TRABAJO DE GRADO
Opción Seminario-Diplomado.

ScaleWise Cloud Services

Corporación Universitaria Remington.
Facultad de Ingeniería
Ingeniería de Sistemas

Estudiantes:
Fray Smith Jaramillo Arbeláez
Juan Camilo Sánchez Quila
Jennifer Cristina Lopez Gomez

Tutor: Juan Pablo Berrio Lopez
Opción de Trabajo de grado Seminario-Diplomado.
2024.

Tabla de Contenidos

Resumen

En este documento se detallará la creación y configuración de un servicio basado en AWS diseñado para una plataforma que permite realizar entregas rápidas a sus clientes. Este servicio integra diversas funcionalidades de AWS para garantizar su eficiencia y escalabilidad.

El desarrollo comienza con la configuración de las VPC (Virtual Private Cloud), las cuales controlarán el entorno de red en el que se implementarán las instancias necesarias. Estas instancias contarán con las configuraciones adecuadas para soportar las aplicaciones y servicios que componen la plataforma. La configuración de las VPC asegura un entorno seguro, segmentado y personalizable, permitiendo una mejor administración de recursos en la nube.

Además, se implementarán balanceadores de carga (Load Balancers) y Auto Scaling Groups. Estas herramientas son esenciales para distribuir de manera equitativa las solicitudes de los clientes entre las instancias disponibles. Los Auto Scaling Groups permitirán ajustar dinámicamente la cantidad de instancias en función de la carga de trabajo, asegurando que la plataforma sea responsiva y eficiente, incluso durante picos de demanda. Este enfoque garantiza una experiencia de usuario estable y un servicio ininterrumpido.

Para optimizar la implementación y ejecución de las aplicaciones, se utilizarán contenedores Docker. Este enfoque simplifica el despliegue, ya que encapsula las aplicaciones junto con sus dependencias, asegurando su portabilidad y consistencia. La virtualización mediante Docker permite trabajar con una infraestructura más ligera y flexible, reduciendo la necesidad de hardware físico y facilitando la centralización de la gestión de servicios.

En resumen, este servicio combina herramientas avanzadas de AWS con tecnologías modernas como Docker para crear una plataforma robusta y eficiente. Su diseño garantiza la capacidad de manejar grandes volúmenes de solicitudes, mantener un servicio estable y adaptarse a las necesidades cambiantes del mercado. La integración de estas soluciones permite maximizar los recursos tecnológicos, ofreciendo a los clientes un servicio de alta calidad y rapidez en las entregas

Palabras clave

AWS, Plataforma de entregas, VPC (Virtual Private Cloud), Instancias, Auto Scaling Groups

Marco conceptual y contextual

¿Qué es AWS (Amazon Web Services)?

AWS (Amazon Web Services) es una plataforma de computación en la nube.

El primer producto (S3) se lanzó en 2006.

AWS ha crecido mucho desde entonces, tanto en tamaño como en gama de productos.

Es, hasta la fecha, el mayor proveedor de nube del mundo.

¿Por qué aprender AWS?

- AWS es el mayor proveedor de nube
- La competencia AWS es popular en el mercado laboral
- Puede hacer la mayoría de las cosas en la nube de AWS
- Gran comunidad/apoyo

1. Cuenta gratuita de AWS

- AWS ofrece un nivel gratuito que le permite explorar y probar sus servicios.

- El nivel gratuito le ofrece un uso limitado de servicios de forma gratuita.

1.1 AWS EC2: servidor de nube virtual

EC2 es un servidor virtual en la nube de AWS. AWS EC2 es la abreviatura de AWS Elastic Cloud Compute. Permite ampliar o reducir fácilmente la capacidad.
Facilita el proceso de aumento y disminución de capacidad. Como resultado, usted puede acceder a los recursos cuando lo necesite. No se necesita ninguna inversión inicial. Sólo pagas por lo que necesitas. EC2 es seguro.

1.2 Tipos de instancias de AWS EC2

Hay diferentes tipos de instancias. Cuál utilizar dependerá de las necesidades. Las instancia se utilizan mejor para diferentes cosas. Por lo tanto, al seleccionar un tipo de instancia, tenga en cuenta las necesidades. Por ejemplo, las necesidades pueden ser un requisito de computación, memoria o almacenamiento. La instancia de propósito general equilibra los recursos informáticos, de memoria y de red.

1.3 ¿Que es una VPC (Virtual private cloud)?

Una VPC de AWS es un servidor que permite lanzar recursos de AWS en una red virtual definida por el usuario, todas las cuentas nuevas de AWS tiene una VPC por defecto , las nuevas instancia EC2 se lanza de la VPC por defecto si no se especifica lo contrario.
Una VPC (Virtual Private Cloud) en AWS es una red virtual privada que se puede utilizar para crear y conectar recursos de computación, como servidores y bases de datos, en un entorno aislado y seguro. Las VPCs pueden ser públicas o privadas, y se pueden configurar de diferentes maneras para adaptarse a las necesidades específicas de cada usuario.
Una VPC permite controlar todos los aspectos del entorno de red virtual, como la selección de un rango de direcciones IP, la creación de subredes, y la configuración de tablas de ruteo y puertas de enlace de red

1.4 Lanzar una instancia

Después de iniciar la instancia, puede conectarse a ella y utilizarla. Inicialmente, el estado de la instancia es pending. Cuando el estado de la instancia es running, la instancia ha empezado a arrancar. Puede que transcurran unos instantes antes de que pueda conectarse a la instancia. Tenga en cuenta que los tipos de instancia bare metal pueden tardar más tiempo en iniciarse.

En función de cómo planee conectarse a la instancia, es posible que desee realizar determinadas configuraciones al inicializar la instancia. Estas configuraciones pueden incluir la especificación de reglas de grupos de seguridad de entrada para cierto tráfico o la asociación de un rol de perfil de instancia. Para obtener más información sobre los métodos de conexión que puede utilizar para conectarse y sus requisitos, consulte Conexión con instancias EC2.
La instancia recibe un nombre de DNS público que puede utilizar para contactar con ella desde Internet. La instancia también recibe un nombre de DNS privado que las demás instancias de la misma VPC pueden utilizar para contactar con ella.

Cuando haya terminado con una instancia, para evitar incurrir en costos innecesarios, asegúrese de terminarla. Para obtener más información, consulte Terminación de las instancias de Amazon EC2.

1.5 Seguridad en la nube de AWS

Una seguridad sólida en el centro de una organización permite la transformación digital y la innovación. AWS ayuda a las organizaciones a desarrollar y convertir la seguridad, la identidad y el cumplimiento en facilitadores empresariales clave. En AWS, la seguridad es la máxima prioridad. AWS está diseñado para ser la infraestructura en la nube global más segura en la que crear, migrar y administrar aplicaciones y cargas de trabajo.

1.6 Grupos de seguridad

Un grupo de seguridad funciona como un firewall virtual para las instancias de EC2 para controlar el tráfico entrante y saliente. Las reglas de entrada controlan el tráfico entrante a la instancia y las reglas de salida controlan el tráfico saliente desde la instancia. Al iniciar una instancia puede especificar uno o varios grupos de seguridad. Si no especifica un grupo de seguridad, Amazon EC2 utiliza el grupo de seguridad predeterminado para la VPC. Una vez lanzada la instancia, puede cambiar sus grupos de seguridad.
El grupo de seguridad solo se puede utilizar en la VPC para la que se creó. Puede asociar cada instancia a varios grupos de seguridad y puede asociar cada grupo de seguridad a varias instancias. Añade reglas a cada grupo de seguridad que permiten que el tráfico a o desde sus instancias asociadas. Puede modificar las reglas de un grupo de seguridad en cualquier momento. Las reglas nuevas y modificadas se aplican automáticamente a todas las instancias asociadas al grupo de seguridad.

1.7 Grupos objetivo

Un grupo objetivo de VPC Lattice es un conjunto de objetivos, o recursos de cómputo, que ejecutan su aplicación o servicio. Los destinos pueden ser EC2 instancias, direcciones IP, funciones Lambda, balanceadores de carga de aplicaciones o pods de Kubernetes. También puede asociar los servicios existentes a sus grupos de destino.

Cada grupo de destino se utiliza para direccionar solicitudes a uno o varios destinos registrados. Cuando crea la regla del oyente, especifica un grupo de destino y condiciones. Cuando se cumple la condición de una regla, el tráfico se reenvía al grupo de destino correspondiente. Puede crear grupos de destino diferentes para los distintos tipos de solicitudes. Por ejemplo, cree un grupo de destino para las solicitudes generales y otros grupos de destino para las solicitudes que incluyan condiciones de regla específicas, como una ruta o un valor de encabezado.
Puede definir la configuración de comprobación de estado de su servicio para cada grupo de destino. Cada grupo de destino utiliza la configuración de comprobación de estado predeterminada, a menos que la anule al crear el grupo de destino o la modifique posteriormente. Después de especificar un grupo de destino en una regla para un oyente, el servicio monitoriza constantemente el estado de todos los destinos registrados en el grupo de destino. El servicio dirige las solicitudes a los destinos registrados que se encuentran en buen estado.
Para especificar un grupo de destino en una regla para un oyente de servicios, el grupo de destino debe estar en la misma cuenta que el servicio. VPC Los grupos objetivo de Lattice son similares a los grupos objetivo proporcionados por Elastic Load Balancing, pero no son intercambiables.
Si un grupo objetivo está configurado con el HTTPS protocolo o utiliza comprobaciones de HTTPS estado, las TLS conexiones a los destinos utilizan la política de seguridad del

agente de escucha. VPCLattice establece TLS las conexiones con los destinos mediante certificados que usted instala en los destinos. VPCLattice no valida estos certificados. Por lo tanto, puede utilizar certificados autofirmados o certificados que hayan caducado. El tráfico entre VPC Lattice y los objetivos se autentica a nivel de paquetes, por lo que no corre el riesgo de sufrir man-in-the-middle ataques o suplantación de identidad, incluso si los certificados de los objetivos no son válidos.

1.8 Crear imagen

Cuando crea una AMI a partir de una instancia, Amazon EC2 apaga la instancia antes de crear la AMI para asegurarse de que todo lo que hay en la instancia está detenido y en un estado constante durante el proceso de creación. Si está seguro de que la instancia está en un estado coherente adecuado para la creación de una AMI, puede informar a Amazon EC2 de que no apague y reinicie la instancia. Algunos sistemas de archivos, como XFS, pueden pausar y reanudar la actividad, de forma que sea seguro crear la imagen sin tener que reiniciar la instancia.

1.9 Plantillas de lanzamiento

Una plantilla de lanzamiento es similar a una configuración de lanzamiento, ya que sirve para especificar la información de configuración de las instancias. Incluye el ID de Amazon Machine Image (AMI), el tipo de instancia, un key pair, los grupos de seguridad y otros parámetros que se utilizan para lanzar EC2 instancias. No obstante, la definición de una plantilla de lanzamiento en lugar de una configuración de lanzamiento le permite tener varias versiones de una plantilla de lanzamiento. Con el control de versiones de plantillas de lanzamiento, puede crear un subconjunto del conjunto completo de parámetros**.**

1.10 Equilibrador de carga

Elastic Load Balancing distribuye automáticamente el tráfico entrante entre varios destinos, por ejemplo, instancias EC2, contenedores y direcciones IP en una o varias zonas de disponibilidad. Monitorea el estado de los destinos registrados y enruta el tráfico solamente a destinos en buen estado. Elastic Load Balancing escala el equilibrador de carga a medida que el tráfico entrante va cambiando con el tiempo. Puede escalarse automáticamente para adaptarse a la mayoría de las cargas de trabajo.

1.11 Grupos de escalado automático

Un grupo de Auto Scaling contiene una colección de EC2 instancias que se tratan como una agrupación lógica con fines de escalado y administración automáticos. Un grupo de Auto Scaling también le permite utilizar las funciones de Amazon EC2 Auto Scaling, como las sustituciones de chequeos de estado y las políticas de escalado. Tanto el

mantenimiento del número de instancias en un grupo de Auto Scaling como el escalado automático son las funciones principales del servicio Amazon EC2 Auto Scaling.

El tamaño de un grupo de Auto Scaling depende del número de instancias que establezca como capacidad deseada. Puede ajustar su tamaño para satisfacer la demanda, ya sea de forma manual o mediante el uso de escalado automático.

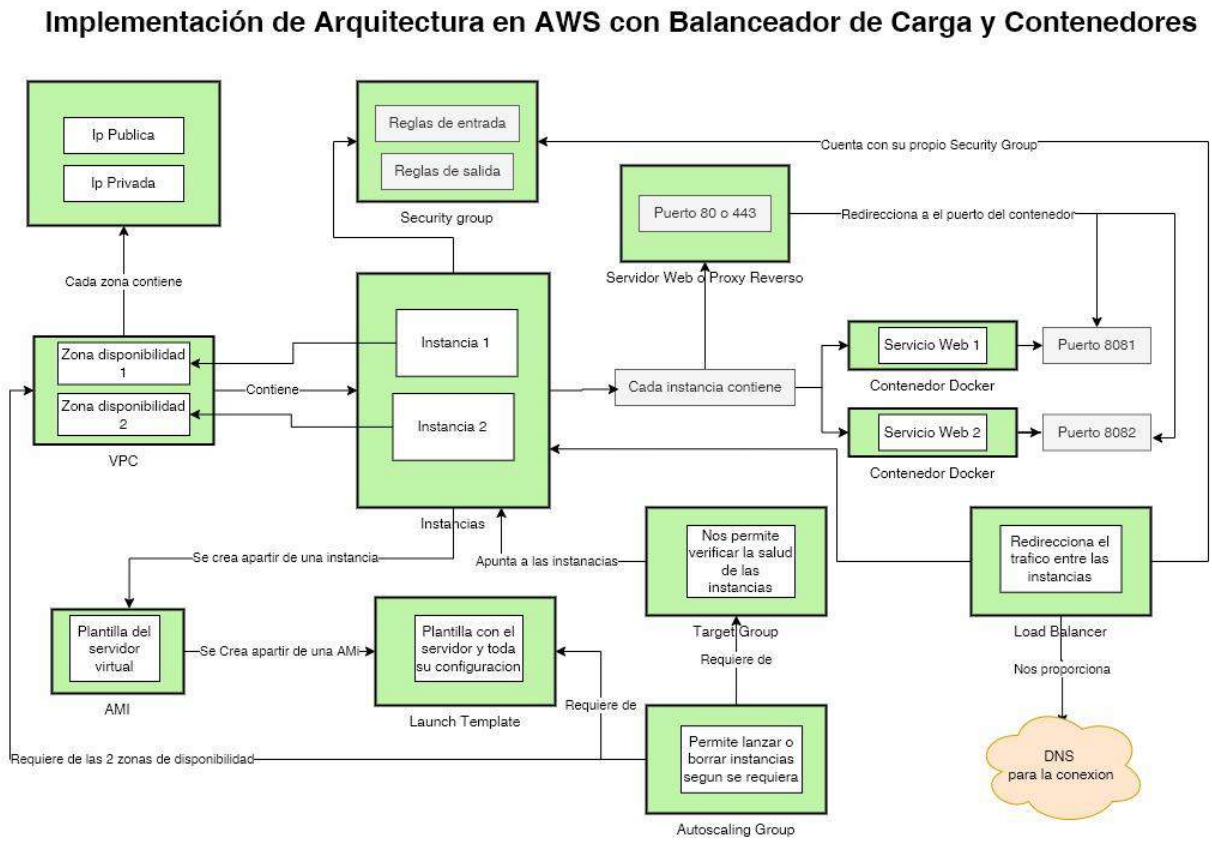
1.12 Docker

Amazon ECS utiliza imágenes de Docker en las definiciones de tareas para lanzar contenedores. Docker es una tecnología que brinda herramientas para crear, ejecutar, probar e implementar aplicaciones distribuidas en contenedores.
Las definiciones de tareas de Amazon ECS utilizan imágenes de Docker para lanzar contenedores en las instancias de contenedor de los clústeres. En esta sección, va a crear una imagen de Docker de una aplicación web simple y la va a probar en su sistema local o en la instancia de Amazon EC2. Luego, enviará la imagen a un registro de contenedores de Amazon ECR para poder utilizarla en una definición de tarea de Amazon ECS.

