



TRABAJO DE GRADO
Opción Seminario-Diplomado.

Implementación de Arquitectura en AWS para la startup Digital Tech BECS

Corporación Universitaria Remington
Ingeniería de Sistemas
Seminario Amazon Web Services

Blanca Elizeth Castiblanco Silva
Juan Pablo Berrio López
Seminario de grado
2024

Tabla de Contenidos

Resumen.....	3
Marco conceptual y contextual	3
Computación en la nube.....	4
¿Qué son los servicios en la nube?	4
¿Qué son los servicios administrados en la nube?	4
Tipos de informática en la nube.....	5
Infraestructura como servicio (IaaS).....	5
Plataforma como servicio (PaaS).....	5
Software como servicio (SaaS).....	5
VPC.....	6
Instancias.....	6
Balanceador de Carga	7
Volúmenes	7
Instantáneas / Snapshot.....	7
Security Groups	8
Auto Scaling Group	8
¿Qué es Docker?	9
Contenedores.....	10
Desarrollo e implementación del aprendizaje.....	11
Entrega I.....	11
Investigue y explique cuáles son los recursos de AWS que permite ejecutar instancias de AWS.....	11
Implemente un servidor web en Amazon Linux.....	12
Video de explicación Servidor Web de Amazon Linux en AWS.....	32
Entrega II	32
Video de explicación implementación de Balanceador de Carga en AWS	54
Implementación de S3.....	54
Implementación de Auto Scaling Group y configuración de políticas	62
Entrega III	68
Docker / Contenedores.....	69
Video implementación de Docker en AWS.....	78
Incluya un diagrama de los recursos usados y como se comunican entre ellos.....	79
Conclusiones	80
Referencias.....	80

Resumen

En este trabajo se da a conocer la implementación de una arquitectura en AWS haciendo uso de los servicios que ofrece este reconocido proveedor de computación en la nube.

Tiene como objetivo optimizar la infraestructura física tecnológica del sistema.

Los servicios más usados en este trabajo son instancias y contenedores, lo que hace que sea una plataforma de alta disponibilidad. Asegura un alto rendimiento, mayor seguridad, eficiencia y escalabilidad en función del crecimiento del trabajo.

Por ser recursos virtuales genera flexibilidad para ajustar la capacidad de almacenamiento, procesamiento y red ya que no se requieren equipos ni infraestructuras físicas.

Gracias a los recursos que ofrece se pueden rastrear costos y rendimiento.

Este trabajo se implementa una infraestructura ágil y flexible alineada con las demandas de la tecnología moderna.

Palabras clave

Seguridad, rendimiento, disponibilidad, crecimiento, sistematización.

Marco conceptual y contextual

Computación en la nube

La informática en la nube es la distribución de recursos de TI bajo demanda a través de Internet mediante un esquema de pago por uso. En lugar de comprar, poseer y mantener servidores y centros de datos físicos, puede acceder a servicios tecnológicos, como capacidad informática, almacenamiento y bases de datos, en función de sus necesidades a través de un proveedor de la nube como Amazon Web Services (AWS).

¿Qué son los servicios en la nube?

Los servicios en la nube son recursos de TI administrados por AWS que se entregan bajo demanda a través de Internet. Tradicionalmente, las organizaciones debían adquirir y configurar todo, desde el hardware de servidores y sistemas de almacenamiento hasta las tecnologías de redes y seguridad, antes de lanzar cualquier sistema digital. El aprovisionamiento y la administración de la infraestructura de TI representan un desafío costoso y complejo, además de restar tiempo a la innovación.

Los servicios en la nube permiten a cualquier persona acceder a la infraestructura de TI necesaria para crear y mantener sistemas digitales, al abstraer infraestructuras complejas de forma que todos puedan desarrollar aplicaciones sofisticadas con rapidez y a escala global. Ejecutar sus aplicaciones en un servidor en la nube es solo el comienzo. Puede utilizar los servicios en la nube para agregar inteligencia artificial y machine learning (IA/ML), en tiempo real y muchas otras capacidades a sus aplicaciones.

¿Qué son los servicios administrados en la nube?

Los servicios en la nube también se denominan servicios administrados en la nube porque AWS administra por completo la infraestructura subyacente. Todo el hardware, los sistemas operativos y otras capas de infraestructura necesarios se almacenan y administran en centros de datos de AWS distribuidos en todo el mundo, con una altamente segura. Compramos y mantenemos todo tipo de recursos de TI y los ponemos a disposición como servicios a los que puede acceder en el código de su aplicación. AWS monitorea y mantiene servidores en la nube, almacenamiento, redes, bases de datos y mucho más, lo que garantiza un rendimiento y un tiempo de actividad constantes.

Además del mantenimiento del hardware, nos ocupamos de todo tipo de tareas operativas de TI, desde el equilibrio de carga y la administración de parches hasta la recuperación ante desastres, entre otras.

Hoy en día, gracias a AWS, cualquier persona, desde estudiantes universitarios hasta equipos empresariales, puede acceder a los servicios en la nube por una fracción del costo que implica administrar la infraestructura local. Cualquier persona puede crear e implementar software sin grandes inversiones iniciales en infraestructura de TI.

Tipos de informática en la nube

Los tres tipos principales de informática en la nube son: infraestructura como servicio, plataforma como servicio y software como servicio. Cada tipo ofrece diferentes niveles de control, flexibilidad y administración para que pueda seleccionar el conjunto de servicios correcto para sus necesidades.

Infraestructura como servicio (IaaS)

La IaaS incluye los bloques de creación básicos para la TI basada en la nube. Generalmente, provee acceso a características de conexión en red, equipos (virtuales o en hardware exclusivo) y espacio de almacenamiento de datos. La IaaS le ofrece el mayor nivel de flexibilidad y control de administración en relación con sus recursos de TI. Es similar a los recursos de TI que muchos desarrolladores y departamentos de TI ya conocen.

Plataforma como servicio (PaaS)

La PaaS elimina la necesidad de administrar la infraestructura subyacente (normalmente hardware y sistemas operativos) y permite enfocarse en la implementación y administración de aplicaciones. Esto contribuye a mejorar el nivel de eficiencia, ya que no debe preocuparse por el aprovisionamiento de recursos, la planificación de la capacidad, el mantenimiento del software, la implementación de parches ni ninguna de las demás arduas tareas que conlleva la ejecución de su aplicación.

Software como servicio (SaaS)

El SaaS le proporciona un producto completo que el proveedor del servicio ejecuta y administra. En la mayoría de los casos, quienes hablan de SaaS en realidad se refieren a aplicaciones de usuario final (como el email basado en la Web). Con una solución basada en SaaS, ya no debe pensar en cómo mantener el servicio ni en cómo administrar la infraestructura subyacente. Solamente debe pensar en cómo utilizar ese software en particular.

VPC

Amazon Virtual Private Cloud (VPC) es un servicio que permite lanzar recursos de AWS en una red virtual aislada de forma lógica que usted defina. Puede controlar todos los aspectos del entorno de red virtual, incluida la selección de su propio rango de direcciones IP, la creación de subredes y la configuración de tablas de ruteo y puertas de enlace de red. Puede utilizar tanto IPv4 como IPv6 para la mayoría de los recursos de la VPC, lo que ayuda a garantizar el acceso seguro y fácil a los recursos y las aplicaciones.

Como uno de los servicios esenciales de AWS, Amazon VPC facilita la personalización de la configuración de red de la VPC. Puede crear una subred con acceso público para los servidores web que tengan acceso a Internet. También permite colocar los sistemas backend, como los servidores de aplicaciones o las bases de datos, en una subred con acceso privado sin acceso a Internet. Amazon VPC le permite utilizar varias capas de seguridad, incluidos los grupos de seguridad y las listas de control de acceso a la red, para ayudar a controlar el acceso a las instancias de Amazon Elastic Compute Cloud (Amazon EC2) en cada subred.

Instancias

Una instancia de computación en la nube es un recurso de servidor que brindan servicios en la nube de terceros. Si bien puede administrar y mantener los recursos del servidor físico en las instalaciones, hacerlo es costoso e ineficiente. Los proveedores de la nube mantienen el hardware en sus centros de datos y les dan acceso virtual a los recursos de computación en forma de instancia. Puede usar la instancia en la nube para ejecutar

cargas de trabajo con uso intensivo de cómputos, como contenedores, bases de datos, microservicios y máquinas virtuales.

Balanceador de Carga

El equilibrador de carga de aplicación funciona al nivel de las solicitudes (capa 7) y dirige el tráfico a los destinos (instancias EC2, contenedores, direcciones IP y funciones de Lambda) según el contenido de la solicitud. El equilibrador de carga de aplicación, ideal para el equilibrio de carga avanzado de tráfico HTTP y HTTPS, proporciona un direccionamiento de solicitudes avanzado destinado a la entrega de arquitecturas de aplicaciones modernas, incluidos microservicios y aplicaciones basadas en contenedores. El equilibrador de carga de aplicación simplifica y mejora el nivel de seguridad de su aplicación. Para ello, se asegura de que se utilicen en todo momento los protocolos y cifradores SSL/TLS más recientes.

Volúmenes

Los volúmenes HDD pasivos (sc1) ofrecen almacenamiento magnético de bajo costo que define el funcionamiento en términos de rendimiento en lugar de IOPS. Con un límite de rendimiento más bajo que el de los volúmenes HDD optimizados para el rendimiento (st1), los volúmenes sc1 representan una buena opción para las cargas de trabajo secuenciales, de datos pasivos y de gran tamaño. Si no necesita acceder a sus datos con frecuencia y quiere ahorrar costos, los volúmenes sc1 proporcionan almacenamiento en bloque económico. No se admiten volúmenes de arranque sc1.

Instantáneas / Snapshot

Las instantáneas de Amazon Elastic Block Store (EBS) proporcionan una solución de protección de datos sencilla y segura diseñada para proteger los datos de almacenamiento en bloques, como los volúmenes de EBS, los volúmenes de arranque y los datos de bloques en las instalaciones. Las instantáneas de EBS son una copia puntual de sus datos y se pueden utilizar para permitir la recuperación de desastres, migrar datos entre regiones y cuentas y mejorar la conformidad de las copias de seguridad.

Puede crear y administrar sus instantáneas de EBS a través de la consola de administración de AWS, la interfaz de la línea de comandos (CLI) de AWS o los SDK de AWS. Las instantáneas de Amazon EBS también se integran con Amazon Data Lifecycle Manager (DLM), que le permite definir políticas que lo ayudan a automatizar la administración del ciclo de vida de las instantáneas. Las instantáneas de EBS se almacenan de forma progresiva, lo que significa que solo se facturarán los bloques almacenados que han cambiado

Security Groups

Un grupo de seguridad controla el tráfico que puede llegar y salir de los recursos a los que está asociado. Por ejemplo, después de asociar un grupo de seguridad con una instancia de EC2, este controla el tráfico entrante y saliente de la instancia.

Cuando crea una VPC, esta viene con un grupo de seguridad predeterminado. Puede crear grupos de seguridad adicionales para una VPC, cada uno con sus propias reglas de entrada y salida. Puede especificar el origen, el rango de puertos y el protocolo para cada regla de entrada. Puede especificar el destino, el rango de puertos y el protocolo para cada regla de salida.

El siguiente diagrama muestra una VPC con una subred, una puerta de enlace de Internet y un grupo de seguridad. La subred contiene una instancia de EC2. El grupo de seguridad está asignado a la instancia. El grupo de seguridad actúa como un firewall virtual. El único tráfico que llega a la instancia es el tráfico permitido por las reglas del grupo de seguridad. Por ejemplo, si el grupo de seguridad contiene una regla que permite el tráfico ICMP a la instancia desde su red, entonces podría hacer ping a la instancia desde su computadora. Si el grupo de seguridad no contiene una regla que permita el tráfico SSH, entonces no podría conectarse a su instancia mediante SSH.

Auto Scaling Group

Puede ajustar manualmente el número de EC2 instancias de su grupo de Auto Scaling en cualquier momento. Este proceso de cambiar el recuento de instancias manualmente se

denomina *escalado manual*. El escalado manual es una alternativa al escalado automático, especialmente si desea realizar cambios de capacidad únicos.

El escalado dinámico escala la capacidad del grupo de escalado automático a medida que se producen cambios de tráfico

Amazon EC2 Auto Scaling admite los siguientes tipos de políticas de escalado dinámico:

- **Escalado de seguimiento de objetivos:** aumenta y reduce la capacidad actual del grupo en función de una CloudWatch métrica de Amazon y un valor objetivo. Funciona de forma similar a los termostatos, que mantienen la temperatura del hogar: seleccionamos una temperatura y el termostato hace el resto.
- **Step scaling** (Escalado por pasos): permite aumentar o reducir la capacidad actual del grupo en función de una serie de ajustes de escalado, denominados *ajustes por pasos*, que variarán en función del tamaño de la interrupción de alarma.
- **Simple scaling** (Escalado sencillo): permite aumentar o reducir la capacidad actual del grupo en función de un único ajuste de escalado con un periodo de recuperación entre cada actividad.

¿Qué es Docker?

Docker es una plataforma de software que le permite crear, probar e implementar aplicaciones rápidamente. Docker empaqueta software en unidades estandarizadas llamadas contenedores que incluyen todo lo necesario para que el software se ejecute, incluidas bibliotecas, herramientas de sistema, código y versión ejecutable. Con Docker, puede implementar y ajustar la escala de aplicaciones rápidamente en cualquier entorno con la certeza de saber que su código se ejecutará.

- La ejecución de Docker en AWS les ofrece a desarrolladores y administradores una manera muy confiable y económica de crear, enviar y ejecutar aplicaciones distribuidas en cualquier escala.

- La frecuencia media de envío de software de los usuarios de Docker es siete veces superior a la de aquellos que no lo usan. Docker le permite entregar servicios aislados con la frecuencia necesaria.
- Las aplicaciones con contenedores facilitan la implementación, la identificación de problemas y el retorno a una fase anterior para remediarlos.
- Las aplicaciones basadas en Docker pueden transferirse a la perfección desde equipos de desarrollo locales a implementaciones de producción en AWS.
- Las aplicaciones con contenedores facilitan la implementación, la identificación de problemas y el retorno a una fase anterior para remediarlos.

Contenedores

Los servicios de contenedores de AWS facilitan la administración de la infraestructura subyacente, ya sea en las instalaciones locales o en la nube, para que usted se pueda centrar en la innovación y en las necesidades de su negocio.

Desarrollo e implementación del aprendizaje

Entrega I

Investigue y explique cuáles son los recursos de AWS que permite ejecutar instancias de AWS.

En Amazon Web Services (AWS), una instancia depende de varios recursos y componentes para operar correctamente. A continuación, se describen los principales recursos y servicios necesarios:

1. Recursos de Cómputo

Instancias EC2: Son las máquinas virtuales que proporcionan el poder de procesamiento.

Tipos de instancias: Determinan la capacidad de la máquina (CPU, memoria, almacenamiento, y optimización de red).

2. Almacenamiento

Amazon Elastic Block Store (EBS): Proporciona almacenamiento persistente para las instancias EC2. Un volumen EBS es como un disco duro que se adjunta a la instancia.

Amazon S3: Almacenamiento para objetos (archivos, backups, datos de aplicaciones).

Instance Store: Almacenamiento efímero directamente vinculado al hardware de la instancia.

3. Redes

Amazon Virtual Private Cloud (VPC): Una red virtual dedicada a tu cuenta AWS.

Subnets: Segmentos de red dentro del VPC que contienen las instancias.

Internet Gateway (IGW): Para proporcionar acceso a Internet a las instancias.

Elastic IPs: Direcciones IP estáticas que pueden asignarse a las instancias.

Security Groups: Reglas de firewall que controlan el tráfico entrante y saliente de las instancias.

Network Access Control Lists (NACLs): Reglas adicionales para controlar el tráfico en las subredes.

4. Administración de Identidad y Seguridad

AWS Identity and Access Management (IAM): Control de acceso a los recursos a través de políticas y roles.

Roles de IAM para instancias: Permiten que las instancias interactúen con otros servicios de AWS de manera segura.

5. Balanceo y Escalado

Elastic Load Balancer (ELB): Distribuye el tráfico entrante entre varias instancias.

Auto Scaling Groups (ASG): Escala automáticamente el número de instancias EC2 según la demanda.

6. Monitoreo y Gestión

Amazon CloudWatch: Monitorea métricas como el uso de CPU, memoria y disco, y configura alertas.

AWS Systems Manager: Gestiona las instancias, automatiza tareas administrativas y soluciona problemas.

7. DNS y Control de Tráfico

Amazon Route 53: Administra el enrutamiento de tráfico a las instancias.

8. Backup y Recuperación

Amazon Backup: Permite realizar copias de seguridad automáticas de volúmenes EBS.

Snapshots de EBS: Copias puntuales de los volúmenes para recuperación de datos.

9. Opciones de Configuración

Key Pairs: Claves SSH para acceder a las instancias.

User Data y Metadata: Scripts y configuraciones que se ejecutan al iniciar una instancia.

Implemente un servidor web en Amazon Linux

Imagen 1. Crear VPC

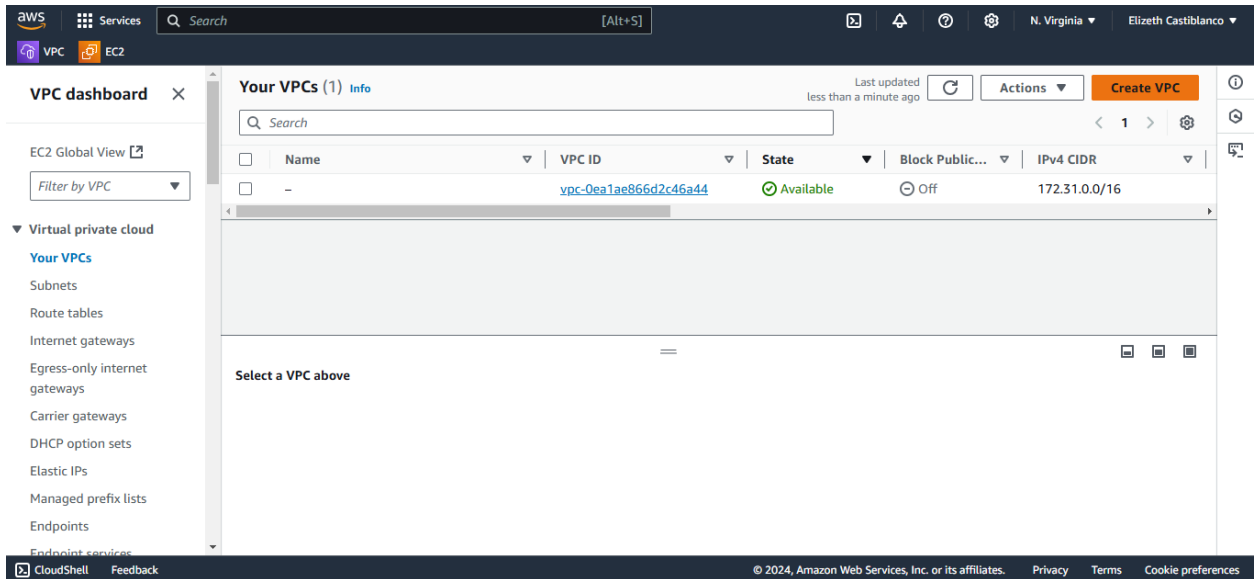


Imagen 2. Asignar nombre

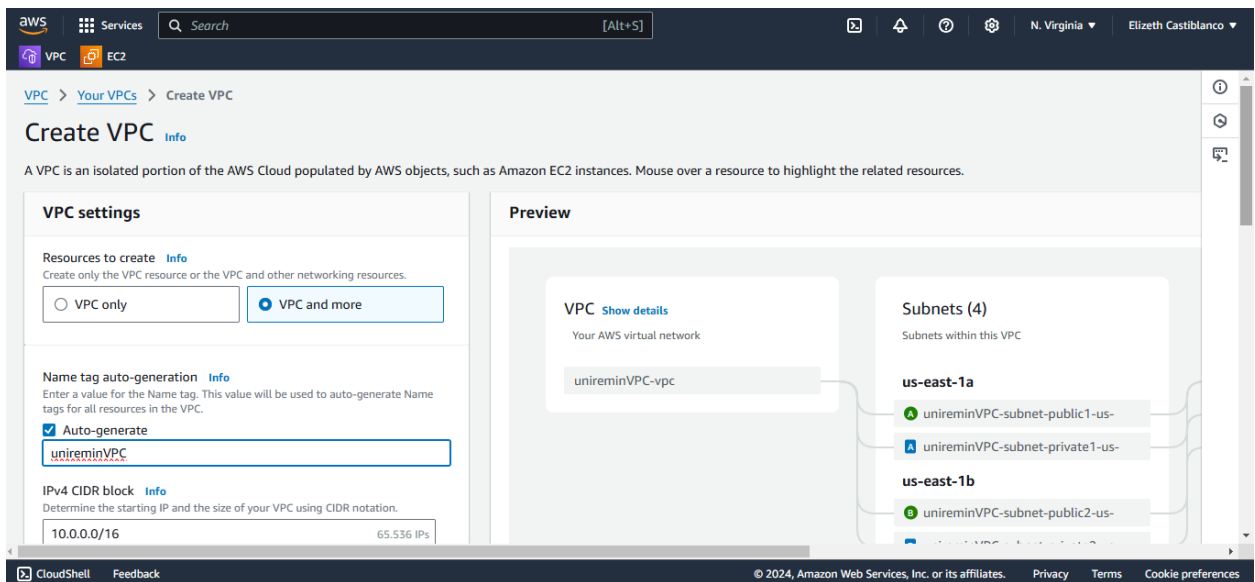


Imagen 3. Elegir recursos de la VPC

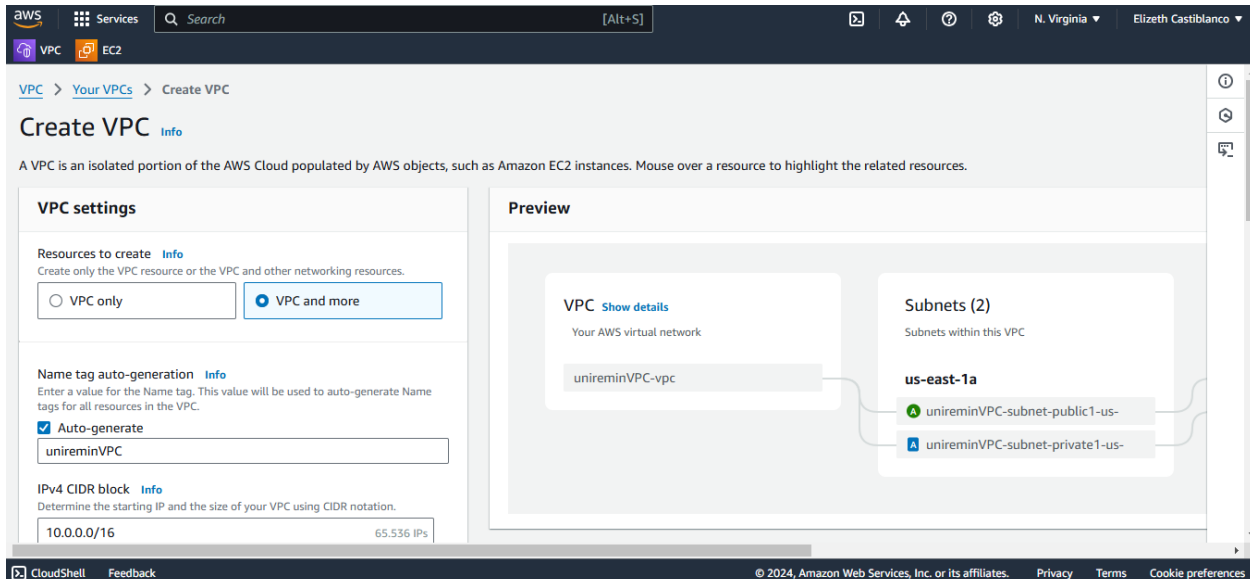


Imagen 4. Elegir número de zonas de disponibilidad, número de subredes públicas y privadas.

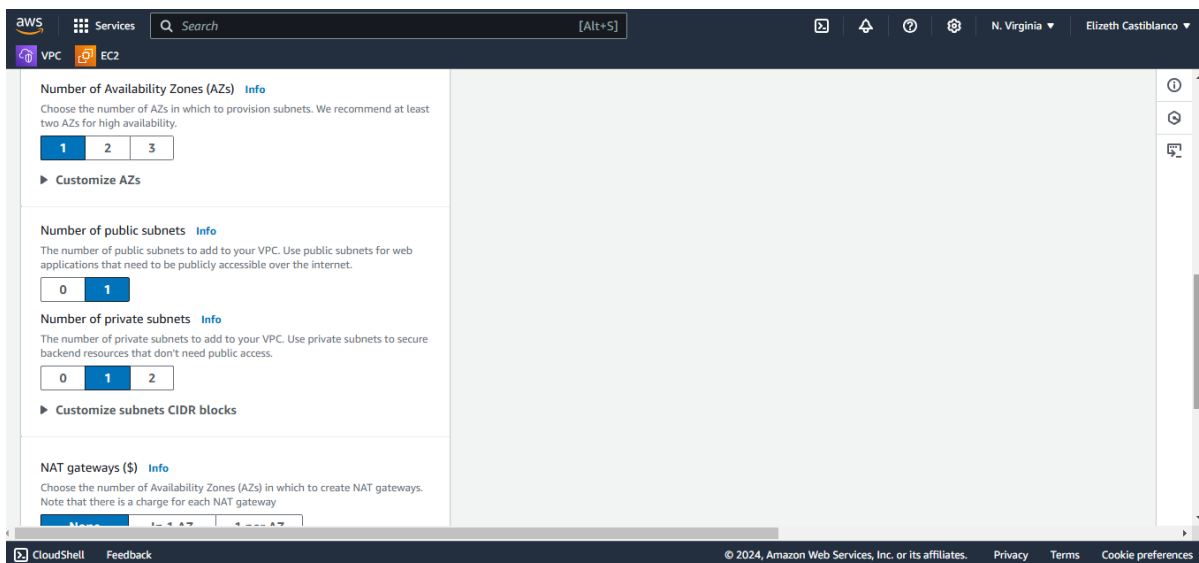


Imagen 5. Ejecución de la creación de la VPC

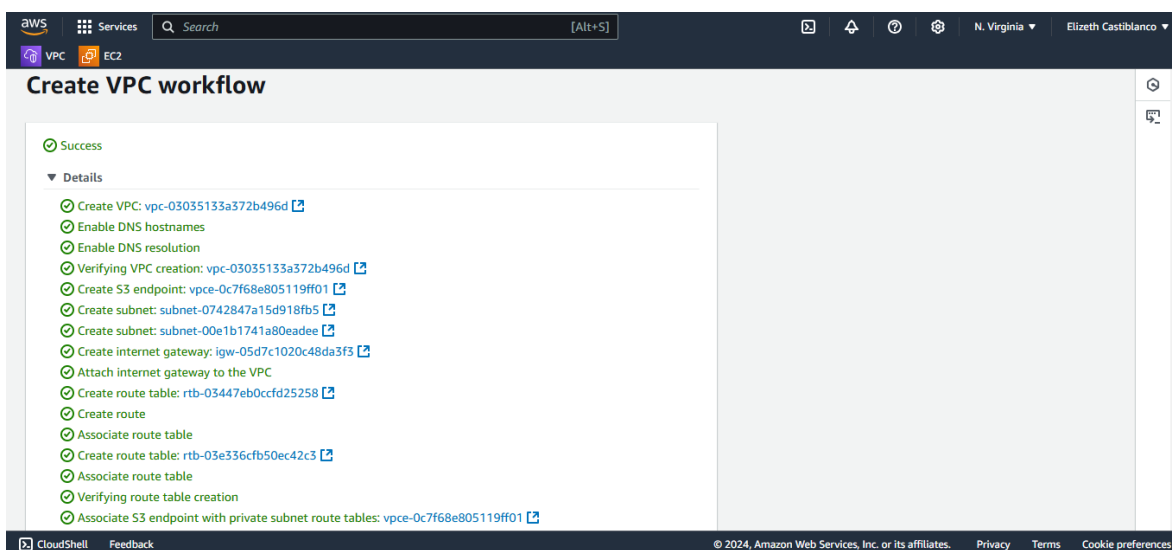


Imagen 6. VPC creada con su respectiva información

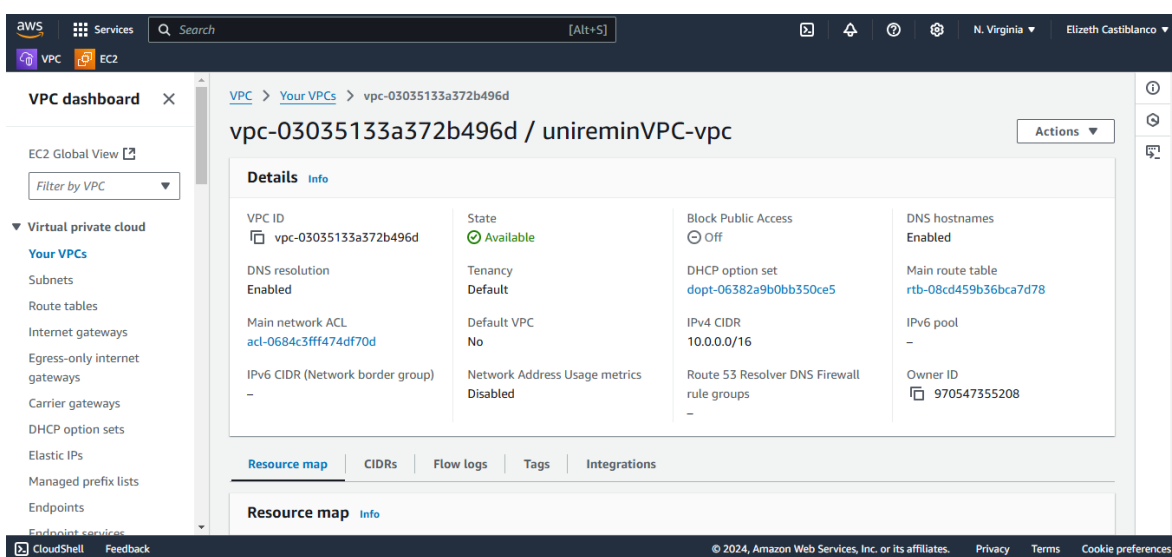


Imagen 7. Creación de Instancia

The screenshot shows the AWS Management Console interface for the 'Instances' page. The top navigation bar includes the AWS logo, a search bar, and the user's name 'Elizeth Castiblanco'. The left sidebar shows the navigation menu with categories like 'Instances', 'Images', and 'Elastic Block Store'. The main content area is titled 'Instances (1) Info' and shows a table with the following data:

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Avail.
ServerU1	i-011860cdef0d6748	Terminated	t2.micro	-	View alarms +	us-east

Below the table, there is a 'Select an instance' section. The footer of the console shows the copyright information: '© 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences'.

Imagen 8. Asignación de nombre

The screenshot shows the 'Launch an instance' wizard in the AWS Management Console. The page is titled 'Launch an instance' and includes a brief introduction: 'Amazon EC2 allows you to create virtual machines, or instances, that run on the AWS Cloud. Quickly get started by following the simple steps below.' The wizard is divided into several sections:

- Name and tags:** The 'Name' field contains 'ServerU2'. There is an 'Add additional tags' link.
- Application and OS Images (Amazon Machine Image):** This section is expanded, showing a search bar with the text 'Search our full catalog including 1000s of application and OS images' and a 'Quick Start' tab.
- Summary:** This section on the right provides a overview of the configuration:
 - Number of instances:** 1
 - Software Image (AMI):** Amazon Linux 2023 AMI 2023.6.2...read more (ami-0455ec754f44f9e4a)
 - Virtual server type (instance type):** t2.micro
 - Firewall (security group):** New security group
 - Storage (volumes):** (This section is partially visible)

At the bottom of the wizard, there are 'Cancel' and 'Launch instance' buttons, along with a 'Preview code' link. The footer of the console shows the copyright information: '© 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences'.

Imagen 9. Elección de Aplicación y Sistema Operativo para lanzar la creación de una máquina virtual.

The screenshot shows the AWS Management Console interface for launching an EC2 instance. The main section is titled "Application and OS Images (Amazon Machine Image)". It includes a search bar and a "Quick Start" tab. Below this, there are several OS options represented by icons: Amazon Linux, macOS, Ubuntu, Windows, Red Hat, SUSE Linux, and Debian. The "Amazon Linux 2023 AMI" is selected. To the right, a "Summary" panel displays the configuration: "Number of instances" is 1, "Software Image (AMI)" is "Amazon Linux 2023 AMI 2023.6.2...", "Virtual server type (instance type)" is "t2.micro", and "Firewall (security group)" is "New security group". A "Launch instance" button is prominently displayed.

