o bases de datos a través del internet. (Zettle, s.f.)

AWS (Amazon web services): es un servicio de computación en la nube que permite el uso de distintas herramientas, para el desarrollo, prueba e implementación de aplicaciones, con una gran seguridad, brinda también servicios de virtualización. Redes virtuales, entre otros servicios. (Marujita, muy tecnológicos, 2023)

EC2: ec2 es un servicio dentro de AWS que ofrece la posibilidad de crear instancias, crear AMI, usar balanceador de carga. (tarlogic, s.f.)

Instancia: la instancia es una máquina virtual creada dentro del servicio de ec2 de AWS con los recursos que requiera la maquina como por ejemplo un sistema operativo ya sea Linux o Windows, se le puede colocar un almacenamiento interno, memoria RAM y otros recursos que podría tener una maquina física (Documentación aws, s.f.).

Vpc (virtual private cloud): es una red virtual, que sería similar a una red física, con los beneficios que trae AWS y que permite hacer pruebas de manera pública a través de una IP pública o hacer pruebas locales con una red privada. (documentación AWS, s.f.)

Ip publica: una IP pública es identificador numérico único que identifica a cada dispositivo en el internet, esa dirección se puede usar para conectar con diferentes maquinas remotamente. (Marujita, muy tecnológicos, 2023)

Balanceador de carga: un balanceador de carga es un recurso que permite distribuir el trafico entre distintas instancias o aplicaciones asociadas al balanceador, para no saturar una sola instancia, configurando previamente el balanceador y las instancias manualmente. (AWS Amazon, s.f.)

Auto scaling group: es un servicio que permite la creación de instancias automáticamente de acuerdo con el consumo que este teniendo una instancia, ya sea en almacenamiento o CPU, configurando que recurso va a estar vigilando; el auto scaling permite gestionar la cantidad de instancias que se podrán crear de manera automática basado en una AMI creada previamente. (Documentación aws, s.f.)

AMI (imagen de máquina de Amazon): es una imagen de una instancia creada a partir de una instancia que ya estaba previamente creada y configurada anteriormente. (Documentación aws, s.f.)

Target group: los target group o grupo de destino son los encargados de redirigir las solicitudes a las instancias mediante protocolos o puertos que se especifiquen. (Documentación aws, s.f.)

Puertos virtuales: los puertos virtuales en internet hacen referencia a distintos números puertos utilizados para hacer conexiones en la red o con servicios como lo es el puerto para la web http que es el puerto 80. (akamai, s.f.)

Docker: es una aplicación que permite empaquetar aplicaciones o software en un contenedor con todas las herramientas necesarias para que la aplicación pueda funcionar correctamente. (10code, 2024)

Nginx: es un servidor web que sirve para hacer proxy reverse y funciona de balanceador de carga. (B G. , 2025)

Proxy reverse: el proxy reverso o proxy inverso tiene la función de recibir las solicitudes de los clientes procesar esas solicitudes y luego enviarlas a los servidores adecuados. (B D. , 2025)

Marco contextual

El objetivo del trabajo es mostrar los temas que se abarcaron en el seminario de Amazon web services (AWS), mostrando varios servicios que fueron utilizados para hacer diferentes pruebas durante los distintos encuentros virtuales y mostrando evidencia visual, como, imágenes donde se pueda ver los resultados que se van dando con el uso de esos servicios, dependiendo los requerimientos solicitados para realizar las distintas pruebas, también se hablará de temas conceptuales explicando que es la computación en la nube, hablar sobre cuáles fueron los orígenes de la computación en la nube, se mostrara una línea de tiempo donde se podrá ver cuales fueron esos hitos que marcaron en distintas fechas la computación en la nube, y que hizo que la computación en la nube tenga tanta relevancia, se mostrara una breve comparación con otros servicios de computación en la nube que existen como lo son Azure y GCP.

Para este trabajo, se mostraran diversas imágenes mostrando el uso de instancias, que fueron creadas para hacer algunas pruebas, evidenciando además el uso del recursos que tiene AWS que es el balanceador de carga, se podrá observar la prueba de estrés que se le hizo a una instancia Linux para activar el balanceador de carga y el auto scaling que se configuro para que fuera creando de manera automática varias instancias en base a una AMI que se creó previamente de ese servidor Linux, si el consumo de CPU supera cierto umbral el auto scaling creara de manera automática otras instancias, hasta un máximo de 4 instancias.

Se mostrará también una instancia con sistema Linux en el cual se hicieron configuraciones para que tenga funcionando unos contenedores o Docker, que contengas unos archivos index que se mostraran en el navegador, se podrá ver los distintos contenedores y los puertos que fueron seleccionados para realizar las pruebas, se mostrara una configuración de proxy reverse que se hizo en el sistema Linux para redireccionar por los puertos a los

que se quiere visitar los sitios, todo esto mediante un repositorio de nginx, se configuraron los puertos en el target group o grupo objetivo para que se pudieran acceder a ellos en AWS.

Desarrollo e implementación del aprendizaje

¿Qué es la virtualización?

La virtualización es una tecnología que se puede usar para crear representaciones virtuales de servidores, almacenamiento, redes y otras máquinas físicas. El software virtual imita las funciones del hardware físico para ejecutar varias máquinas virtuales a la vez en una única máquina física. También potencia los servicios de computación en la nube que ayudan a las organizaciones a administrar la infraestructura de manera más eficaz. (AWS Amazon, s.f.)

Inicios de la virtualización

En la década de 1960, múltiples ingenieros informáticos se dieron la tarea de encontrar una manera que los ordenadores hicieran múltiples tareas simultáneamente, ya que en ese tiempo los ordenadores solo podían ejecutar un programa a la vez, sin embargo, con el tiempo los ingenieros encontraron la manera de crear un sistema que realizara varios programas a la vez; lo que se conoció como “tiempo compartido”.

En la década de 1970 la virtualización comenzó a aplicarse en mainframes, (computadoras de alto rendimiento), ya que los ingenieros desarrollaron sistemas que permitían que múltiples máquinas virtuales ejecutarse en un solo mainframe. En 1990 la virtualización empezó aplicarse a ordenadores personales, que de ahí mismo fueron surgiendo empresas tales como VMware y Virtual PC, que permitía a sus usuarios ejecutar múltiples sistemas operativos en un solo ordenador.

Desde entonces la virtualización ha ido evolucionando y se ha convertido en una herramienta importante en la industria de la informática. Hoy en día la virtualización se ha ido utilizando en una variedad de aplicaciones, desde la virtualización de servidores hasta la virtualización de escritorios. (redhat, 2024)

¿Qué es la computación en la nube?

Es un modelo que entrega servicios a través de TI, (tecnología de información), a través de internet, es decir los usuarios pueden acceder a cualquiera de estos recursos de manera remota desde cualquier dispositivo con conexión a internet, gracias a esta herramienta, los usuarios pueden disfrutar a cualquier hora y en cualquier lugar sus servicios, ya sea en Smartphone y Tablet, también es posible, almacenar archivos, trabajar y crear copias de seguridad sin ocupar espacio en el dispositivo. (wikipedia, s.f.)