1.13 Proxy inverso NGINX

El uso de proxy se suele utilizar para distribuir la carga entre varios servidores, mostrar sin problemas contenido de diferentes sitios web o pasar solicitudes para su procesamiento a servidores de aplicaciones a través de protocolos distintos de HTTP. Cuando NGINX envía una solicitud a través de un proxy, la envía a un servidor proxy específico, obtiene la respuesta y la envía de vuelta al cliente. Es posible enviar solicitudes a través de un proxy a un servidor HTTP (otro servidor NGINX o cualquier otro servidor) o a un servidor que no sea HTTP (que pueda ejecutar una aplicación desarrollada con un marco específico, como PHP o Python) utilizando un protocolo específico.
El proxy inverso de Nginx actúa como un servidor intermedio que intercepta las solicitudes de los clientes y las reenvía al servidor backend correspondiente y, posteriormente, reenvía una respuesta del servidor al cliente. El proxy inverso ofrece varios beneficios como capa abstracta por encima de los servidores ascendentes

Desarrollo e implementación del aprendizaje



Entrega 1

Para garantizar que una instancia de AWS (como las instancias EC2) funcione adecuado, diferentes recursos deben trabajar en conjunto. A continuación, explico cada uno de estos recursos y cómo contribuyen al funcionamiento de una instancia:

1. **Instancias EC2 (Elastic Compute Cloud)**
El recurso principal es EC2, que proporciona capacidad de computación escalable en la nube. Una instancia EC2 es una máquina virtual que se ejecuta en los servidores físicos de AWS.
Sin embargo, una instancia EC2 necesita varios recursos adicionales para funcionar correctamente:

2. **VPC (Virtual Private Cloud)**
Una VPC es una red privada virtual dentro de AWS donde operan los recursos de tu cuenta, incluida la instancia EC2.

La VPC proporciona aislamiento de red y permite configurar reglas de comunicación, como acceso interno o externo.

Cada instancia debe ejecutarse dentro de una VPC.

3. **Subredes**
Una subred es un segmento dentro de una VPC donde se despliegan los recursos.

Puede ser una subred pública (accesible desde Internet) o privada (aislada del tráfico externo).

En tu caso, como se necesita acceso desde Internet, la instancia se coloca en una subred pública.

4. **Internet Gateway**
Un Internet Gateway es un recurso que conecta la VPC a Internet, permitiendo que las instancias en una subred pública reciban tráfico entrante y envíen tráfico saliente.

Es esencial para permitir que los usuarios se conecten a una instancia EC2 desde fuera de AWS.

5. **Tablas de enrutamiento**
Las tablas de rutas se utilizan para definir cómo el tráfico fluye dentro de la red.

Por ejemplo, para que una instancia en una subred pública se comunique con Internet, la tabla de enrutamiento de la subred debe contener una entrada que dirija el tráfico a través del Internet Gateway.

6. **Grupos de seguridad (Security Groups)**
Los Grupos de seguridad actúan como cortafuegos virtuales para controlar el tráfico que entra o sale de las instancias.

Es necesario configurar las reglas del grupo de seguridad para permitir el tráfico requerido (por ejemplo, el puerto 22 para SSH o el puerto 80 para HTTP).

7. **Clave SSH o autenticación**
Para acceder a la instancia, necesitas un par de claves SSH u otro método de autenticación configurado en la instancia.
Esto garantiza un acceso seguro a la instancia.

8. **Elastic IP (opcional)**
Si deseas asignar una dirección IP pública fija a tu instancia, puedes usar una Elastic IP.

Esto facilita el acceso a la instancia desde Internet sin que su dirección IP cambie tras un reinicio.

9. **Almacenamiento (EBS o Instance Store)**
Las instancias necesitan almacenamiento para funcionar, que puede ser proporcionado por:

EBS (Elastic Block Store): Discos persistentes que retienen los datos incluso si la instancia se detiene.
Instance Store: Almacenamiento temporal que se pierde cuando la instancia se detiene.
Estos recursos trabajan juntos para garantizar que las instancias EC2 operen de manera efectiva en la nube. Aunque EC2 es el recurso principal, los demás son igualmente esenciales para crear un entorno funcional.

Entrega 1
figura 1.1.
Figura 1. Creación de un servidor web desde cero en Amazon AWS.
Se ingresa a la pagina principal de AWS con las credenciales

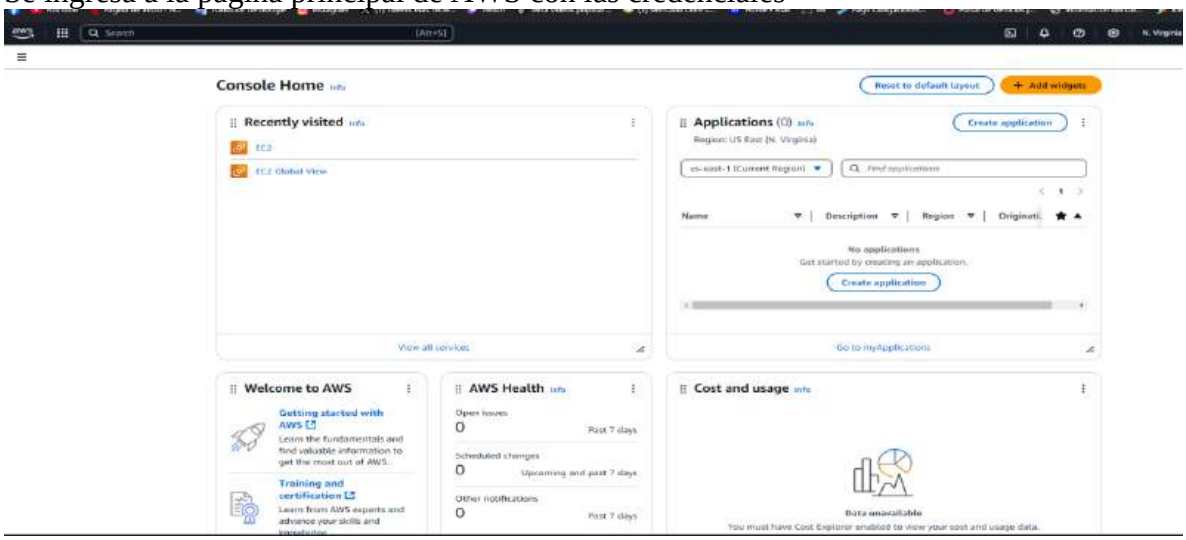


Figura 2. Luego vamos a todos los servicios:

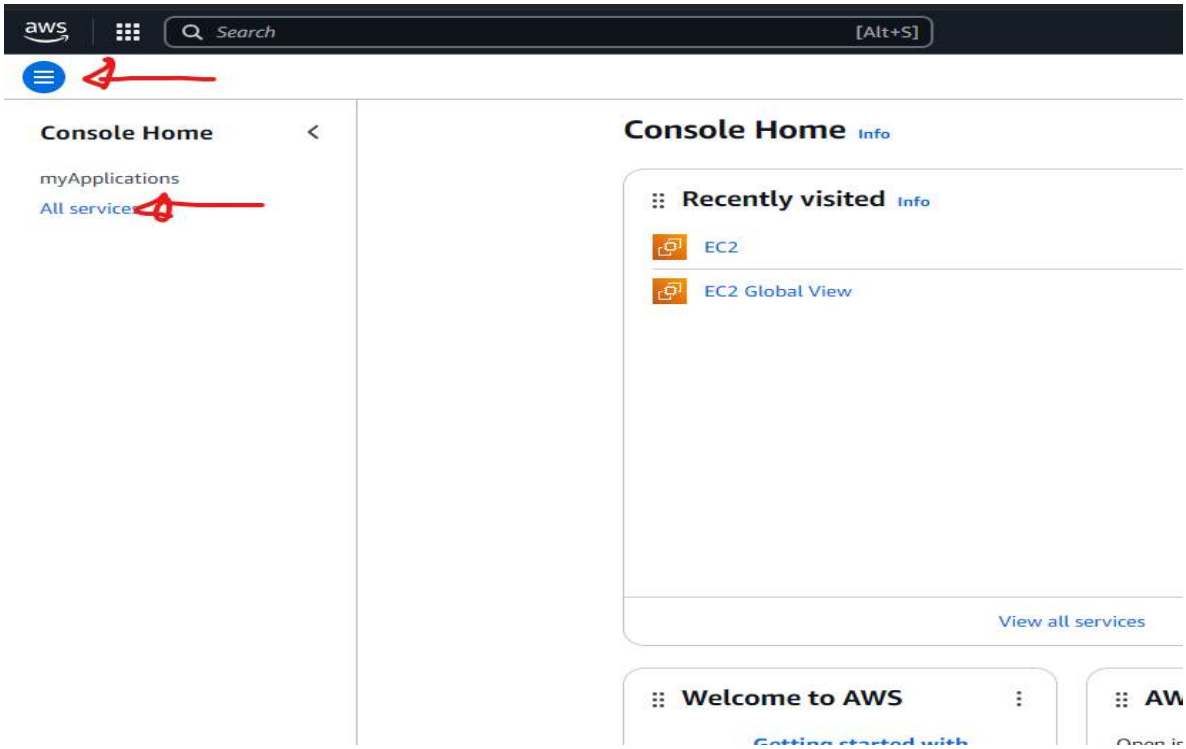


Figura 3. Luego bajamos hasta la parte de Networking & Content Delivery y buscamos la opción de VPC

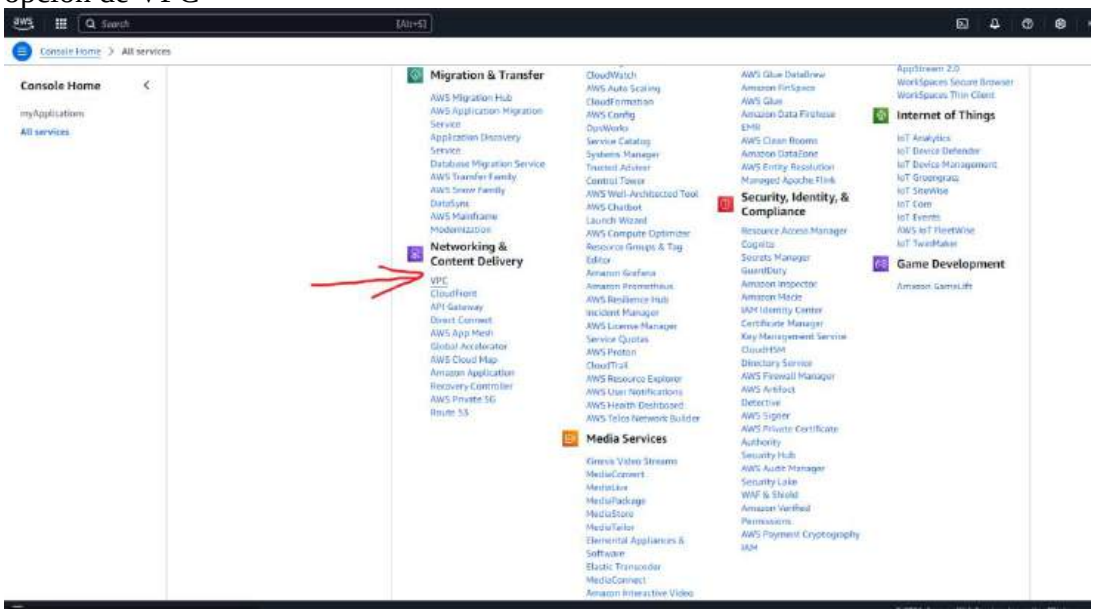


Figura 4. Luego vamos a donde dice “Your VPCs” y Create VPC
Aquí podemos elegir el nombre de la VPC, las zonas de disponibilidad y las subredes publicas y privadas