Imagen 10. Elección de versión de sistema operativo gratuito

This screenshot shows the "Amazon Machine Image (AMI)" selection screen. The list of AMIs is expanded, showing several options. The "Amazon Linux 2023 AMI" is selected, marked with a blue checkmark. Other visible AMIs include "Amazon Linux 2 AMI (HVM) - Kernel 5.10, SSD Volume Type", "Deep Learning OSS Nvidia Driver AMI GPU PyTorch 2.5 (Amazon Linux 2023)", "Deep Learning OSS Nvidia Driver AMI GPU TensorFlow 2.16 (Amazon Linux 2)", and "Deep Learning Base OSS Nvidia Driver GPU AMI (Amazon Linux 2023)". The "Summary" panel on the right remains the same as in the previous image, showing the instance type as "t2.micro" and the "Launch instance" button.

Imagen 11. Elección de tipo de instancia gratuita (recursos de hardware de nuestra máquina virtual).

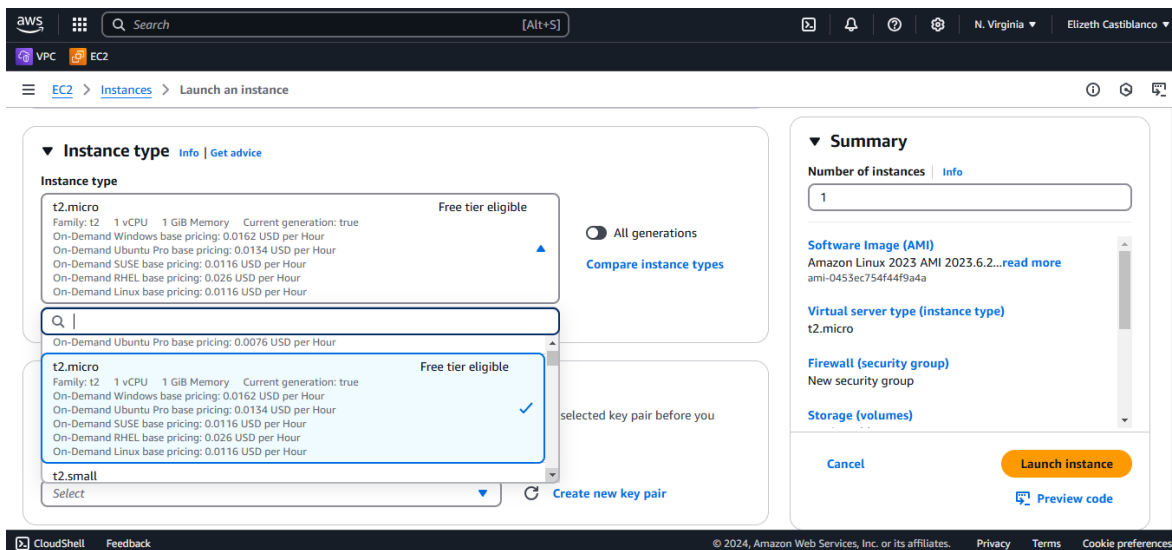


Imagen 12. Creación del Certificado de Seguridad (permite logear luego de crear la instancia).

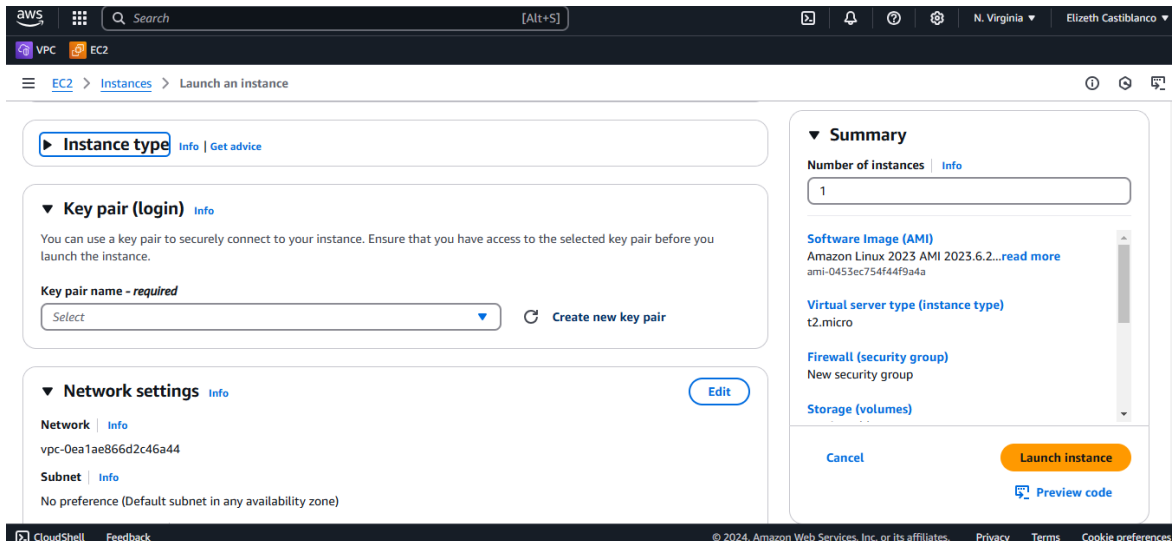


Imagen 13. Selección de tipo de certificado (algoritmo descifrado que se va a utilizar) y formato (PuTTY aplicación ayuda).

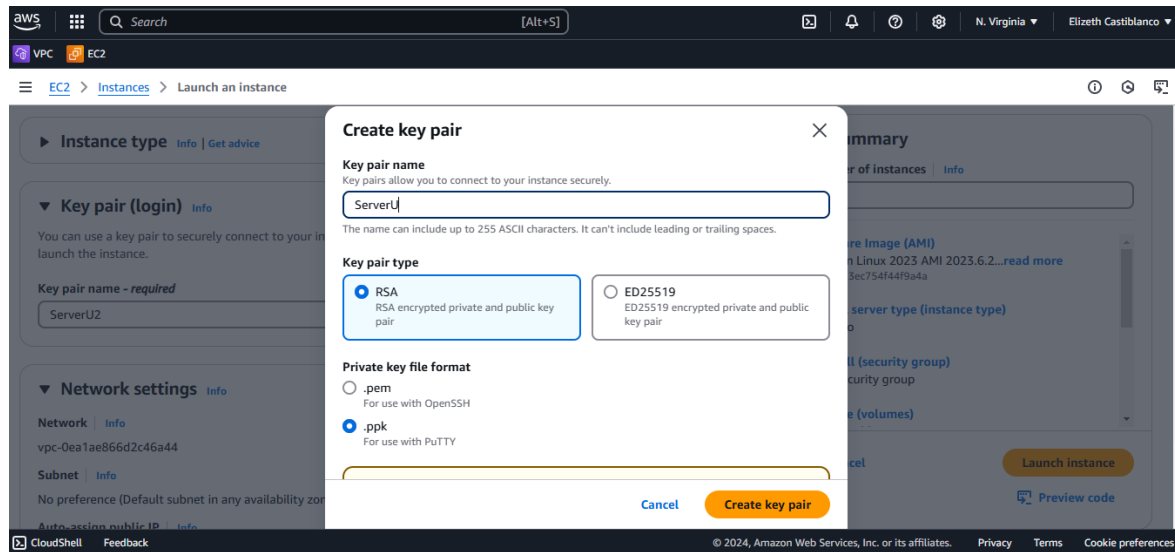


Imagen 14. Configuración de red, elección de VPC creada.

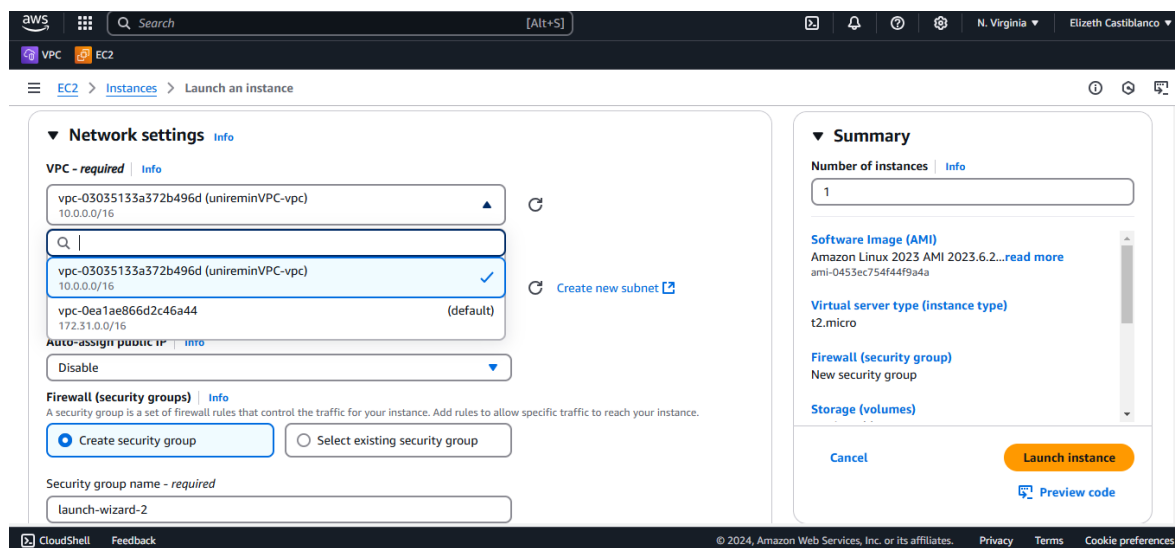


Imagen 15. Elección de Subred (Publica ya que es un servidor web).

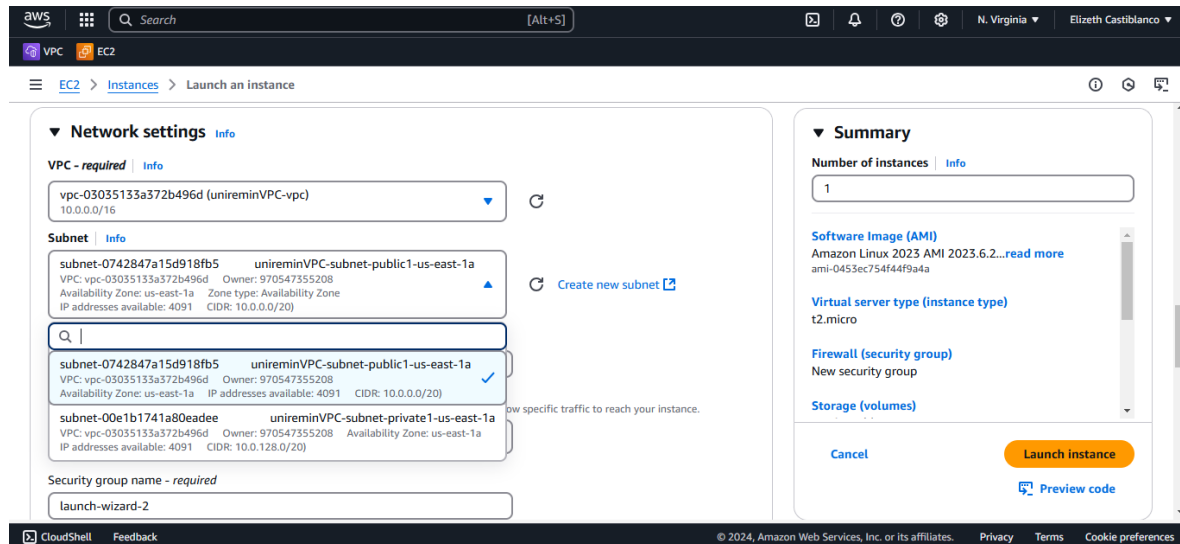


Imagen 16. Habilitar la opción de asignación automática de IP Pública (para que se vea en internet).

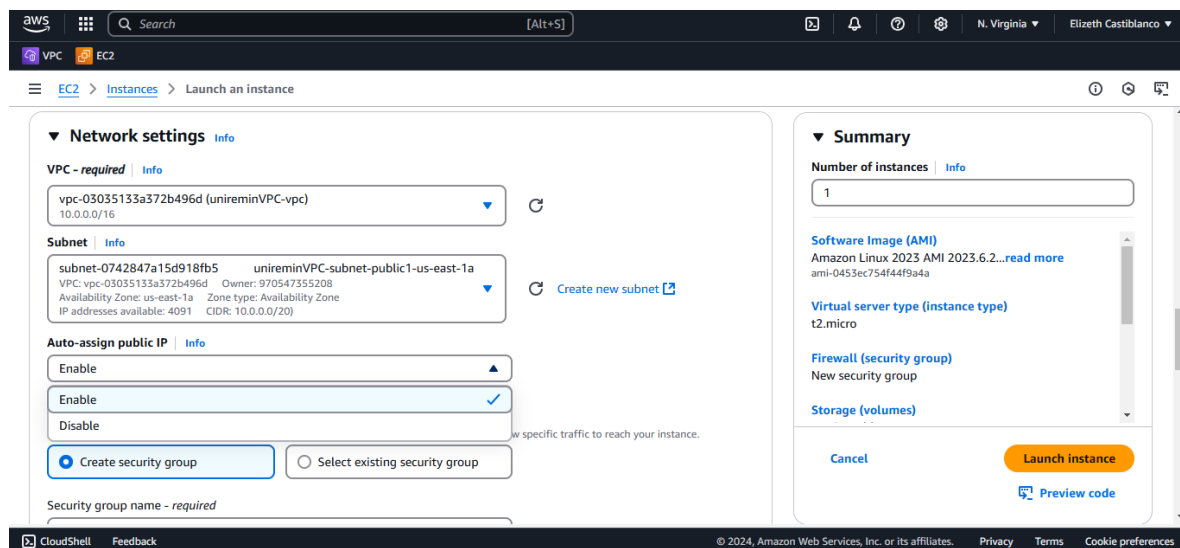


Imagen 17. Asignación de nombre al grupo de seguridad (para mantener las reglas siempre dentro de ese grupo) y las reglas de entrada se dejan por defecto (tipo ssh, puerto 22).

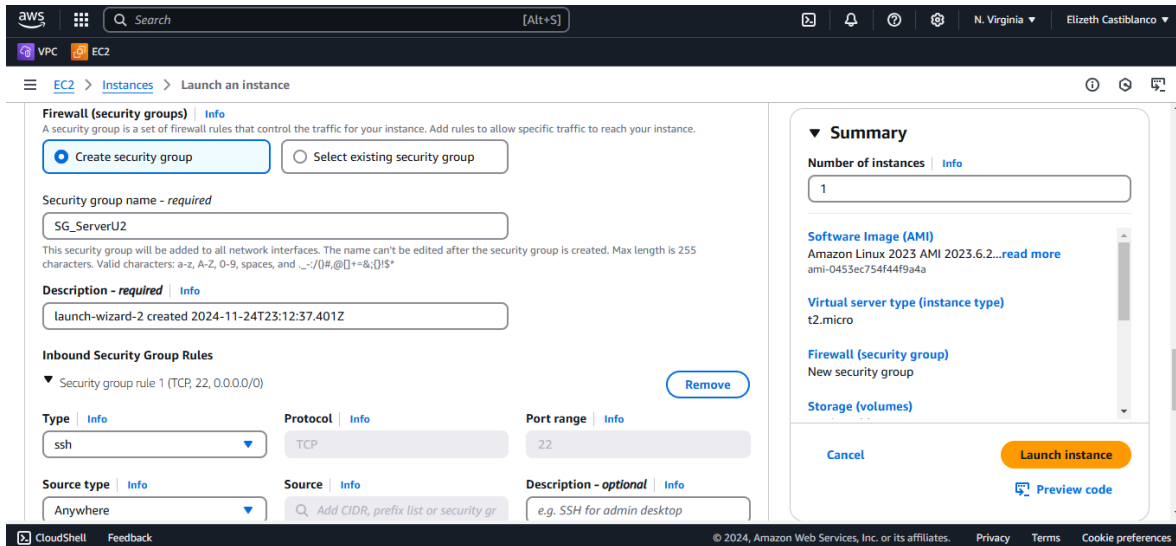


Imagen 18. Características de Almacenamiento se dejan por defecto.

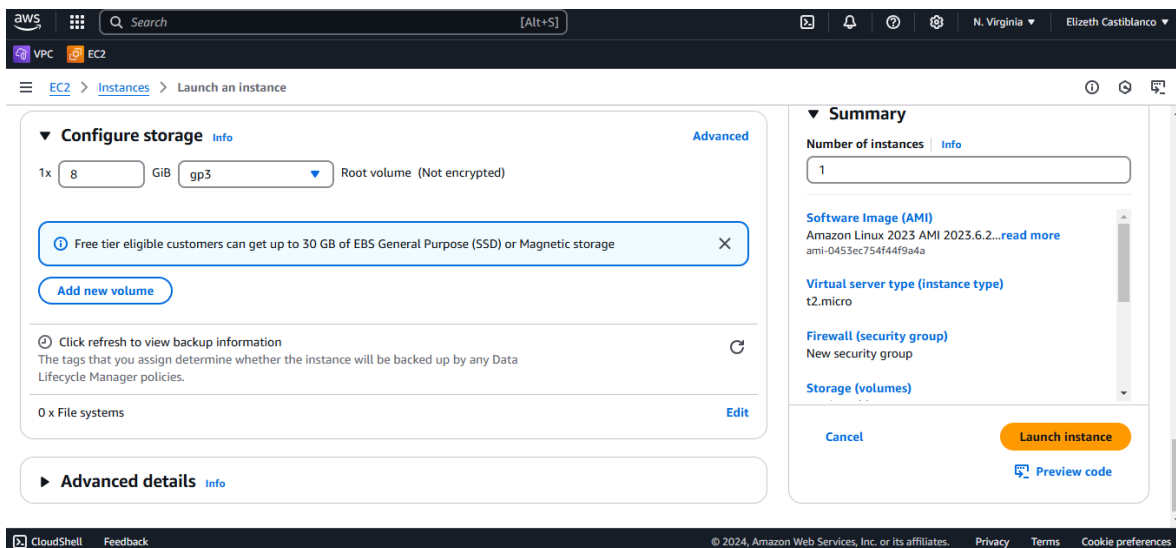


Imagen 19. Lanzamiento de Instancia exitoso.

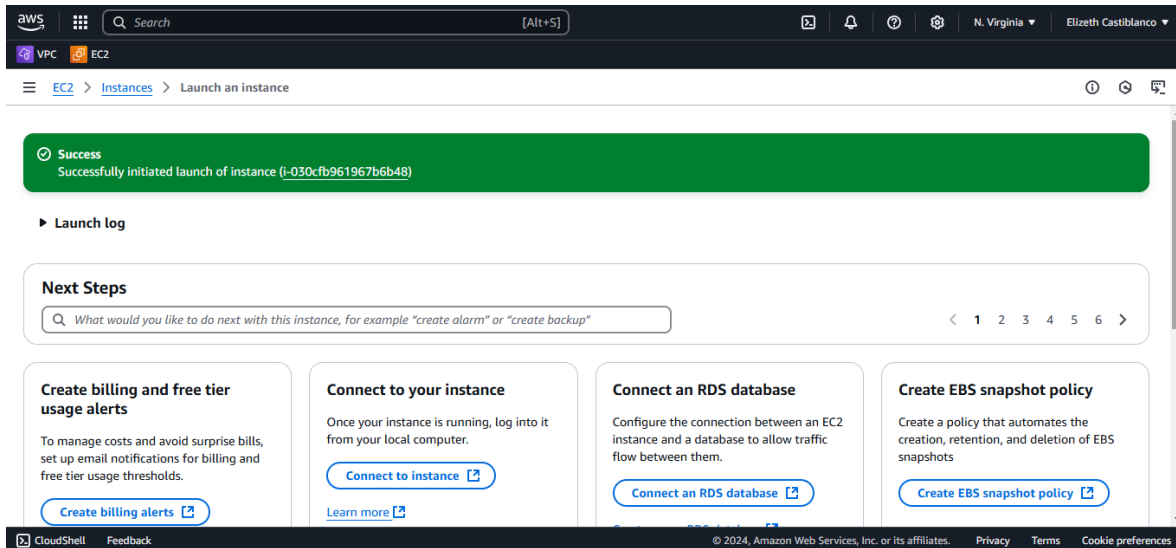


Imagen 20. Estado de la Instancia Inicializando.

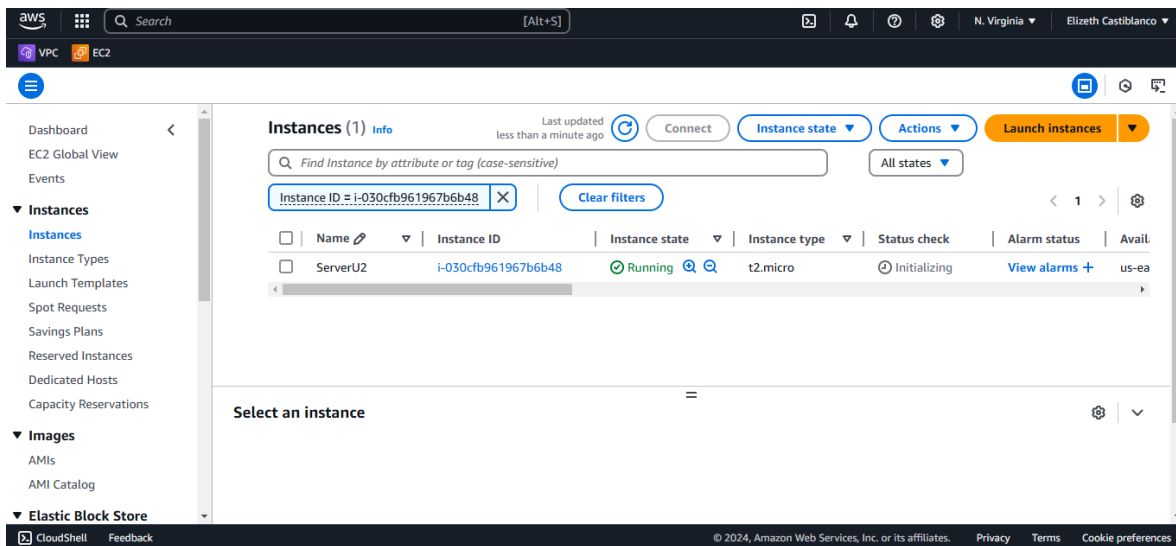


Imagen 21. Información de la Instancia

Instance summary for i-030cfb961967b6b48 (ServerU2) Info

Updated less than a minute ago

[Connect](#) [Instance state](#) [Actions](#)

Instance ID i-030cfb961967b6b48	Public IPv4 address 54.234.142.115 open address	Private IPv4 addresses 172.31.86.245
IPv6 address -	Instance state Running	Public IPv4 DNS ec2-54-234-142-115.compute-1.amazonaws.com open address
Hostname type IP name: ip-172-31-86-245.ec2.internal	Private IP DNS name (IPv4 only) ip-172-31-86-245.ec2.internal	Elastic IP addresses -
Answer private resource DNS name IPv4 (A)	Instance type t2.micro	AWS Compute Optimizer finding Opt-in to AWS Compute Optimizer for recommendations.
Auto-assigned IP address 54.234.142.115 [Public IP]	VPC ID vpc-0ea1ae866d2c46a44	

Imagen 22. Instancia Inicializada 2/2 (corriendo)

Instances (1/2) Info

Last updated less than a minute ago [Connect](#) [Instance state](#) [Actions](#) [Launch instances](#)

Find Instance by attribute or tag (case-sensitive) [All states](#)

Instance state (client): [Clear filters](#)

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Avail
ServerU2	i-030cfb961967b6b48	Running	t2.micro	2/2 checks passed	View alarms	us-east-1
ServerU1	i-011860cfd6f0d6748	Terminated	t2.micro	-	View alarms	us-east-1

i-030cfb961967b6b48 (ServerU2)

[Details](#) [Status and alarms](#) [Monitoring](#) [Security](#) [Networking](#) [Storage](#) [Tags](#)

Instance summary Info

Imagen 23. Conectar Instancia

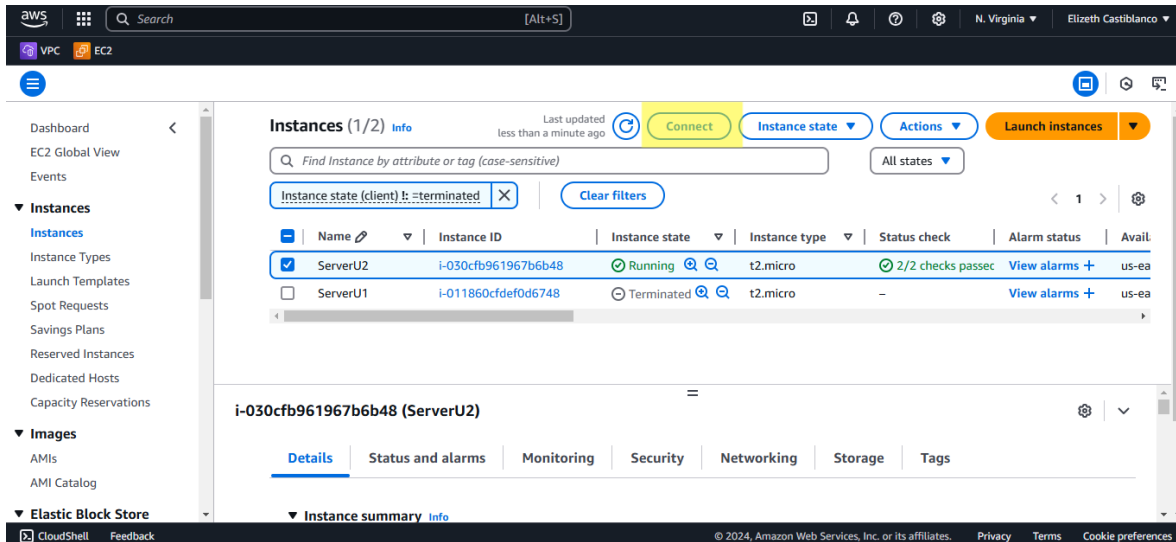


Imagen 24. Copiar el nombre de la máquina para conectarse desde Windows.

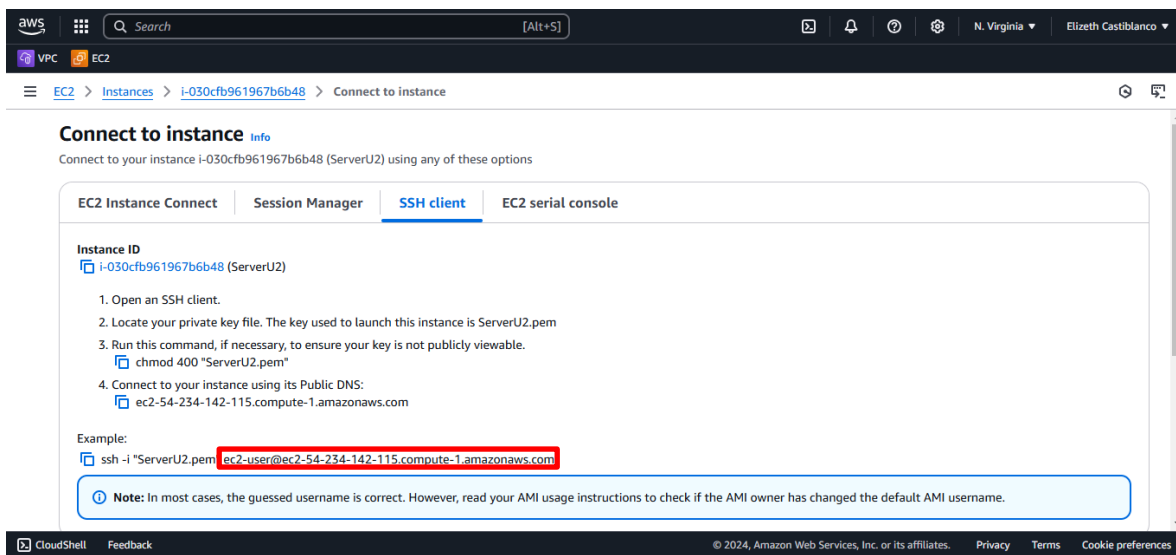


Imagen 25. Usando la aplicación PuTTY copio el nombre de la maquina en Host Name y se deja el puerto 22.

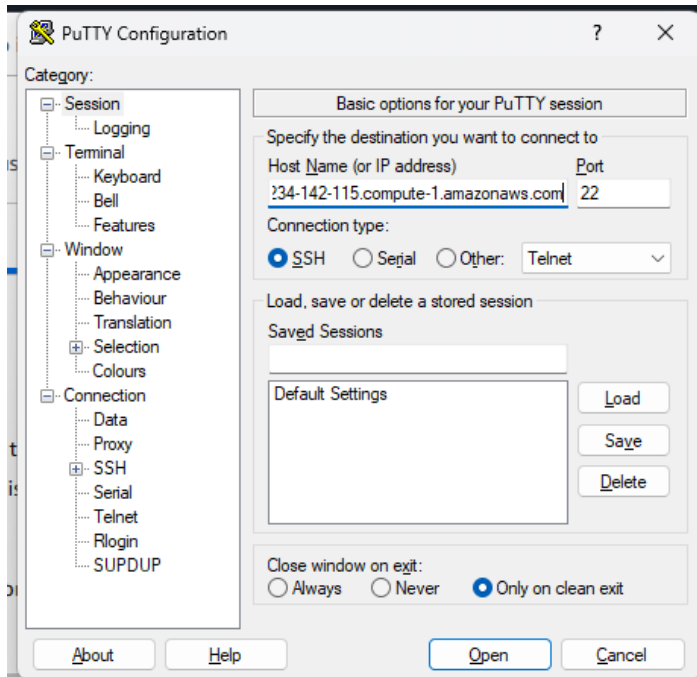


Imagen 26. Opcion SSH, Auth, Credentials. Opcion Private key se carga el certificado descargado anteriormente.

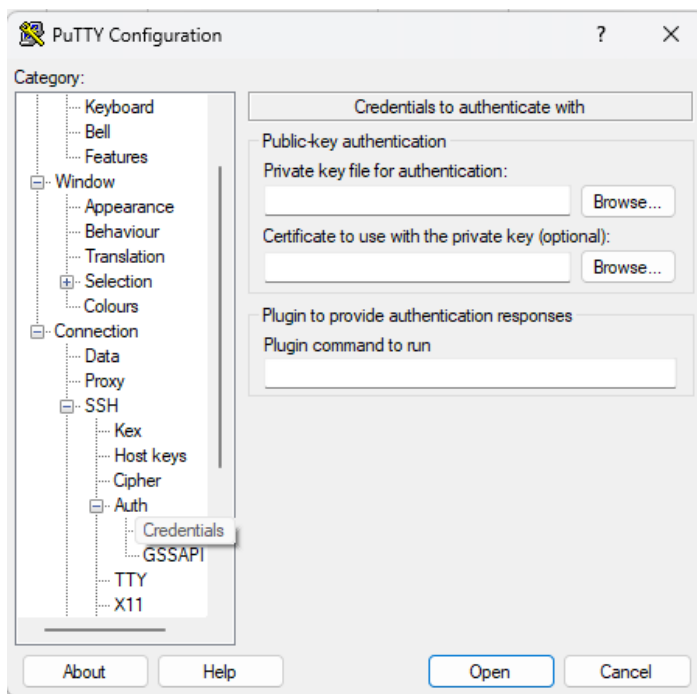


Imagen 27. Cargar el certificado de seguridad descargado.

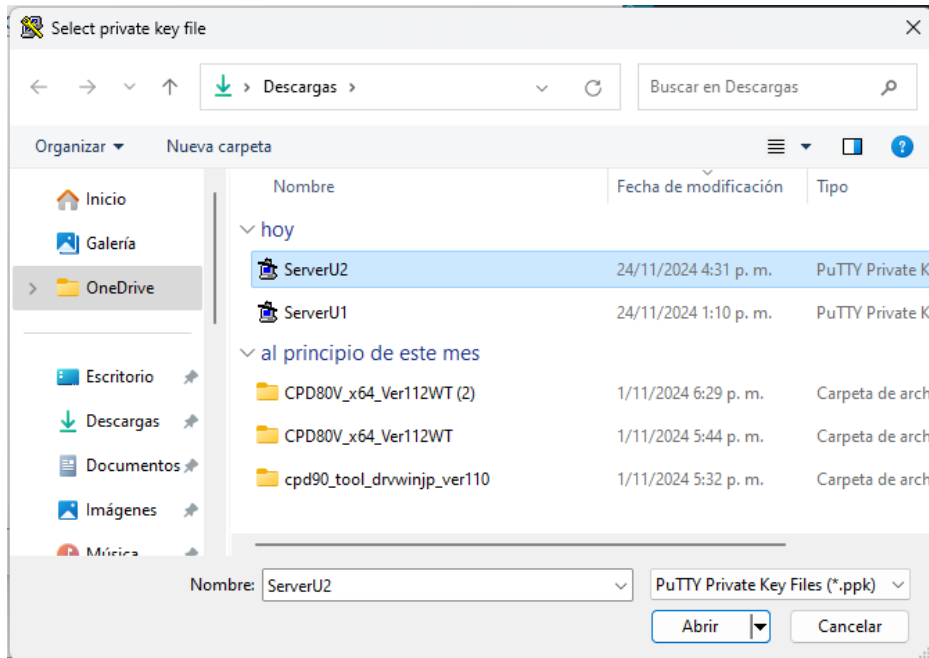


Imagen 28. Aceptar alerta de seguridad.