Inicio de la computación en la nube

El inicio de la computación en la nube nos lleva al señor HERB GROSH, quién fue uno de los primeros en usar ese concepto y quien además dijo afirmar en la década de 1950 que una mejor manera para que las economías se adaptaran mejor si confiaban en el almacenamiento centralizado; que es donde entra el señor JHON McCarthy, quien introdujo el concepto de inteligencia artificial y también mencionó sobre el tiempo compartido, que entre la década de 1960 y 1970 fue implementado en muchas empresas de EE.UU, ya que este ofrecía transferencias de horas, muy parecido al que usamos en la actualidad AMAZON AWS o GLOOGLE CLOUD. (infranet working, 2017)

Línea de tiempo

Inicio de la virtualización:

1960: creación de un sistema que fuera capaz de realizar múltiples tareas simultáneamente.

1970: La virtualización empezó aplicarse en computadoras de alto rendimiento (mainframes).

1990: La virtualización empezó aplicarse en ordenadores personales y surgen nuevas empresas como: VMware y Virtual PC.

1995: Se lanza el primer servicio de almacenamiento en la nube llamado NetStorage.

1999: Se lanza (enlace unavailable), código de estado HTTP lo que significa que el servidor del sitio web simplemente no está disponible en este momento, una de las primeras aplicaciones de software como servicio (SaaS).

2002: Se lanza Amazon Web Services (AWS), una de las primeras plataformas de infraestructura como servicio (IaaS).

2006: Se lanza Google Apps, una suite de aplicaciones de productividad en la nube.

2008: Se lanza Microsoft Azure, una plataforma de computación en la nube.

2010: La madurez de la computación en la nube, así como también se lanza iCloud, un servicio de almacenamiento en la nube de Apple.

2015: Se lanza Amazon Lambda, un servicio de computación sin servidor.

2020: En este año se logran lanzar 2 cosas innovadoras de la computación en la nube híbrida y multicloud, se lanza Azure Arc, un servicio de computación híbrida de Microsoft y Google Anthos, un servicio de computación híbrida de Google. (time toast, s.f.)

DIFERENCIAS ENTRE AWS, AZURE Y GCP

AWS, AZURE y GCP, son los tres principales proveedores de servicios en la nube, sin embargo, cada uno cuenta con diferentes funcionamientos, aunque también tienen un poco de similitud como, por ejemplo: empezamos con AWS, es el gigante de la nube, debido a que cuenta con mayor variedad de servicios, tales como, computación, almacenamiento y machine learning. (subcampo de inteligencia artificial que usa algoritmos para aprender de los datos), en otras palabras, es un supermercado de servicios en la nube, algo muy servicial y práctico para sus usuarios, por otro lado, AZURE, está integrado con servicios de Microsoft como Windows y Office. Es como un proveedor de la nube especializado en Microsoft, lo cual es ideas para empresas que ya utilizaban estos productos.

GCP, es un proveedor de servicios en la nube que está enfocado en la inteligencia artificial y machine learning, lo que lo hace ideal para proyectos que requieren tecnología. Algunas similitudes que podemos encontrar, es que los tres proveedores cuentan con flexibilidad, almacenamiento en la nube, seguridad etc. Esto permite que sus usuarios puedan crear y configurar sus propias máquinas virtuales y redes, sin tener que preocuparse por la infraestructura y así mismo también se permite recuperar sus datos de manera segura. Todo esto con enfocarse en la seguridad y cumplimiento. (paradigma digital, 2019)

Evidencia de realización de prácticas del seminario.

The screenshot shows the AWS Management Console interface for EC2 instances. The main content area displays a table of instances:

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS
LINUXSERVER1	i-02f90cf11aaba24f2	Running	t2.micro	2/2 checks passed	View alarms +	us-east-1a	ec2-3-90-39-40.comput...
LINUXSERVER2	i-05c0d6925459f2041	Running	t2.micro	2/2 checks passed	View alarms +	us-east-1b	-
i-058f668ade92e9f5e	i-058f668ade92e9f5e	Running	t2.micro	2/2 checks passed	View alarms +	us-east-1a	-

The instance i-058f668ade92e9f5e is selected, and its details are shown below:

- Instance summary info**
 - Instance ID: i-058f668ade92e9f5e
 - IPV6 address: -
 - Hostname type: IP name: ip-10-0-139-79.ec2.internal
 - Answer private resource DNS name: -
- Public IPv4 address**: -
- Private IPv4 addresses**: 10.0.139.79
- Instance state**: Running
- Private IP DNS name (IPv4 only)**: ip-10-0-139-79.ec2.internal
- Instance type**: t2.micro
- Public IPv4 DNS**: -
- Elastic IP addresses**: -

Evidencia de las distintas instancias creadas, dos creadas manualmente con sistema Linux, las otras creadas a través de la AMI de la instancia LINUXSERVER1 con ayuda del auto escalado.

The screenshot shows the AWS Management Console interface for Amazon Machine Images (AMIs). The main content area displays a table of AMIs:

Name	AMI name	AMI ID	Source	Owner	Visibility	Status
serverconhttpd	serverconhttpd	ami-003288090e256010f	925204439332/serverconhttpd	925204439332	Private	Available

Below the table, there is a 'Select an AMI' section.

AMI que se creó de la instancia LINUXSERVER1 que había sido configurada con el servicio de apache para la prueba.

The screenshot displays the AWS Management Console interface for a Security Group. The breadcrumb navigation shows 'EC2 > Security Groups > sg-099fd1b8c981b05e1 - SERVERLINUX1'. The left-hand navigation pane is expanded to 'Network & Security' > 'Security Groups'. The main content area shows the details for the Security Group 'sg-099fd1b8c981b05e1 - SERVERLINUX1'. The 'Details' section includes:

- Security group name:** SERVERLINUX1
- Security group ID:** sg-099fd1b8c981b05e1
- Description:** launch-wizard-1 created 2025-03-22T03:01:08.424Z
- VPC ID:** vpc-0693db237798d7dfe
- Owner:** 925204439332
- Inbound rules count:** 2 Permission entries
- Outbound rules count:** 1 Permission entry

Below the details, there are tabs for 'Inbound rules', 'Outbound rules', 'Sharing - new', 'VPC associations - new', and 'Tags'. The 'Inbound rules (2)' tab is active, showing a table of rules:

Name	Security group rule ID	IP version	Type	Protocol	Port range	Source
-	sgr-0e397b02404428eb1	IPv4	HTTP	TCP	80	0.0.0.0/0
-	sgr-05f00b8dd226acca8	IPv4	SSH	TCP	22	0.0.0.0/0

El security group seleccionado para la AMI y el auto escalado fue el security group creado para la instancia de LINUXSERVER1.