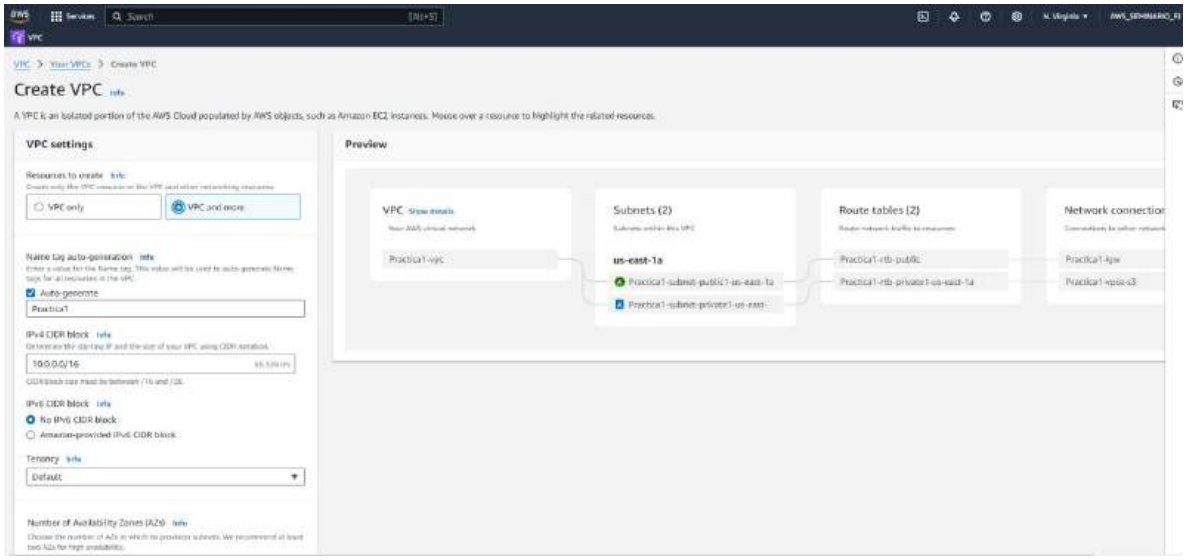


Figura 5. Una vez configurado a gusto, dar clic en Create VPC

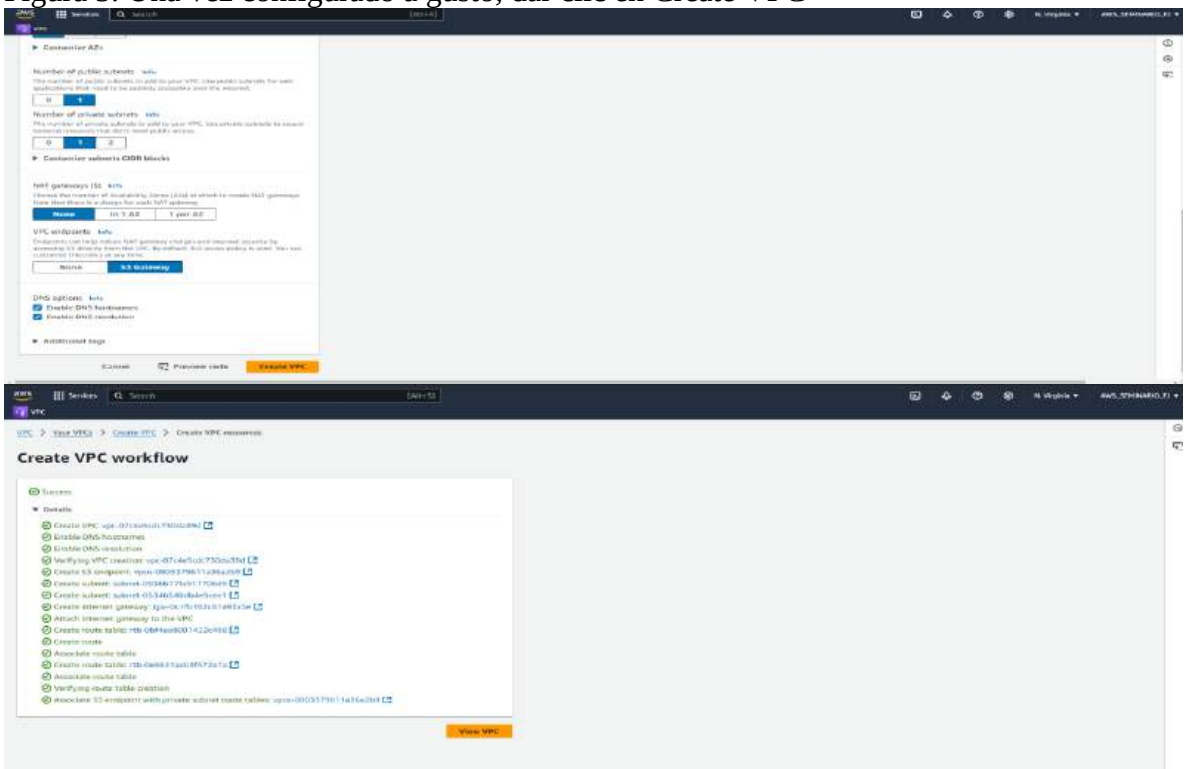


Figura 6. Una vez creado el VPC, debemos crear la instancia (Maquina virtual), para ello vamos a All Services y luego a EC2

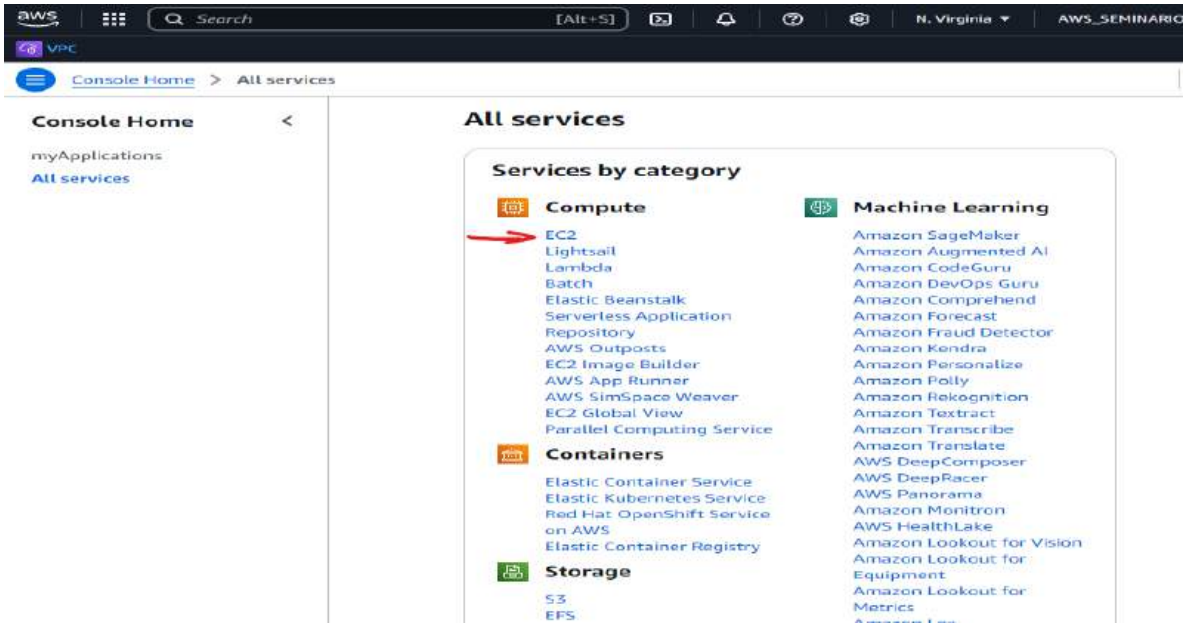


Figura 7. Luego vamos a Instancias, y luego a Launch Instances

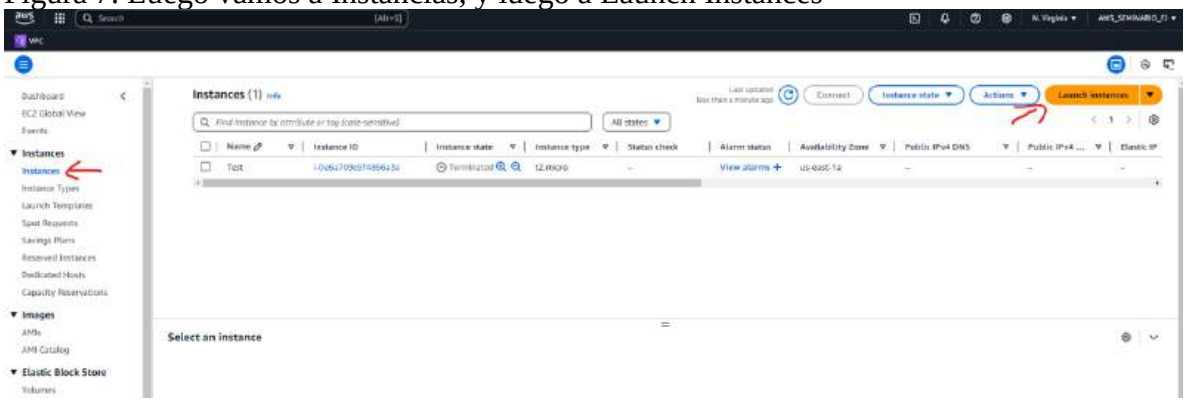


Figura 8. Aquí debemos realizar toda la configuración relacionada a nuestra instancia; Lo primero definir un nombre para la instancia, en nuestro caso InstanciaPractica1, luego se selecciona el sistema operativo con el que vamos a trabajar, para esta ocasión elegiremos Amazon Linux 2023 el cual es gratis, esto lo sabemos porque esta marcado como “Free tier eligible” y seleccionamos la arquitectura del sistema operativo X64

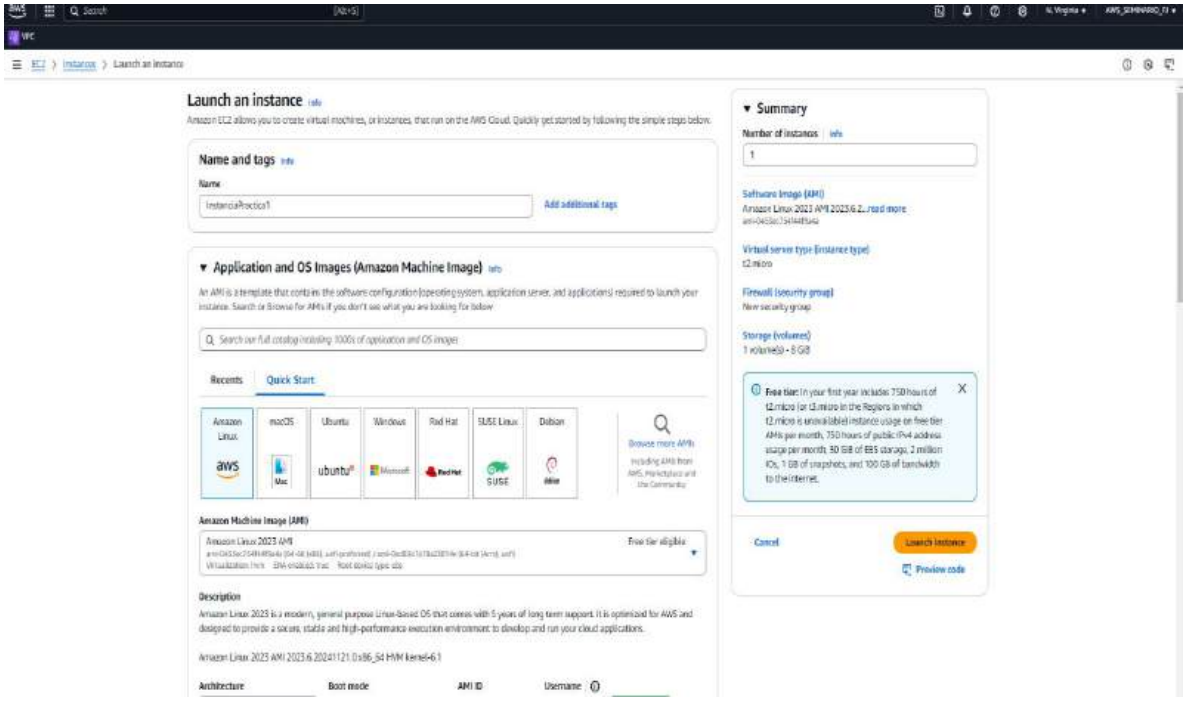


Figura 9. En Instance Type, se debe elegir las características de hardware de nuestra maquina virtual, en nuestro caso elegimos t2.micro que esta marcada como gratis (Free tier eligible)

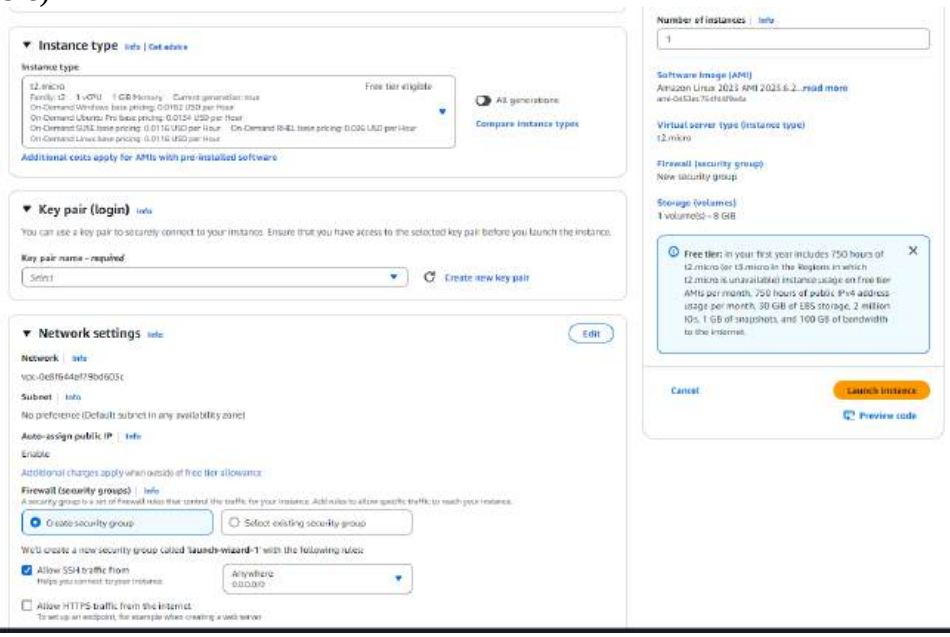


Figura 10. En Key Pair (Login) debemos crear o seleccionar el certificado de seguridad según lo necesitemos, en nuestro caso es un certificado con extensión .ppk debido a que nos conectaremos a una maquina Linux mediante el software Putty y debemos guardarlo en nuestro computador.