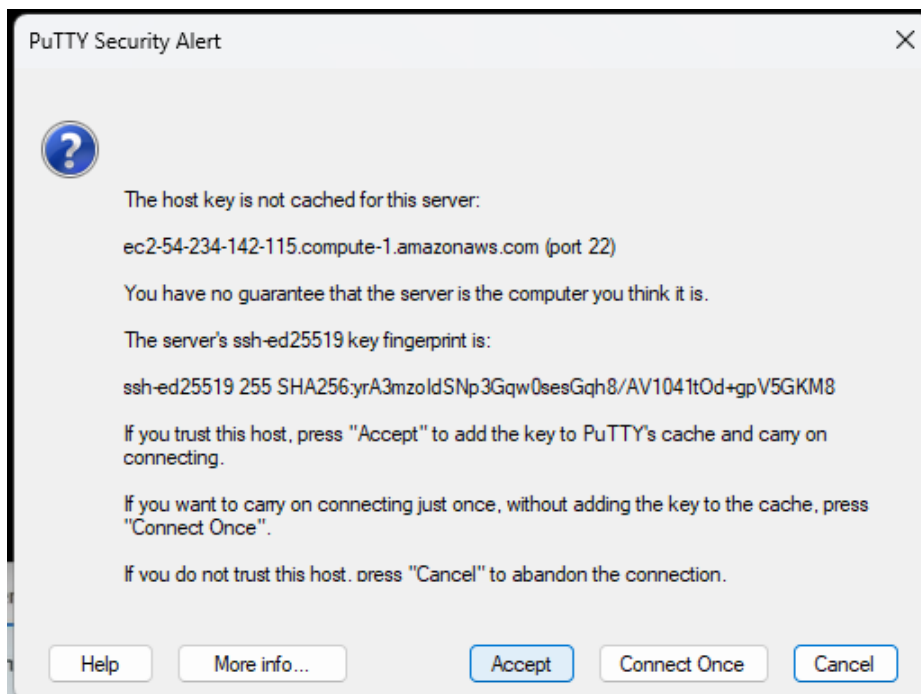


Imagen 29. Acceso a Linux a través de la consola.

```

ec2-user@ip-172-31-86-245:~
Using username "ec2-user".
Authenticating with public key "ServerU2"

#_
~\##### Amazon Linux 2023
~~\#####\
~~\###|
~~\#/ https://aws.amazon.com/linux/amazon-linux-2023
~~V~' ->
~~~~
~~~.
~~~m/
[ec2-user@ip-172-31-86-245 ~]$

```

Imagen 30. Permisos de Administrador (Para loguear como administrador).

```

root@ip-172-31-86-245/home/ec2-user
[ec2-user@ip-172-31-86-245 ~]$ sudo su
[root@ip-172-31-86-245 ec2-user]# yum install httpd
Last metadata expiration check: 0:21:01 ago on Sun Nov 24 21:47:27 2024.
Dependencies resolved.
=====
Package                Arch      Version                Repository      Size
=====
Installing:
httpd                  x86_64    2.4.62-1.amzn2023     amazonlinux    48 k
Installing dependencies:
apr                    x86_64    1.7.2-2.amzn2023.0.2  amazonlinux    129 k
apr-util               x86_64    1.6.3-1.amzn2023.0.1  amazonlinux    98 k
generic-logos-httpd   noarch    18.0.0-12.amzn2023.0.3 amazonlinux    19 k
httpd-core             x86_64    2.4.62-1.amzn2023     amazonlinux    1.4 M
httpd-filesystem      noarch    2.4.62-1.amzn2023     amazonlinux    14 k
httpd-tools           x86_64    2.4.62-1.amzn2023     amazonlinux    81 k
libbrotli              x86_64    1.0.9-4.amzn2023.0.2  amazonlinux    315 k
mailcap               noarch    2.1.49-3.amzn2023.0.3 amazonlinux    33 k
Installing weak dependencies:
apr-util-openssl       x86_64    1.6.3-1.amzn2023.0.1  amazonlinux    17 k
mod_http2              x86_64    2.0.27-1.amzn2023.0.3 amazonlinux    166 k
mod_lua                x86_64    2.4.62-1.amzn2023     amazonlinux    61 k
Transaction Summary

```

Imagen 31. Instalación de aplicación Apache (yum: gestor de paquetes de Amazon Linux, install: comando de la orden, httpd: apache).

```

root@ip-172-31-86-245:/home/ec2-user
[ec2-user@ip-172-31-86-245 ~]$ sudo su
[root@ip-172-31-86-245 ec2-user]# yum install httpd
Last metadata expiration check: 0:21:01 ago on Sun Nov 24 21:47:27 2024.
Dependencies resolved.
=====
Package                Arch      Version                Repository            Size
=====
Installing:
httpd                  x86_64    2.4.62-1.amzn2023     amazonlinux           48 k
Installing dependencies:
apr                    x86_64    1.7.2-2.amzn2023.0.2  amazonlinux           129 k
apr-util               x86_64    1.6.3-1.amzn2023.0.1  amazonlinux           98 k
generic-logos-httpd   noarch    18.0.0-12.amzn2023.0.3 amazonlinux           19 k
httpd-core             x86_64    2.4.62-1.amzn2023     amazonlinux           1.4 M
httpd-filesystem      noarch    2.4.62-1.amzn2023     amazonlinux           14 k
httpd-tools            x86_64    2.4.62-1.amzn2023     amazonlinux           81 k
libbrotli              x86_64    1.0.9-4.amzn2023.0.2  amazonlinux           315 k
mailcap                noarch    2.1.49-3.amzn2023.0.3 amazonlinux           33 k
Installing weak dependencies:
apr-util-openssl       x86_64    1.6.3-1.amzn2023.0.1  amazonlinux           17 k
mod_http2              x86_64    2.0.27-1.amzn2023.0.3 amazonlinux           166 k
mod_lua                x86_64    2.4.62-1.amzn2023     amazonlinux           61 k
=====
Transaction Summary

```

Imagen 32. Instalar paquete de 2.3 M

```

root@ip-172-31-86-245:/home/ec2-user
Install 12 Packages
Total download size: 2.3 M
Installed size: 6.9 M
Is this ok [y/N]: y
Downloading Packages:
(1/12): apr-util-openssl-1.6.3-1.amzn2023.0.1.x 318 kB/s | 17 kB 00:00
(2/12): apr-util-1.6.3-1.amzn2023.0.1.x86_64.rp 1.4 MB/s | 98 kB 00:00
(3/12): generic-logos-httpd-18.0.0-12.amzn2023. 975 kB/s | 19 kB 00:00
(4/12): apr-1.7.2-2.amzn2023.0.2.x86_64.rpm 1.5 MB/s | 129 kB 00:00
(5/12): httpd-2.4.62-1.amzn2023.x86_64.rpm 2.0 MB/s | 48 kB 00:00
(6/12): httpd-filesystem-2.4.62-1.amzn2023.noar 627 kB/s | 14 kB 00:00
(7/12): httpd-core-2.4.62-1.amzn2023.x86_64.rpm 22 MB/s | 1.4 MB 00:00
(8/12): httpd-tools-2.4.62-1.amzn2023.x86_64.rp 1.7 MB/s | 81 kB 00:00
(9/12): libbrotli-1.0.9-4.amzn2023.0.2.x86_64.r 7.8 MB/s | 315 kB 00:00
(10/12): mailcap-2.1.49-3.amzn2023.0.3.noarch.r 1.7 MB/s | 33 kB 00:00
(11/12): mod_lua-2.4.62-1.amzn2023.x86_64.rpm 2.9 MB/s | 61 kB 00:00
(12/12): mod_http2-2.0.27-1.amzn2023.0.3.x86_6 5.2 MB/s | 166 kB 00:00
-----
Total 10 MB/s | 2.3 MB 00:00
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.

```

Imagen 33. Consultar el estado del servicio.

```

root@ip-172-31-86-245:/home/ec2-user

Installed:
  apr-1.7.2-2.amzn2023.0.2.x86_64
  apr-util-1.6.3-1.amzn2023.0.1.x86_64
  apr-util-openssl-1.6.3-1.amzn2023.0.1.x86_64
  generic-logos-httpd-18.0.0-12.amzn2023.0.3.noarch
  httpd-2.4.62-1.amzn2023.x86_64
  httpd-core-2.4.62-1.amzn2023.x86_64
  httpd-filesystem-2.4.62-1.amzn2023.noarch
  httpd-tools-2.4.62-1.amzn2023.x86_64
  libbrotli-1.0.9-4.amzn2023.0.2.x86_64
  mailcap-2.1.49-3.amzn2023.0.3.noarch
  mod_http2-2.0.27-1.amzn2023.0.3.x86_64
  mod_lua-2.4.62-1.amzn2023.x86_64

Complete!
[root@ip-172-31-86-245 ec2-user]# status httpd
bash: status: command not found
[root@ip-172-31-86-245 ec2-user]# systemctl status httpd
○ httpd.service - The Apache HTTP Server
   Loaded: loaded (/usr/lib/systemd/system/httpd.service; disabled; preset: d
   Active: inactive (dead)
   Docs: man:httpd.service(8)
lines 1-4/4 (END)

```

Imagen 34. Activar el estado del servicio y consultarlo nuevamente (debe queda corriendo).

```

root@ip-172-31-86-245:/home/ec2-user

^C
[root@ip-172-31-86-245 ec2-user]# ^C
[root@ip-172-31-86-245 ec2-user]# systemctl start httpd
[root@ip-172-31-86-245 ec2-user]# systemctl status httpd
● httpd.service - The Apache HTTP Server
   Loaded: loaded (/usr/lib/systemd/system/httpd.service; disabled; preset: d
   Active: active (running) since Sun 2024-11-24 22:14:30 UTC; 14s ago
   Docs: man:httpd.service(8)
 Main PID: 26676 (httpd)
  Status: "Total requests: 0; Idle/Busy workers 100/0; Requests/sec: 0; Bytes
   Tasks: 177 (limit: 1111)
  Memory: 12.9M
    CPU: 60ms
  CGroup: /system.slice/httpd.service
          └─26676 /usr/sbin/httpd -DFOREGROUND
            └─26677 /usr/sbin/httpd -DFOREGROUND
              └─26678 /usr/sbin/httpd -DFOREGROUND
                └─26679 /usr/sbin/httpd -DFOREGROUND
                  └─26680 /usr/sbin/httpd -DFOREGROUND