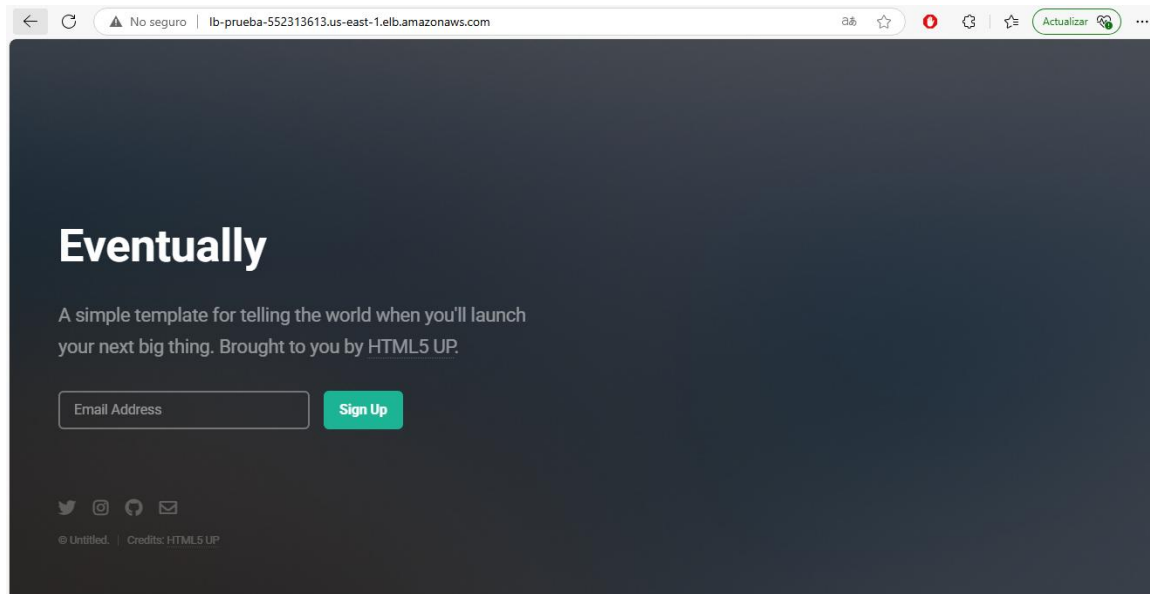
The screenshot displays the AWS Management Console interface for a Load Balancer. The breadcrumb navigation shows 'EC2 > Load balancers'. The left-hand navigation pane is expanded to 'Load Balancing' > 'Load Balancers'. The main content area shows the details for the Load Balancer 'lb-prueba'. The 'Load balancers (1/1)' section includes a table of load balancers:

Name	DNS name	State	VPC ID	Availability Zones	Type	Date created
lb-prueba	lb-prueba-552313613.us-e...	Active	vpc-0693db237798d7dfe	2 Availability Zones	application	March 23, 2025, 19:51 (UT...

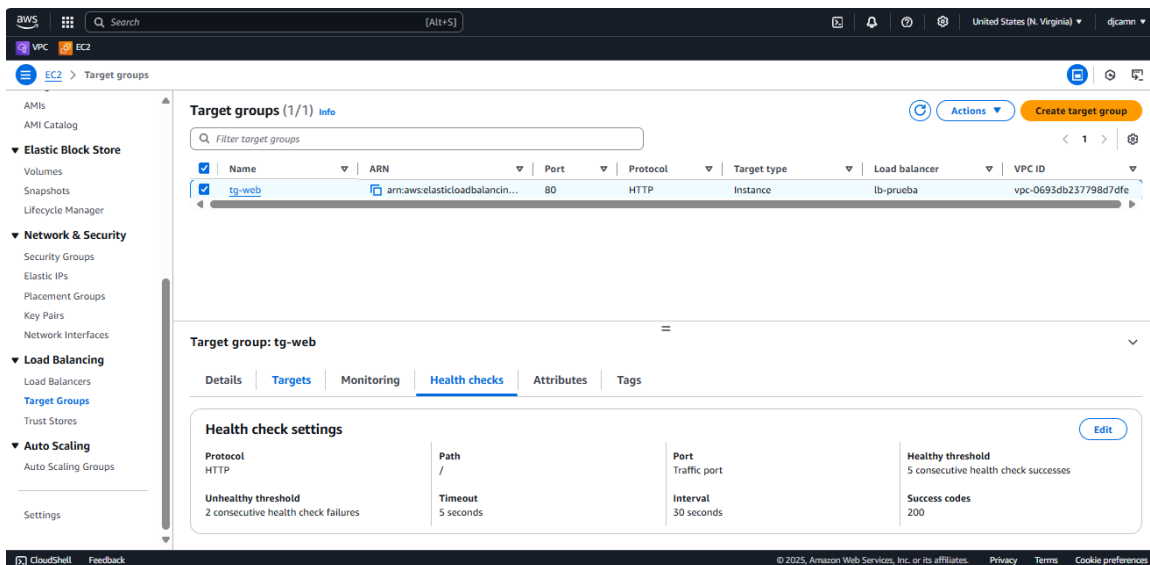
Below the table, the 'Load balancer: lb-prueba' details are shown, including:

- Load balancer type:** Application
- Status:** Active
- VPC:** vpc-0693db237798d7dfe
- Load balancer IP address type:** IPv4
- Scheme:** Internet-facing
- Hosted zone:** Z35SXDOTRQ7K7K
- Availability Zones:** subnet-056dcf51d7355ea0f (us-east-1a), subnet-0f1837fedcd44781f (us-east-1b)
- Date created:** March 23, 2025, 19:51 (UTC-05:00)

Balanceador de carga creado para la prueba y que se utilizó en el auto escalado, configurado con las vpc privadas y que solo pudiera entrar a través del DNS, ya que las instancias que se crearan con el auto escalado serían creadas con una VPC PRIVADA.



Evidencia de que load balance está enviando a las instancias que tienen configurado el servicio de apache.



Target group creado para la comunicación, se utilizó con el load balancer y se configuró la carpeta raíz, el puerto del tráfico, el código de confirmación a recibir, se habilito el puerto 80 http.

The screenshot shows the AWS Management Console for a Target Group named 'tg-web'. The 'Registered targets' section displays the following data:

Target ID	Health	Unhealthy	Unused	Initial	Draining
3	3 Healthy	0 Unhealthy	0 Unused	0 Initial	0 Draining

Below this, the 'Distribution of targets by Availability Zone (AZ)' section is visible, with a note: 'Select values in this table to see corresponding filters applied to the Registered targets table below.'

Evidencia de la comprobación que se muestra en el TG creado de las máquinas virtuales que se van creando, en el momento mostraba 3 ya que se había realizado una prueba de estrés de la instancia creada por el servicio de auto escalado.

The screenshot shows the AWS Management Console for a Target Group named 'tg-web'. The 'Registered targets' section displays the following data:

Target ID	Health	Unhealthy	Unused	Initial	Draining
1	1 Healthy	0 Unhealthy	0 Unused	0 Initial	0 Draining

Below this, the 'Distribution of targets by Availability Zone (AZ)' section is visible, with a note: 'Select values in this table to see corresponding filters applied to the Registered targets table below.'

Tg funcionando con la configuración creada con el menor de objetivos creados, se configuro con un mínimo de 1 y un máximo de 4, se cambió para dos y creo una segunda instancia, pero al ser poco el consumo de la CPU se regresó a 1 y la configuración coloco como mínimo 1.