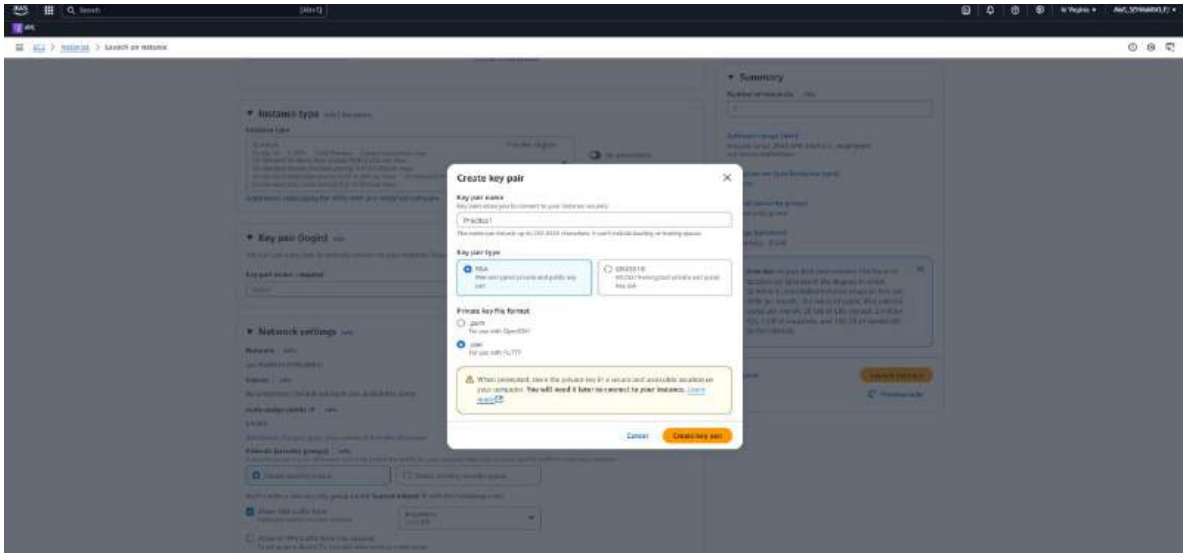


Figura 11. Luego sigue configurar la red (Network Settings)
Aquí crearemos o seleccionaremos el security group donde se establecen las reglas de entrada y salida de nuestra instancia, es una especie de cortafuegos
También se configurará el almacenamiento (Configure Storage) el cual dejaremos por defecto en 8 gb gp3. Luego damos clic en Launch Instance

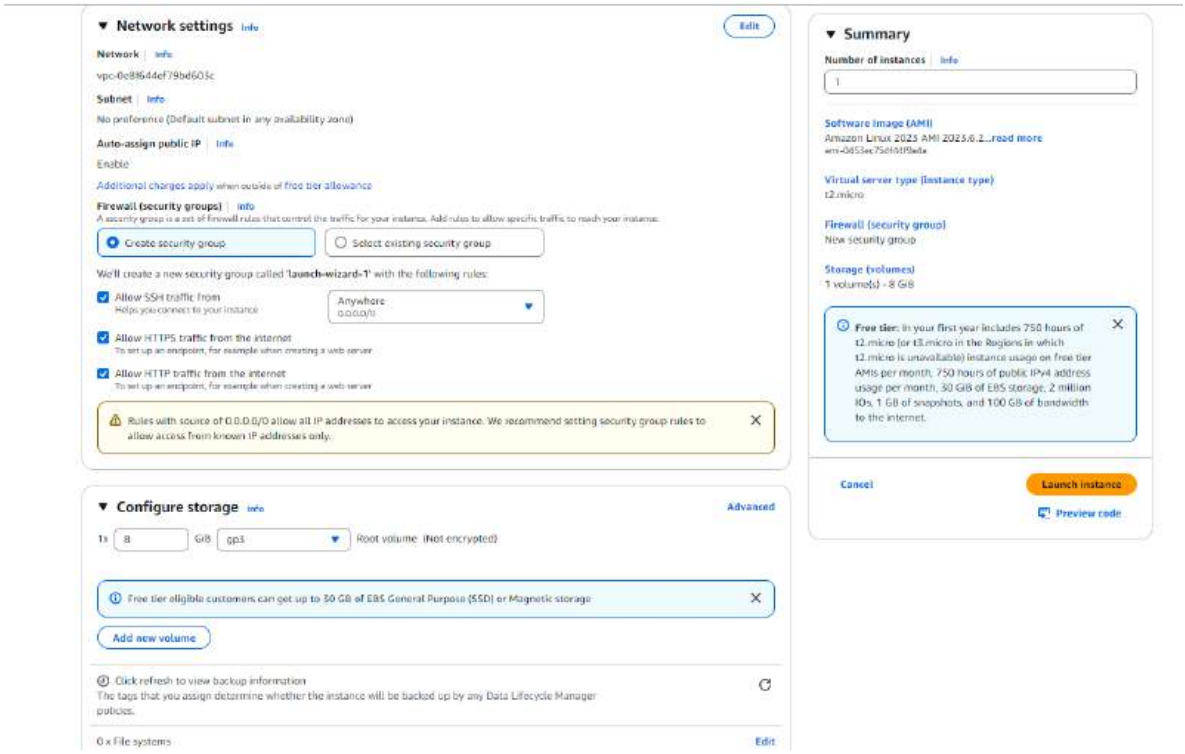


Figura 12. Aquí en instances, verificamos que nuestra instancia llamada InstanciaPractica1 esta creada y corriendo.

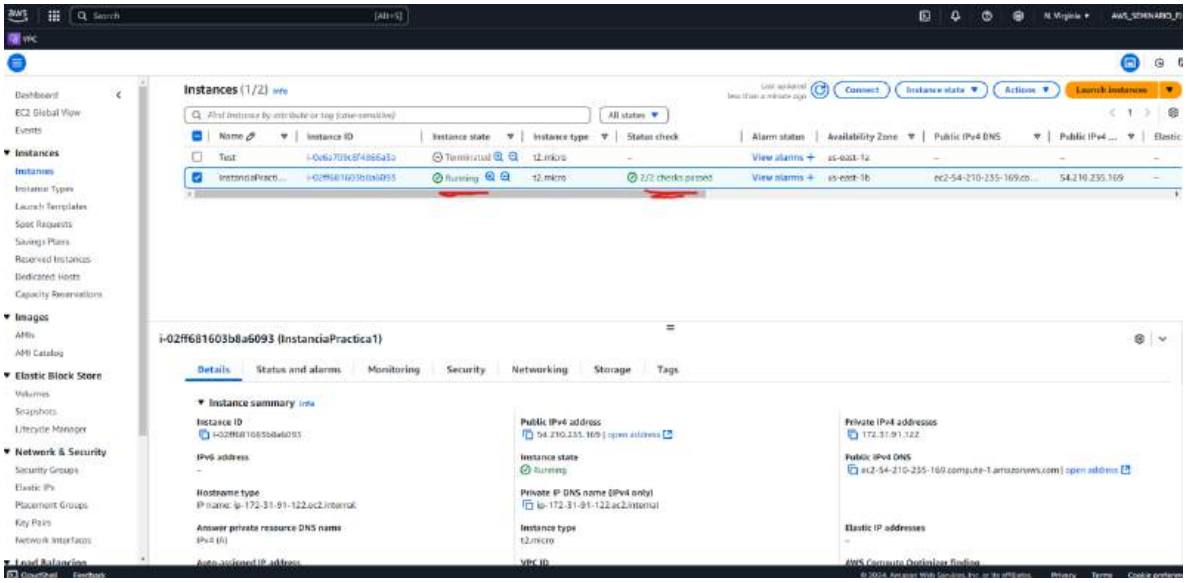


Figura 13. Si damos clic a nuestra instancia veremos información detallada, como la ip privada y publica

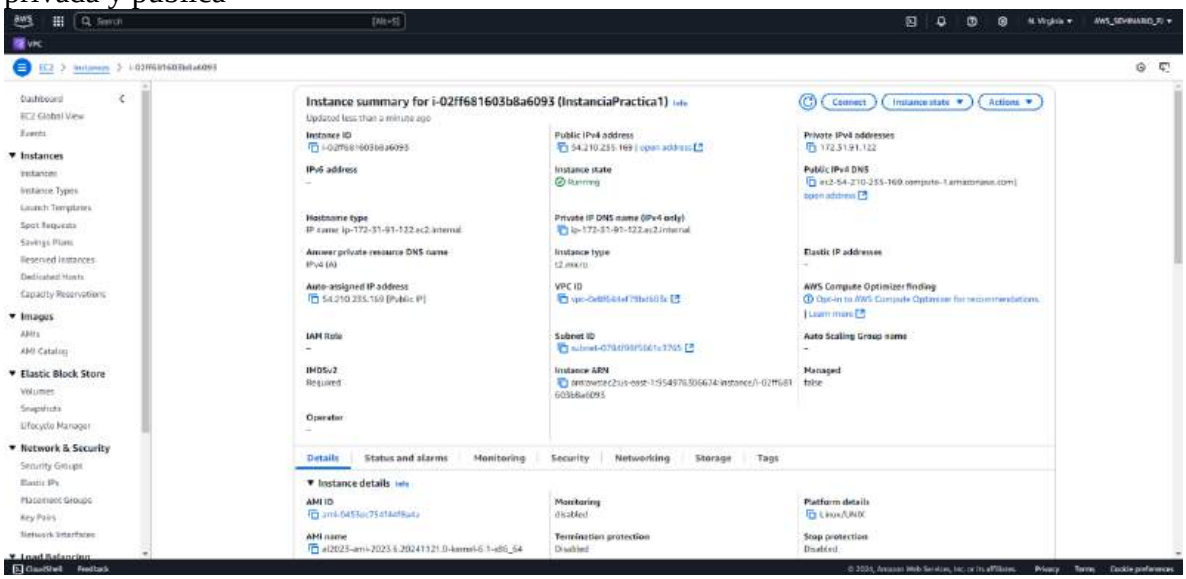


Figura 14. Tambien podemos ver la información de conexión a nuestra instancia :

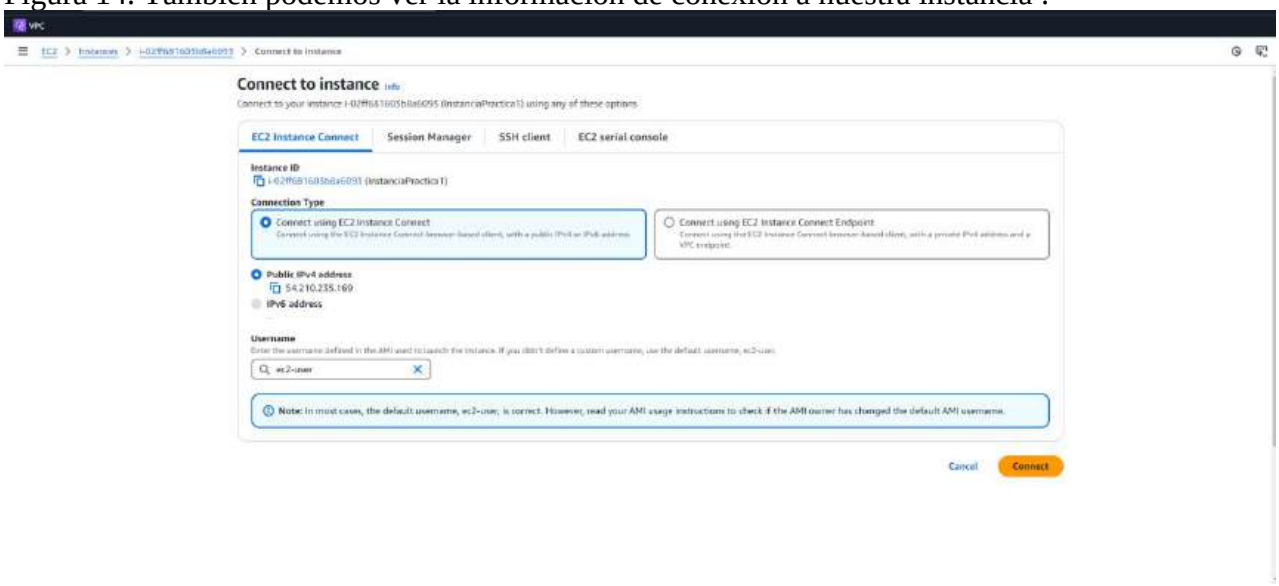


Figura 15. Para conectarnos a nuestra instancia utilizaremos el protocolo SSH por medio del programa Putty en el cual ingresamos la ip publica y nos conectamos por el puerto 22:

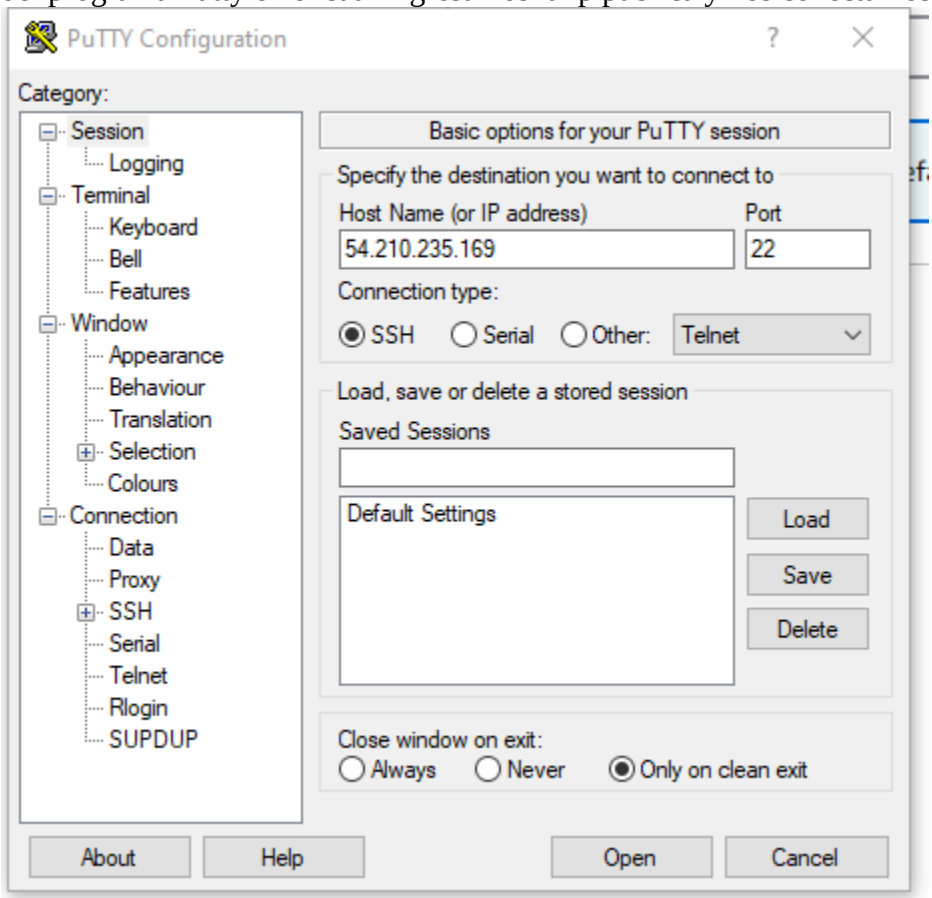
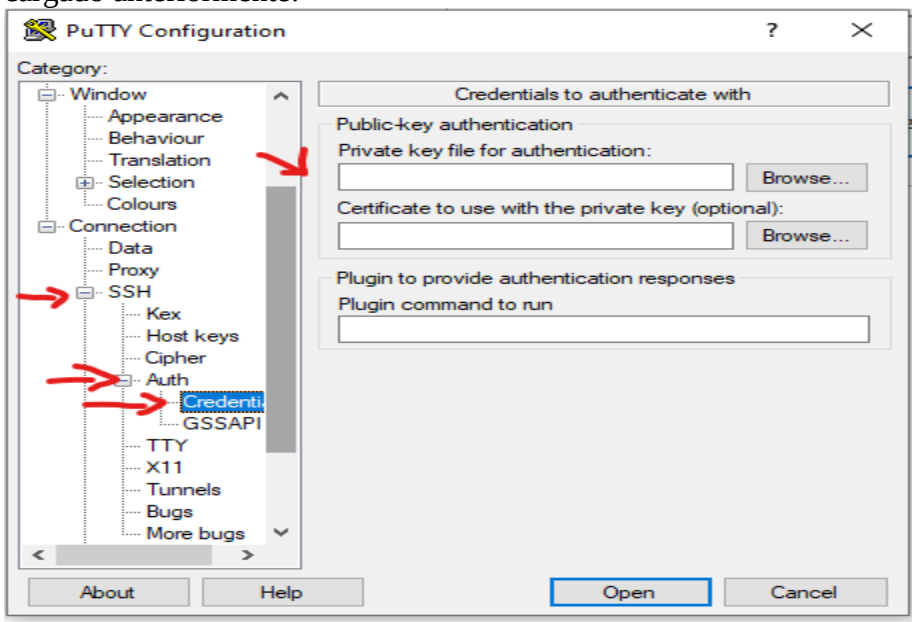