Nov 24 22:14:30 ip-172-31-86-245.ec2.internal systemd[1]: Starting httpd.servic
Nov 24 22:14:30 ip-172-31-86-245.ec2.internal systemd[1]: Started httpd.servic
Nov 24 22:14:30 ip-172-31-86-245.ec2.internal httpd[26676]: Server configured,
lines 1-19/19 (END)

```

Imagen 35. Editar y agregar regla de seguridad en la configuración de la Instancia

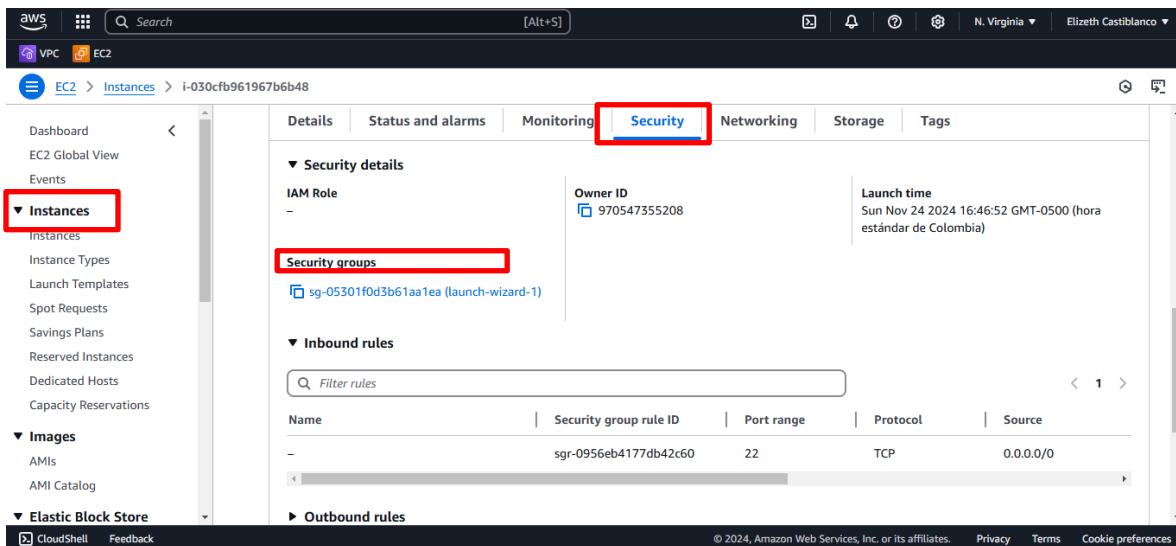


Imagen 36. Agregar regla, colocar número de puerto y desde donde quiero aceptar que se conecten al servicio web (en este caso Todos (0.0.0.0/0)).

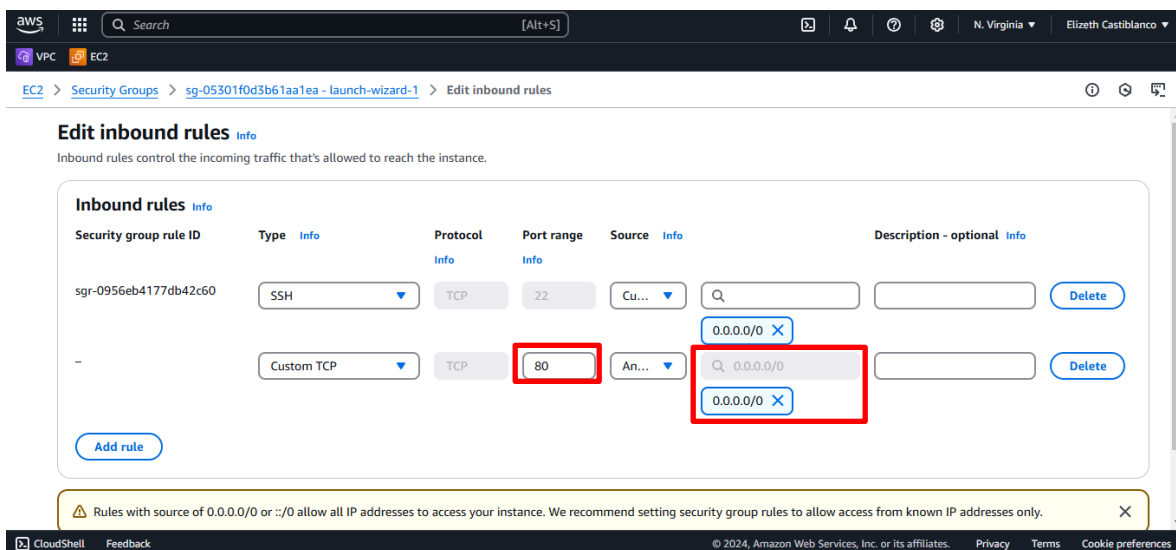


Imagen 37. Reglas de seguridad actualizadas.

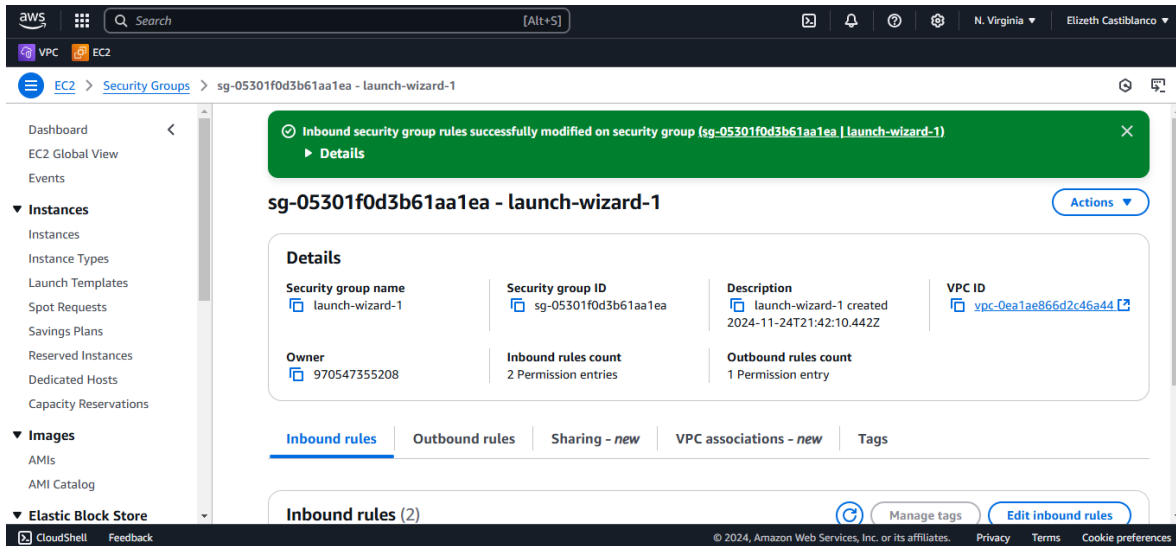


Imagen 38. Copiar la IPv4 asignada en la información de la instancia.

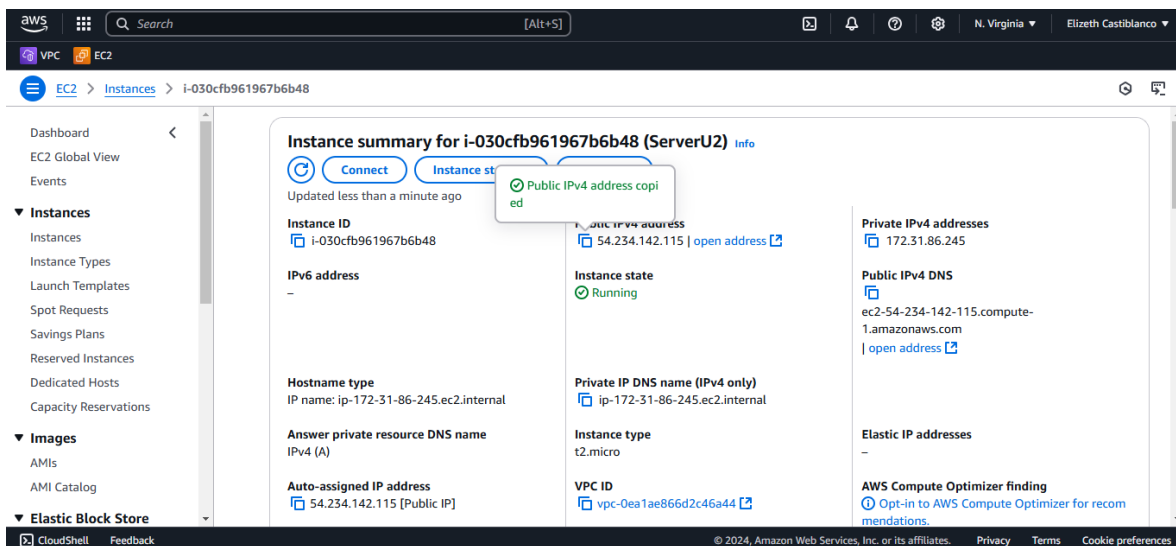
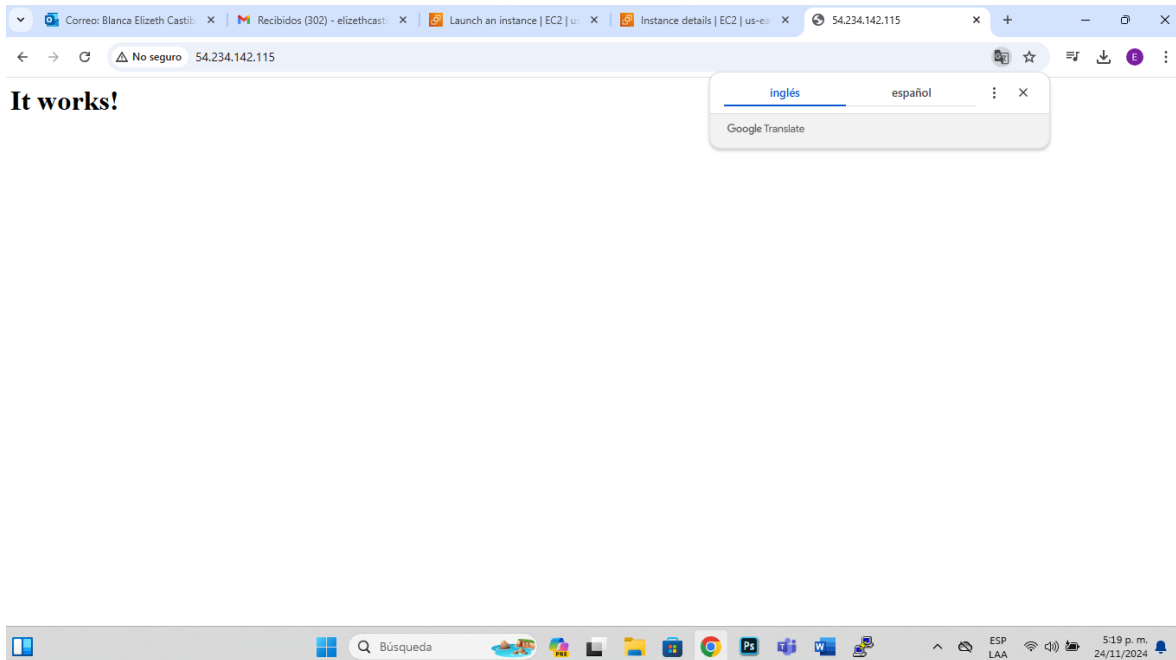


Imagen 39. Pegar la IP en el navegador, ejecutar y ya estaría corriendo correctamente el servidor web.



Video de explicación Servidor Web de Amazon Linux en AWS

https://youtu.be/56m_94_09QM

Entrega II

Imagen 1. Crear Snapshot a partir del volumen de la instancia creada (es una copia del disco duro).

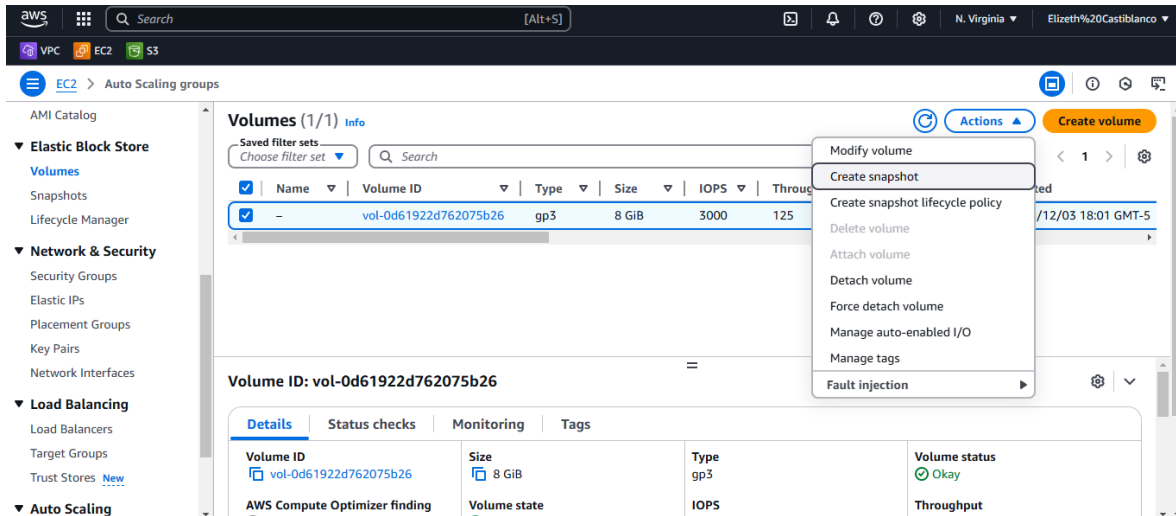


Imagen 2. Asignación de nombre y la configuración que viene por defecto se deja

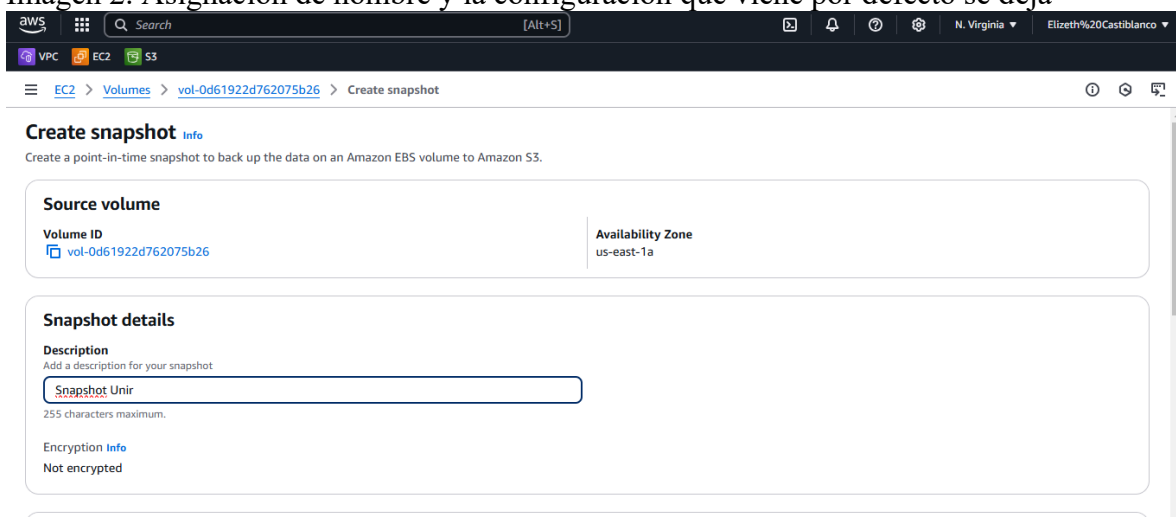


Imagen 3. Snapshot creado

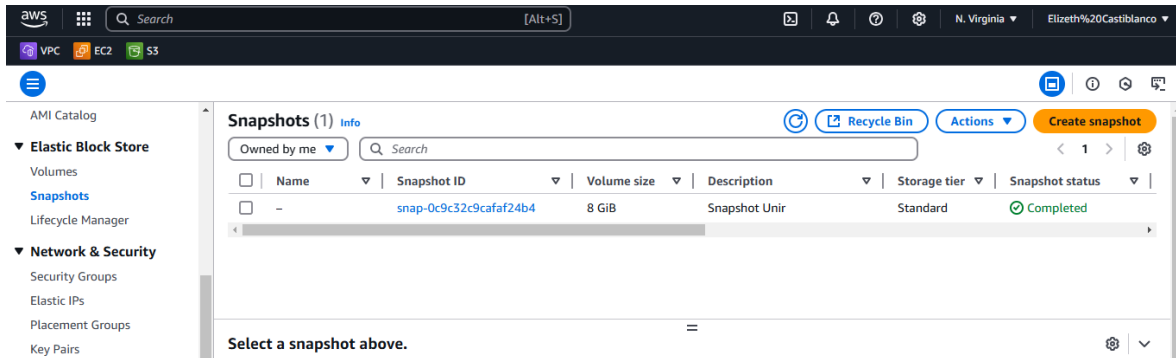


Imagen 4. Creación de etiquetas en volúmenes (Ayuda a filtrar rápidamente información).

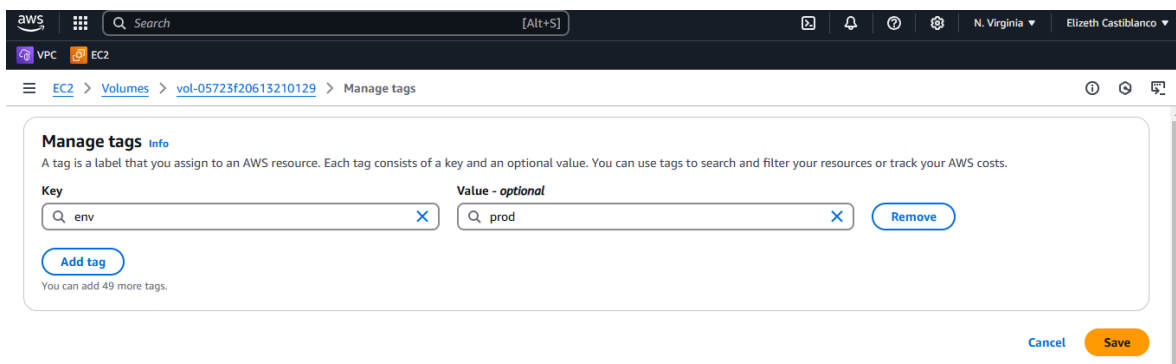


Imagen 5. Creación de AMI (imagen personalizada) a partir del snapshot creado anteriormente (Cuando se crean nuevas instancias, se puede seleccionar esa imagen para ahorrarse la configuración web del servidor).

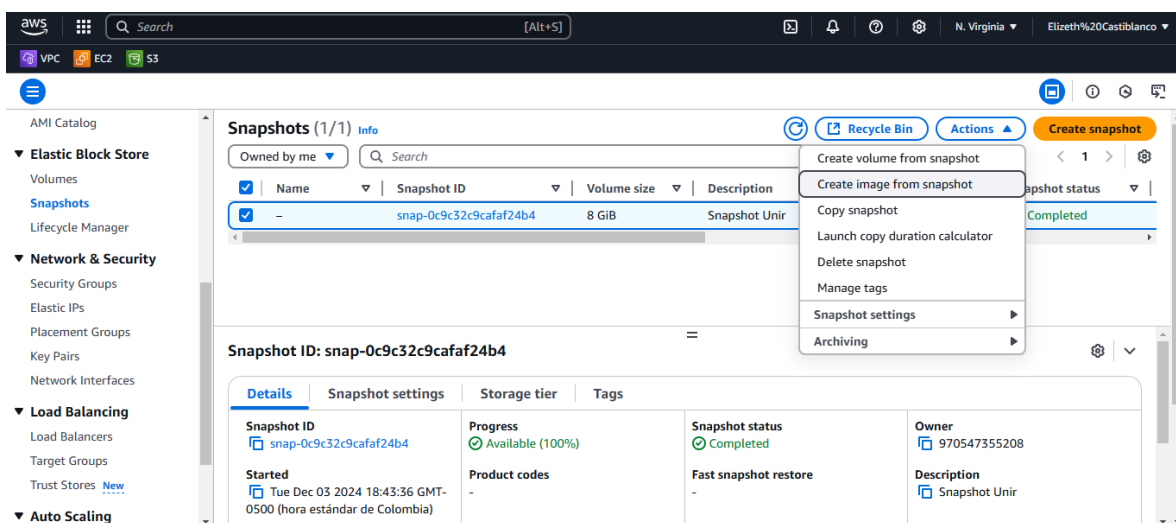


Imagen 6. Asignación de nombre y la configuración que viene por defecto se deja

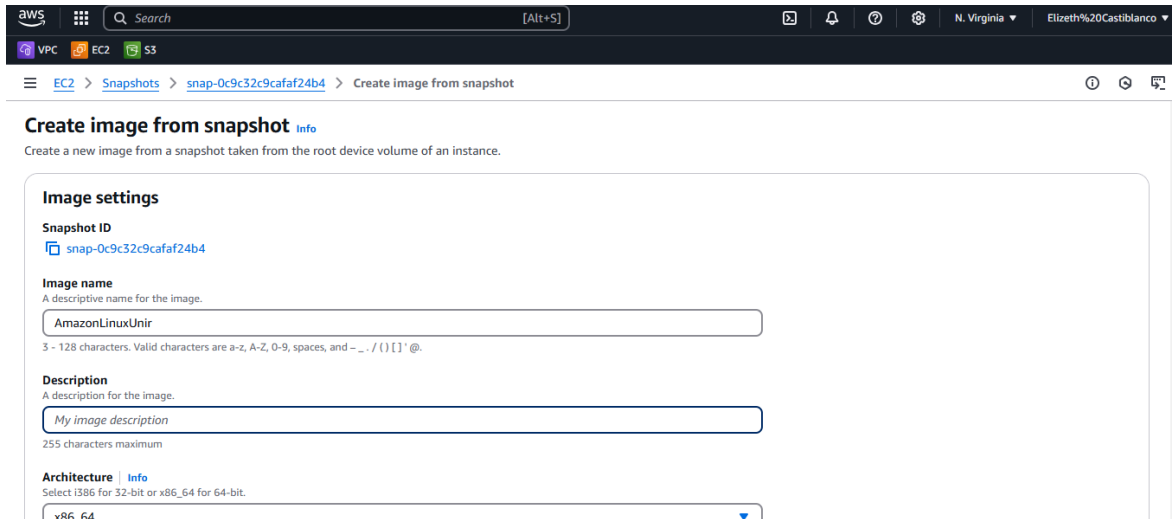


Imagen 7. AMI creada

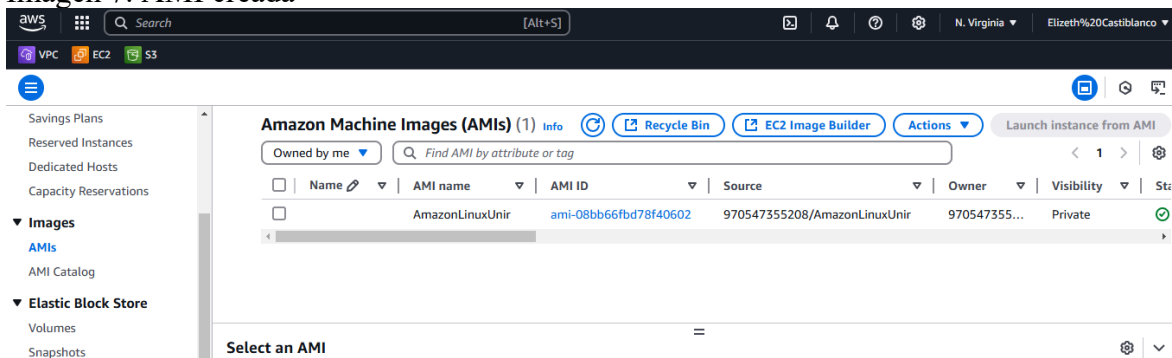


Imagen 8. Crear la primera instancia usando la AMI creada

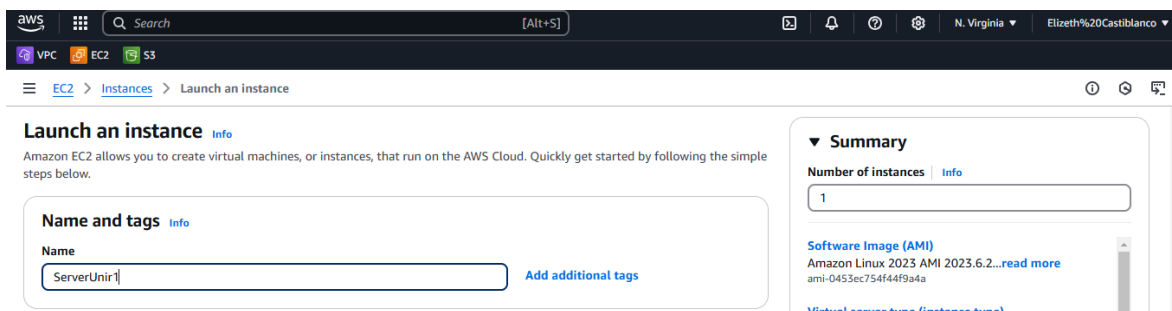


Imagen 9. Se elige la AMI creada

EC2 > Instances > Launch an instance

Recents | **My AMIs** | Quick Start

Owned by me Shared with me

[Browse more AMIs](#)
Including AMIs from AWS, Marketplace and the Community

Amazon Machine Image (AMI)

AmazonLinuxUnir
ami-08bb66fbd78f40602
2024-12-03T23:48:17.000Z Virtualization: hvm ENA enabled: true Root device type: ebs

Description
-

Architecture	AMI ID
x86_64	ami-08bb66fbd78f40602

Summary

Number of instances [Info](#)
1

Software Image (AMI)
AmazonLinuxUnir
ami-08bb66fbd78f40602

Virtual server type (instance type)
t2.micro

Firewall (security group)
New security group

Storage (volumes)

[Cancel](#) [Launch instance](#) [Preview code](#)

Imagen 10. Se elige el almacenamiento y certificado de seguridad

EC2 > Instances > Launch an instance

Instance type [Info](#) | [Get advice](#)

Instance type

t2.micro Free tier eligible

Family: t2 1 vCPU 1 GiB Memory Current generation: true
On-Demand Windows base pricing: 0.0162 USD per Hour
On-Demand Ubuntu Pro base pricing: 0.0134 USD per Hour
On-Demand SUSE base pricing: 0.0116 USD per Hour
On-Demand RHEL base pricing: 0.026 USD per Hour
On-Demand Linux base pricing: 0.0116 USD per Hour

All generations [Compare instance types](#)

[Additional costs apply for AMIs with pre-installed software](#)

Key pair (login) [Info](#)

You can use a key pair to securely connect to your instance. Ensure that you have access to the selected key pair before you launch the instance.

Key pair name - *required*

ServerUnir [Create new key pair](#)

Summary

Number of instances [Info](#)
1

Software Image (AMI)
AmazonLinuxUnir
ami-08bb66fbd78f40602

Virtual server type (instance type)
t2.micro

Firewall (security group)
New security group

Storage (volumes)

[Cancel](#) [Launch instance](#) [Preview code](#)

Imagen 11. Se elige la misma VPC, la subred publica y se activa la asignación de IP pública.

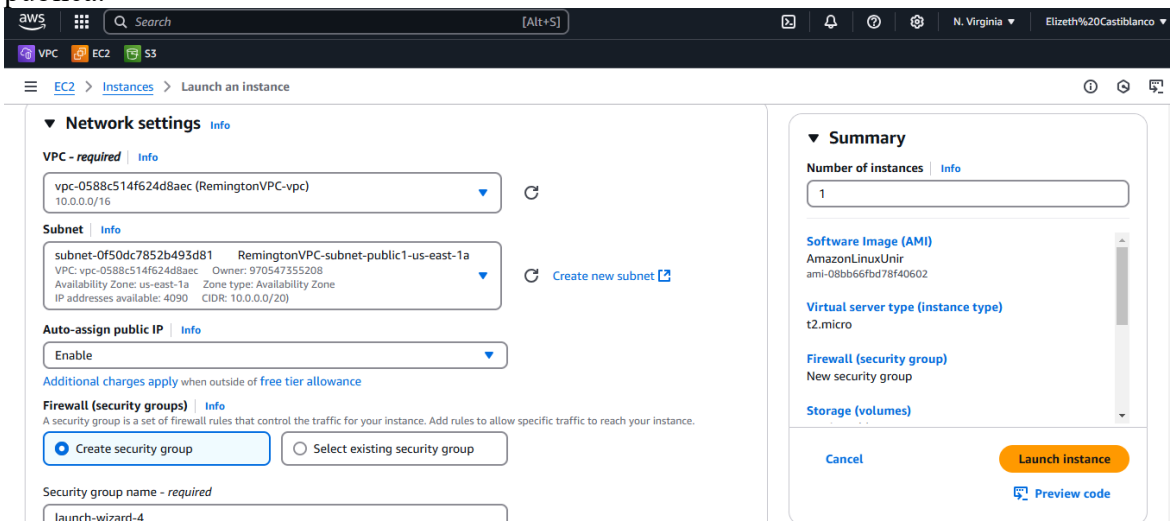


Imagen 12. Se crea una regla de seguridad con el puerto 80 y las demás configuraciones que vienen por defecto se dejan

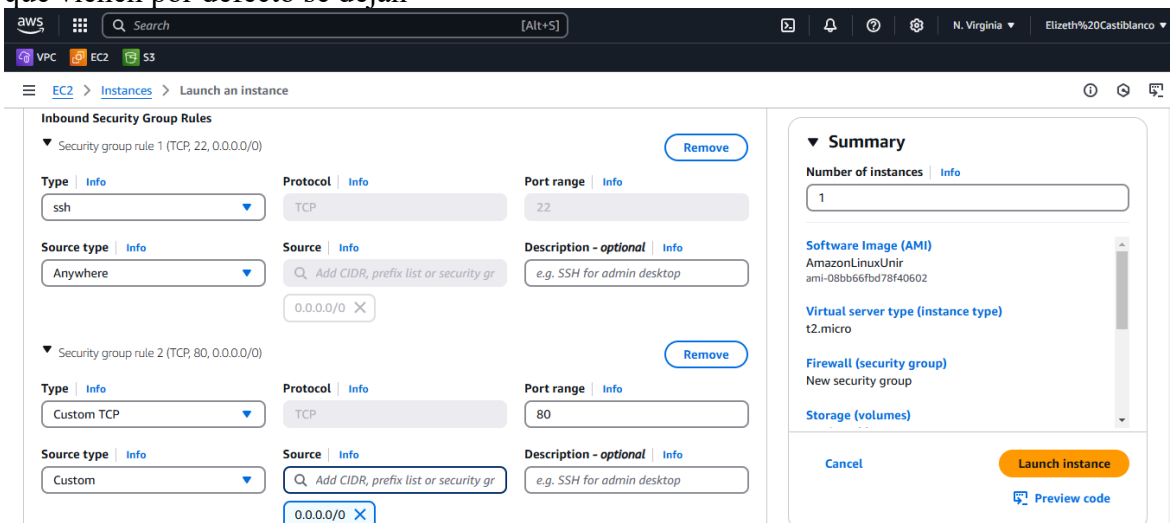


Imagen 13. Instancia inicializando

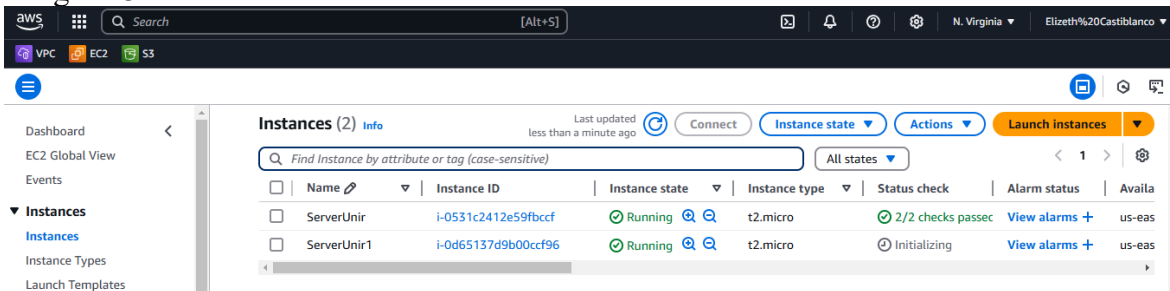


Imagen 14. Instancia en estado 2/2

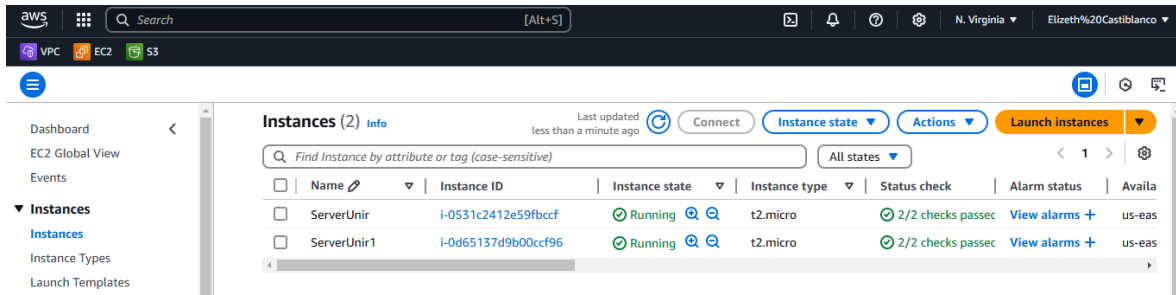
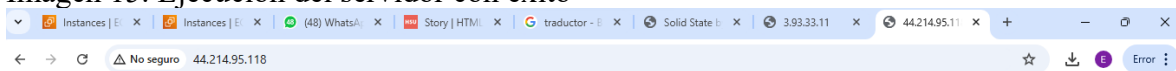


Imagen 15. Ejecución del servidor con éxito



SITIO WEB DE PRUEBA AMAZON LINUX

Imagen 16. Creación de segunda instancia con la misma configuración de la anterior

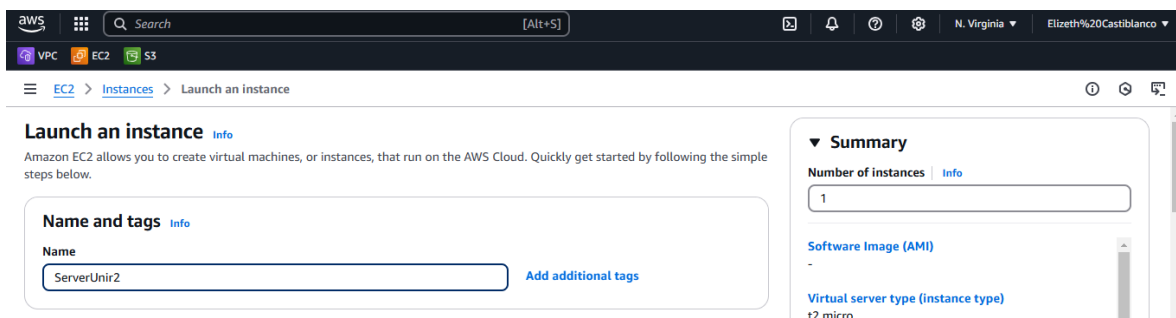


Imagen 17. Instancia inicializando

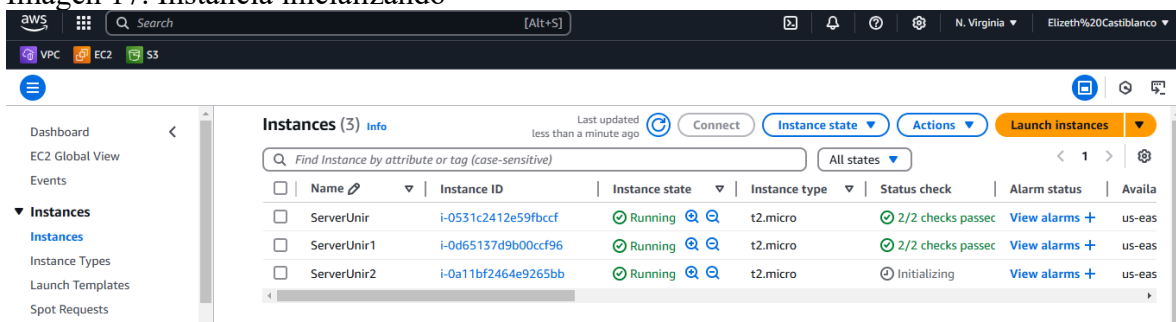


Imagen 18. Instancia en estado 2/2

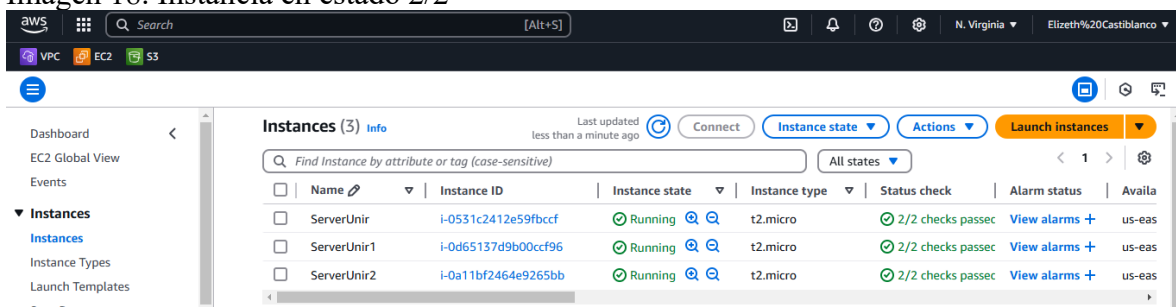
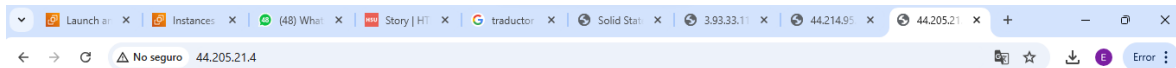


Imagen 19. Ejecución del servidor con éxito



SITIO WEB DE PRUEBA AMAZON LINUX



Imagen 20. Crear Balanceador de carga

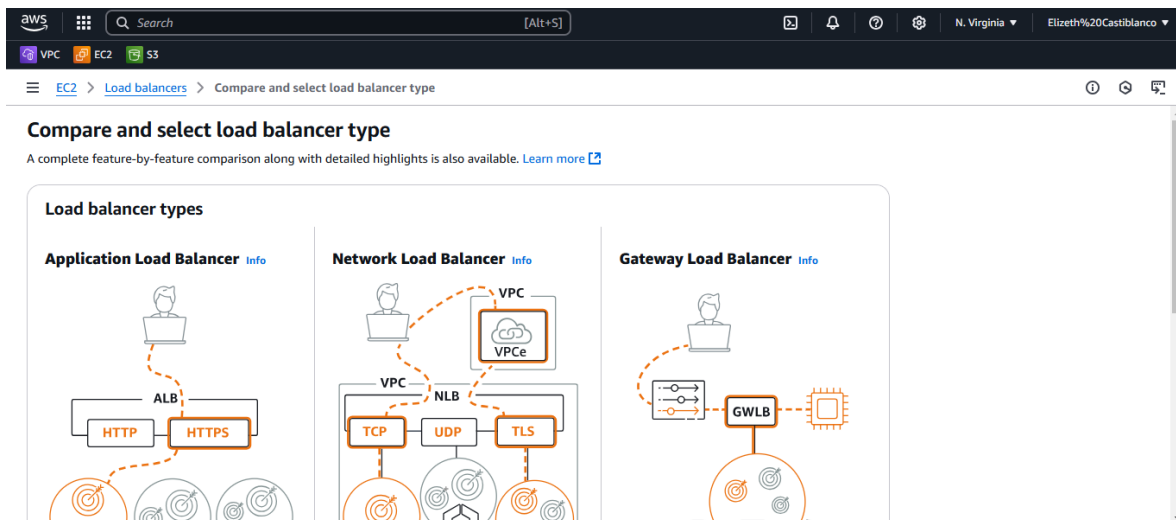


Imagen 21. Asignación de nombre

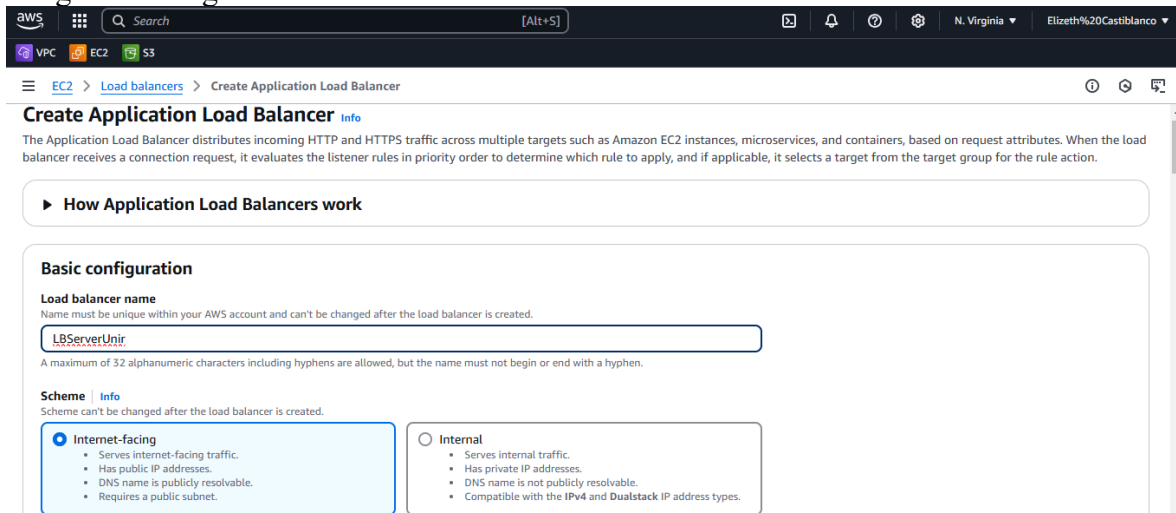


Imagen 22. Configuración de IP y VPC

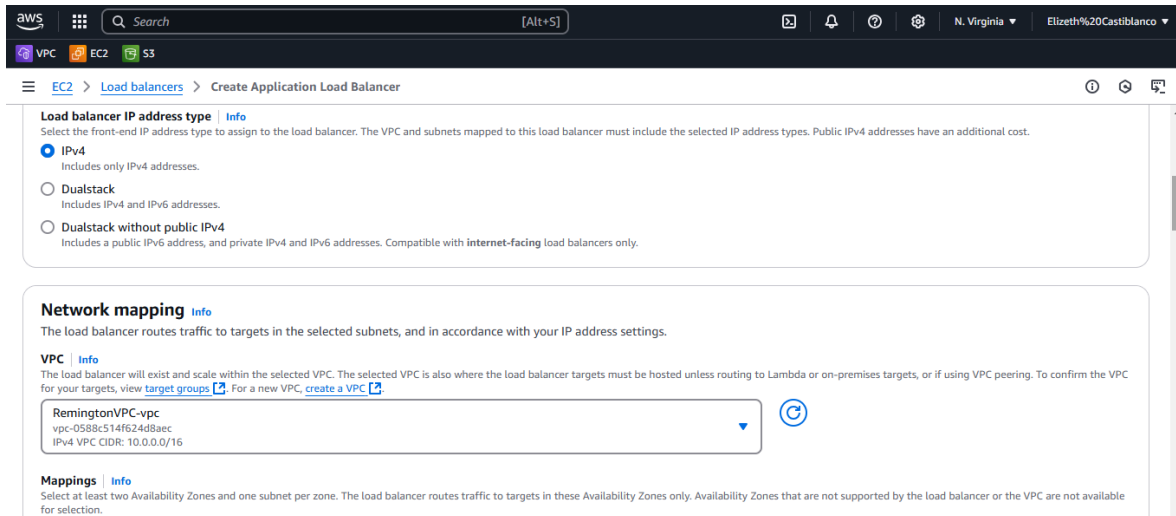


Imagen 23. Elección de zonas de disponibilidad y redes públicas (mínimo dos)

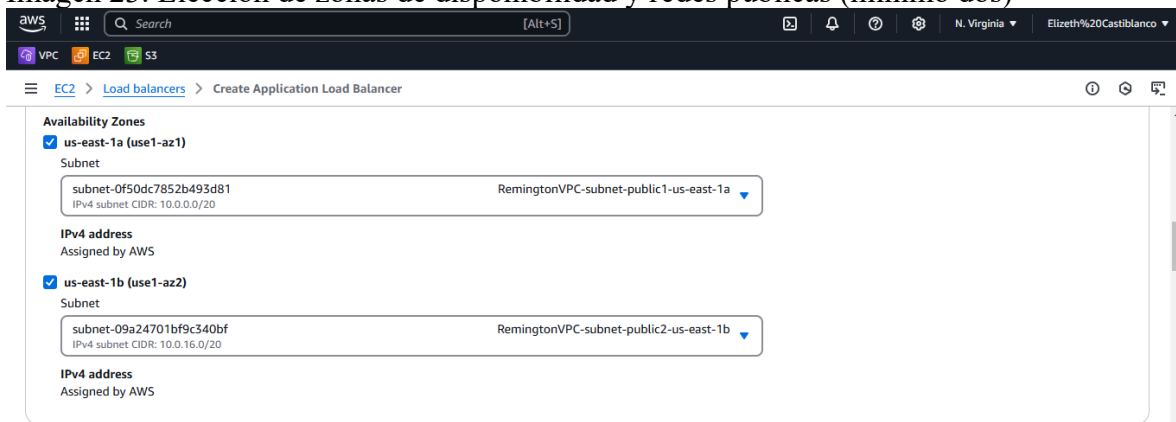


Imagen 24. Crear grupo de seguridad independiente para el balanceador de carga