AS-svrlinux1 Capacity overview

arn:aws:autoscaling:us-east-1:925204439332:autoScalingGroup:4e4a4da1-55d5-48a1-b6b1-05927dac6368:autoScalingGroupName/AS-svrlinux1

Desired capacity	Scaling limits (Min - Max)	Desired capacity type	Status
1	1 - 4	Units (number of instances)	-

Date created
Mon Mar 24 2025 10:45:46 GMT-0500 (hora estándar de Colombia)

Launch template

Launch template	AMI ID	Instance type	Owner
lt-0e0efdcxc1cdd52c plantilla-linux-svr1	ami-003288090e256010f	t2.micro	arn:aws:iam:925204439332:root
Version	Security groups	Security group IDs	Create time
Default	-	sg-099fd1b8c981b05e1	Mon Mar 24 2025 10:40:43 GMT-0500 (hora estándar de Colombia)
Description	Storage (volumes)	Key pair name	Request Spot Instances
-	-	CLAVELINUXSERVER1	No

Configuración de auto scaling, de instancias mínimas y máximas agregadas, se configuro para que, si el rendimiento de la CPU aumentaba a más del 40%, para que se creara un máximo de 4 instancias y cuando el rendimiento comience a bajar en la CPU del 40% se comiencen a eliminar las instancias.

Status	Description	Cause	Start time	End time
Successful	Terminating EC2 instance: i-058f668ade92e9f5e	At 2025-03-24T22:11:18Z a monitor alarm TargetTracking-AS-svrlinux1-AlarmLow-192cc2eb-8cf9-4192-acb8-ca0029eb413a in state ALARM triggered policy Target Tracking Policy changing the desired capacity from 2 to 1. At 2025-03-24T22:11:21Z an instance was taken out of service in response to a difference between desired and actual capacity, shrinking the capacity from 2 to 1. At 2025-03-24T22:11:21Z instance i-058f668ade92e9f5e was selected for termination.	2025 March 24, 05:11:21 PM -0500	2025 March 24, 05:16:43 PM -0500
Successful	Launching a new EC2 instance: i-05c0d6925459f2041	At 2025-03-24T22:06:00Z a user request update of AutoScalingGroup constraints to min: 1, max: 4, desired: 2 changing the desired capacity from 1 to 2. At 2025-03-24T22:06:09Z an instance was started in response to a difference between desired and actual capacity, increasing the capacity from 1 to 2.	2025 March 24, 05:06:12 PM -0500	2025 March 24, 05:06:19 PM -0500
Successful	Terminating EC2 instance: i-0b774cc75fcaee5f	At 2025-03-24T18:36:18Z a monitor alarm TargetTracking-AS-svrlinux1-AlarmLow-192cc2eb-8cf9-4192-acb8-ca0029eb413a in state ALARM triggered policy Target Tracking Policy changing the desired capacity from 2 to 1. At 2025-03-24T18:36:27Z an instance was taken out of service in response to a difference between desired and actual capacity, shrinking the capacity from 2 to 1. At 2025-03-24T18:36:27Z instance i-0b774cc75fcaee5f was selected for termination.	2025 March 24, 01:36:27 PM -0500	2025 March 24, 01:43:10 PM -0500
Successful	Terminating EC2 instance: i-0266d9b3e6dc3bd49	At 2025-03-24T18:31:18Z a monitor alarm TargetTracking-AS-svrlinux1-AlarmLow-192cc2eb-8cf9-4192-acb8-ca0029eb413a in state ALARM triggered policy Target Tracking Policy changing the desired capacity from 3 to 2. At 2025-03-24T18:31:26Z an instance was taken out of service in response to a difference between desired and actual capacity, shrinking the capacity from 3 to 2. At 2025-03-24T18:31:26Z instance i-0266d9b3e6dc3bd49 was selected for termination.	2025 March 24, 01:31:26 PM -0500	2025 March 24, 01:37:50 PM -0500
Successful	Terminating EC2 instance: i-0b790afd56d894f48	At 2025-03-24T18:16:08Z a monitor alarm TargetTracking-AS-svrlinux1-AlarmLow-192cc2eb-8cf9-4192-acb8-ca0029eb413a in state ALARM triggered policy Target Tracking Policy changing the desired capacity from 4 to 3. At 2025-03-24T18:16:14Z an instance was taken out of service in response to a difference between desired and actual capacity, shrinking the capacity from 4 to 3. At 2025-03-24T18:16:14Z instance i-0b790afd56d894f48 was selected for termination.	2025 March 24, 01:16:14 PM -0500	2025 March 24, 01:21:56 PM -0500
		At 2025-03-24T17:55:50Z a monitor alarm TargetTracking-AS-svrlinux1-AlarmHigh-922d47bb-e663-	2025 March	2025 March

Evidencia de la creación de instancias y eliminación de las instancias de manera automatizada por el auto escaldo.

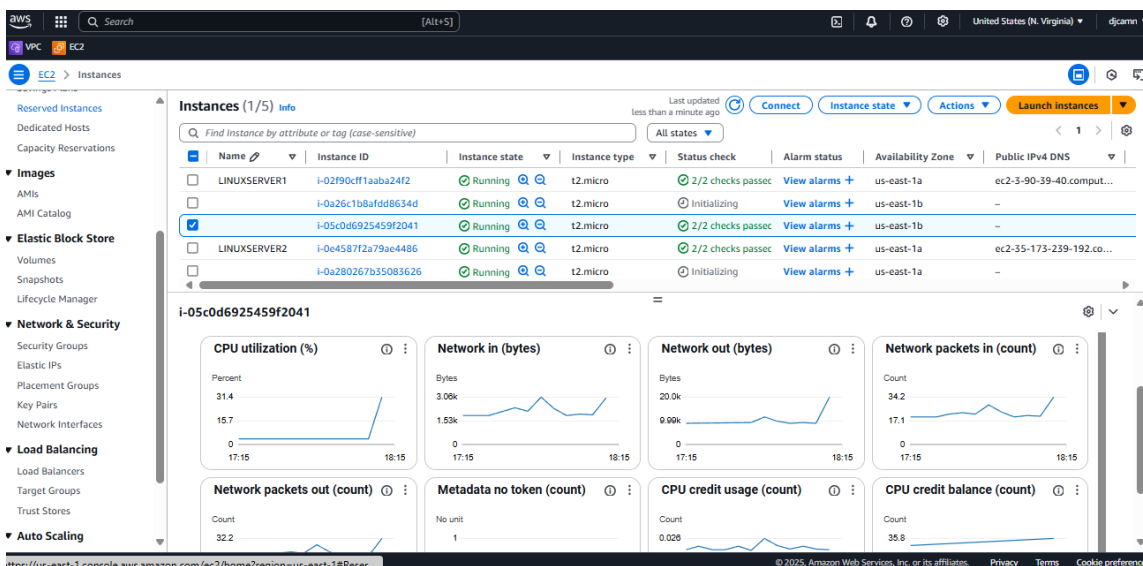
```

[ec2-user@ip-10-0-144-137 ~]$ ^C
[ec2-user@ip-10-0-144-137 ~]$ while ;; do ;; done &
[1] 5903
[ec2-user@ip-10-0-144-137 ~]$ for i in {1..4}; do while ;; do ;; done & done
[2] 5963
[3] 5964
[4] 5965
[5] 5966
[ec2-user@ip-10-0-144-137 ~]$ top
top - 23:15:24 up 1:09, 1 user, load average: 4.61, 2.01, 0.77
Tasks: 108 total, 6 running, 102 sleeping, 0 stopped, 0 zombie
%Cpu(s): 99.0 us, 0.0 sy, 0.0 ni, 0.0 id, 0.0 wa, 0.0 hi, 0.0 si, 1.0 st
MiB Mem : 949.5 total, 602.6 free, 134.4 used, 212.4 buff/cache
MiB Swap: 0.0 total, 0.0 free, 0.0 used, 674.8 avail Mem

  PID USER      PR  NI   VIRT   RES   SHR  S  %CPU  %MEM     TIME+ COMMAND
 5903 ec2-user  20   0 224032   1428    0 R   20.3   0.1   0:47.41 bash
 5964 ec2-user  20   0 224032   1428    0 R   19.9   0.1   0:29.56 bash
 5965 ec2-user  20   0 224032   1428    0 R   19.9   0.1   0:29.57 bash
 5966 ec2-user  20   0 224032   1428    0 R   19.9   0.1   0:29.57 bash
 5963 ec2-user  20   0 224032   1428    0 R   19.6   0.1   0:29.56 bash
 2040 apache    20   0 1084988 8472 5392 S   0.3   0.9   0:00.78 httpd
    1 root      20   0 105732 17012 10596 S   0.0   1.7   0:01.03 systemd
    2 root      20   0 0         0         0 S   0.0   0.0   0:00.00 kthreadd