Figura 16. Luego en el menú de la izquierda SSH > Ath > Credentials y allí en la parte de Private key file for authentication cargamos el certificado de seguridad que habíamos cargado anteriormente:



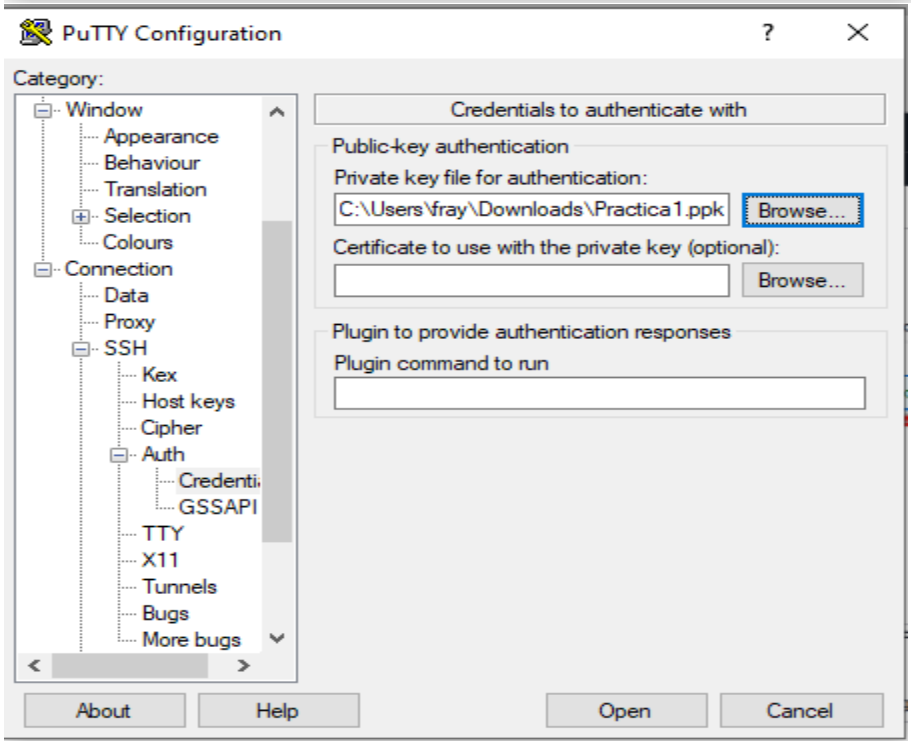


Figura 17. Luego aceptamos la alerta de seguridad de putty y estaremos conectados a nuestra isntancia el Amazon Linux

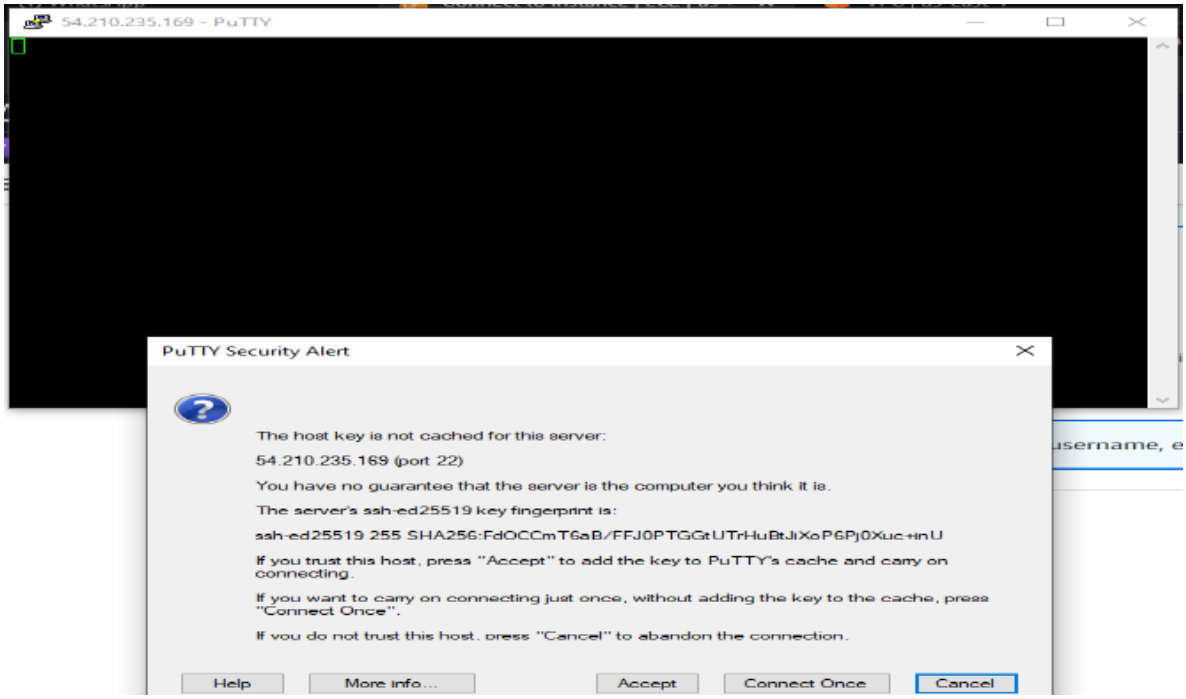


Figura 18. Aquí debemos ingresar con el usuario que vimos anteriormente en la parte de coneccion:

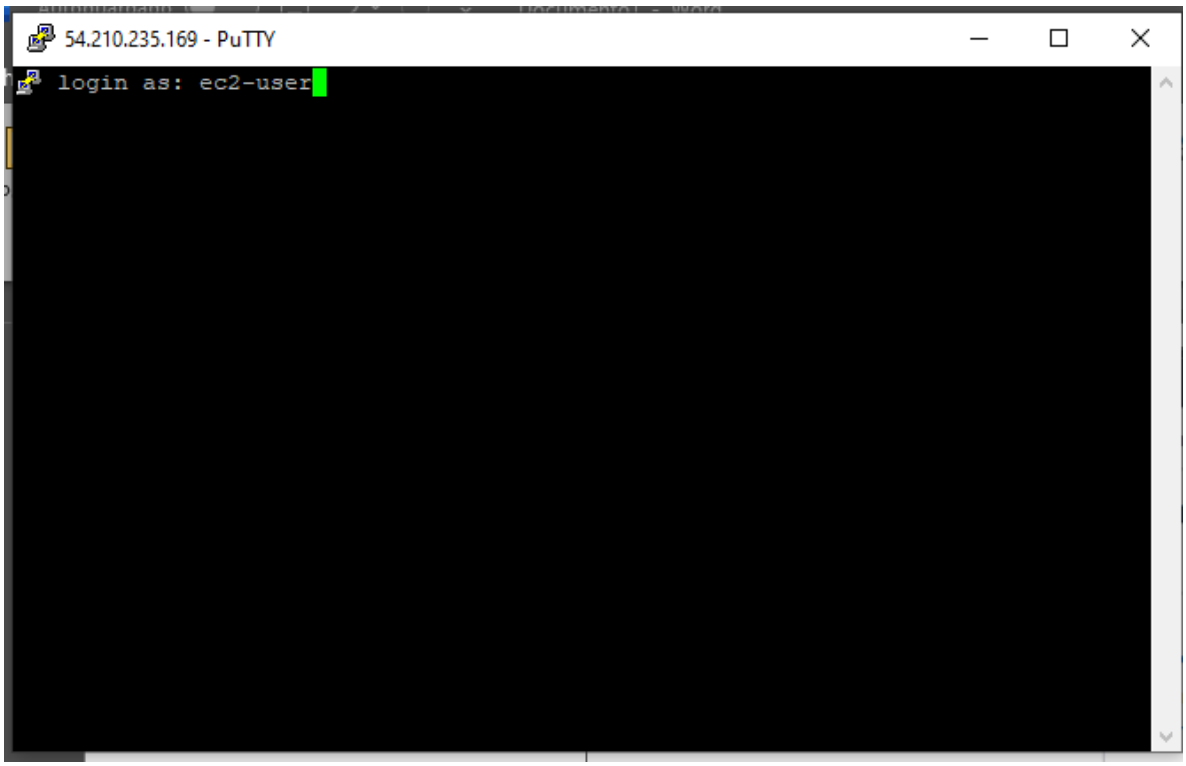


Figura 19. Una vez acepte las credenciales veremos algo como lo siguiente:

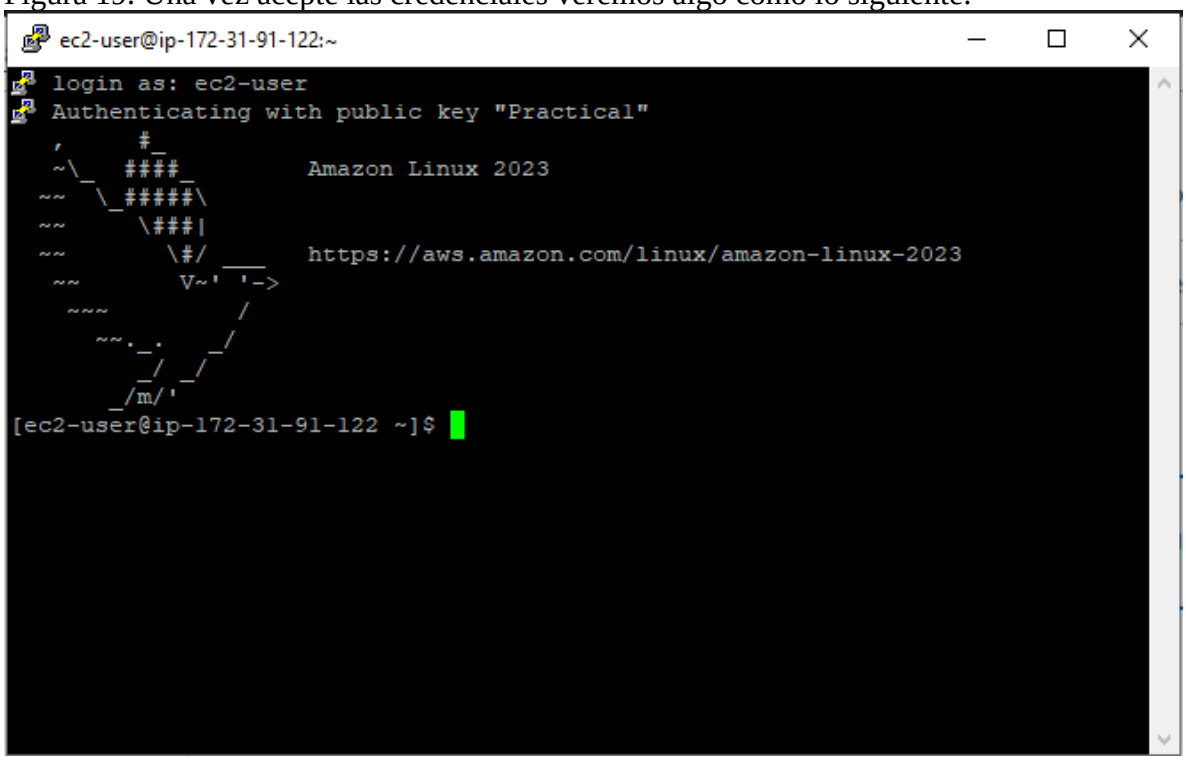
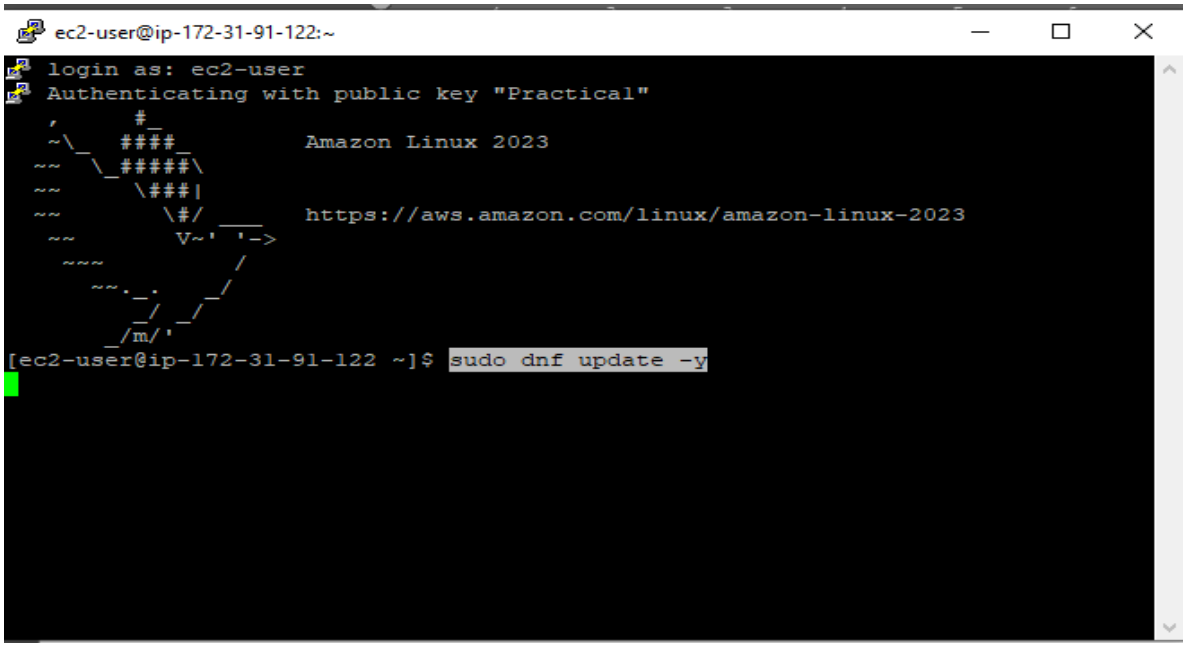


Figura 20. Desde aquí demos hacer las configuraciones que se requieran para crear nuestro servidor web. Lo primero que se realizará es actualizar el sismeta operativo para posteriormente instalar un software llamado Apache el cual nos permite crear un servidor web en linux.