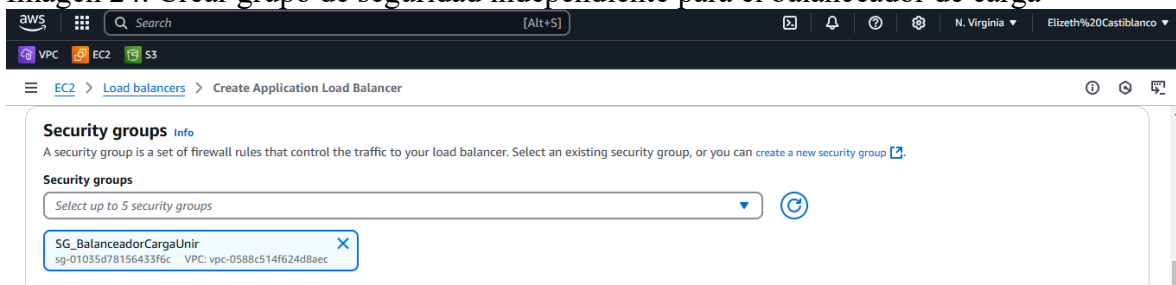


Imagen 25. Asignación de reglas (puertos desde donde se va a escuchar y para donde salen las instrucciones)

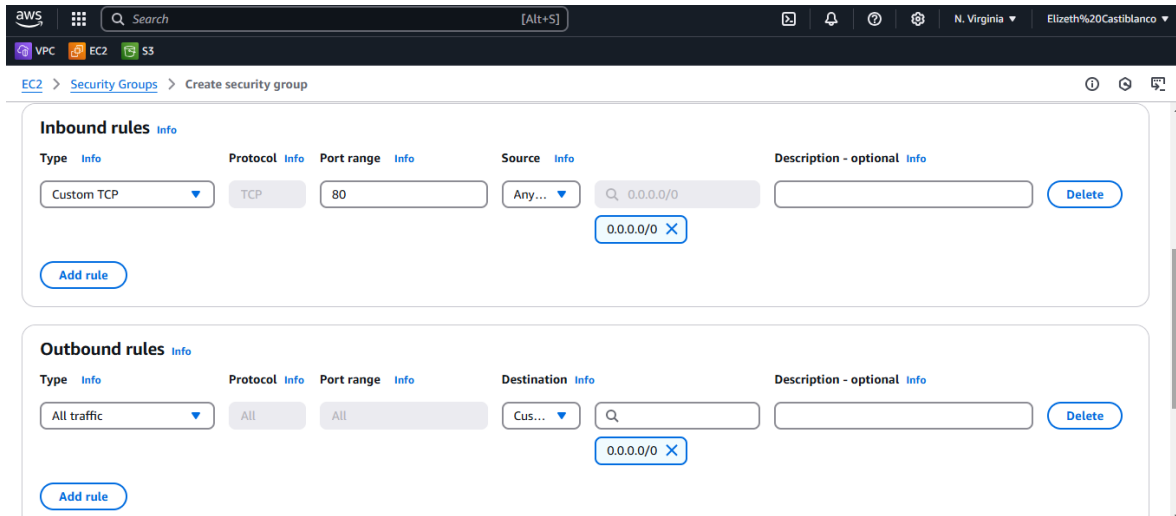


Imagen 26. Creación de grupo de objetivos-Target Group (cuando ingresan las peticiones se envían a los objetivos o máquinas de destino). Se elige la misma VPC y se deja la demás configuración que viene por defecto.

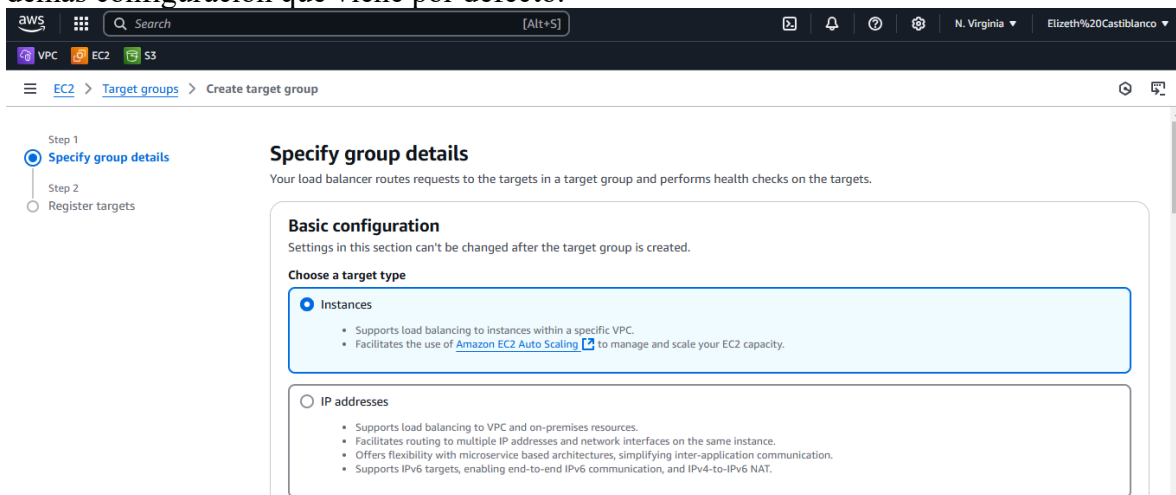


Imagen 27. Configuración del grupo de objetivos-Target Group

Target group name

TGSerUnir

A maximum of 32 alphanumeric characters including hyphens are allowed, but the name must not begin or end with a hyphen.

Protocol : Port

Choose a protocol for your target group that corresponds to the Load Balancer type that will route traffic to it. Some protocols now include anomaly detection for the targets and you can set mitigation options once your target group is created. This choice cannot be changed after creation

HTTP 80

1-65535

IP address type

Only targets with the indicated IP address type can be registered to this target group.

IPv4

Each instance has a default network interface (eth0) that is assigned the primary private IPv4 address. The instance's primary private IPv4 address is the one that will be applied to the target.

IPv6

Each instance you register must have an assigned primary IPv6 address. This is configured on the instance's default network interface (eth0). [Learn more](#)

VPC

Select the VPC with the instances that you want to include in the target group. Only VPCs that support the IP address type selected above are available in this list.

RemingtonVPC-vpc

vpc-0588c514f624d8aac

IPv4 VPC CIDR: 10.0.0.0/16

Imagen 28. Configuración del grupo de objetivos-Target Group

Protocol version

HTTP1

Send requests to targets using HTTP/1.1. Supported when the request protocol is HTTP/1.1 or HTTP/2.

HTTP2

Send requests to targets using HTTP/2. Supported when the request protocol is HTTP/2 or gRPC, but gRPC-specific features are not available.

gRPC

Send requests to targets using gRPC. Supported when the request protocol is gRPC.

Health checks

The associated load balancer periodically sends requests, per the settings below, to the registered targets to test their status.

Health check protocol

HTTP

Health check path

Use the default path of "/" to perform health checks on the root, or specify a custom path if preferred.

/

Up to 1024 characters allowed.

Imagen 29. Se agregan las instancias al grupo de objetivos

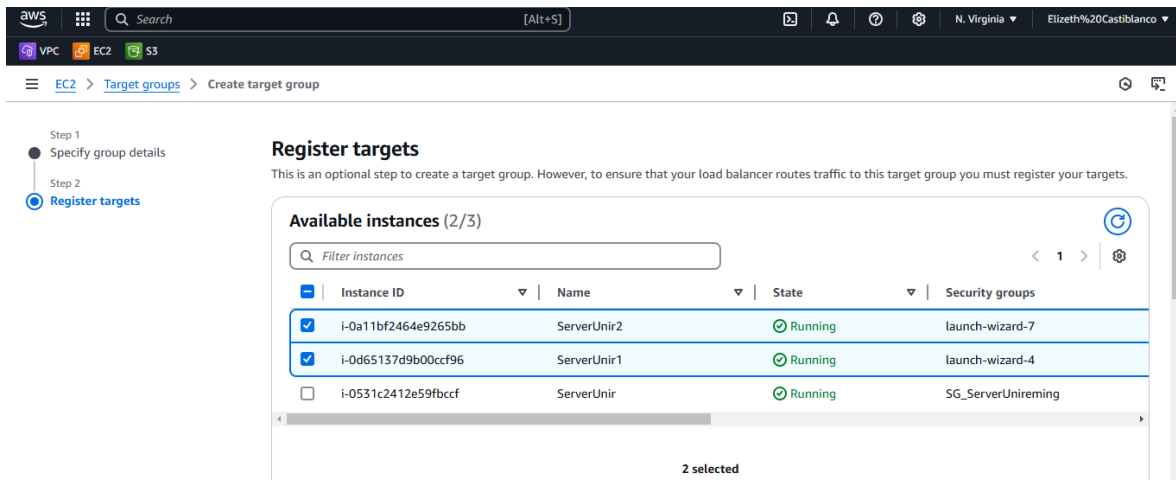


Imagen 30. Las instancias aparecen como objetivos

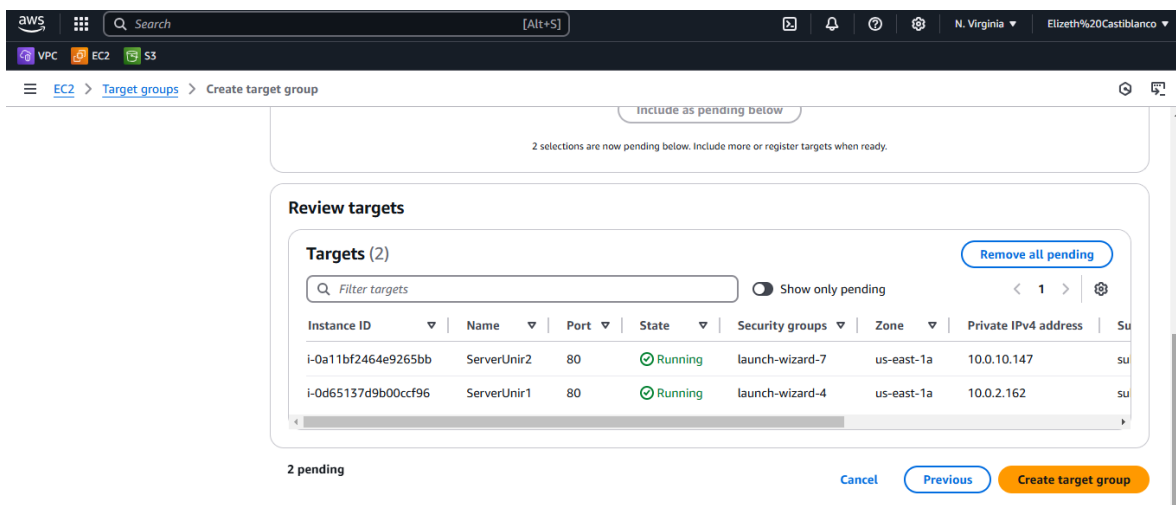


Imagen 31. Queda creado el Target Group y empieza a validar en la ruta que le indicamos (/) cada uno de los servidores e ira indicando si responden o no aumentando el número en el detalle Saludable y/o No Saludable.

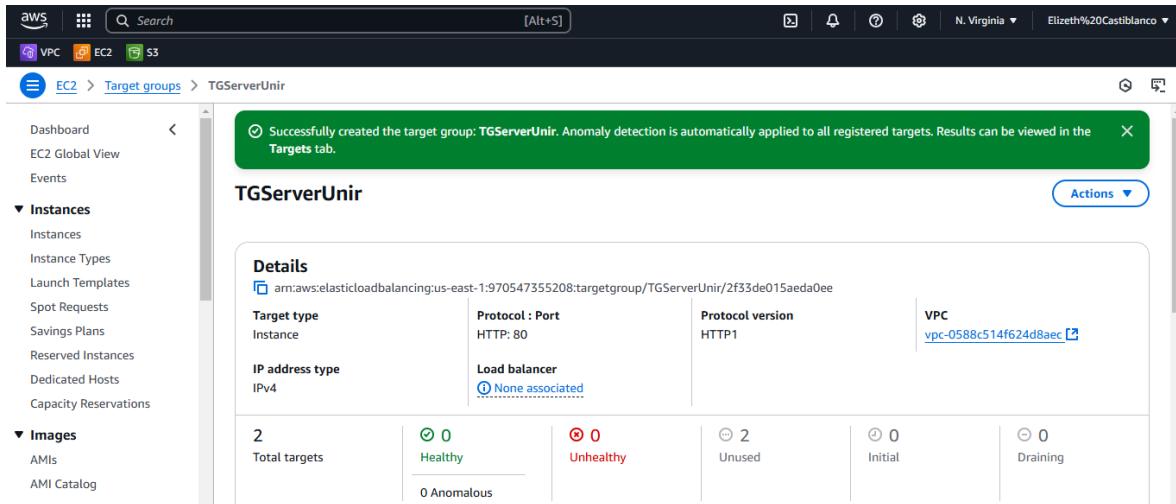


Imagen 32. Se elige el grupo de objetivos creado anteriormente y el resto de configuración se deja por defecto

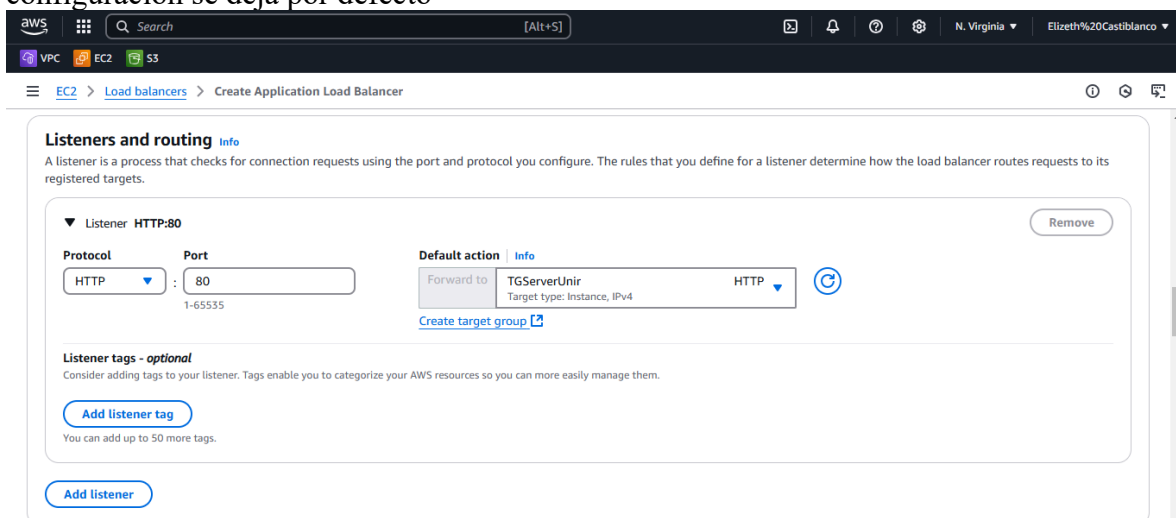


Imagen 33. Balanceador de carga inicializando

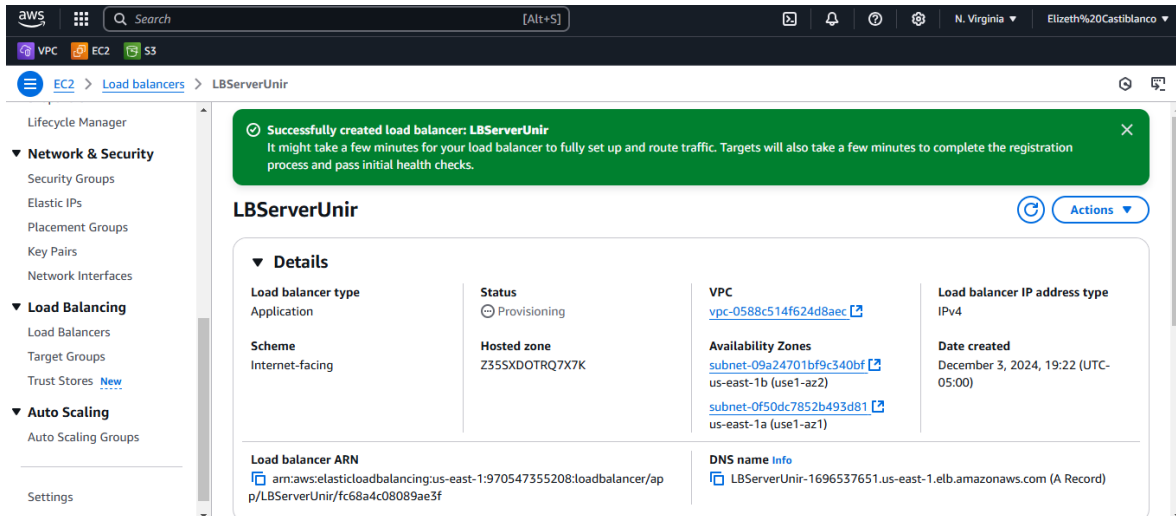


Imagen 34. Balanceador de carga activo

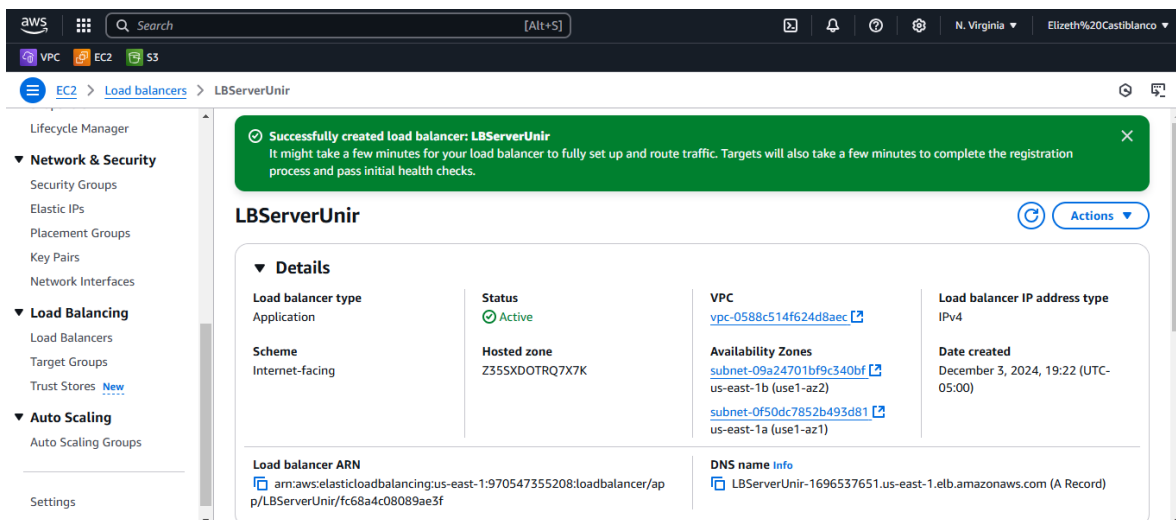


Imagen 35. Una vez el Balanceador de carga se activa el grupo de objetivos inicia a cargar el estado de las instancias, en este caso las dos están en estado Saludable.

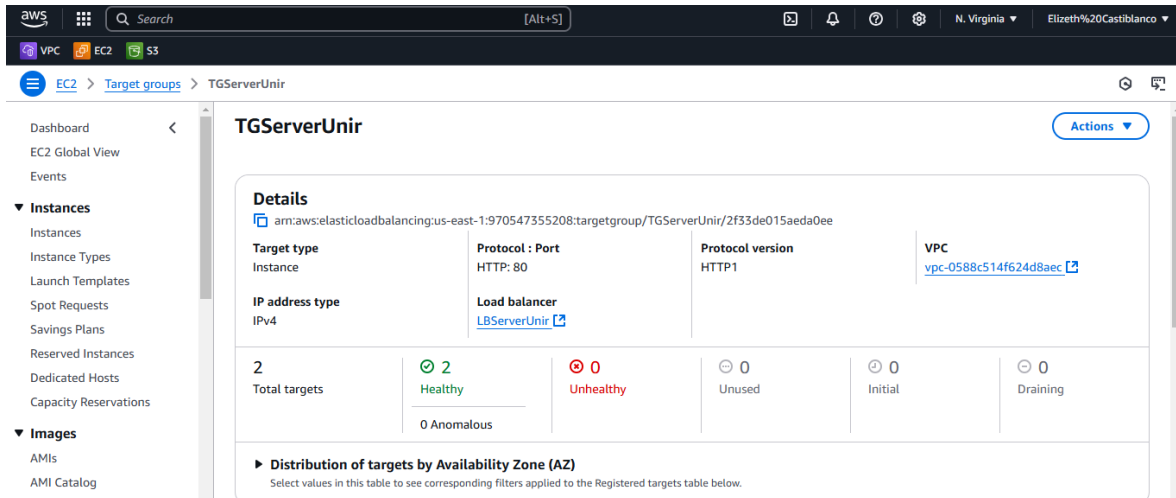


Imagen 36. Se ingresa ahora con el DNS del Balanceador de Carga (ya no con las IP de las instancias) al navegador y se evidencia que el servidor está funcionando correctamente.

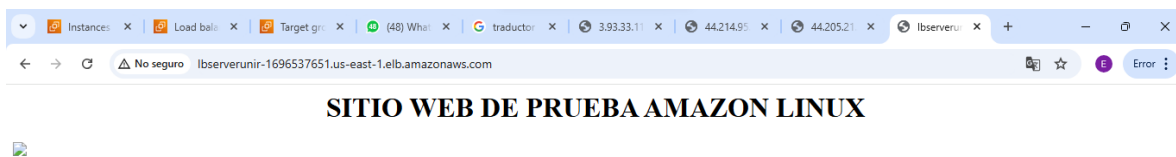


Imagen 37. Crear un Auto Scaling Group (cuando alguna de las instancias deje de funcionar por alguna razón se crean nuevas instancias automáticamente con la misma configuración de la AMI para que el servidor web no deje de responder en ningún momento).

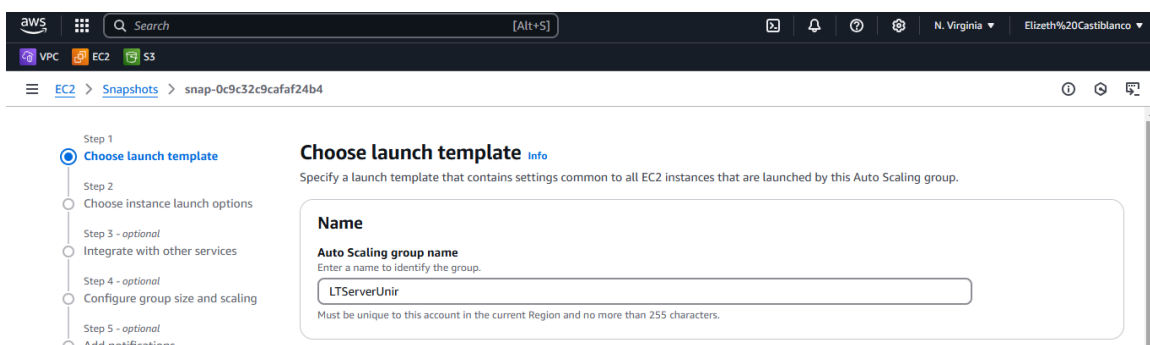


Imagen 38. Configuración del grupo de escalado automatico

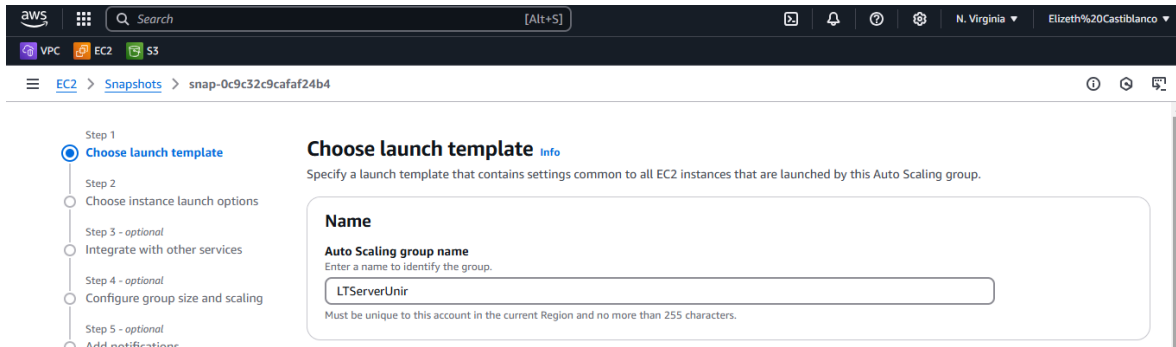


Imagen 39. Se elige la AMI creada al inicio

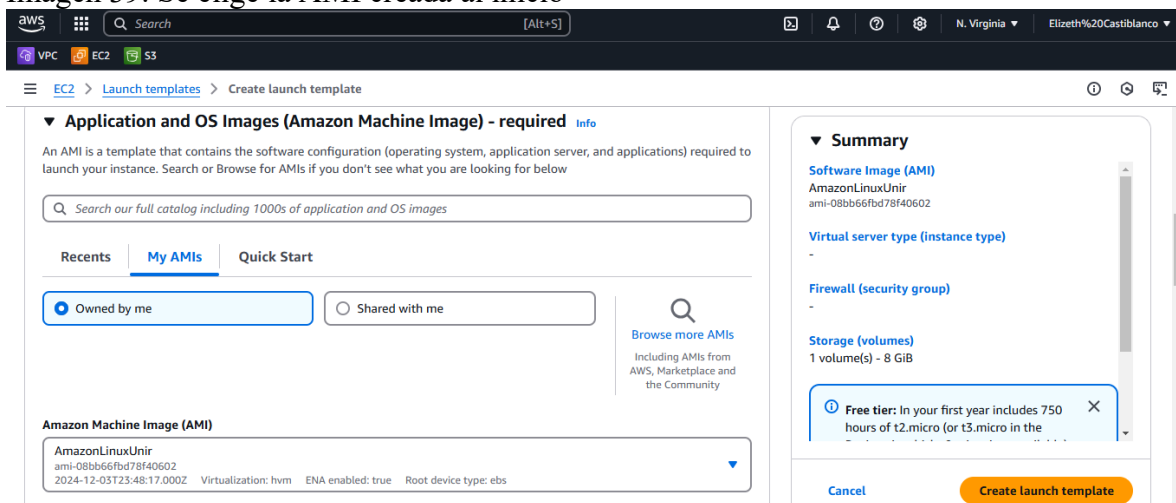


Imagen 40. Se configura el almacenamiento y se elige el mismo certificado de seguridad con el que se ha trabajado

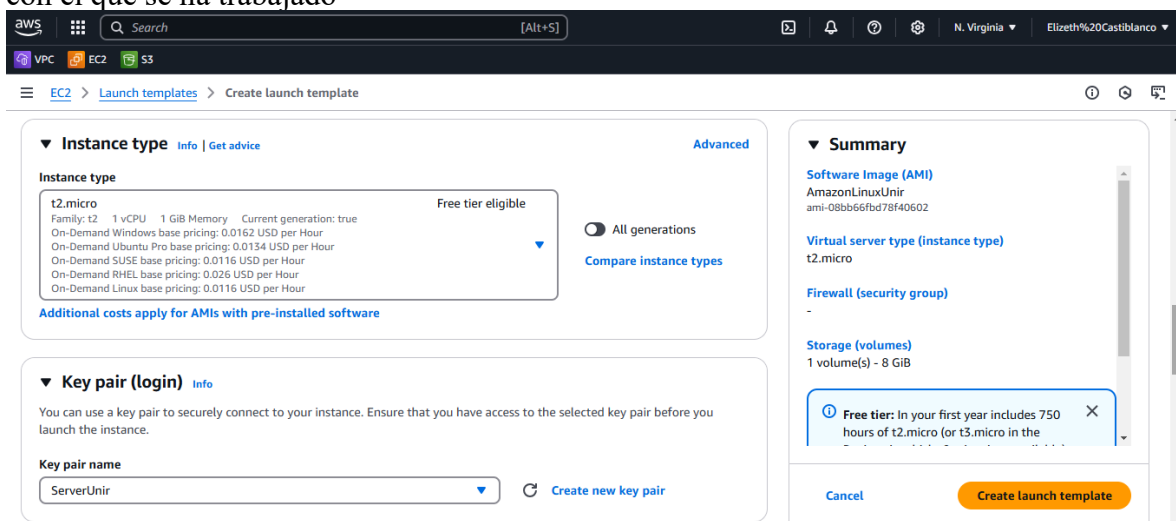


Imagen 41. Se elige el grupo de seguridad creado para el Balanceador de carga y se activan las IP publicas

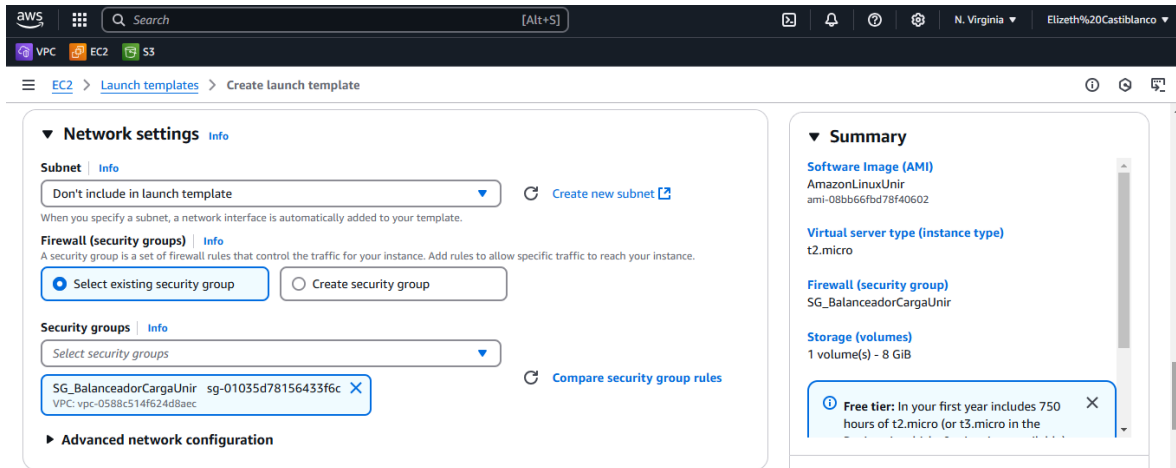


Imagen 42. Crear plantilla de lanzamiento

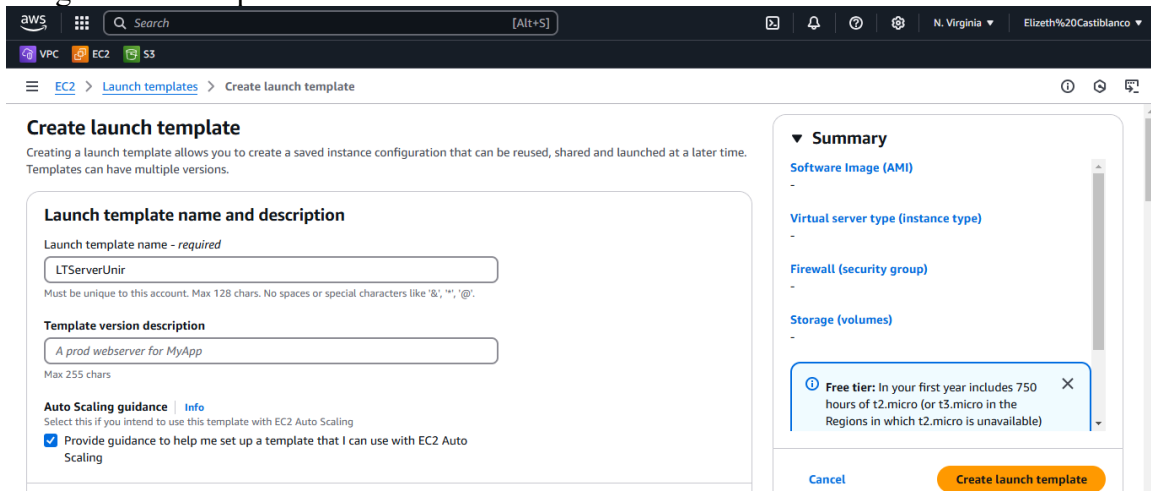


Imagen 43. Configuración plantilla de lanzamiento

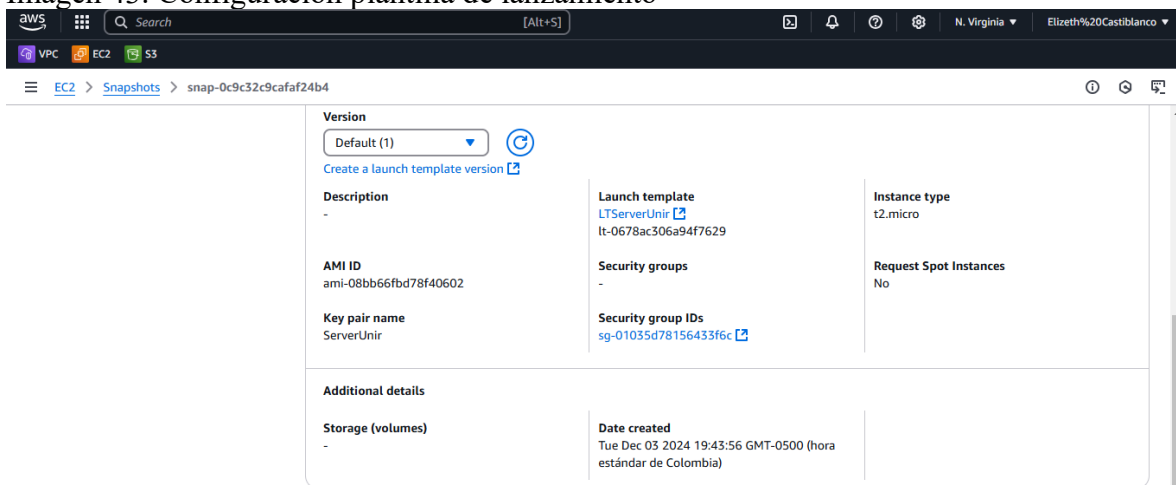


Imagen 44. Se elige la plantilla de lanzamiento creada para seguir configurando el grupo de escalado automático

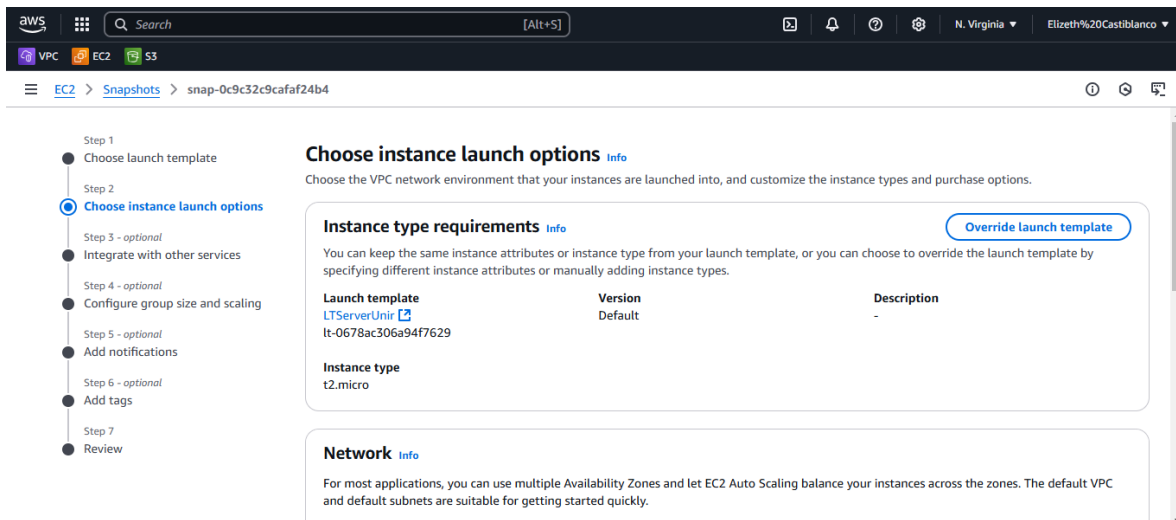


Imagen 45. Se configuran las dos zonas de disponibilidad de la VPC

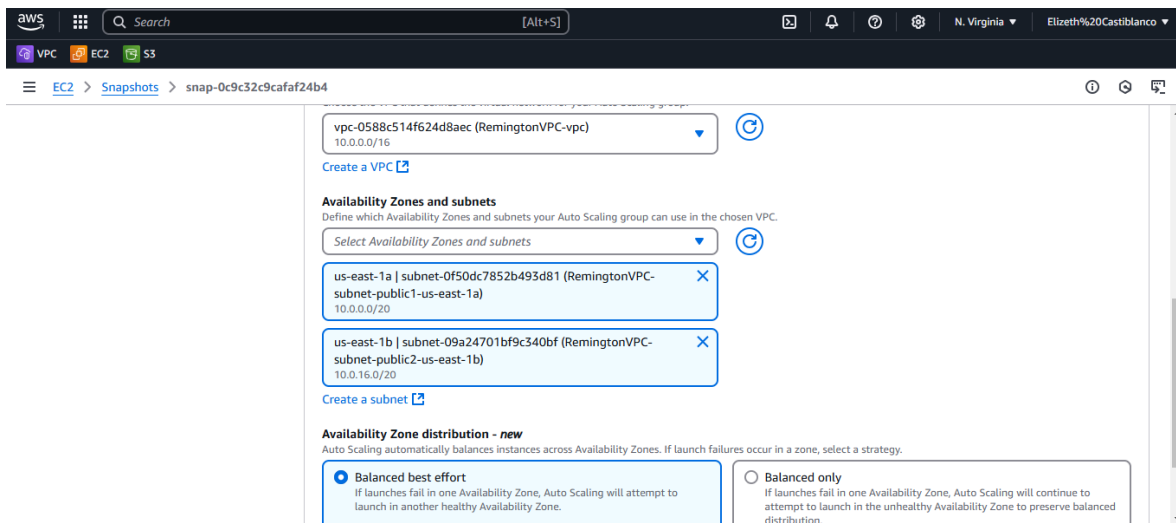


Imagen 46. Se continua con la configuración por defecto

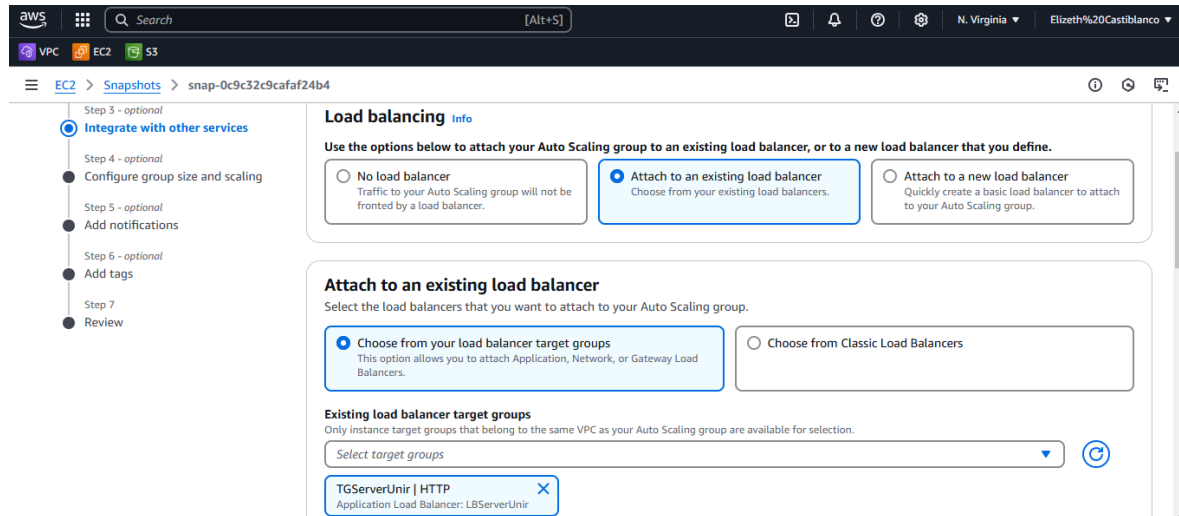


Imagen 47. Se configura la cantidad de instancias que se desea se creen cuando alguna deja de funcionar, en este caso será de mínimo 2 máximo 5 y deseadas 3

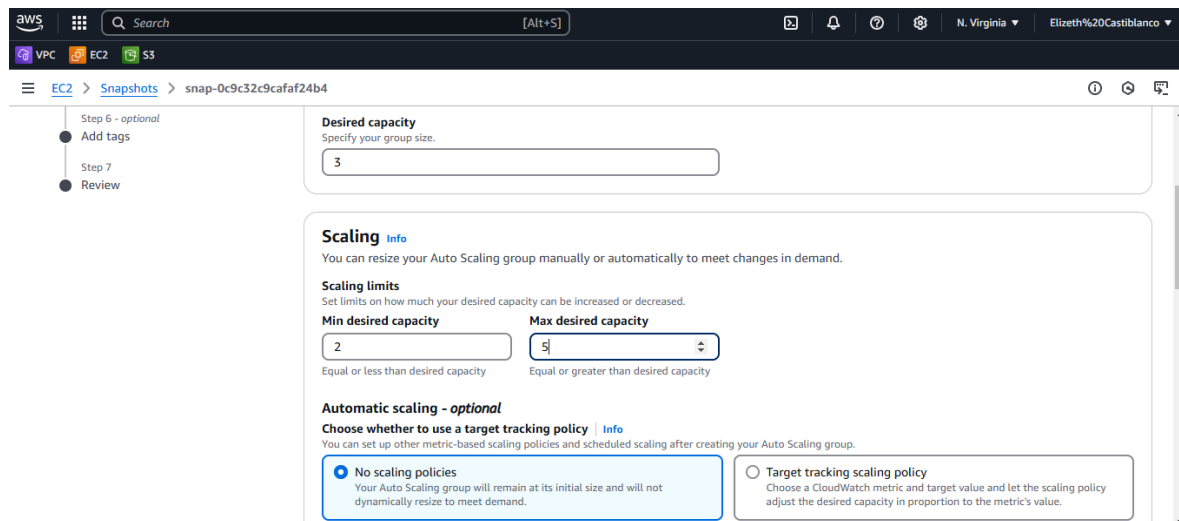


Imagen 48. Se continua con la configuración por defecto