```

Evidencia de la prueba de estrés realizada al servidor creado por el auto escalado para aumentar la CPU y se crearan otras instancias automáticamente.



Evidencia de las instancias que se fueron creando, automáticamente por el auto escalamiento.

aws [Search] [Alt+S] United States (N. Virginia) djcamn

EC2 > Target groups > tg-web

tg-web

arn:aws:elasticloadbalancing:us-east-1:925204439332:targetgroup/tg-web/6c5841e4b15d9597

Target type Instance **Protocol : Port** HTTP: 80 **Protocol version** HTTP1 **VPC** vpc-0693db237798d7dfe

IP address type IPv4 **Load balancer** lb-prueba

3 Total targets 3 Healthy 0 Unhealthy 0 Unused 0 Initial 0 Draining

0 Anomalous

Distribution of targets by Availability Zone (AZ)
Select values in this table to see corresponding filters applied to the Registered targets table below.

Targets | Monitoring | Health checks | Attributes | Tags

Registered targets (3) info Anomaly mitigation: Not applicable Deregister Register targets

Target groups route requests to individual registered targets using the protocol and port number specified. Health checks are performed on all registered targets according to the target group's health check settings. Anomaly detection is automatically applied to HTTP/HTTPS target groups with at least 3 healthy targets.

Filter targets

https://us-east-1.console.aws.amazon.com/console/home?region=us-east-1 © 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

aws [Search] [Alt+S] United States (N. Virginia) djcamn

EC2 > Auto Scaling groups > AS-svrlinux1

Status	Description	Cause	Start time	End time
Successful	Terminating EC2 instance: i-0a26c1b8afd8634d	At 2025-03-24T23:56:04Z a monitor alarm TargetTracking-AS-svrlinux1-AlarmLow-192ce2eb-8cf9-4192-acb8-ca0029eb413a in state ALARM triggered policy Target Tracking Policy changing the desired capacity from 2 to 1. At 2025-03-24T23:56:15Z an instance was taken out of service in response to a difference between desired and actual capacity, shrinking the capacity from 2 to 1. At 2025-03-24T23:56:15Z instance i-0a26c1b8afd8634d was selected for termination.	2025 March 24, 06:56:15 PM -05:00	2025 March 24, 07:02:58 PM -05:00
Successful	Terminating EC2 instance: i-0a280267b35083626	At 2025-03-24T23:53:04Z a monitor alarm TargetTracking-AS-svrlinux1-AlarmLow-192ce2eb-8cf9-4192-acb8-ca0029eb413a in state ALARM triggered policy Target Tracking Policy changing the desired capacity from 3 to 2. At 2025-03-24T23:53:06Z an instance was taken out of service in response to a difference between desired and actual capacity, shrinking the capacity from 3 to 2. At 2025-03-24T23:53:06Z instance i-0a280267b35083626 was selected for termination.	2025 March 24, 06:53:06 PM -05:00	2025 March 24, 06:59:29 PM -05:00
Successful	Terminating EC2 instance: i-05c0d6925459f2041	At 2025-03-24T23:51:27Z a monitor alarm TargetTracking-AS-svrlinux1-AlarmLow-192ce2eb-8cf9-4192-acb8-ca0029eb413a in state ALARM triggered policy Target Tracking Policy changing the desired capacity from 4 to 3. At 2025-03-24T23:51:32Z an instance was taken out of service in response to a difference between desired and actual capacity, shrinking the capacity from 4 to 3. At 2025-03-24T23:51:32Z instance i-05c0d6925459f2041 was selected for termination.	2025 March 24, 06:51:32 PM -05:00	2025 March 24, 06:57:14 PM -05:00
Successful	Launching a new EC2 instance: i-0ef3f743f26859c5	At 2025-03-24T23:25:50Z a monitor alarm TargetTracking-AS-svrlinux1-AlarmHigh-922d47bb-e663-46d9-a995-c5bea47060ac in state ALARM triggered policy Target Tracking Policy changing the desired capacity from 3 to 4. At 2025-03-24T23:26:01Z an instance was started in response to a difference between desired and actual capacity, increasing the capacity from 3 to 4.	2025 March 24, 06:26:03 PM -05:00	2025 March 24, 06:31:09 PM -05:00
Successful	Launching a new EC2 instance: i-0a26c1b8afd8634d	At 2025-03-24T23:19:50Z a monitor alarm TargetTracking-AS-svrlinux1-AlarmHigh-922d47bb-e663-46d9-a995-c5bea47060ac in state ALARM triggered policy Target Tracking Policy changing the desired capacity from 1 to 3. At 2025-03-24T23:20:00Z an instance was started in response to a difference between desired and actual capacity, increasing the capacity from 1 to 3.	2025 March 24, 06:20:02 PM -05:00	2025 March 24, 06:25:08 PM -05:00
Successful	Launching a new EC2 instance: i-0a26c1b8afd8634d	At 2025-03-24T23:19:50Z a monitor alarm TargetTracking-AS-svrlinux1-AlarmHigh-922d47bb-e663-46d9-a995-c5bea47060ac in state ALARM triggered policy Target Tracking Policy changing the desired capacity from 1 to 3. At 2025-03-24T23:20:00Z an instance was started in response to a difference between desired and actual capacity, increasing the capacity from 1 to 3.	2025 March 24, 06:20:02 PM -05:00	2025 March 24, 06:25:08 PM -05:00

https://us-east-1.console.aws.amazon.com/console/home?region=us-east-1 © 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

The screenshot shows the AWS Management Console interface. On the left, there's a navigation menu with 'EC2' selected. The main area displays 'Instancias (1/2) Información'. A table lists two instances: 'LINUXSERVER1' (ID: i-02f90cff1aaba24f2, State: Detenida) and 'LINUXSERVER2' (ID: i-0e4587f2a79ae4486, State: En ejecución). Below the table, the details for 'i-0e4587f2a79ae4486 (LINUXSERVER2)' are shown, including its ID, public IP address (107.22.135.255), private IP address (10.0.10.182), and state (En ejecución).