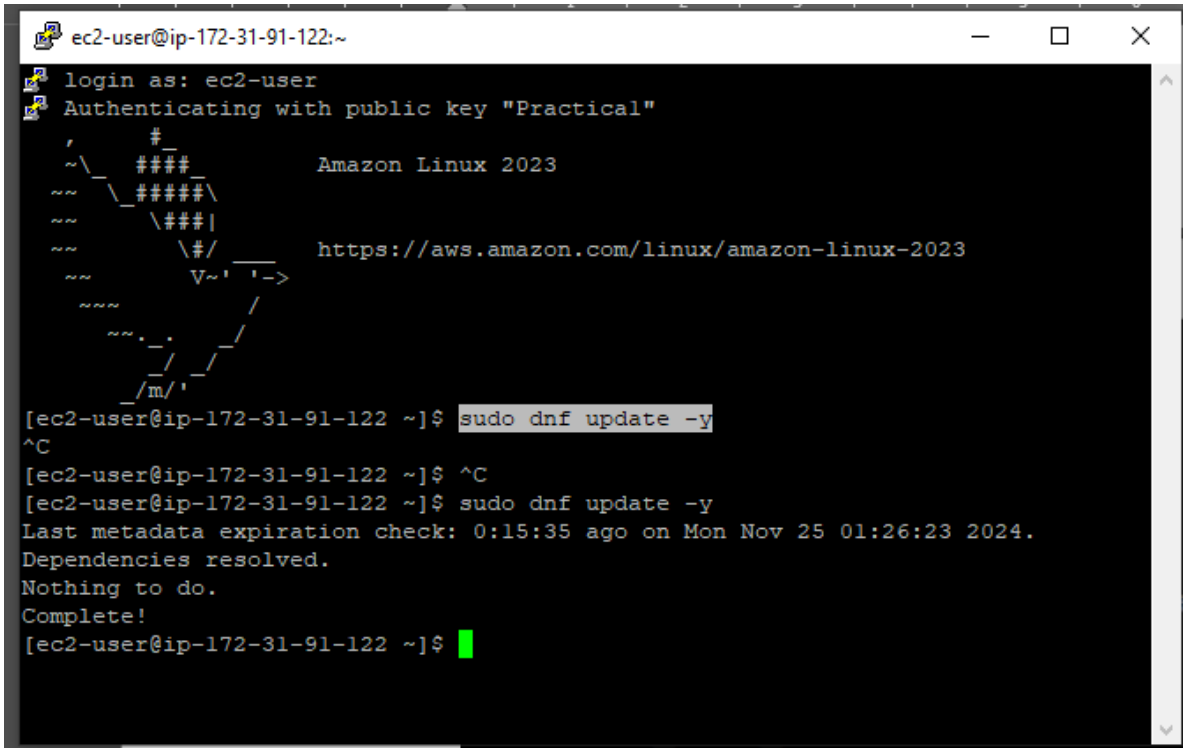
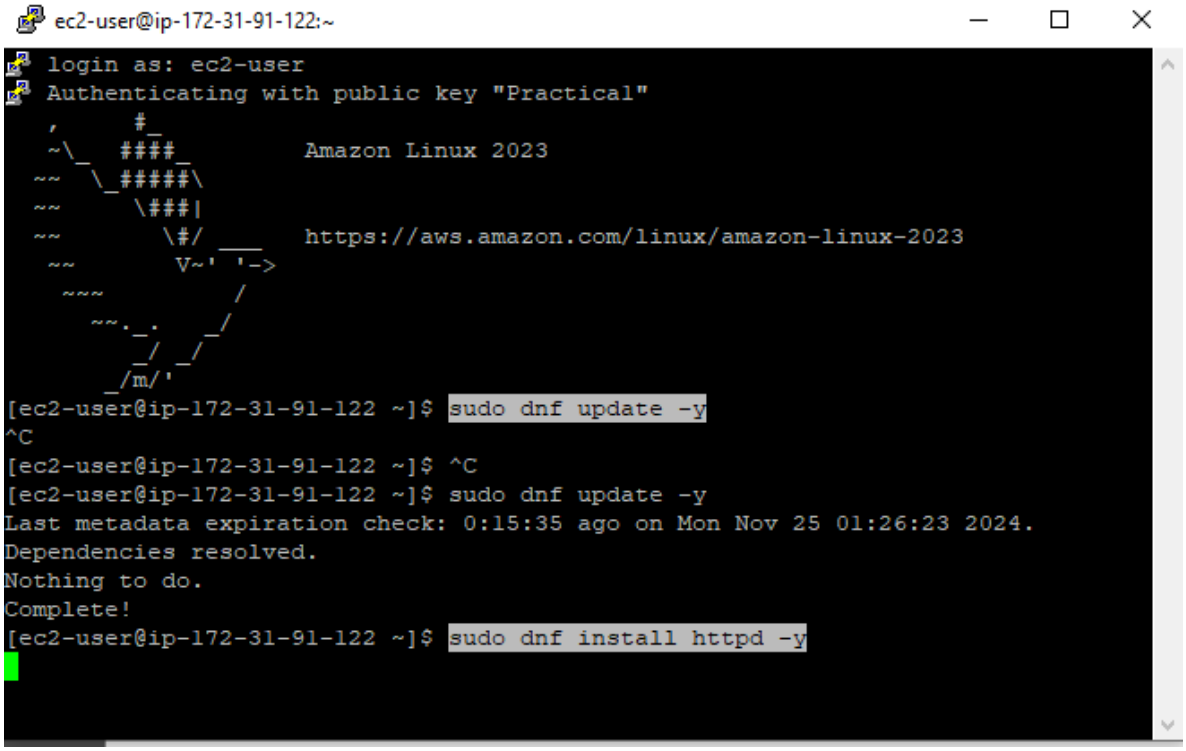
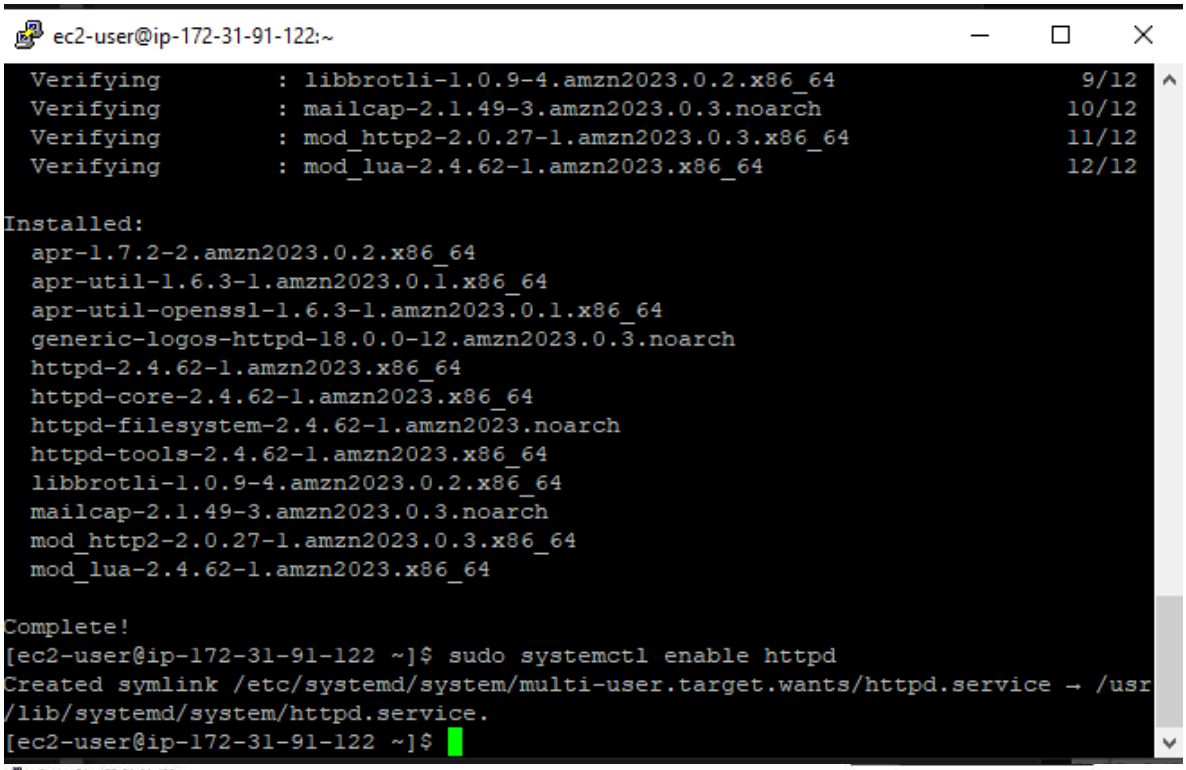


Figura 21. Luego instalamos apache, habilitamos el servicio, luego iniciamos el servicio, y verificamos que este funcionando correctamente:




ec2-user@ip-172-31-91-122:~
Verifying : libbrotli-1.0.9-4.amzn2023.0.2.x86_64 9/12
Verifying : mailcap-2.1.49-3.amzn2023.0.3.noarch 10/12
Verifying : mod_http2-2.0.27-1.amzn2023.0.3.x86_64 11/12
Verifying : mod_lua-2.4.62-1.amzn2023.x86_64 12/12
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!
[ec2-user@ip-172-31-91-122 ~]$ sudo systemctl enable httpd
Created symlink /etc/systemd/system/multi-user.target.wants/httpd.service → /usr/lib/systemd/system/httpd.service.
[ec2-user@ip-172-31-91-122 ~]$

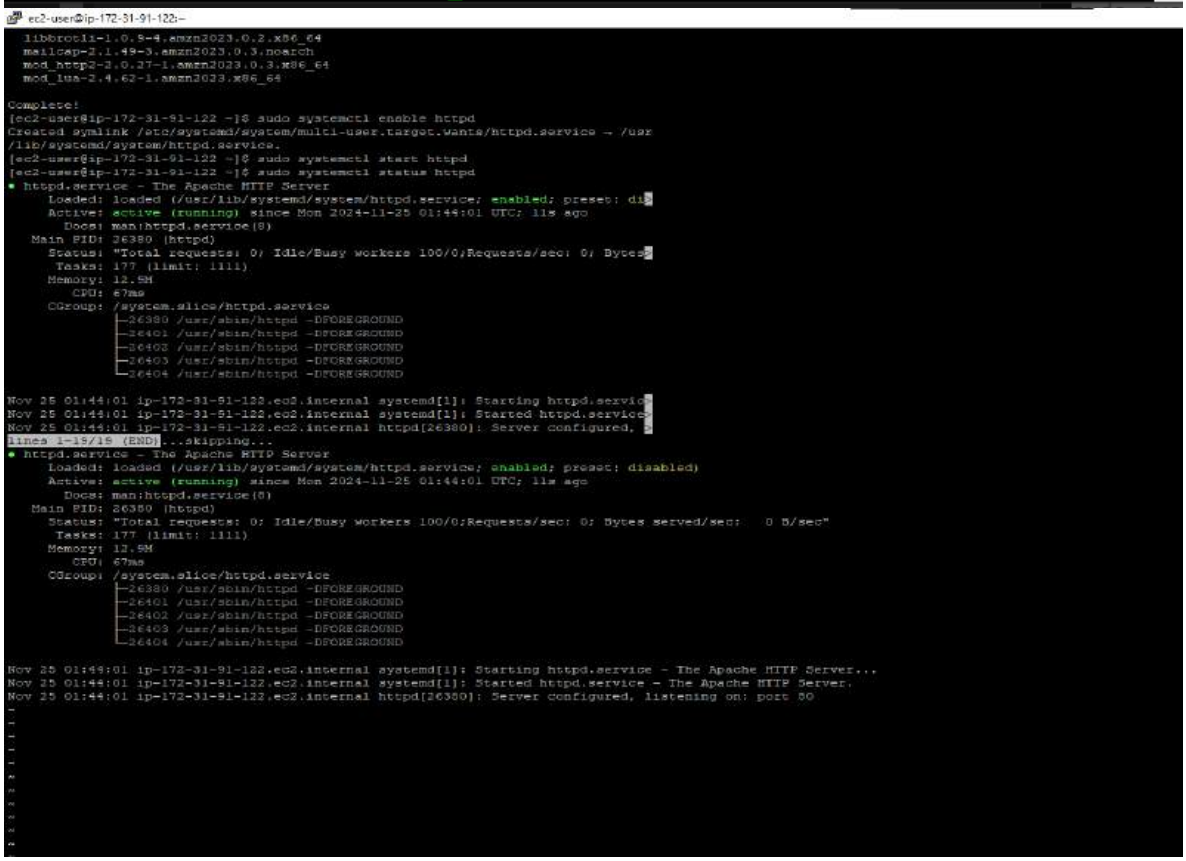

ec2-user@ip-172-31-91-122:~
Complete!
[ec2-user@ip-172-31-91-122 ~]$ sudo systemctl enable httpd
Created symlink /etc/systemd/system/multi-user.target.wants/httpd.service → /usr/lib/systemd/system/httpd.service.
[ec2-user@ip-172-31-91-122 ~]$ sudo systemctl start httpd
[ec2-user@ip-172-31-91-122 ~]$ sudo systemctl status httpd
● httpd.service - The Apache HTTP Server
Loaded: loaded (/usr/lib/systemd/system/httpd.service; enabled; preset: disabled)
Active: active (running) since Mon 2024-11-25 01:44:01 UTC; 11s ago
Docs: man:httpd.service(8)
Main PID: 26380 (httpd)
Status: "Total requests: 0; Idle/Busy workers 100/0;Requests/sec: 0; Bytes served/sec: 0 B/sec"
Tasks: 177 (limit: 1111)
Memory: 12.9M
CPU: 67ms
CGroup: /system.slice/httpd.service
├─26380 /usr/sbin/httpd -DFOREGROUND
├─26401 /usr/sbin/httpd -DFOREGROUND
├─26402 /usr/sbin/httpd -DFOREGROUND
├─26403 /usr/sbin/httpd -DFOREGROUND
└─26404 /usr/sbin/httpd -DFOREGROUND
Nov 25 01:44:01 ip-172-31-91-122.ec2.internal systemd[1]: Starting httpd.service - The Apache HTTP Server...
Nov 25 01:44:01 ip-172-31-91-122.ec2.internal systemd[1]: Started httpd.service - The Apache HTTP Server.
Nov 25 01:44:01 ip-172-31-91-122.ec2.internal httpd[26380]: Server configured, listening on: port 80
lines 1-19/19 (END)...skipping...
● httpd.service - The Apache HTTP Server
Loaded: loaded (/usr/lib/systemd/system/httpd.service; enabled; preset: disabled)
Active: active (running) since Mon 2024-11-25 01:44:01 UTC; 11s ago
Docs: man:httpd.service(8)
Main PID: 26380 (httpd)
Status: "Total requests: 0; Idle/Busy workers 100/0;Requests/sec: 0; Bytes served/sec: 0 B/sec"
Tasks: 177 (limit: 1111)
Memory: 12.9M
CPU: 67ms
CGroup: /system.slice/httpd.service
├─26380 /usr/sbin/httpd -DFOREGROUND
├─26401 /usr/sbin/httpd -DFOREGROUND
├─26402 /usr/sbin/httpd -DFOREGROUND
├─26403 /usr/sbin/httpd -DFOREGROUND
└─26404 /usr/sbin/httpd -DFOREGROUND
Nov 25 01:44:01 ip-172-31-91-122.ec2.internal systemd[1]: Starting httpd.service - The Apache HTTP Server...
Nov 25 01:44:01 ip-172-31-91-122.ec2.internal systemd[1]: Started httpd.service - The Apache HTTP Server.
Nov 25 01:44:01 ip-172-31-91-122.ec2.internal httpd[26380]: Server configured, listening on: port 80