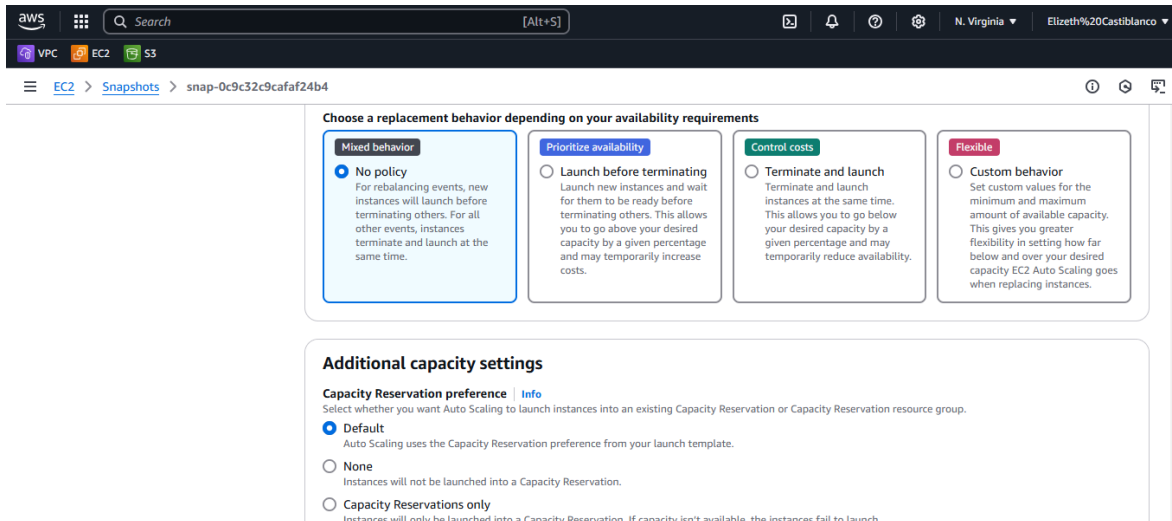


Imagen 49. Se continua con la configuración por defecto

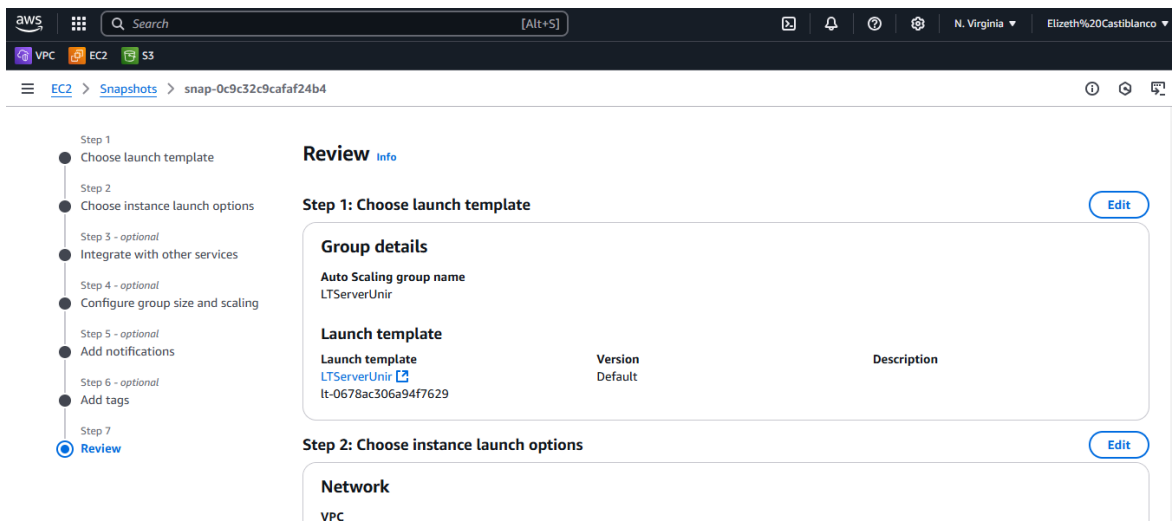


Imagen 50. Se crea el grupo de escalado automático con éxito

LTServerUnir created successfully

Auto Scaling groups (1) Info

Search your Auto Scaling groups

<input type="checkbox"/>	Name	Launch template/configuration	Instances	Status	Desired capacity	Min	Max	
<input type="checkbox"/>	LTServerUnir	LTServerUnir Version Default	0	Updating capacity...	3	2	5	u...

0 Auto Scaling groups selected

Select an Auto Scaling group

Imagen 51. Una vez creado el Auto Scaling Group, se crean automáticamente el número de instancias que se halla configurado (en este caso fueron tres)

Instances (6) Info

Find Instance by attribute or tag (case-sensitive)

<input type="checkbox"/>	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availa
<input type="checkbox"/>		i-00349166ce497a223	Running	t2.micro	2/2 checks passed	View alarms +	us-eas
<input type="checkbox"/>	ServerUnir	i-0531c2412e59fbccf	Running	t2.micro	2/2 checks passed	View alarms +	us-eas
<input type="checkbox"/>	ServerUnir1	i-0d65137d9b00ccf96	Running	t2.micro	2/2 checks passed	View alarms +	us-eas
<input type="checkbox"/>	ServerUnir2	i-0a11bf2464e9265bb	Running	t2.micro	2/2 checks passed	View alarms +	us-eas
<input type="checkbox"/>		i-0f17035e2e0834c2f	Running	t2.micro	2/2 checks passed	View alarms +	us-eas
<input type="checkbox"/>		i-01e3e1606e14ec499	Running	t2.micro	2/2 checks passed	View alarms +	us-eas

Select an instance

Imagen 52. El grupo de objetivos marca las 5 instancias en estado Saludable.

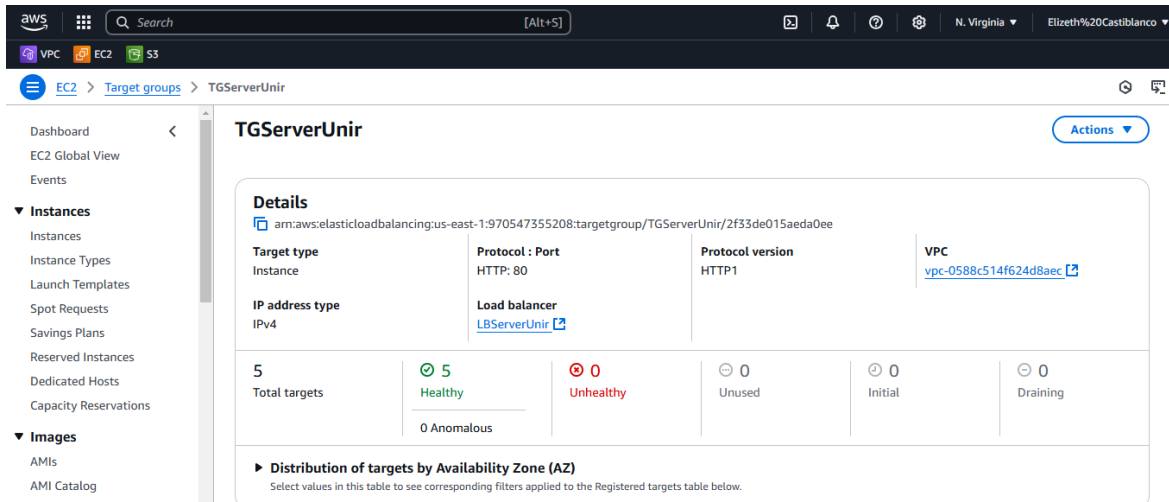


Imagen 53. Se hace la comprobación del Auto Scaling Groups para verificar que trabaje correctamente, por lo tanto, eliminamos una de las 5 instancias manualmente

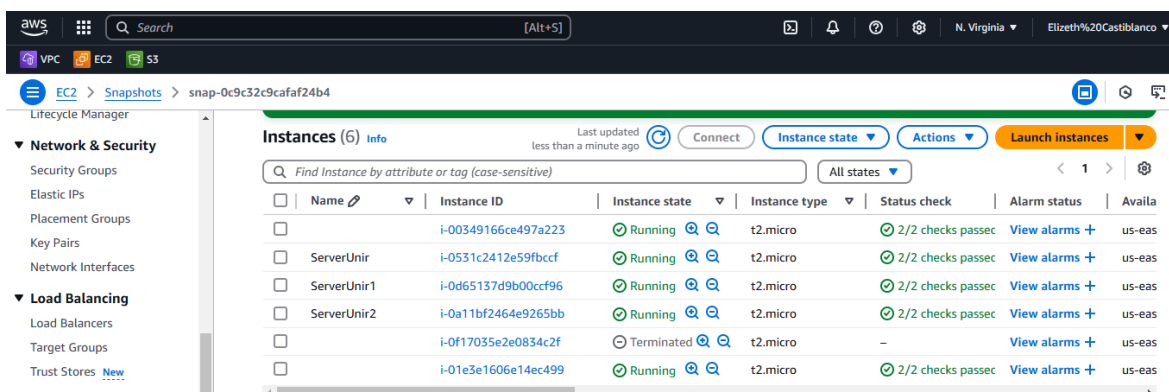


Imagen 54. En la actividad del Auto Scaling ya aparece la instancia eliminada

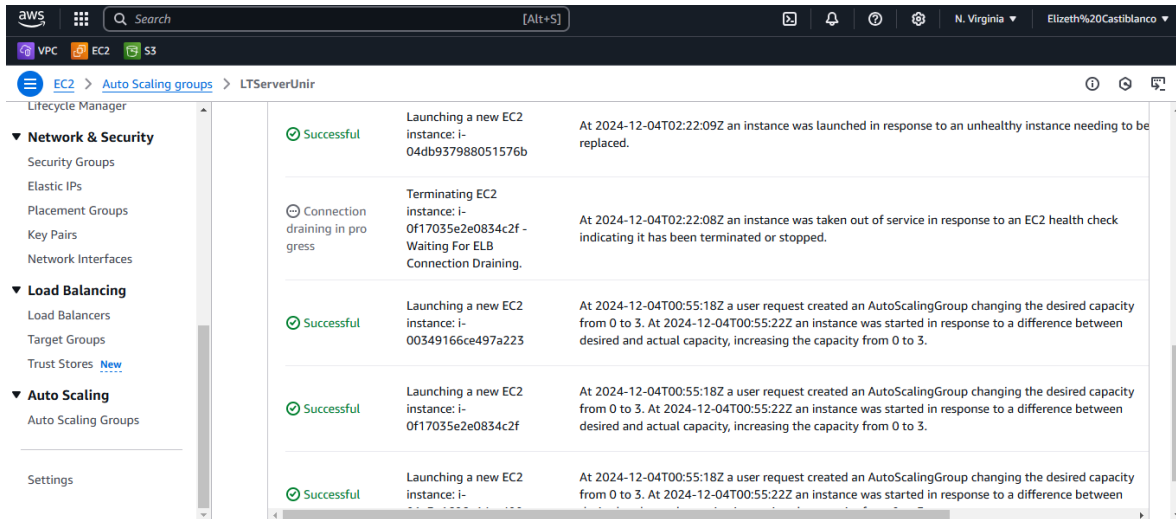
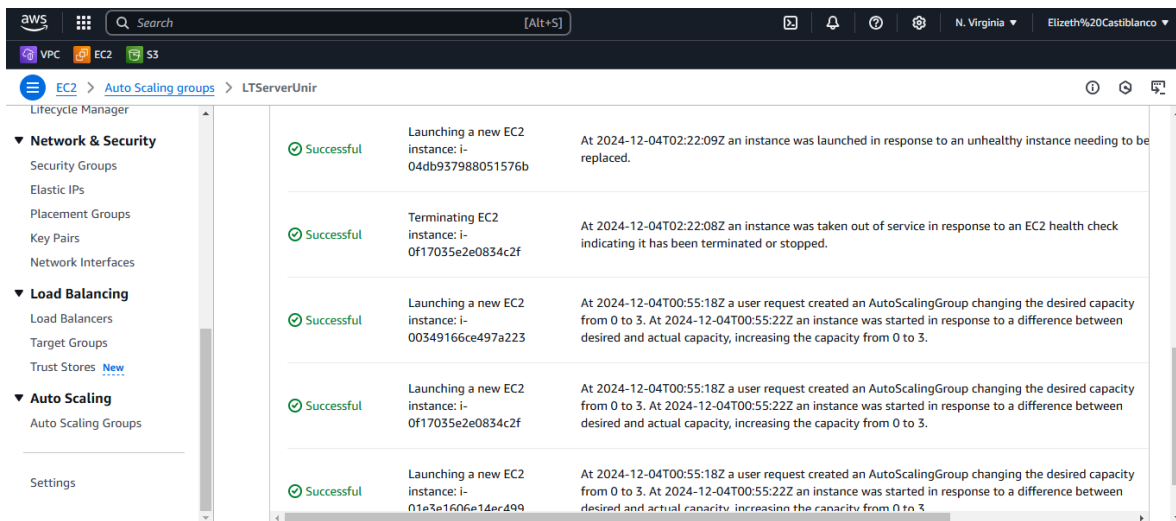


Imagen 55. Finalmente se crea una nueva instancia automáticamente completando siempre las 3 instancias deseadas



Video de explicación implementación de Balanceador de Carga en AWS

https://youtu.be/bddliaoIG_o

Implementación de S3

Imagen 1. S3 Almacenamiento instantáneo que me brinda objetos

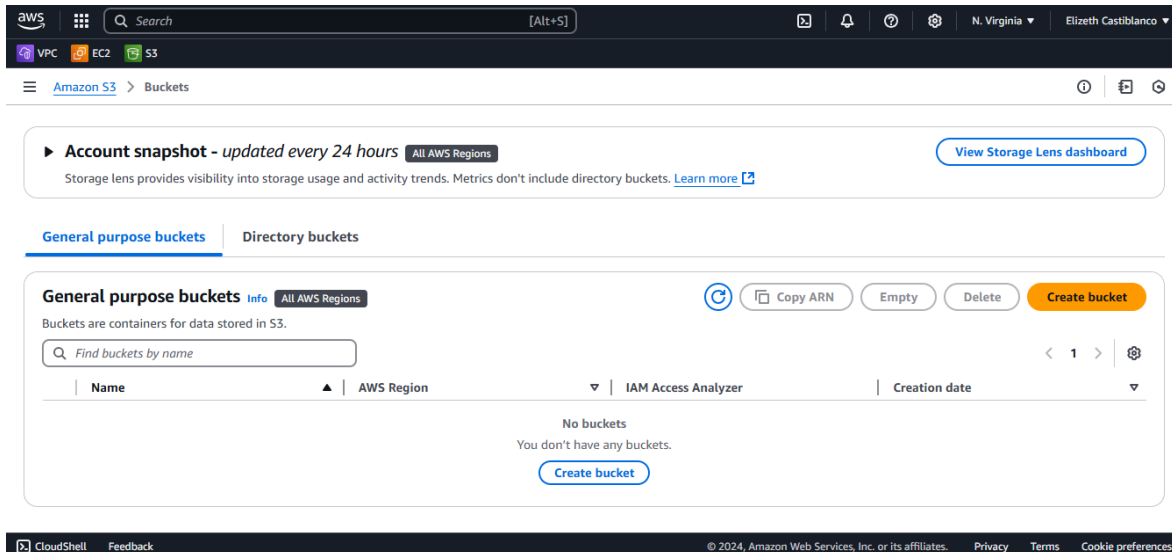


Imagen 2. Asignación de nombre

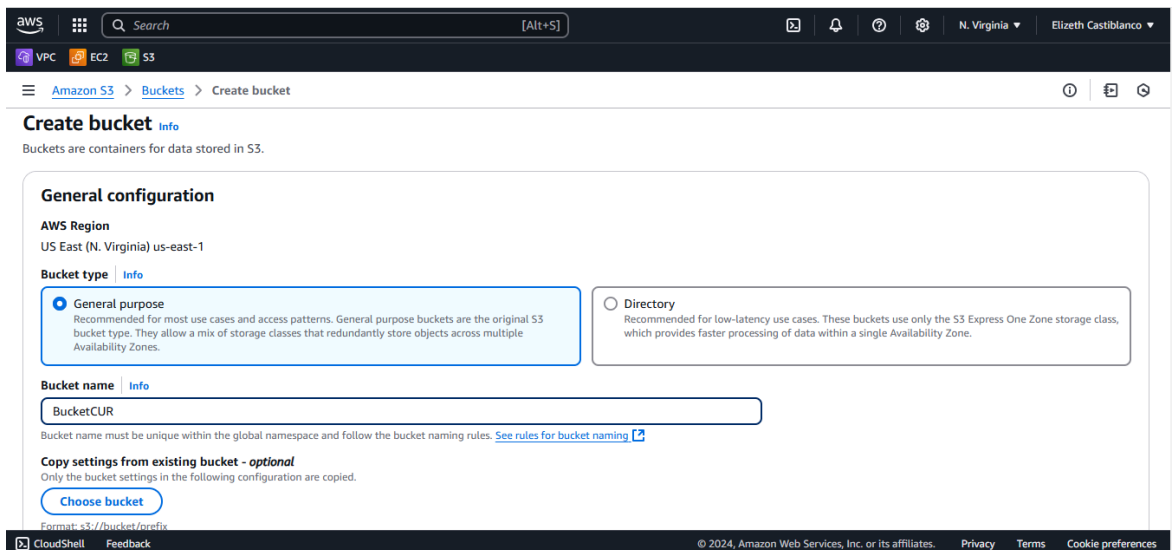


Imagen 3. Manejar los permisos y definir si los objetos que se están manejando son públicos o privados

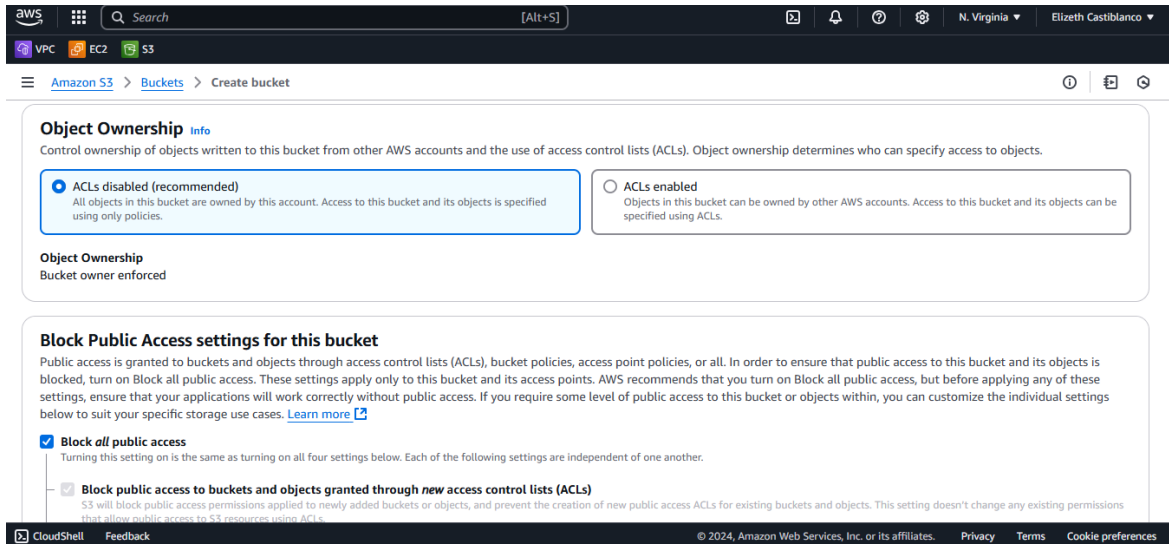


Imagen 4. Bloquear acceso publico

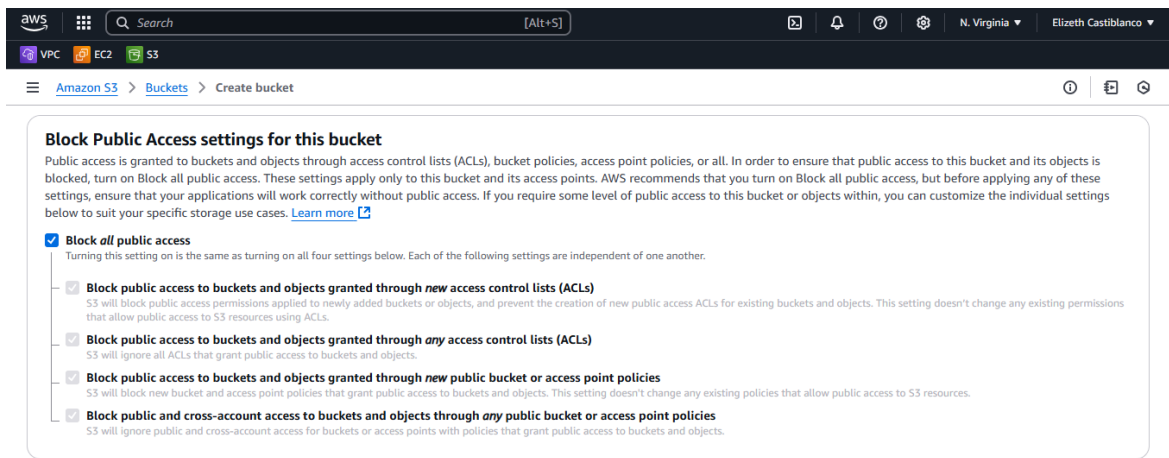


Imagen 5. Crear bucket

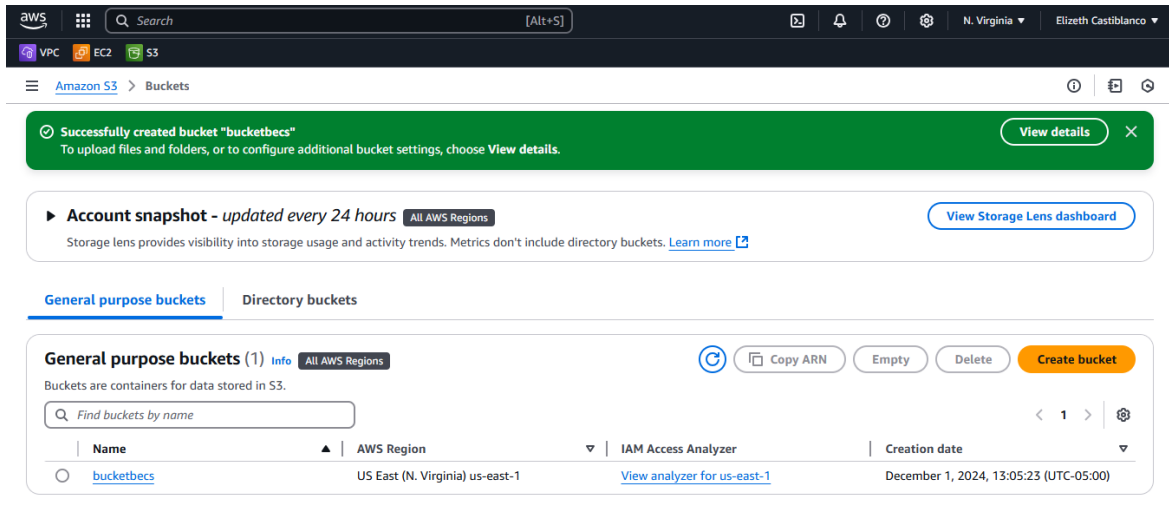


Imagen 6. Crear objetos

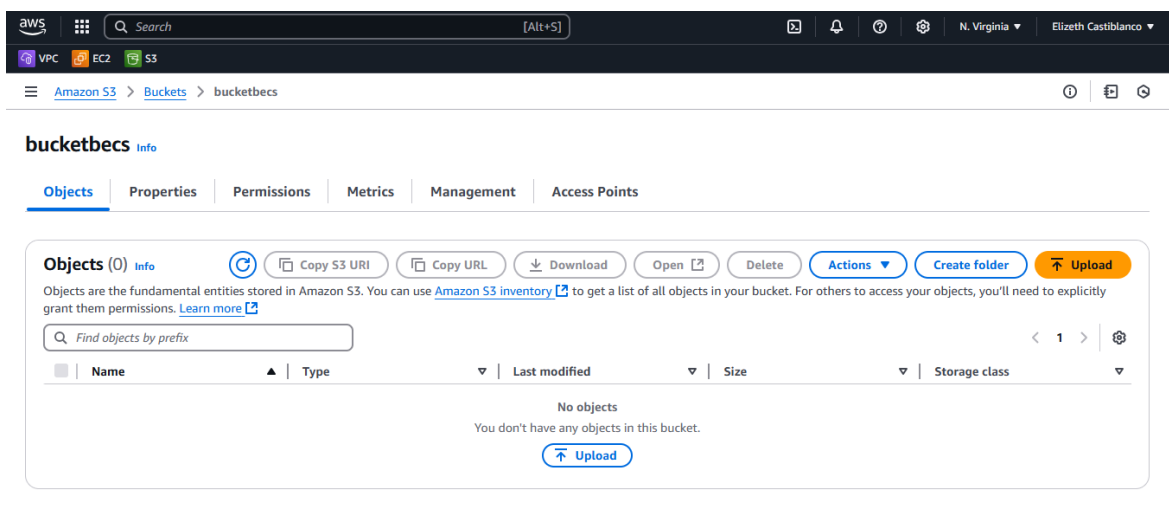


Imagen 7. Subir archivos como objetos

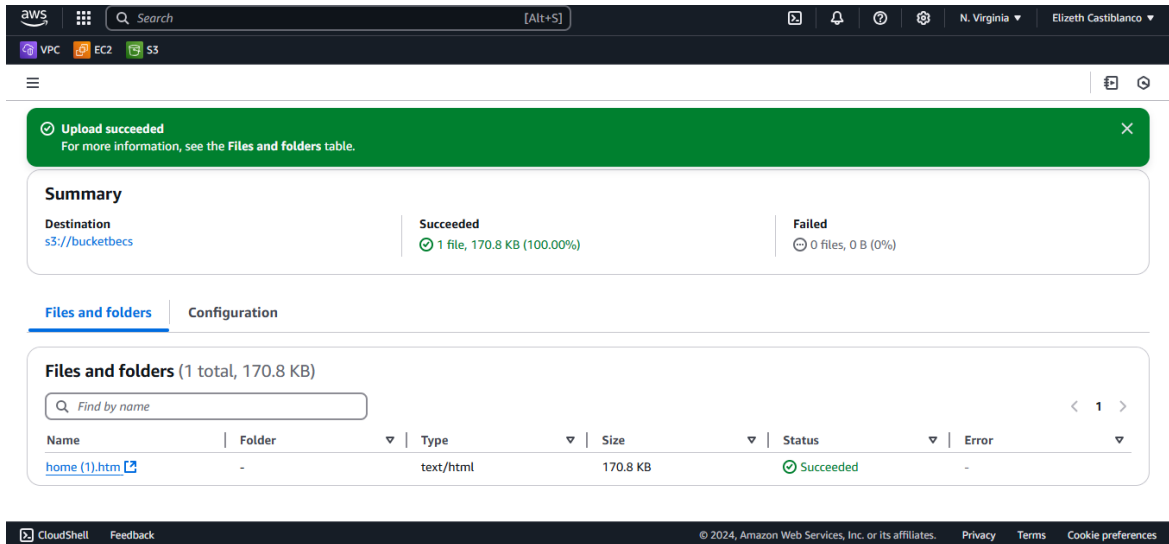


Imagen 8. Dentro de las características del objeto ejecutar la URL

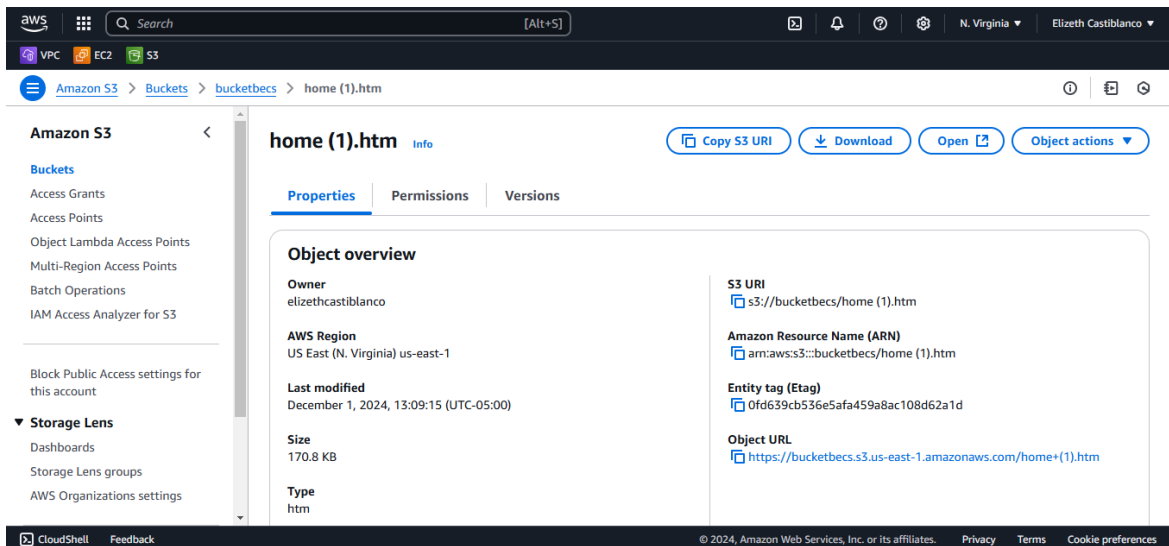


Imagen 9. Teniendo en cuenta que el objeto no tiene permiso de acceso público no se verá.

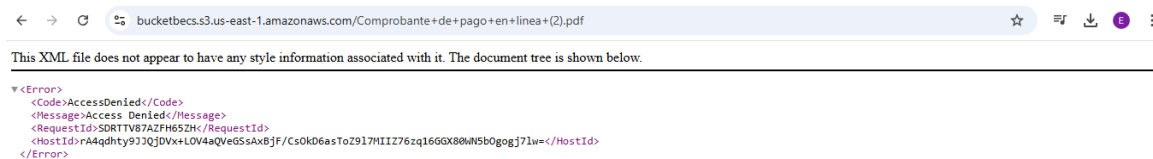


Imagen 10. Desbloquear el acceso público en el bucket

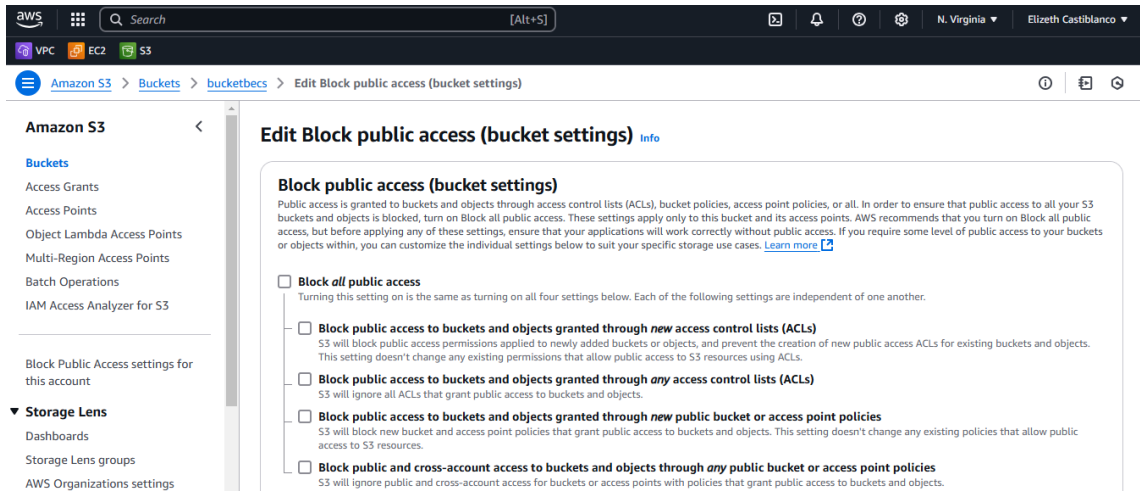


Imagen 11. Dar permisos de acceso público al objeto y ejecutar

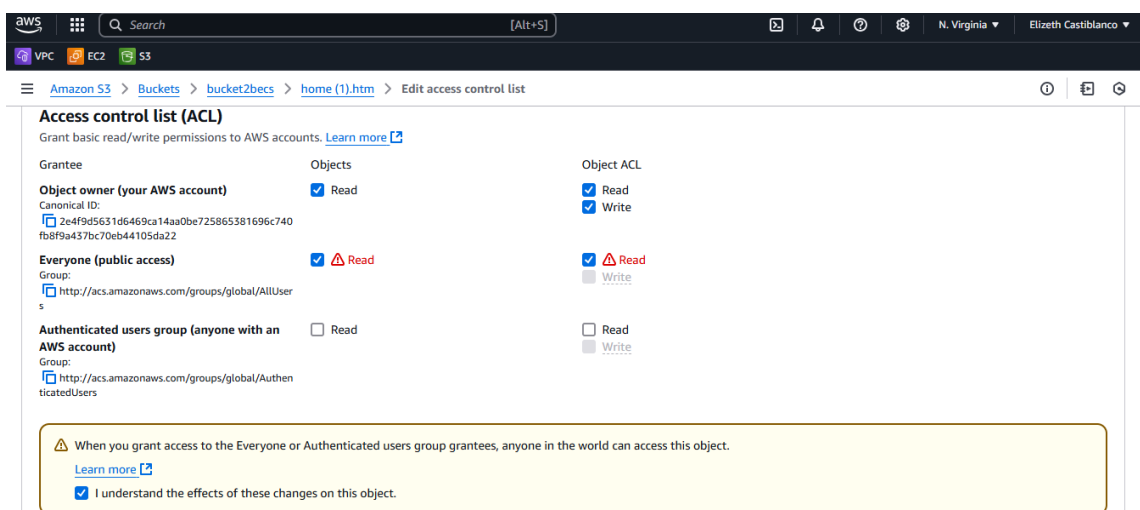


Imagen 12. Verificar la correcta ejecución

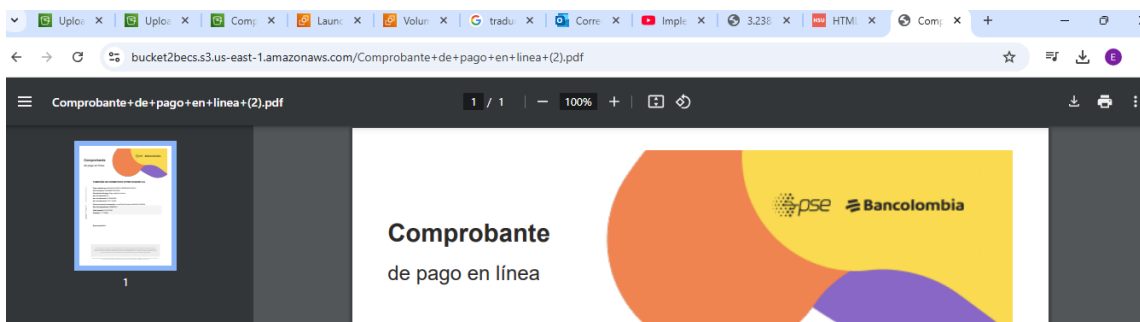


Imagen 13. Usar plantilla para crear como objetos

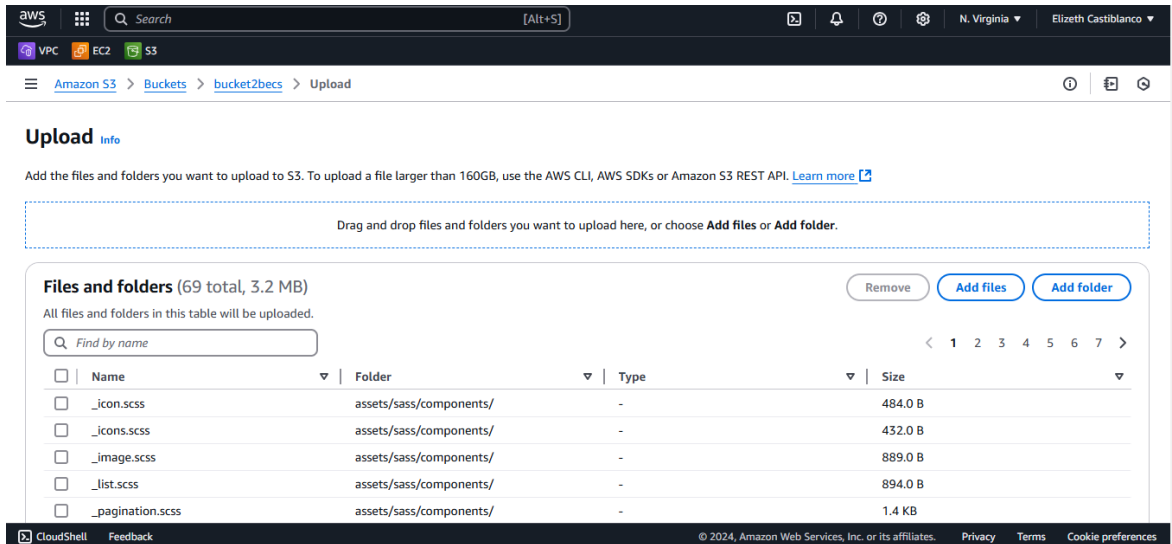


Imagen 14. Garantizar acceso público de lectura y subir

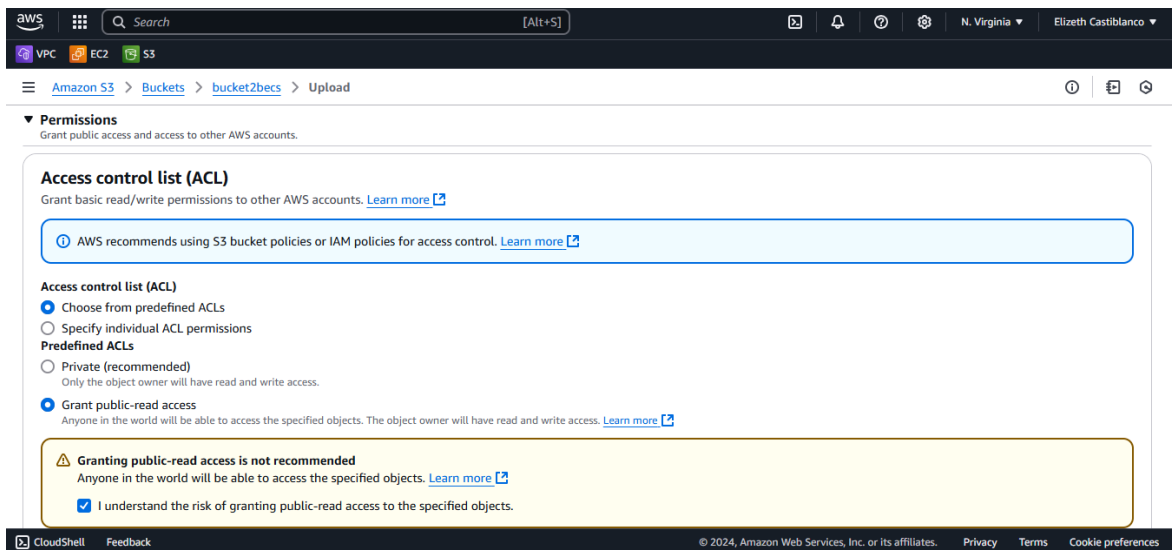
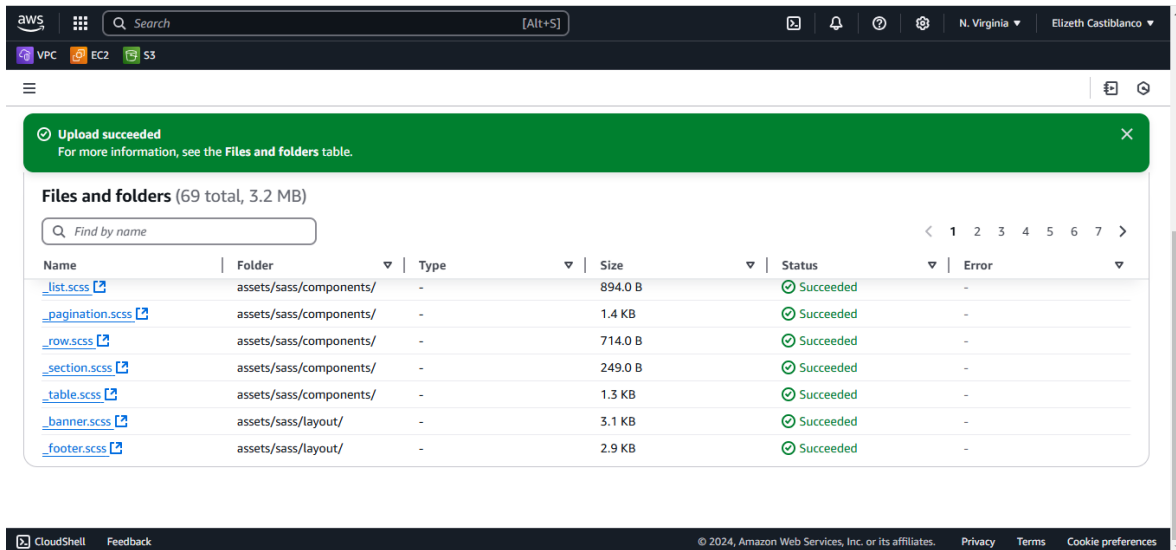


Imagen 15. Plantilla cargada correctamente



Upload succeeded
For more information, see the Files and folders table.

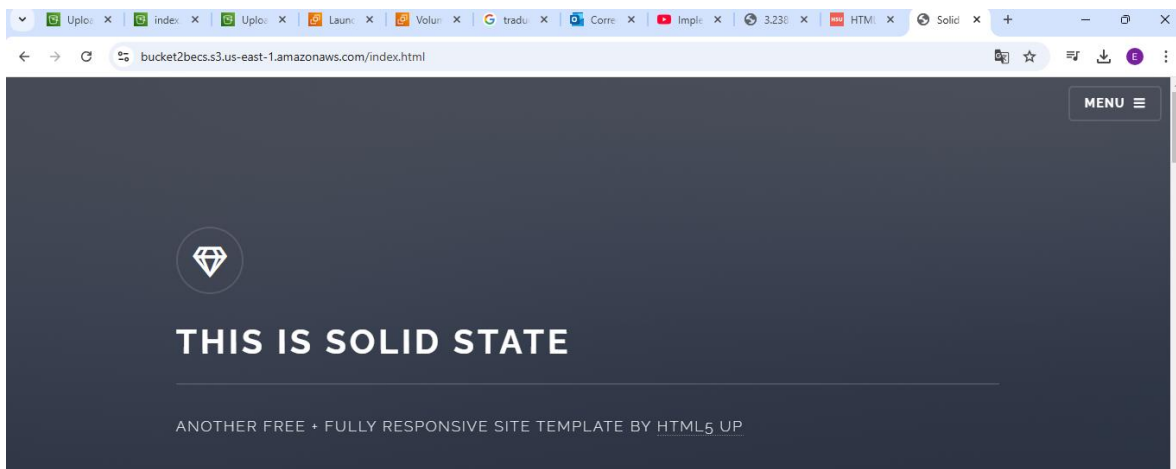
Files and folders (69 total, 3.2 MB)

Find by name

Name	Folder	Type	Size	Status	Error
_list.scss	assets/sass/components/	-	894.0 B	✔ Succeeded	-
_pagination.scss	assets/sass/components/	-	1.4 KB	✔ Succeeded	-
_row.scss	assets/sass/components/	-	714.0 B	✔ Succeeded	-
_section.scss	assets/sass/components/	-	249.0 B	✔ Succeeded	-
_table.scss	assets/sass/components/	-	1.3 KB	✔ Succeeded	-
_banner.scss	assets/sass/layout/	-	3.1 KB	✔ Succeeded	-
_footer.scss	assets/sass/layout/	-	2.9 KB	✔ Succeeded	-


CloudShell Feedback © 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

Imagen 16. Ejecutar URL y verificar



bucket2becs.s3.us-east-1.amazonaws.com/index.html

MENU



THIS IS SOLID STATE

ANOTHER FREE + FULLY RESPONSIVE SITE TEMPLATE BY [HTML5 UP](#)

Implementación de Auto Scaling Group y configuración de políticas

Imagen 1. Ahora que se tienen únicamente las instancias creadas automáticamente por el Auto Scaling Group (ya no tienen IP pública) se crea una nueva instancia con la misma configuración de las anteriores sin usar la AMI creada, para conectarse por ssh y administrar las demás.

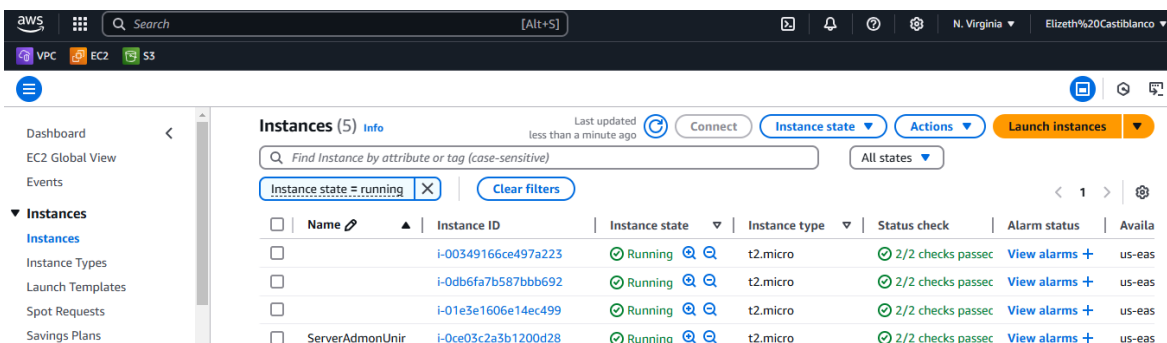


Imagen 2. Se crea una política dinámica de escalado dentro del Auto Scaling Group para que cuando el uso de CPU sea más o menos del 25% escale.

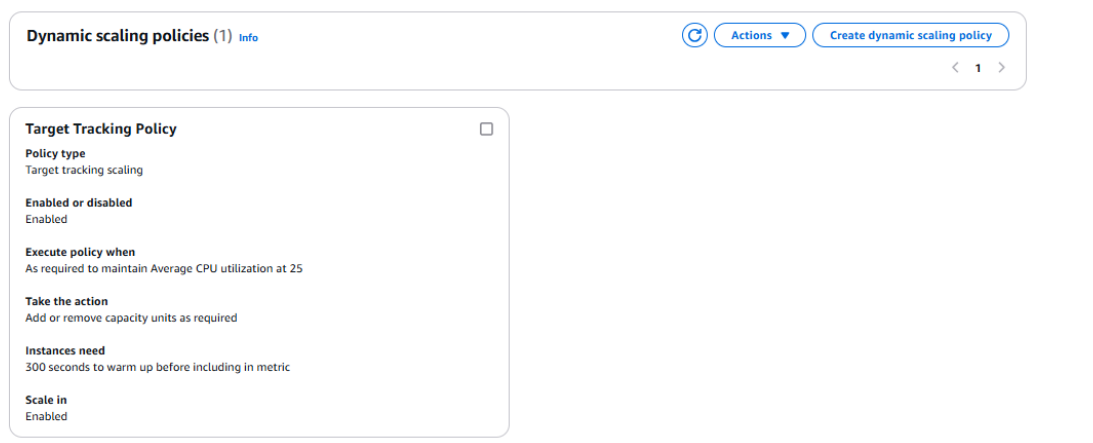


Imagen 3. Se conecta desde la instancia y se instala un administrador de recursos.