Para la realización de la segunda prueba se utilizó la instancia con el nombre Linuxserver2, una instancia que anteriormente se había creado, pero solo se le había probado con el servicio de httpd, se instaló el servicio de Docker y nginx para poder realizar la prueba.

The screenshot shows a terminal window with the following content:

```

[ec2-user@ip-10-0-10-182 ~]$ docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED    STATUS    PORTS
d54c557b53fa  httpd    "httpd-foreground"      19 hours ago Up 19 hours 0.0.0
.0:8082->80/tcp, :::8082->80/tcp  app2
6089adb6e776  httpd    "httpd-foreground"      19 hours ago Up 19 hours 0.0.0
.0:8081->80/tcp, :::8081->80/tcp  app1
690be293e4cc  httpd    "httpd-foreground"      20 hours ago Up 19 hours 0.0.0
.0:8080->80/tcp, :::8080->80/tcp  apachev1
[ec2-user@ip-10-0-10-182 ~]$ docker stop app2
[ec2-user@ip-10-0-10-182 ~]$ docker stop app1
[ec2-user@ip-10-0-10-182 ~]$ docker stop apachev1
[ec2-user@ip-10-0-10-182 ~]$ docker run --name app2 -p 8082:80 httpd -t
nginx: [emerg] unexpected end of file, expecting "]" in /etc/nginx/nginx.conf:27
nginx: configuration file /etc/nginx/nginx.conf test failed
[ec2-user@ip-10-0-10-182 ~]$ nano /etc/nginx/nginx.conf
[ec2-user@ip-10-0-10-182 ~]$ systemctl restart nginx
[ec2-user@ip-10-0-10-182 ~]$ docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED    STATUS    PORTS
d54c557b53fa  httpd    "httpd-foreground"      20 hours ago Up 20 hours 0.0.0.0:8082->
80/tcp, :::8082->80/tcp  app2
6089adb6e776  httpd    "httpd-foreground"      20 hours ago Up 20 hours 0.0.0.0:8081->
80/tcp, :::8081->80/tcp  app1
690be293e4cc  httpd    "httpd-foreground"      20 hours ago Up 19 hours 0.0.0.0:8080->
80/tcp, :::8080->80/tcp  apachev1
[ec2-user@ip-10-0-10-182 ~]$

```

Se crearon 3 contenedores para esta actividad, con diferentes puertos, el contenedor 1, funciona en el puerto 8080, el contenedor 2 está funcionando en el puerto 8081 y el contenedor 3 funcionando en el puerto 8082.

```

ec2-107-22-135-255.compute-1.amazonaws.com (ec2-user)
Terminal Sessions View X server Tools Games Settings Macros Help
Quick connect...
/home/ec2-user/
Name
..
.ssh
app1
app2
.bash_history
.bash_logout
.bash_profile
.bashrc
Remote monitoring
Follow terminal folder

.0:8081->80/tcp, ::8081->80/tcp app1
690be293e4cc httpd "httpd-foreground" 20 hours ago Up 19 hours 0.0.0
.0:8080->80/tcp, ::8080->80/tcp apachev1
[root@ip-10-0-10-182 nginx]# nginx -t
nginx: [emerg] unexpected end of file, expecting ";" in /etc/nginx/nginx.conf:27
nginx: configuration file /etc/nginx/nginx.conf test failed
[root@ip-10-0-10-182 nginx]# nano nginx.conf
[root@ip-10-0-10-182 nginx]# systemctl restart nginx
[root@ip-10-0-10-182 nginx]# docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS
d54c557b53fa httpd "httpd-foreground" 20 hours ago Up 20 hours 0.0.0.0:8082->
80/tcp, ::8082->80/tcp app2
6089adb6e776 httpd "httpd-foreground" 20 hours ago Up 20 hours 0.0.0.0:8081->
80/tcp, ::8081->80/tcp app1
690be293e4cc httpd "httpd-foreground" 20 hours ago Up 19 hours 0.0.0.0:8080->
80/tcp, ::8080->80/tcp apachev1
[root@ip-10-0-10-182 nginx]# ls
conf.d fastcgi_params.default nginx.conf uwsgi_params
default.d koi-utf nginx.conf.BK uwsgi_params.default
fastcgi.conf koi-win nginx.conf.default win-utf
fastcgi.conf.default mime.types scgi_params
fastcgi_params mime.types.default scgi_params.default
[root@ip-10-0-10-182 nginx]#

```

Se configuro el archivo `nginx.conf`, pero con anterioridad a configurar el archivo se creó una copia del archivo con el nombre de `nginx.conf.BK`, similar al nombre explicado en la clase.

```

ec2-107-22-135-255.compute-1.amazonaws.com (ec2-user)
Terminal Sessions View X server Tools Games Settings Macros Help
Quick connect...
/home/ec2-user/
Name
..
.ssh
app1
app2
.bash_history
.bash_logout
.bash_profile
.bashrc
Remote monitoring
Follow terminal folder

GNU nano 5.8 nginx.conf
events {}

http {
    server {
        listen 80;
        server_name www.adidas.com;
        location / {
            proxy_pass http://localhost:8080;
        }
    }

    server {
        listen 80;
        server_name www.miropa.com;
        location / {
            proxy_pass http://localhost:8081;
        }
    }

    server {
        listen 80;
        server_name www.pizzahoy.com;
        location / {
            proxy_pass http://localhost:8082;
        }
    }
}

```

Se utilizaron las siguientes líneas de comando en el documento `nginx.conf` para hacer el redireccionamiento para que la IP publica de la instancia mostrara lo que estaba almacenado en los distintos contenedores, dependiendo el nombre que se colocara en el navegador, para que el proxy reverso haga el trabajo de redireccionar. Los nombres que se utilizaron para la prueba fueron: `www.adidas.com`, `www.miropa.com` y www.pizzahoy.com.

Configuración de las reglas de entrada, donde se adicionaron los puertos por los cuales se iba hacer las peticiones para acceder a los sitios que se configuraron en los contenedores.

Nombre del grupo de seguridad	ID del grupo de seguridad	Descripción	ID de la VPC
SERVERLINUX2	sg-0662d53705a981e99	launch-wizard-1 created 2025-03-22T03:39:37.317Z	vpc-0693db237798d7dfc

Propietario	Número de reglas de entrada	Número de reglas de salida
925204439332	5 Entradas de permisos	1 Entrada de permiso