Figura 22. Tambien debemos verificar que en Security groups de AWS tengamos las reglas configuradas correctamente para que podamos acceder a nuestro servidor web desde internet, debemos permitir acceso al puerto HTTP 80 a todos los usuarios o solo a las ip que nosotros queramos:

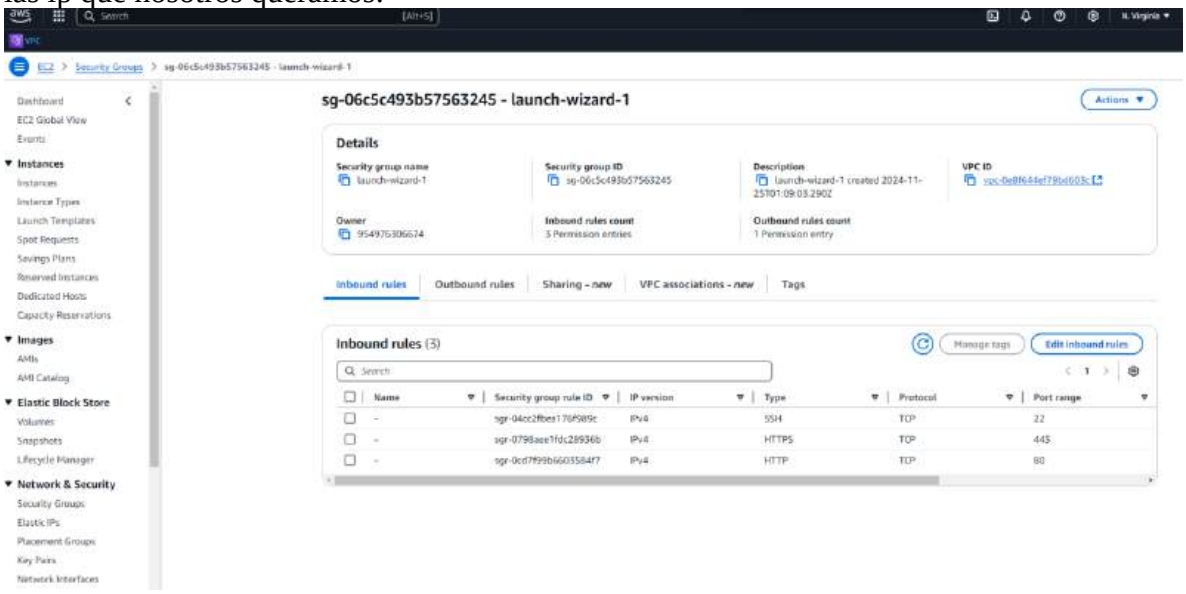
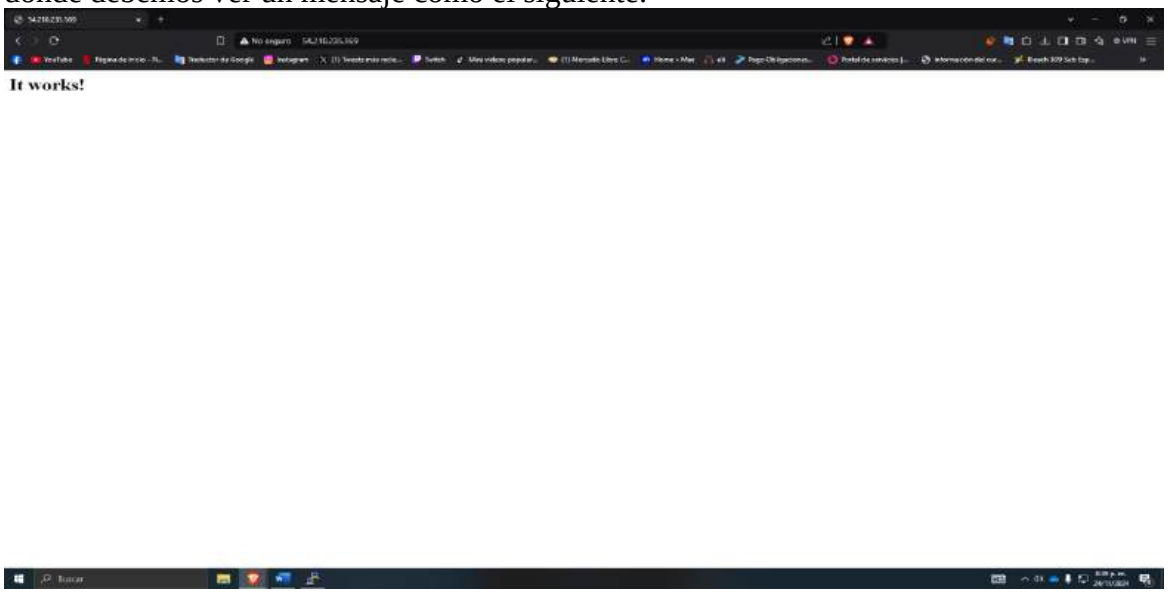


Figura 23. Para verificar que nuestro servidor web esta funcionando correctamente lo unico que debemos hacer es conectarnos a nuestra IP publica desde cualquier navegador donde debemos ver un mensaje como el siguiente:



Entrega 2

1. Crear una VPC con dos subredes públicas

VPC: Define un rango de direcciones CIDR (por ejemplo, 10.0.0.0/16).
Subredes públicas: Crea dos subredes públicas en diferentes zonas de disponibilidad (por ejemplo, 10.0.1.0/24 y 10.0.2.0/24).
Gateway de Internet: Asocia un Gateway de Internet a la VPC para habilitar el acceso a internet.
Tablas de rutas: Configura la tabla de rutas para las subredes públicas para enrutar el tráfico hacia el Gateway de Internet.

2. Grupos de seguridad:

Grupo de seguridad para el Load Balancer:

Los balanceadores de carga de aplicación se utiliza para aplicaciones web cuando el destino son aplicaciones que se ejecutan en protocolo HTTP y HTTPS. Hay otro balanceadores de carga que se llama de red donde se puede poner cualquier protocolo ejm telefonía IP.

Los grupos de seguridad para el Load Balancer :

Permitir tráfico entrante en los puertos 80 (HTTP) y 443 (HTTPS) desde cualquier origen (0.0.0.0/0).
Permitir todo el tráfico saliente.
Grupo de seguridad para las instancias:
Permitir tráfico entrante desde el grupo de seguridad del Load Balancer en el puerto correspondiente a la aplicación (por ejemplo, 80).
Permitir todo el tráfico saliente.

3. AMI (Amazon Machine Image)

Crea una AMI a partir de una instancia preconfigurada con el software y las dependencias necesarias.

4. Plantilla de lanzamiento (Launch Template):

Es el que me va permitir definir en un archivo cuales con las características de hardware y de sistema operativo que van a tener la maquina: a nivel de sistema operativo se hace con la AMI , en la parte de Launch Template Le vamos a definIr la CPU, RAM, así:

Configura una plantilla de lanzamiento que incluya:

AMI previamente creada.
Tipo de instancia (por ejemplo, t2.micro).
Grupo de seguridad para las instancias.
Par de claves (opcional, para acceso SSH).
Configuración de red asociada a las subredes públicas.

5. Instancias

Lanza dos instancias utilizando la plantilla de lanzamiento configurada. (Esto también puede ser gestionado mediante Auto Scaling, se puede observar en el siguiente paso).

6. Grupos de destino (Target Groups):

Es un grupo objetivo, son los destinos donde vamos a enviar las peticiones.

Crea un grupo de destino para las instancias.
Tipo de destino: Instancias.
Puerto: 80 (o el puerto utilizado por la aplicación).
Configura comprobaciones de estado para asegurar que el tráfico solo se envíe a instancias saludables

7. Load Balancer:

Es mejorar la eficiencia y la tolerancia a fallos de la aplicación, El listeners es la configuración que va a ir a buscar el balanceador de carga, es el que escucha las peticiones de los puertos

Crea un Application Load Balancer (ALB):
Asocia el grupo de seguridad del Load Balancer.
Configura listeners en los puertos 80 y 443.
Asocia las subredes públicas.
Conecta el Load Balancer al grupo de destino.

8. Grupo de Auto Scaling:
Es una configuración automática para ese balanceador de carga

Crea un grupo de Auto Scaling:
Asocia la plantilla de lanzamiento.
Define un mínimo de 2 instancias y un máximo según sea necesario.
Distribuye las instancias en las subredes públicas.
Asocia el grupo de destino para garantizar que las nuevas instancias se registren automáticamente en el Load Balancer.

Entrega 2
figura 2.1.

Figura 1. Creamos una VPC seleccionamos la opción “VPC or more” la cual tendrá la opción de 2 zonas disponible, esto será necesario para el funcionamiento del balanceador de carga, la demás configuración se puede cambiar de acuerdo a la necesidad.

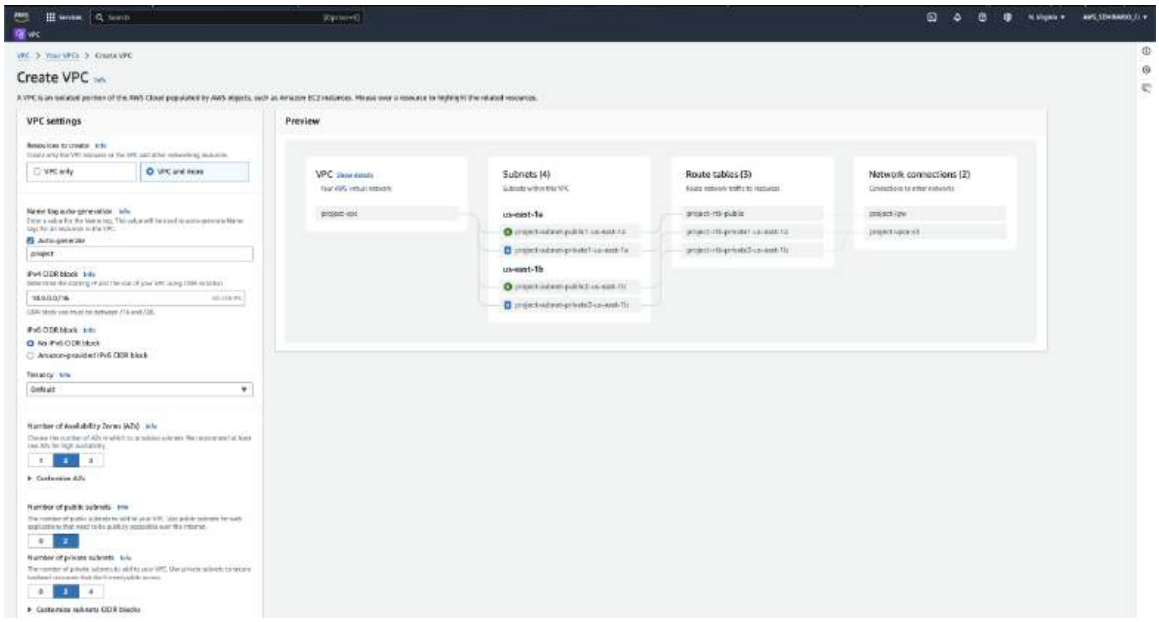


Figura 2: Luego crearemos una instancia con todas las configuraciones necesarias, lo ideal es que esta configuración quede completa ya que así podremos crear una AMI la cual nos permitirá crear otras instancias de una forma más fácil al usar nuestra propia AMI y nos ayudara más adelante para crear el balanceador de carga de forma automática.

Procederemos a crear al AMI en base a nuestra instancia:

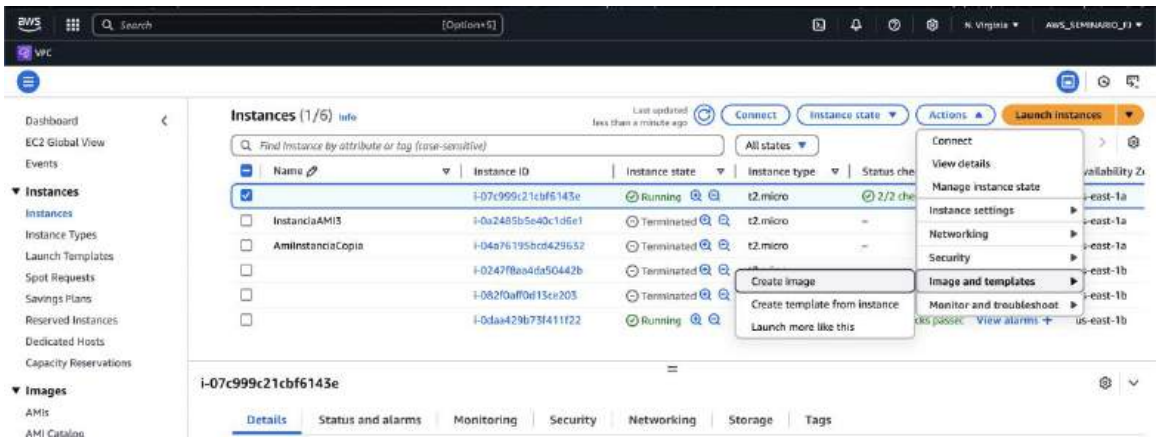


Figura 3: Le daremos un nombre y el espacio de almacenamiento que usara, en este caso tomara por defecto los de la instancia actual.