```

ec2-user@ip-10-0-28-122:~
Total download size: 183 k
Installed size: 432 k
Is this ok [y/N]: y
Downloading Packages:
htop-3.2.1-87.amzn2023.0.3.x86_64.rpm      3.3 MB/s | 183 kB    00:00
-----
Total                                1.8 MB/s | 183 kB    00:00
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
  Preparing      :                                1/1
  Installing    : htop-3.2.1-87.amzn2023.0.3.x86_64 1/1
  Running scriptlet: htop-3.2.1-87.amzn2023.0.3.x86_64 1/1
  Verifying     : htop-3.2.1-87.amzn2023.0.3.x86_64 1/1

Installed:
  htop-3.2.1-87.amzn2023.0.3.x86_64

Complete!
[root@ip-10-0-7-105 ec2-user]# htop
[root@ip-10-0-7-105 ec2-user]# ssh -i ec2-user@10.0.28.122
Warning: Identity file ec2-user@10.0.28.122 not accessible: No such file or dire

```

Imagen 6. Se dan los permisos al certificado

```

ec2-user@ip-10-0-28-122:~
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@                WARNING: UNPROTECTED PRIVATE KEY FILE!                @
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
Permissions 0644 for 'cert.PEM' are too open.
It is required that your private key files are NOT accessible by others.
This private key will be ignored.
Load key "cert.PEM": bad permissions
ec2-user@10.0.28.122: Permission denied (publickey,gssapi-keyex,gssapi-with-mic)
.
[root@ip-10-0-7-105 ec2-user]# chmod 777 cert.PEM
[root@ip-10-0-7-105 ec2-user]# chmod 400 cert.PEM
[root@ip-10-0-7-105 ec2-user]# ssh -i cert.PEM ec2-user@10.0.28.122

#
~\  ###_      Amazon Linux 2023
~~ \_###_\
~~  \###|
~~   \#/      https://aws.amazon.com/linux/amazon-linux-2023
~~
~~~
~~ \_#_
~~  /m/

Last login: Tue Dec  3 23:05:03 2024 from 103.219.234.219
[ec2-user@ip-10-0-28-122 ~]$

```

Imagen 7. Se instala una aplicación para subir el uso de CPU de una instancia y hacer la prueba de estrés

```

root@ip-10-0-7-105:/home/ec2-user
Preparing      :                               1/1
Installing     : htop-3.2.1-87.amzn2023.0.3.x86_64 1/1
Running scriptlet: htop-3.2.1-87.amzn2023.0.3.x86_64 1/1
Verifying      : htop-3.2.1-87.amzn2023.0.3.x86_64 1/1

Installed:
  htop-3.2.1-87.amzn2023.0.3.x86_64

Complete!
[root@ip-10-0-28-122 ec2-user]# ^C
[root@ip-10-0-28-122 ec2-user]# ^C
[root@ip-10-0-28-122 ec2-user]# ^C
[root@ip-10-0-28-122 ec2-user]# ^C
[root@ip-10-0-28-122 ec2-user]# ^C
[root@ip-10-0-28-122 ec2-user]# ^C
[root@ip-10-0-28-122 ec2-user]# ^C
[root@ip-10-0-28-122 ec2-user]# ^C
[root@ip-10-0-28-122 ec2-user]# exit
exit
[ec2-user@ip-10-0-28-122 ~]$ exit
logout
Connection to 10.0.28.122 closed.
[root@ip-10-0-7-105 ec2-user]# htop
[root@ip-10-0-7-105 ec2-user]# yum install stress

```

Imagen 8. Se ejecuta el comando para aumentar el rendimiento de CPU

```

root@ip-10-0-7-105:/home/ec2-user
Total download size: 34 k
Installed size: 68 k
Is this ok [y/N]: y
Downloading Packages:
stress-1.0.7-2.amzn2023.0.1.x86_64.rpm      635 kB/s | 34 kB    00:00
-----
Total                                      318 kB/s | 34 kB    00:00
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
  Preparing      :                               1/1
  Installing     : stress-1.0.7-2.amzn2023.0.1.x86_64 1/1
  Running scriptlet: stress-1.0.7-2.amzn2023.0.1.x86_64 1/1
  Verifying      : stress-1.0.7-2.amzn2023.0.1.x86_64 1/1

Installed:
  stress-1.0.7-2.amzn2023.0.1.x86_64

Complete!
[root@ip-10-0-7-105 ec2-user]# stress --cpu 4 --timeout 600
stress: info: [27157] dispatching hogs: 4 cpu, 0 io, 0 vm, 0 hdd

```

Imagen 9. Se verifica como el uso de CPU de la instancia aumento al 99.3%

```

root@ip-10-0-7-105:/home/ec2-user
top - 21:11:24 up 51 min, 2 users, load average: 2.59, 2.49, 1.60
Tasks: 106 total, 4 running, 102 sleeping, 0 stopped, 0 zombie
%Cpu(s): 99.3 us, 0.0 sy, 0.0 ni, 0.0 id, 0.0 wa, 0.0 hi, 0.0 si, 0.7 st
MiB Mem : 949.5 total, 525.3 free, 128.4 used, 295.8 buff/cache
MiB Swap: 0.0 total, 0.0 free, 0.0 used. 678.6 avail Mem

  PID USER      PR  NI  VIRT  RES  SHR  S  %CPU  %MEM    TIME+  COMMAND
 27468 root        20   0   3516   112    0  R   33.2   0.0   0:33.45 stress
 27470 root        20   0   3516   112    0  R   33.2   0.0   0:33.45 stress
 27469 root        20   0   3516   112    0  R   32.9   0.0   0:33.45 stress
   1 root        20   0 105724 16996 10508  S   0.0   1.7   0:00.84 systemd
   2 root        20   0     0     0     0  S   0.0   0.0   0:00.00 kthreadd
   3 root         0  -20     0     0     0  I   0.0   0.0   0:00.00 rcu_gp
   4 root         0  -20     0     0     0  I   0.0   0.0   0:00.00 rcu_part+
   5 root         0  -20     0     0     0  I   0.0   0.0   0:00.00 slub_fl+
   6 root         0  -20     0     0     0  I   0.0   0.0   0:00.00 netns
   8 root         0  -20     0     0     0  I   0.0   0.0   0:00.00 kworker+
  10 root         0  -20     0     0     0  I   0.0   0.0   0:00.00 mm_perc+
  11 root        20   0     0     0     0  I   0.0   0.0   0:00.00 rcu_tas+
  12 root        20   0     0     0     0  I   0.0   0.0   0:00.00 rcu_tas+
  13 root        20   0     0     0     0  I   0.0   0.0   0:00.00 rcu_tas+
  14 root        20   0     0     0     0  S   0.0   0.0   0:00.14 ksofttir+
  15 root        20   0     0     0     0  I   0.0   0.0   0:00.06 rcu_pre+
  16 root        rt   0     0     0     0  S   0.0   0.0   0:00.01 migrati+

```

Imagen 10. Se verifica en la actividad del Auto Scaling Group que el proceso de detención de la instancia inicia

Event Type	Instance ID	Description	Start Time	End Time
Terminating EC2 instance	i-0f38bcbcb1cb8dc79	At 2024-12-08T21:08:29Z a monitor alarm TargetTracking-AGServerUnir-AlarmLow-08a46a2-bd68-4b8b-ab24-5a05f96c253 in state ALARM triggered policy Target Tracking Policy changing the desired capacity from 3 to 2. At 2024-12-08T21:08:37Z 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 2024-12-08T21:08:37Z instance i-0f38bcbcb1cb8dc79 was selected for termination.	2024 December 08, 04:08:37 PM -05:00	
Successful	i-036f9c43f55229e8	At 2024-12-08T20:07:47Z an instance was launched in response to an unhealthy instance needing to be replaced.	2024 December 08, 03:07:49 PM -05:00	2024 December 08, 03:07:49 PM -05:00
Successful	i-0318f19703d613054	At 2024-12-08T20:07:47Z an instance was taken out of service in response to an EC2 health check indicating it has been terminated or stopped.	2024 December 08, 03:07:47 PM -05:00	2024 December 08, 03:12 PM -05:00
Successful	i-0f38bcbcb1cb8dc79	At 2024-12-08T20:03:32Z a user request created an AutoScalingGroup changing the desired capacity from 0 to 3. At 2024-12-08T20:03:37Z an instance was started in response to a difference between desired and actual capacity, increasing the capacity from 0 to 3.	2024 December 08, 03:05:39 PM -05:00	2024 December 08, 03:03 PM -05:00
Successful	i-0318f19703d613054	At 2024-12-08T20:03:32Z a user request created an AutoScalingGroup changing the desired capacity from 0 to 3. At 2024-12-08T20:03:37Z an instance was started in response to a difference between desired and actual capacity, increasing the capacity from 0 to 3.	2024 December 08, 03:03:39 PM -05:00	2024 December 08, 03:03 PM -05:00
Successful	i-0d19e25b9fdb4126	At 2024-12-08T20:03:32Z a user request created an AutoScalingGroup changing the desired capacity from 0 to 3. At 2024-12-08T20:03:37Z an instance was started in response to a difference between desired and actual capacity, increasing the capacity from 0 to 3.	2024 December 08, 03:03:39 PM -05:00	2024 December 08, 03:03 PM -05:00

Imagen 11. Se verifica en el Target Group que ya dejó de funcionar una instancia

The screenshot shows the AWS Management Console interface for a Target Group named "TGServerUnir". The left sidebar contains navigation options like Dashboard, EC2 Global View, Events, Instances, and Images. The main content area shows the "Details" for the target group, including its ARN, target type (Instance), protocol (HTTP: 80), IP address type (IPv4), and VPC ID. Below the details, there is a summary of target health: 5 total targets, 4 healthy, 0 unhealthy, 0 anomalous, 0 unused, 0 initial, and 1 draining. A section titled "Distribution of targets by Availability Zone (AZ)" is also visible.

Imagen 12. Se verifica en el rendimiento de la instancia como el uso de CPU aumentó

The screenshot displays the "Instances" page in the AWS Management Console. A table lists several instances, with "ServerUnirAdmon" (Instance ID: i-07ee4659358a2133f) selected. Below the table, performance metrics for this instance are shown in four graphs: CPU utilization (%), Network in (bytes), Network out (bytes), and Network packets in (count). The CPU utilization graph shows a significant spike to 88.7% at 21:10. The other graphs show network activity over the same period.

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	P
	i-0f38bcbcb1cb8dc79	Running	t2.micro	2/2 checks passee	View alarms +	us-east-1b	ec2-54-237-194-78.co...	5
	i-036fc9c43f55229e8	Running	t2.micro	2/2 checks passee	View alarms +	us-east-1b	ec2-3-90-6-90.compute...	3
ServerUnir	i-0531c2412e59fbccf	Running	t2.micro	2/2 checks passee	View alarms +	us-east-1a	ec2-3-93-33-11.comput...	3
ServerUnirAdmon	i-07ee4659358a2133f	Running	t2.micro	2/2 checks passee	View alarms +	us-east-1a	ec2-34-237-144-145.co...	3
ServerUnir2	i-083de775d2ba7bbf4	Running	t2.micro	2/2 checks passee	View alarms +	us-east-1a	ec2-3-215-174-90.com...	3
	i-0d19e25b9fdba4126	Running	t2.micro	2/2 checks passee	View alarms +	us-east-1a	ec2-44-193-2-245.com...	4
ServerUnir1	i-0fab895cc440b4cb6	Running	t2.micro	2/2 checks passee	View alarms +	us-east-1a	ec2-44-200-157-90.co...	4

Imagen 13. Nuevamente en la actividad de la instancia se verifica que el proceso de finalización de la instancia terminó

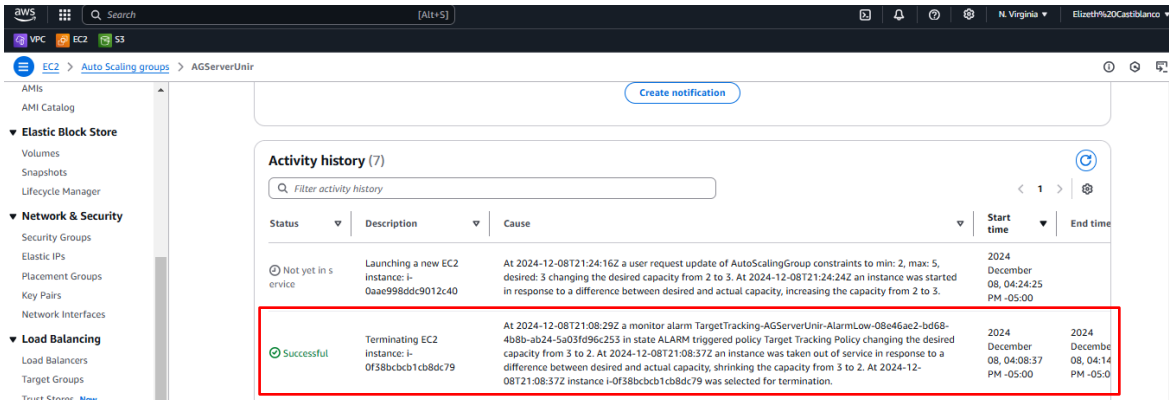


Imagen 14. Inicia automáticamente una nueva instancia en remplazo de la que dejo de funcionar por uso de CPU mayor al 25%

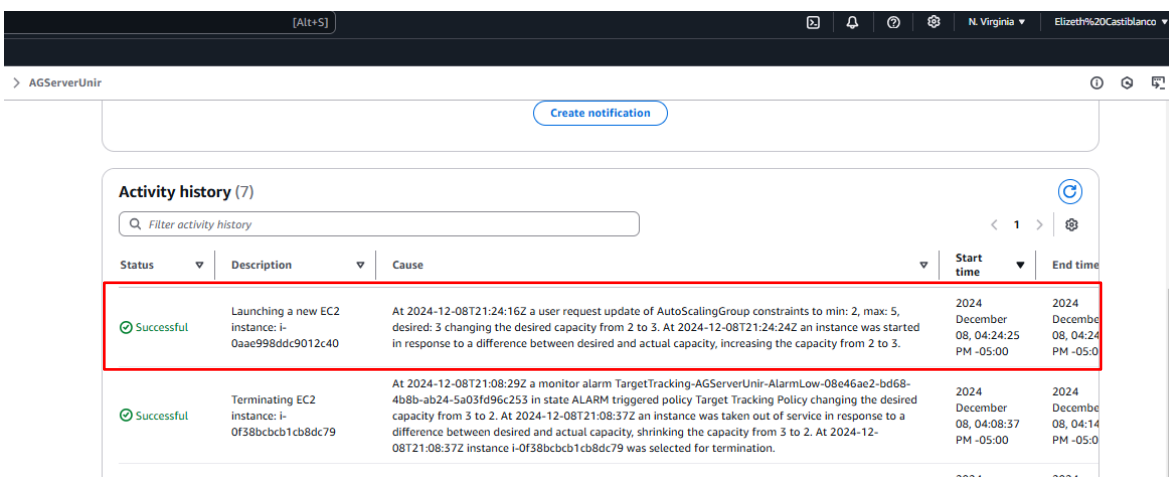
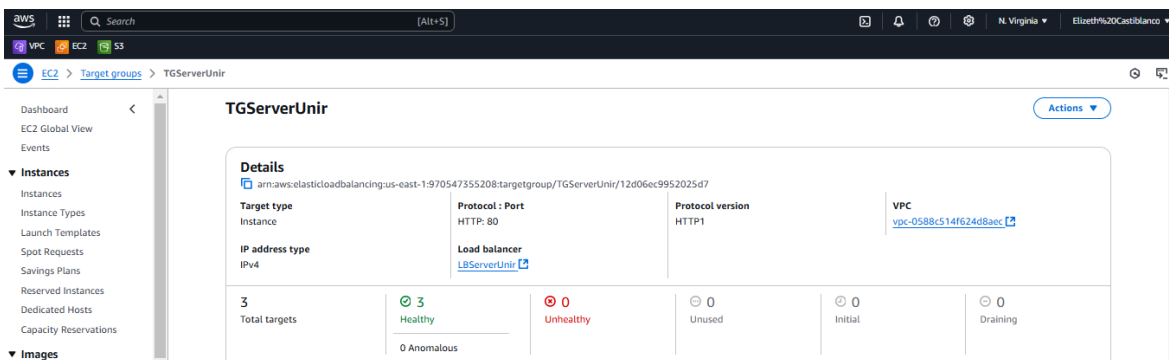


Imagen 15. Finalmente quedan las tres instancias que deben existir por defecto (según la configuración que se hizo) en estado saludable



Entrega III


```

root@ip-10-0-3-1:/home/ec2-user
Status: "Total requests: 8388; Idle/Busy workers 100/0;Requests/sec: 0.016
Tasks: 230 (limit: 1111)
Memory: 25.0M
CPU: 5min 51.872s
CGroup: /system.slice/httpd.service
├─ 25934 /usr/sbin/httpd -DFOREGROUND
├─ 333341 /usr/sbin/httpd -DFOREGROUND
├─ 333342 /usr/sbin/httpd -DFOREGROUND
├─ 333343 /usr/sbin/httpd -DFOREGROUND
├─ 333344 /usr/sbin/httpd -DFOREGROUND
└─ 347365 /usr/sbin/httpd -DFOREGROUND

Dec 03 23:05:54 ip-10-0-3-1.ec2.internal systemd[1]: Starting httpd.service - T
Dec 03 23:05:54 ip-10-0-3-1.ec2.internal systemd[1]: Started httpd.service - Th
Dec 03 23:05:54 ip-10-0-3-1.ec2.internal httpd[25934]: Server configured, liste
Dec 08 00:00:09 ip-10-0-3-1.ec2.internal systemd[1]: Reloading httpd.service - >
Dec 08 00:00:09 ip-10-0-3-1.ec2.internal httpd[25934]: Server configured, liste
Dec 08 00:00:09 ip-10-0-3-1.ec2.internal systemd[1]: Reloaded httpd.service - T
lines 2-24/24 (END)

[root@ip-10-0-3-1 ec2-user]# systemctl disable httpd
Removed "/etc/systemd/system/multi-user.target.wants/httpd.service".
[root@ip-10-0-3-1 ec2-user]# systemctl stop httpd
[root@ip-10-0-3-1 ec2-user]# dnf install nginx -y