Reglas de entrada (5)																																										
<table border="1"> <thead> <tr> <th>Nombre</th> <th>ID de la regla del gr...</th> <th>Versión de IP</th> <th>Tipo</th> <th>Protocolo</th> <th>Intervalo de puertos</th> <th>Origen</th> </tr> </thead> <tbody> <tr> <td>-</td> <td>sgr-0cad381b4197c4469</td> <td>IPv4</td> <td>HTTP</td> <td>TCP</td> <td>80</td> <td>0.0.0.0</td> </tr> <tr> <td>-</td> <td>sgr-0c9588a289849a060</td> <td>IPv4</td> <td>TCP personalizado</td> <td>TCP</td> <td>8081</td> <td>0.0.0.0</td> </tr> <tr> <td>-</td> <td>sgr-06b24f49ea030dc70</td> <td>IPv4</td> <td>TCP personalizado</td> <td>TCP</td> <td>8082</td> <td>0.0.0.0</td> </tr> <tr> <td>-</td> <td>sgr-07c716e98b6fd1d93</td> <td>IPv4</td> <td>SSH</td> <td>TCP</td> <td>22</td> <td>0.0.0.0</td> </tr> <tr> <td>-</td> <td>sgr-0725255a58db7c1b6</td> <td>IPv4</td> <td>TCP personalizado</td> <td>TCP</td> <td>8080</td> <td>0.0.0.0</td> </tr> </tbody> </table>	Nombre	ID de la regla del gr...	Versión de IP	Tipo	Protocolo	Intervalo de puertos	Origen	-	sgr-0cad381b4197c4469	IPv4	HTTP	TCP	80	0.0.0.0	-	sgr-0c9588a289849a060	IPv4	TCP personalizado	TCP	8081	0.0.0.0	-	sgr-06b24f49ea030dc70	IPv4	TCP personalizado	TCP	8082	0.0.0.0	-	sgr-07c716e98b6fd1d93	IPv4	SSH	TCP	22	0.0.0.0	-	sgr-0725255a58db7c1b6	IPv4	TCP personalizado	TCP	8080	0.0.0.0
Nombre	ID de la regla del gr...	Versión de IP	Tipo	Protocolo	Intervalo de puertos	Origen																																				
-	sgr-0cad381b4197c4469	IPv4	HTTP	TCP	80	0.0.0.0																																				
-	sgr-0c9588a289849a060	IPv4	TCP personalizado	TCP	8081	0.0.0.0																																				
-	sgr-06b24f49ea030dc70	IPv4	TCP personalizado	TCP	8082	0.0.0.0																																				
-	sgr-07c716e98b6fd1d93	IPv4	SSH	TCP	22	0.0.0.0																																				
-	sgr-0725255a58db7c1b6	IPv4	TCP personalizado	TCP	8080	0.0.0.0																																				

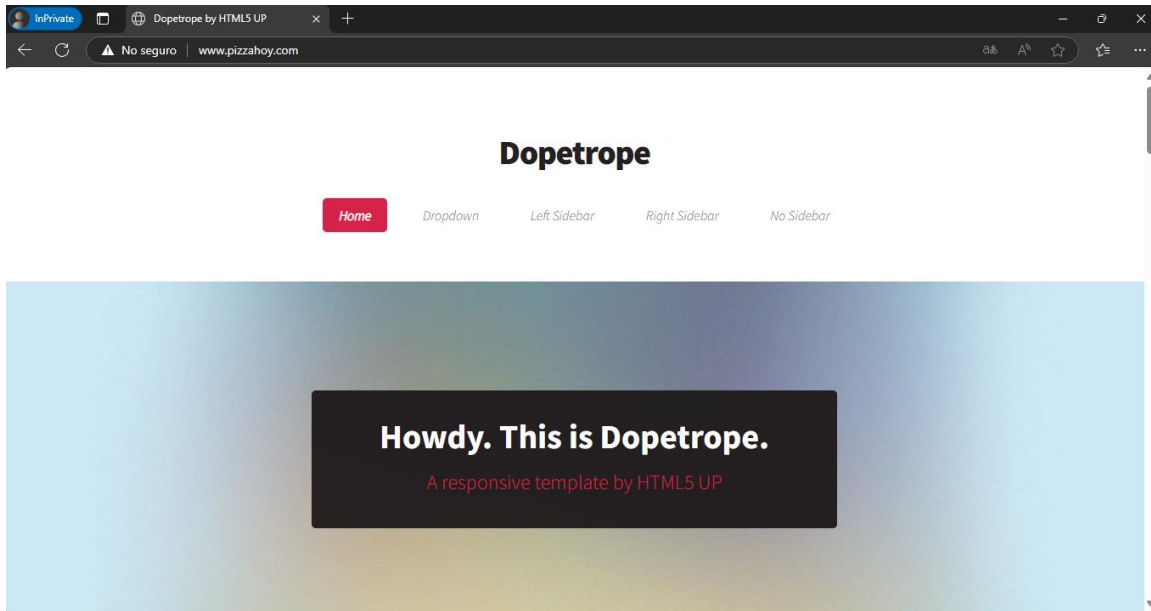
Configuración de las reglas de entrada, donde se adicionaron los puertos por los cuales se iba hacer las peticiones para acceder a los sitios que se configuraron en los contenedores.

```

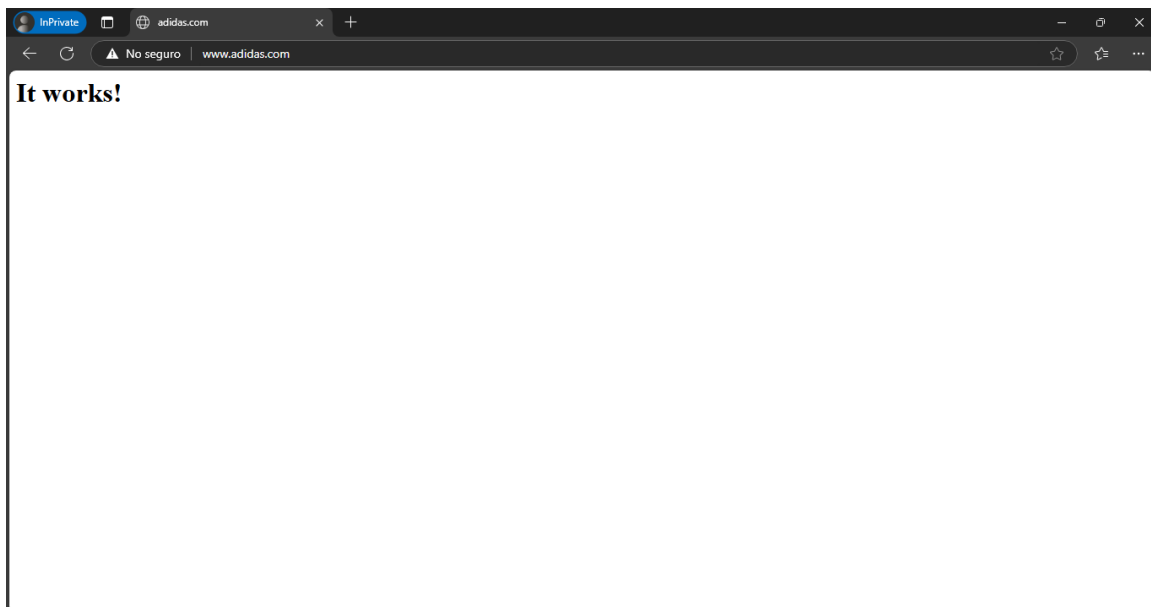
1 # Copyright (c) 1993-2009 Microsoft Corp.
2 #
3 # This is a sample HOSTS file used by Microsoft TCP/IP for Windows.
4 #
5 # This file contains the mappings of IP addresses to host names. Each
6 # entry should be kept on an individual line. The IP address should
7 # be placed in the first column followed by the corresponding host name.
8 # The IP address and the host name should be separated by at least one
9 # space.
10 #
11 # Additionally, comments (such as these) may be inserted on individual
12 # lines or following the machine name denoted by a '#' symbol.
13 #
14 # For example:
15 #
16 #       102.54.94.97       rhino.acme.com       # source server
17 #       38.25.63.10      x.acme.com         # x client host
18
19 # localhost name resolution is handled within DNS itself.
20 # 127.0.0.1       localhost
21 # ::1            localhost
22 107.22.135.255 www.miropa.com
23 107.22.135.255 www.pizzahoy.com
24 107.22.135.255 www.adidas.com
25