```

Imagen 4. Se verifica el estado del servicio nginx (en este caso esta inactivo)

```

root@ip-10-0-3-1:/home/ec2-user
Installing      : nginx-core-1:1.26.2-1.amzn2023.0.1.x86_64      5/6
Installing      : nginx-1:1.26.2-1.amzn2023.0.1.x86_64      6/6
Running scriptlet: nginx-1:1.26.2-1.amzn2023.0.1.x86_64      6/6
Verifying       : gperftools-libs-2.9.1-1.amzn2023.0.3.x86_64  1/6
Verifying       : libunwind-1.4.0-5.amzn2023.0.2.x86_64      2/6
Verifying       : nginx-1:1.26.2-1.amzn2023.0.1.x86_64      3/6
Verifying       : nginx-core-1:1.26.2-1.amzn2023.0.1.x86_64  4/6
Verifying       : nginx-filesystem-1:1.26.2-1.amzn2023.0.1.noarch  5/6
Verifying       : nginx-mimetypes-2.1.49-3.amzn2023.0.3.noarch  6/6

Installed:
gperftools-libs-2.9.1-1.amzn2023.0.3.x86_64
libunwind-1.4.0-5.amzn2023.0.2.x86_64
nginx-1:1.26.2-1.amzn2023.0.1.x86_64
nginx-core-1:1.26.2-1.amzn2023.0.1.x86_64
nginx-filesystem-1:1.26.2-1.amzn2023.0.1.noarch
nginx-mimetypes-2.1.49-3.amzn2023.0.3.noarch

Complete!
[root@ip-10-0-3-1 ec2-user]# systemctl status nginx
○ nginx.service - The nginx HTTP and reverse proxy server
   Loaded: loaded (/usr/lib/systemd/system/nginx.service; disabled; preset: di
   Active: inactive (dead)
lines 1-3/3 (END)

```

Imagen 5. Se arranca el servicio nginx y se ejecuta el comando enable para que las próximas veces arranque automáticamente

```
root@ip-10-0-3-1:/home/ec2-user
Verifying      : nginx-mimetypes-2.1.49-3.amzn2023.0.3.noarch 6/6

Installed:
gperftools-libs-2.9.1-1.amzn2023.0.3.x86_64
libunwind-1.4.0-5.amzn2023.0.2.x86_64
nginx-1:1.26.2-1.amzn2023.0.1.x86_64
nginx-core-1:1.26.2-1.amzn2023.0.1.x86_64
nginx-filestream-1:1.26.2-1.amzn2023.0.1.noarch
nginx-mimetypes-2.1.49-3.amzn2023.0.3.noarch

Complete!
[root@ip-10-0-3-1 ec2-user]# systemctl status nginx
o nginx.service - The nginx HTTP and reverse proxy server
   Loaded: loaded (/usr/lib/systemd/system/nginx.service; disabled; preset: disabled)
   Active: inactive (dead)

lines 1-3/3 (END)

[root@ip-10-0-3-1 ec2-user]# systemctl start nginx
[root@ip-10-0-3-1 ec2-user]# systemctl enable nginx
Failed to enable unit: Unit file nginx.service does not exist.
[root@ip-10-0-3-1 ec2-user]# systemctl enable nginx
Created symlink /etc/systemd/system/multi-user.target.wants/nginx.service - /usr/lib/systemd/system/nginx.service.
[root@ip-10-0-3-1 ec2-user]#
```

Imagen 6. Se confirma que el servidor funciona correctamente

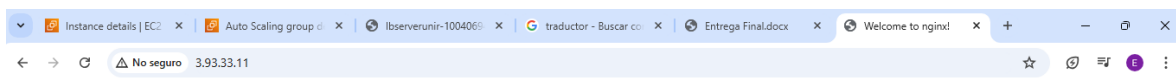


Imagen 7. Se instala Docker

```

root@ip-10-0-3-1:/home/ec2-user
Verifying      : nginx-mimetypes-2.1.49-3.amzn2023.0.3.noarch      6/6

Installed:
  gperftools-libs-2.9.1-1.amzn2023.0.3.x86_64
  libunwind-1.4.0-5.amzn2023.0.2.x86_64
  nginx-1:1.26.2-1.amzn2023.0.1.x86_64
  nginx-core-1:1.26.2-1.amzn2023.0.1.x86_64
  nginx-filesystem-1:1.26.2-1.amzn2023.0.1.noarch
  nginx-mimetypes-2.1.49-3.amzn2023.0.3.noarch

Complete!
[root@ip-10-0-3-1 ec2-user]# systemctl status nginx
○ nginx.service - The nginx HTTP and reverse proxy server
   Loaded: loaded (/usr/lib/systemd/system/nginx.service; disabled; preset: disabled)
   Active: inactive (dead)

lines 1-3/3 (END)

[root@ip-10-0-3-1 ec2-user]# systemctl start nginx
[root@ip-10-0-3-1 ec2-user]# systemctl enable nginx
Failed to enable unit: Unit file nginx.service does not exist.
[root@ip-10-0-3-1 ec2-user]# systemctl enable nginx
Created symlink /etc/systemd/system/multi-user.target.wants/nginx.service → /usr/lib/systemd/system/nginx.service.
[root@ip-10-0-3-1 ec2-user]# yum install docker -y

```

Imagen 8. Se activa el servicio y se ejecuta el comando enable para que arranque automáticamente las próximas veces

```

root@ip-10-0-3-1:/home/ec2-user
Verifying      : libnetfilter_conntrack-1.0.8-2.amzn2023.0.2.x86_64      6/10
Verifying      : libnfnetlink-1.0.1-19.amzn2023.0.2.x86_64             7/10
Verifying      : libnftnl-1.2.2-2.amzn2023.0.2.x86_64                 8/10
Verifying      : pigz-2.5-1.amzn2023.0.3.x86_64                       9/10
Verifying      : runc-1.1.14-1.amzn2023.0.1.x86_64                    10/10

Installed:
  containerd-1.7.23-1.amzn2023.0.1.x86_64
  docker-25.0.6-1.amzn2023.0.2.x86_64
  iptables-libs-1.8.8-3.amzn2023.0.2.x86_64
  iptables-nft-1.8.8-3.amzn2023.0.2.x86_64
  libcgroup-3.0-1.amzn2023.0.1.x86_64
  libnetfilter_conntrack-1.0.8-2.amzn2023.0.2.x86_64
  libnfnetlink-1.0.1-19.amzn2023.0.2.x86_64
  libnftnl-1.2.2-2.amzn2023.0.2.x86_64
  pigz-2.5-1.amzn2023.0.3.x86_64
  runc-1.1.14-1.amzn2023.0.1.x86_64

Complete!
[root@ip-10-0-3-1 ec2-user]# systemctl start docker
[root@ip-10-0-3-1 ec2-user]# systemctl enable docker
Created symlink /etc/systemd/system/multi-user.target.wants/docker.service → /usr/lib/systemd/system/docker.service.
[root@ip-10-0-3-1 ec2-user]#

```

Imagen 9. Una vez Docker se ejecuta, se debe instalar un contenedor. Para ello se necesita primero montar una imagen que ejecuta un apache

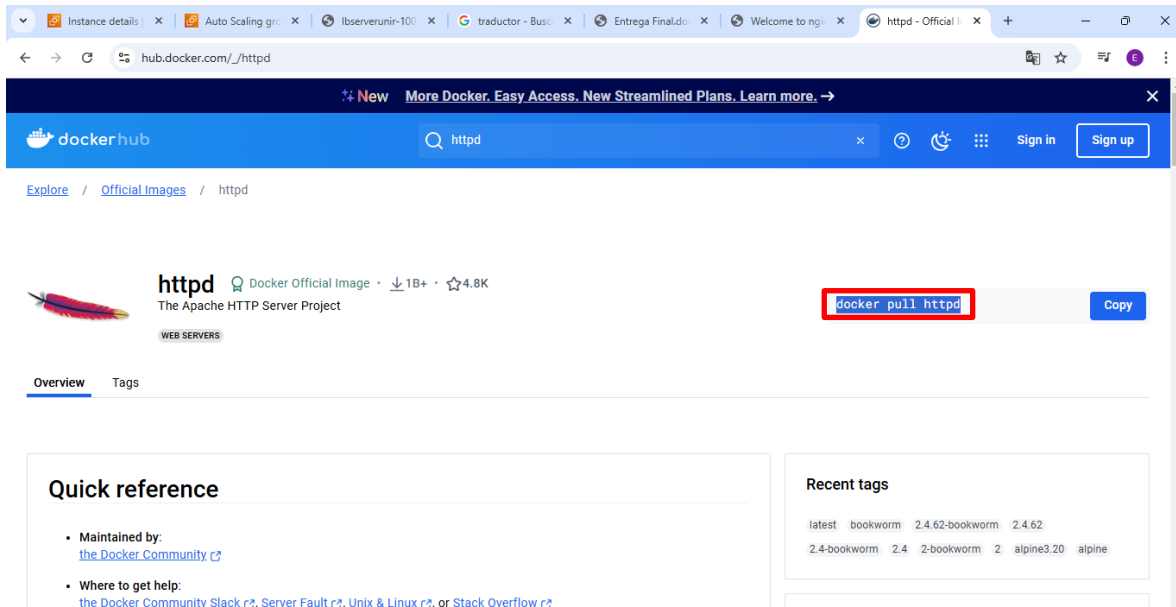


Imagen 10. Se ejecuta el código para subirla

```

root@ip-10-0-3-1:/home/ec2-user
Dec 09 23:48:43 ip-10-0-3-1.ec2.internal systemd[1]: Starting docker.service - D
Dec 09 23:48:43 ip-10-0-3-1.ec2.internal dockerd[488426]: time="2024-12-09T23:48>
Dec 09 23:48:43 ip-10-0-3-1.ec2.internal dockerd[488426]: time="2024-12-09T23:48>
Dec 09 23:48:44 ip-10-0-3-1.ec2.internal dockerd[488426]: time="2024-12-09T23:48>
Dec 09 23:48:44 ip-10-0-3-1.ec2.internal dockerd[488426]: time="2024-12-09T23:48>
Dec 09 23:48:44 ip-10-0-3-1.ec2.internal dockerd[488426]: time="2024-12-09T23:48>
Dec 09 23:48:44 ip-10-0-3-1.ec2.internal dockerd[488426]: time="2024-12-09T23:48>
Dec 09 23:48:44 ip-10-0-3-1.ec2.internal systemd[1]: Started docker.service - Do
lines 1-20/20 (END)

[root@ip-10-0-3-1 ec2-user]#
[root@ip-10-0-3-1 ec2-user]# docker pull httpd
Using default tag: latest
latest: Pulling from library/httpd
bc0965b23a04: Pull complete
d7ad38c6dd97: Pull complete
4f4fb700ef54: Pull complete
79b49624e34b: Pull complete
7d9f97915db2: Pull complete
9bd25d4f7b77: Pull complete
Digest: sha256:f4c5139eda466e45814122d9bd8b886d8ef6877296126c09b76dbad72b03c336
Status: Downloaded newer image for httpd:latest
docker.io/library/httpd:latest
[root@ip-10-0-3-1 ec2-user]#

```

Imagen 11. Se verifican las imágenes que se tienen guardadas de forma local

```

root@ip-10-0-3-1:/home/ec2-user
Dec 09 23:48:44 ip-10-0-3-1.ec2.internal dockerd[488426]: time="2024-12-09T23:48:44.123456789" level=info msg="Starting daemon"
Dec 09 23:48:44 ip-10-0-3-1.ec2.internal dockerd[488426]: time="2024-12-09T23:48:44.123456789" level=info msg="Starting daemon"
Dec 09 23:48:44 ip-10-0-3-1.ec2.internal dockerd[488426]: time="2024-12-09T23:48:44.123456789" level=info msg="Starting daemon"
Dec 09 23:48:44 ip-10-0-3-1.ec2.internal dockerd[488426]: time="2024-12-09T23:48:44.123456789" level=info msg="Starting daemon"
Dec 09 23:48:44 ip-10-0-3-1.ec2.internal systemd[1]: Started docker.service - Docker Engine
lines 1-20/20 (END)

[root@ip-10-0-3-1 ec2-user]#
[root@ip-10-0-3-1 ec2-user]# docker pull httpd
Using default tag: latest
latest: Pulling from library/httpd
bc0965b23a04: Pull complete
d7ad38c6dd97: Pull complete
4f4fb700ef54: Pull complete
79b49624e34b: Pull complete
7d9f97915db2: Pull complete
9bd25d4f7b77: Pull complete
Digest: sha256:f4c5139eda466e45814122d9bd8b886d8ef6877296126c09b76dbad72b03c336
Status: Downloaded newer image for httpd:latest
docker.io/library/httpd:latest
[root@ip-10-0-3-1 ec2-user]# docker images

```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
httpd	latest	494b2b45fd74	4 months ago	147MB

```

[root@ip-10-0-3-1 ec2-user]#

```

Imagen 12. Ahora se instala un contenedor:

-v: ruta (Disco duro) donde quiero que el servidor vaya y busque los archivos de mi aplicación, existe dentro del contenedor (la ruta se copia de la documentación de la misma imagen descargada anteriormente).

httpd: nombre de la imagen descargada

--restart always: hacer que el contenedor cuando se reinicie arranque de una vez y también arranque los contenedores existentes

-p 8080: puerto de ejecución del contenedor

-p 80: puerto donde la máquina virtual escucha

Finalmente debe aparecer el código de verificación

```

root@ip-10-0-3-1/
79b49624e34b: Pull complete
7d9f97915db2: Pull complete
9bd25d4f7b77: Pull complete
Digest: sha256:f4c5139eda466e45814122d9bd8b886d8ef6877296126c09b76dbad72b03c336
Status: Downloaded newer image for httpd:latest
docker.io/library/httpd:latest
[root@ip-10-0-3-1 ec2-user]# docker images
REPOSITORY TAG IMAGE ID CREATED SIZE
httpd latest 494b2b45fd74 4 months ago 147MB
[root@ip-10-0-3-1 ec2-user]# pwd
/home/ec2-user
[root@ip-10-0-3-1 ec2-user]# cd /
[root@ip-10-0-3-1 /]# ls
bin dev home lib64 media opt root sbin sys usr
boot etc lib local mnt proc run srv tmp var
[root@ip-10-0-3-1 /]# mkdir appunir
[root@ip-10-0-3-1 /]# chmod 777 appunir/
[root@ip-10-0-3-1 /]# doker run -dit --name appunir --restart always -p 8080:80 -v /appunir:/usr/local/apache2/ht
docs/ httpd
bash: doker: command not found
[root@ip-10-0-3-1 /]# docker run -dit --name appunir --restart always -p 8080:80 -v /appunir:/usr/local/apache2/h
tdocs/ httpd
e4a4354adf7a4adbd3161059fcc38f5d0263e2f96ab21959b57963bfcd146
[root@ip-10-0-3-1 /]#

```

Imagen 13. Se debe detener el servicio de nginx y el contenedor. Se vuelve a iniciar el contenedor y con el comando docker ps se verifica que el contenedor se está ejecutando.

```

root@ip-10-0-3-1/appunir
tcp appunir
[root@ip-10-0-3-1 /]# cd appunir/
[root@ip-10-0-3-1 appunir]# ls
[root@ip-10-0-3-1 appunir]# nano index.html
[root@ip-10-0-3-1 appunir]# systemctl stop nginx
[root@ip-10-0-3-1 appunir]# docker stop ^C
[root@ip-10-0-3-1 appunir]# docker stop e4a4354adf7a4adbd3161059fcc38f5d0263e2f96ab21959b57963bfcd146
e4a4354adf7a4adbd3161059fcc38f5d0263e2f96ab21959b57963bfcd146
[root@ip-10-0-3-1 appunir]# docker start e4a4354adf7a4adbd3161059fcc38f5d0263e2f96ab21959b57963bfcd146
e4a4354adf7a4adbd3161059fcc38f5d0263e2f96ab21959b57963bfcd146
[root@ip-10-0-3-1 appunir]# docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS
e4a4354adf7a4adbd3161059fcc38f5d0263e2f96ab21959b57963bfcd146 httpd "httpd-foreground" 28 minutes ago Up 3 seconds 0.0.0.0:8080->80/tcp, :::8080->80/t
cp appunir

```

Imagen 14. Se verifican los permisos del archivo de la aplicación

```

root@ip-10-0-3-1/appunir
[root@ip-10-0-3-1 appunir]# docker stop ^C
[root@ip-10-0-3-1 appunir]# docker stop e4a4354adf7a4adbd3161059fcc38f5d0263e2f96ab21959b57963bfcd146
e4a4354adf7a4adbd3161059fcc38f5d0263e2f96ab21959b57963bfcd146
[root@ip-10-0-3-1 appunir]# docker start e4a4354adf7a4adbd3161059fcc38f5d0263e2f96ab21959b57963bfcd146
e4a4354adf7a4adbd3161059fcc38f5d0263e2f96ab21959b57963bfcd146
[root@ip-10-0-3-1 appunir]# docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS
e4a4354adf7a4adbd3161059fcc38f5d0263e2f96ab21959b57963bfcd146 httpd "httpd-foreground" 28 minutes ago Up 3 seconds 0.0.0.0:8080->80/tcp, :::8080->80/t
cp appunir
[root@ip-10-0-3-1 appunir]# ls
index.html
[root@ip-10-0-3-1 appunir]# ls -la
total 4
drwxrwxrwx. 2 root root 24 Dec 10 00:43
dr-xr-xr-x. 19 root root 252 Dec 10 00:14 ..
-rw-r--r--. 1 root root 32 Dec 10 00:43 index.html
[root@ip-10-0-3-1 appunir]# chmod 777 index.html
[root@ip-10-0-3-1 appunir]# docker run -dit --name appunir80 --restart always -p 80:80 -v /appunir:/usr/local/apac
he2/htdocs/ httpd
57b3db69ee44de1ea74ece569463bd524bcc01e84016dc4319b853949d075bdd
[root@ip-10-0-3-1 appunir]# docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS
57b3db69ee44de1ea74ece569463bd524bcc01e84016dc4319b853949d075bdd httpd "httpd-foreground" 27 seconds ago Up 25 seconds 0.0.0.0:80->80/tcp, :::80->80/t
cp appunir80
e4a4354adf7a4adbd3161059fcc38f5d0263e2f96ab21959b57963bfcd146 httpd "httpd-foreground" 31 minutes ago Up 2 minutes 0.0.0.0:8080->80/tcp, :::8080->80/t
tcp appunir
[root@ip-10-0-3-1 appunir]#

```

Imagen 15. Se crea otro contenedor que se ejecute en el puerto 80 y se crean mas sitios web para hacer la prueba

```

root@ip-10-0-3-1/appunir3
-rw-r--r-- 1 root root 32 Dec 10 00:43 index.html
[root@ip-10-0-3-1 appunir1]# chmod 777 index.html
[root@ip-10-0-3-1 appunir1]# docker run -dit --name appunir80 --restart always -p 80:80 -v /appunir:/usr/local/apache2/htdocs/ httpd
57b3db69eee4de1ea74ece569463bd524bcc01e84016dc4319b853949d075bdd
[root@ip-10-0-3-1 appunir1]# docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED        STATUS        PORTS
NAMES
57b3db69eee4   httpd    "httpd-foreground"      27 seconds ago Up 25 seconds 0.0.0.0:80->80/tcp, :::80->80/tcp
appunir80
e4a4354adfad   httpd    "httpd-foreground"      31 minutes ago Up 2 minutes  0.0.0.0:8080->80/tcp, :::8080->80/tcp
appunir
[root@ip-10-0-3-1 appunir1]# cd ..
[root@ip-10-0-3-1 /]# mkdir appunir2
[root@ip-10-0-3-1 /]# mkdir appunir3
[root@ip-10-0-3-1 /]# chmod 777 appunir2
[root@ip-10-0-3-1 /]# chmod 777 appunir3
[root@ip-10-0-3-1 /]# cd appunir2
[root@ip-10-0-3-1 appunir2]# nano index.html
[root@ip-10-0-3-1 appunir2]# cd ..
[root@ip-10-0-3-1 /]# cd appunir3
[root@ip-10-0-3-1 appunir3]# nano index.html
[root@ip-10-0-3-1 appunir3]# docker run -dit --name appunir2 --restart always -p 8082:80 -v /appunir:/usr/local/apache2/htdocs/ httpd
e899725e32a975c15fc9c40b96dfdee21995057ac58ac5121450ddea2d370516
[root@ip-10-0-3-1 appunir3]# docker run -dit --name appunir3 --restart always -p 8083:80 -v /appunir:/usr/local/apache2/htdocs/ httpd
995ecd48702d936777ba5baf16dde2f155aaa9cb4150b7d42100d0ea4bd40492
[root@ip-10-0-3-1 appunir3]#

```

Imagen 16. Con los tres contenedores se realiza la prueba (los otros dos contenedores se crearon mal por lo tanto no se usan)

```

root@ip-10-0-3-1/etc/nginx
57b3db69eee4   httpd    "httpd-foreground"      41 minutes ago Up 41 minutes 0.0.0.0:80->80/tcp, :::80->80/tcp
appunir80
e4a4354adfad   httpd    "httpd-foreground"      About an hour ago Up 44 minutes 0.0.0.0:8080->80/tcp, :::8080->80/tcp
appunir
[root@ip-10-0-3-1 appcur3]# docker stop 57b3db69eee4
57b3db69eee4
[root@ip-10-0-3-1 appcur3]# docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED        STATUS        PORTS
NAMES
f98c5d2bcbdd   httpd    "httpd-foreground"      3 minutes ago Up 3 minutes 0.0.0.0:8085->80/tcp, :::8085->80/tcp
apcur3
0870aad576bf   httpd    "httpd-foreground"      4 minutes ago Up 4 minutes 0.0.0.0:8084->80/tcp, :::8084->80/tcp
apcur2
995ecd48702d   httpd    "httpd-foreground"      29 minutes ago Up 29 minutes 0.0.0.0:8083->80/tcp, :::8083->80/tcp
appunir3
e899725e32a9   httpd    "httpd-foreground"      30 minutes ago Up 30 minutes 0.0.0.0:8082->80/tcp, :::8082->80/tcp
appunir2
e4a4354adfad   httpd    "httpd-foreground"      About an hour ago Up 45 minutes 0.0.0.0:8080->80/tcp, :::8080->80/tcp
appunir
[root@ip-10-0-3-1 appcur3]# systemctl start nginx
[root@ip-10-0-3-1 appcur3]# cde /etc/nginx/
bash: cde: command not found
[root@ip-10-0-3-1 appcur3]# cd /etc/nginx/
[root@ip-10-0-3-1 nginx]# ls
conf.d      fastcgi.conf.default  koi-utf      mime.types.default  scgi_params      uwsgi_params.default
default.d   fastcgi_params        koi-win      nginx.conf           scgi_params.default  win-utf
fastcgi.conf  fastcgi_params.default  mime.types   nginx.conf.default  uwsgi_params
[root@ip-10-0-3-1 nginx]# cp nginx.conf nginx.bk
[root@ip-10-0-3-1 nginx]# nano nginx.conf

```

Imagen 17. Se configura el nginx que es el que se va a encargar de manejar/repartir las peticiones a los contenedores creados, lo que genera una alta disponibilidad y se hace antes un backup

```

root@ip-10-0-3-1/etc/nginx
57b3db69eee4 httpd "httpd-foreground" 41 minutes ago Up 41 minutes 0.0.0.0:80->80/tcp, :::80->80/tcp
  appunir80
e4a4354adfad httpd "httpd-foreground" About an hour ago Up 44 minutes 0.0.0.0:8080->80/tcp, :::8080->80/tcp
  appunir
[root@ip-10-0-3-1 appcur3]# docker stop 57b3db69eee4
57b3db69eee4
[root@ip-10-0-3-1 appcur3]# docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED         STATUS         PORTS
MES
f98c5d2bcbd4   httpd    "httpd-foreground"      3 minutes ago  Up 3 minutes  0.0.0.0:8085->80/tcp, :::8085->80/tcp  ap
pcur3
0870aad576bf   httpd    "httpd-foreground"      4 minutes ago  Up 4 minutes  0.0.0.0:8084->80/tcp, :::8084->80/tcp  ap
pcur2
995ecd48702d   httpd    "httpd-foreground"      29 minutes ago Up 29 minutes  0.0.0.0:8083->80/tcp, :::8083->80/tcp  ap
punir3
e899725e32a9   httpd    "httpd-foreground"      30 minutes ago Up 30 minutes  0.0.0.0:8082->80/tcp, :::8082->80/tcp  ap
punir2
e4a4354adfad   httpd    "httpd-foreground"      About an hour ago Up 45 minutes  0.0.0.0:8080->80/tcp, :::8080->80/tcp  ap
punir
[root@ip-10-0-3-1 appcur3]# systemctl start nginx
[root@ip-10-0-3-1 appcur3]# cde /etc/nginx/
bash: cde: command not found
[root@ip-10-0-3-1 appcur3]# cd /etc/nginx/
[root@ip-10-0-3-1 nginx]# ls
conf.d      fastcgi.conf.default  koi-utf      mime.types.default  scgi_params      uwsgi_params.default
default.d   fastcgi_params        koi-win      nginx.conf           scgi_params.default  win-utf
fastcgi.conf  fastcgi_params.default  mime.types  nginx.conf.default  uwsgi_params
[root@ip-10-0-3-1 nginx]# cp nginx.conf nginx.bk
[root@ip-10-0-3-1 nginx]# nano nginx.conf

```

Imagen 18. Se elimina la configuración que trae por defecto nginx y se ingresa una nueva configuración básica que se requiere para un Proxy Inverso (depende de cada servidor)

```

GNU nano 5.8                               nginx.conf                               Modified
events {}

http {
    upstream backend {
        server localhost:8080;
        server localhost:8084;
        server localhost:8085;
    }

    server {
        listen 80;
        server_name nginx;
        location / {
            proxy_pass http://backend;
        }
    }
}

```

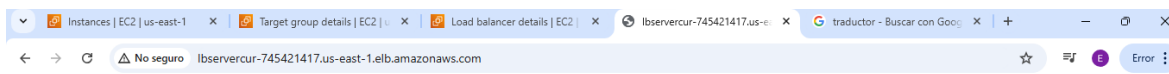
Imagen 19. Se reinicia el servicio de nginx

```

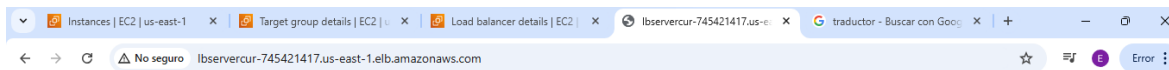
root@ip-10-0-3-1/etc/nginx
e4a4354adfad httpd "httpd-foreground" About an hour ago Up 44 minutes 0.0.0.0:8080->80/tcp, :::8080->80/tcp
P appunir
[root@ip-10-0-3-1 appcur3]# docker stop 57b3db69eee4
57b3db69eee4
[root@ip-10-0-3-1 appcur3]# docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NA
MES
f98c5d2bcbdf httpd "httpd-foreground" 3 minutes ago Up 3 minutes 0.0.0.0:8085->80/tcp, :::8085->80/tcp ap
pcur3
0870aad576bf httpd "httpd-foreground" 4 minutes ago Up 4 minutes 0.0.0.0:8084->80/tcp, :::8084->80/tcp ap
pcur2
995ecd48702d httpd "httpd-foreground" 29 minutes ago Up 29 minutes 0.0.0.0:8083->80/tcp, :::8083->80/tcp ap
punir3
e899729e32a9 httpd "httpd-foreground" 30 minutes ago Up 30 minutes 0.0.0.0:8082->80/tcp, :::8082->80/tcp ap
punir2
e4a4354adfad httpd "httpd-foreground" About an hour ago Up 45 minutes 0.0.0.0:8080->80/tcp, :::8080->80/tcp ap
punir
[root@ip-10-0-3-1 appcur3]# systemctl start nginx
[root@ip-10-0-3-1 appcur3]# cde /etc/nginx/
bash: cde: command not found
[root@ip-10-0-3-1 appcur3]# cd /etc/nginx/
[root@ip-10-0-3-1 nginx]# ls
conf.d fastcgi.conf.default koi-utf mime.types.default scgi_params uwsgi_params.default
default.d fastcgi_params koi-win nginx.conf scgi_params.default win-utf
fastcgi.conf fastcgi_params.default mime.types nginx.conf.default uwsgi_params
[root@ip-10-0-3-1 nginx]# cp nginx.conf nginx.bk
[root@ip-10-0-3-1 nginx]# nano nginx.conf
[root@ip-10-0-3-1 nginx]# systemctl restart nginx
[root@ip-10-0-3-1 nginx]#

```

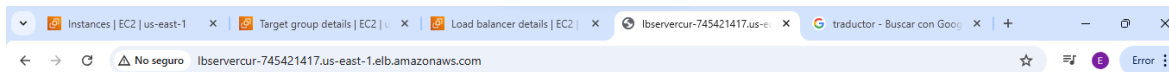
Imagen 20-21-22. Y finalmente debe iniciar a repartir las peticiones a cada uno de los contenedores creados



PRUEBA II USO CONTENEDORES AWS



PRUEBA USO DE CONTENEDORES AWS

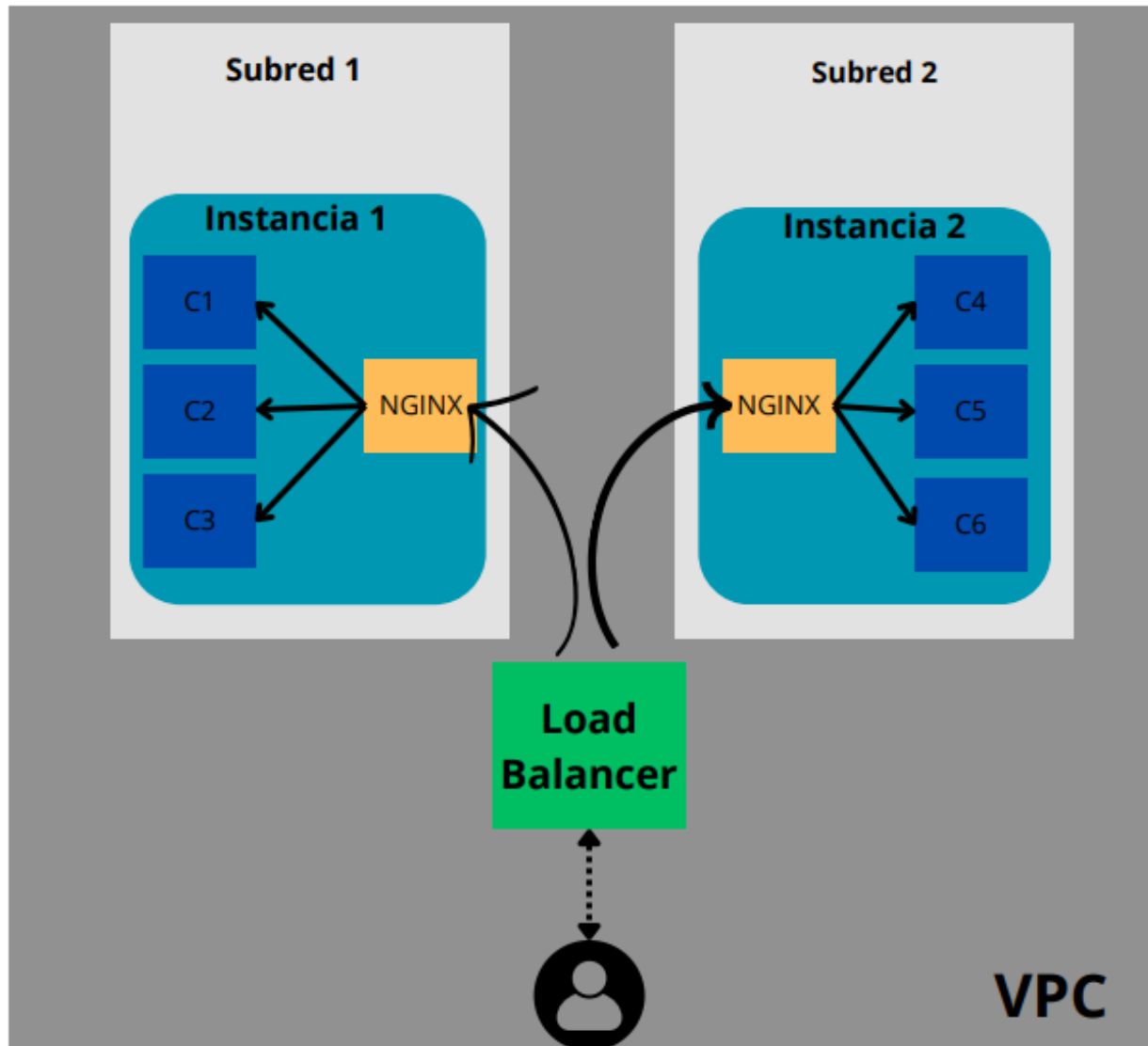


PRUEBA III USO DE CONTENEDORES AWS

Video implementación de Docker en AWS

<https://youtu.be/77fKx1tnRKQ>

Incluya un diagrama de los recursos usados y como se comunican entre ellos



Link diagrama: https://www.canva.com/design/DAGY8OUfapQ/JrpjtXGrpwtqPMq9KF-Z3A/view?utm_content=DAGY8OUfapQ&utm_campaign=designshare&utm_medium=link2&utm_source=uniquelinks&utm_id=hcf72428ca8

Conclusiones

La implementación de esta arquitectura ofrece servicios de alta disponibilidad y permite trabajar de manera mucho más eficiente, escalable, automatizada y monitorizada.

El aprendizaje y adaptación son necesarios, por lo que se requiere tiempo y dedicación para poder empezar a hacer uso de estos servicios y sacar el mayor provecho.

El control de costos puede terminar siendo favorable ya que si se administran correctamente los gastos pueden ser más bajos.

Referencias

(Amazon Web Services, 2024)

(Amazon Web Services, 2024)