```

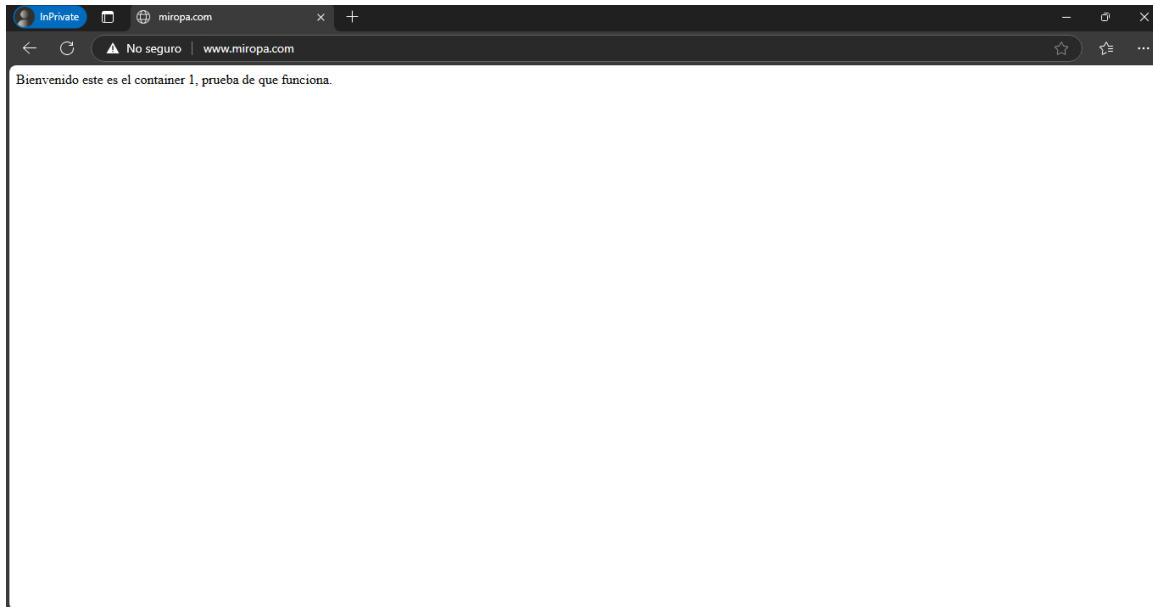
Cambios hechos en el archivo host del sistema operativo local para la prueba, en el cual se incluyeron los nombres a los que irían a la IP publica de la instancia.



Evidencia de resultado de la prueba con la dirección www.pizzahoy.com.



Evidencia de la prueba realizada con la dirección www.adidas.com.



Evidencia de la prueba realizada con la dirección www.miropa.com.

Conclusiones

Con el presente trabajo, pude evidenciar los conocimientos que adquirí sobre lo que es Amazon web services, conocer y aprender todo lo que la plataforma de AWS permite hacer, y algunas funcionalidades que pueden ayudar a una empresa a economizar en costos gracias a la gran variedad de funciones en la nube que tiene la plataforma, evitando que una empresa tenga que infraestructura en infraestructura física, se puede aprender más sobre la computación en la nube, sobre su historia, además gracias al seminario y a las diversas actividades realizadas en el mismo, pude poner en práctica los conocimientos adquiridos en los diferentes temas tratados en el seminario.

Referencias

- 10code*. (10 de 06 de 2024). Obtenido de 10code.es: <https://10code.es/docker-para-que-sirve/>
- akamai*. (s.f.). Obtenido de www.akamai.com:
<https://www.akamai.com/es/glossary/what-are-ports>
- AWS Amazon*. (s.f.). Obtenido de [AWS.amazon.com](http://aws.amazon.com): <https://aws.amazon.com/es/what-is/virtualization/>
- AWS Amazon*. (s.f.). Obtenido de aws.amazon.com:
<https://aws.amazon.com/es/elasticloadbalancing/#:~:text=Elastic%20Load%20Balancing%20%28ELB%29%20distribuye%20autom%C3%A1ticamente%20el%20tr%C3%A1fico,en%20una%20o%20varias%20zonas%20de%20disponibilidad%20%28AZ%29.>
- B, D. (18 de 03 de 2025). *hostinger*. Obtenido de www.hostinger.com:
<https://www.hostinger.com/es/tutoriales/como-configurar-proxy-inverso-nginx>
- B, G. (13 de 03 de 2025). *hostinger*. Obtenido de www.hostinger.com:
<https://www.hostinger.com/es/tutoriales/que-es-nginx>
- documentación AWS*. (s.f.). Obtenido de docs.aws.amazon.com:
https://docs.aws.amazon.com/es_es/vpc/latest/userguide/how-it-works.html
- Documentación aws*. (s.f.). Obtenido de docs.aws.amazon.com:
https://docs.aws.amazon.com/es_es/AWSEC2/latest/UserGuide/Instances.html
- infranet working*. (20 de 06 de 2017). Obtenido de blog.infranetworking.com:
<https://blog.infranetworking.com/computacion-en-la-nube-la-historia/>
- Marujita. (07 de 07 de 2023). *muy tecnológicos*. Obtenido de muytecnologicos.com:
<https://muytecnologicos.com/diccionario-tecnologico/ip-publica>
- Marujita. (07 de 07 de 2023). *muy tecnológicos*. Obtenido de muytecnologicos.com:
<https://muytecnologicos.com/diccionario-tecnologico/amazon-web-services-aws>
- paradigma digital*. (04 de 04 de 2019). Obtenido de www.paradigmadigital.com:
<https://www.paradigmadigital.com/dev/comparativa-servicios-cloud-aws-azure-gcp/>
- redhat*. (09 de 12 de 2024). Obtenido de www.redhat.com:
<https://www.redhat.com/es/topics/virtualization/what-is-virtualization>
- tarlogic*. (s.f.). Obtenido de www.tarlogic.com: <https://www.tarlogic.com/es/glosario-ciberseguridad/instancia-amazon-ec2/>
- time toast*. (s.f.). Obtenido de www.timetoast.com:
<https://www.timetoast.com/timelines/linea-de-tiempo-de-la-evolucion-de-la-computacion-en-la-nube>
- wikipedia*. (s.f.). Obtenido de www.wikipedia.com:
https://es.m.wikipedia.org/wiki/computacion_en_la_nube
- Zettle, K. (s.f.). *www.atlassian.com*. Obtenido de [atlassian](http://atlassian.com):
<https://www.atlassian.com/es/microservices/cloud-computing#:~:text=La%20computaci%C3%B3n%20en%20la%20nube%20es%2>

Ola%20entrega,de%20software%2C%20a%20trav%C3%A9s%20de%20Internet
%20%28la%20nube%29.