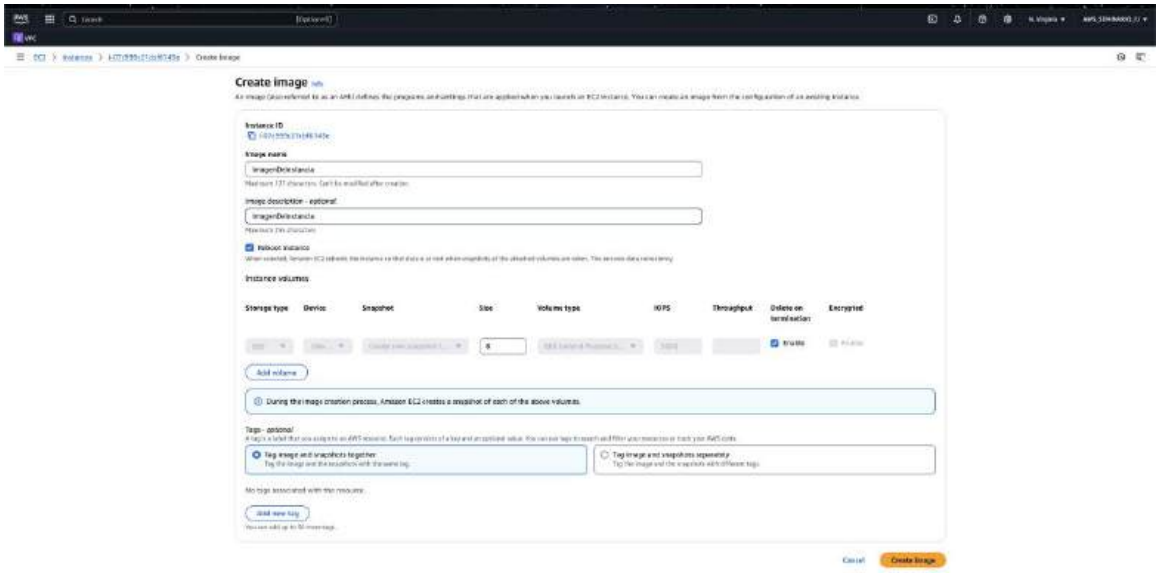


Figura 4: Antes de comenzar con la creación automática del balanceador de carga crearemos otra instancia manualmente, usando nuestra AMI.

Una vez completado este proceso, podemos ver que tenemos 2 instancias que han iniciado el mismo servicio, pero en diferentes servidores, por lo que si entramos a la IP publica de cada instancia desde un navegador veremos la misma información.

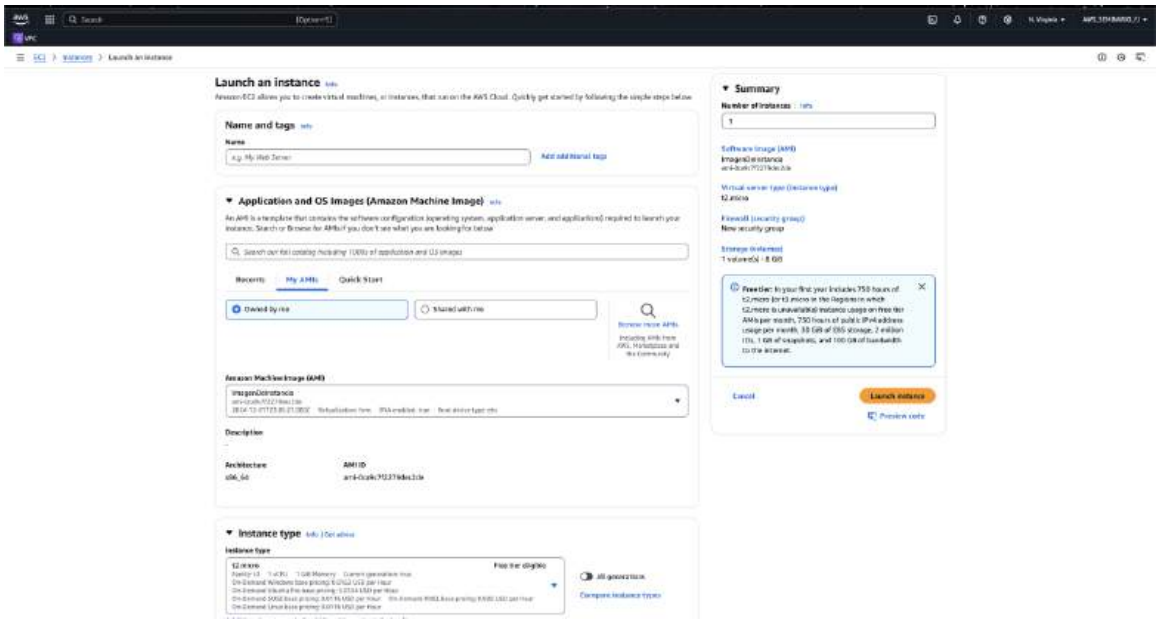


Figura 5: Ahora crearemos un Security Group para controlar las comunicaciones de internet de las instancias dentro del balanceador de carga.

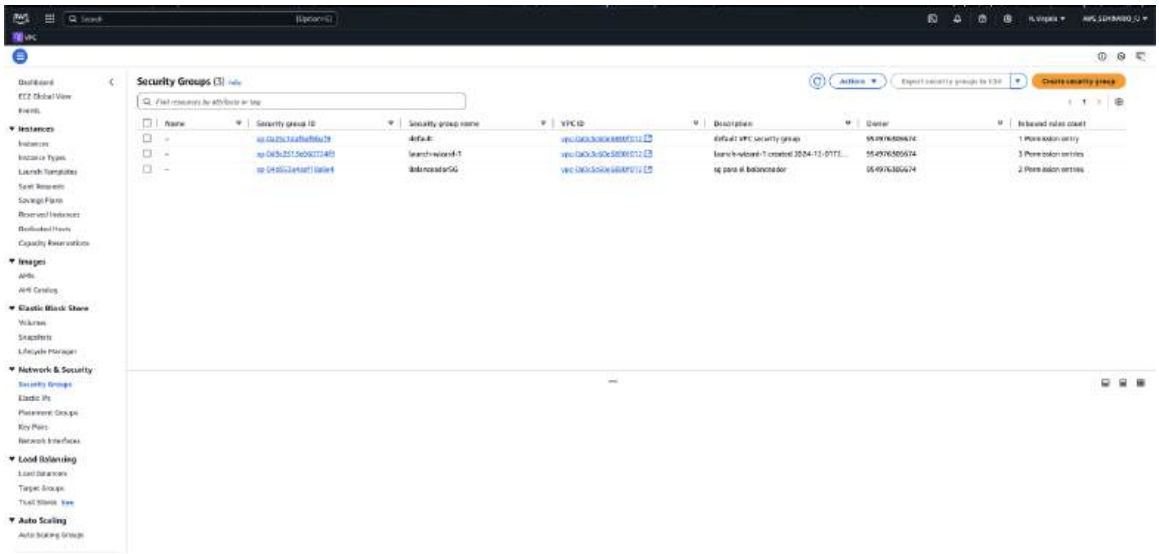


Figura 6: En este caso indicaremos que se permitirá cualquier conexión de salida, pero solo se permitirán las conexiones de entrada por el puerto 80 y 22 desde cualquier IP.

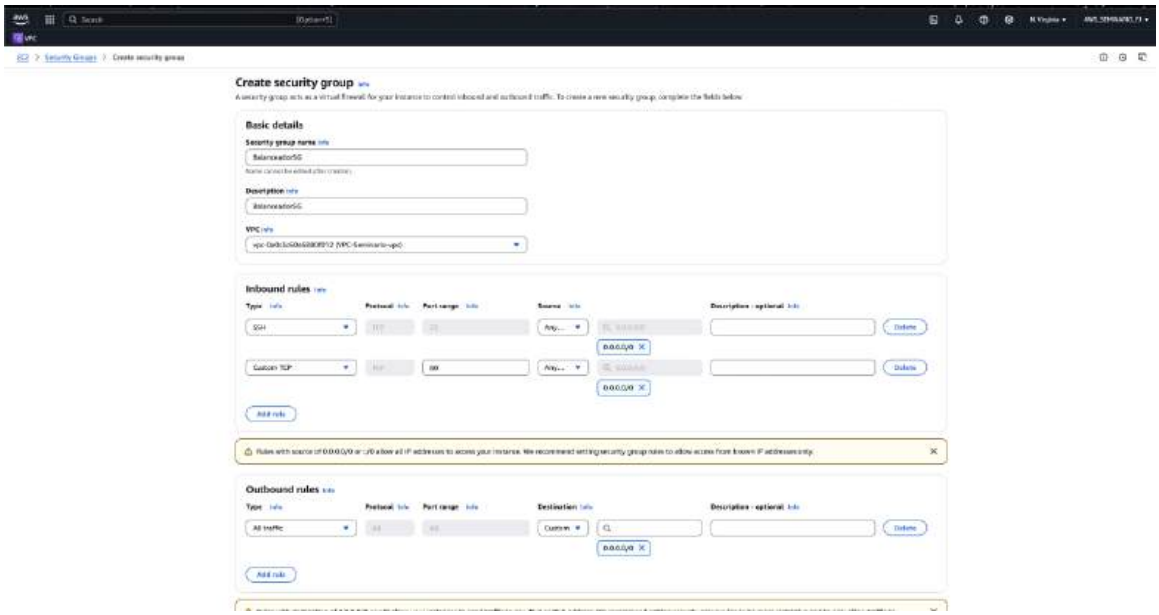


Figura 7: Crearemos un Target Group desde las opciones.

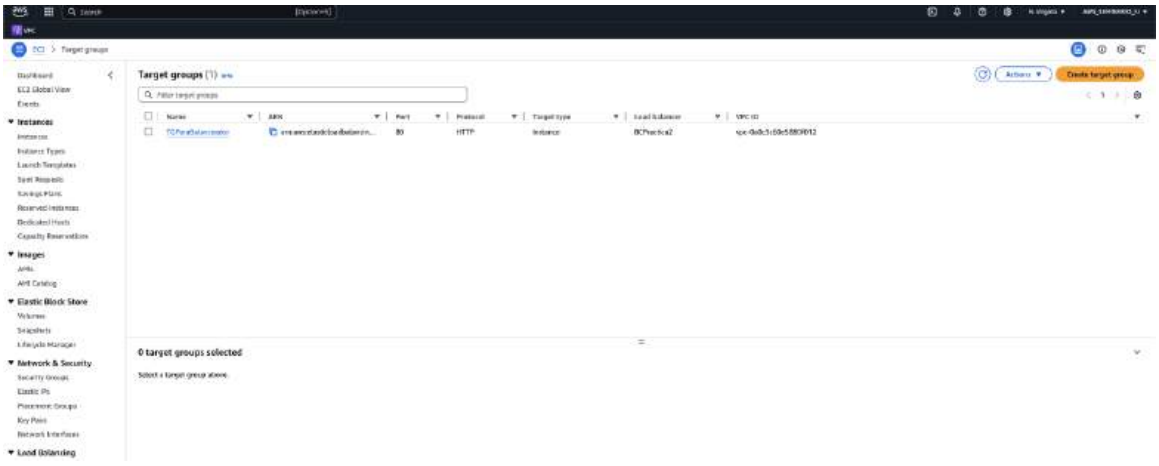


Figura 8: Indicaremos que el target type serán las instancias que hemos creado, y las transacciones que se van a enrutas en nuestro balanceador de carga serán aquellas que pasen por el puerto 80.

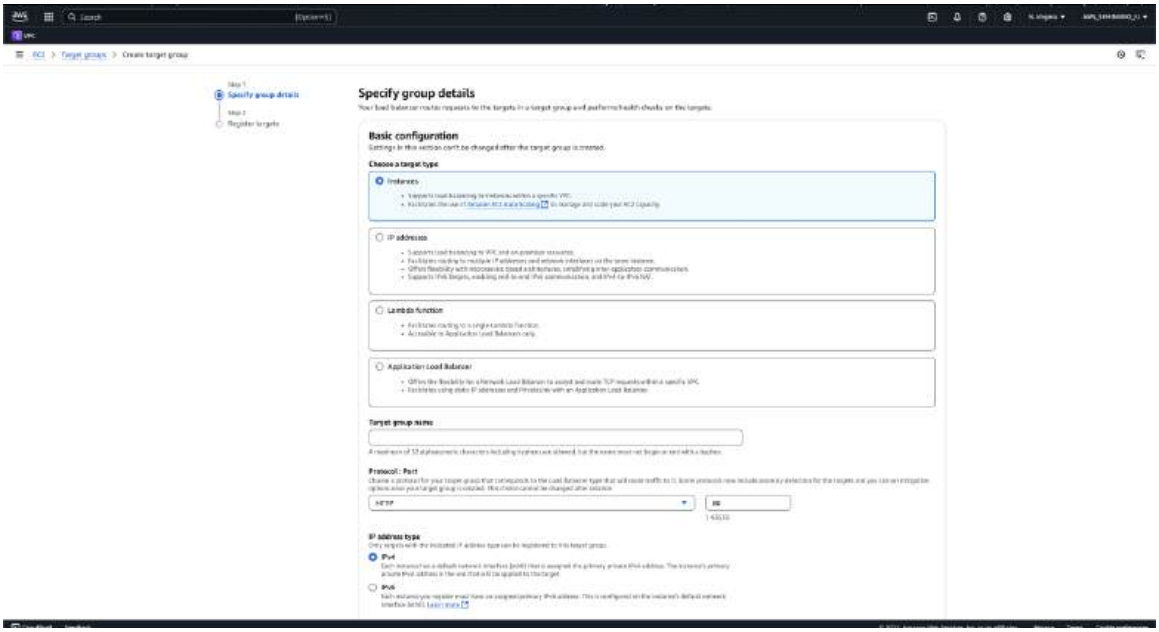


Figura 9: Continuando con la creación del target Group, indicaremos la VPC en la cual se encuentran las instancias para nuestro balanceador de carga, y finalmente indicaremos la ruta desde la cual se realizará el health check de nuestras instancias esto nos ayudará a verificar el estado de nuestras instancias para tomar ciertas medidas en caso de fallos.

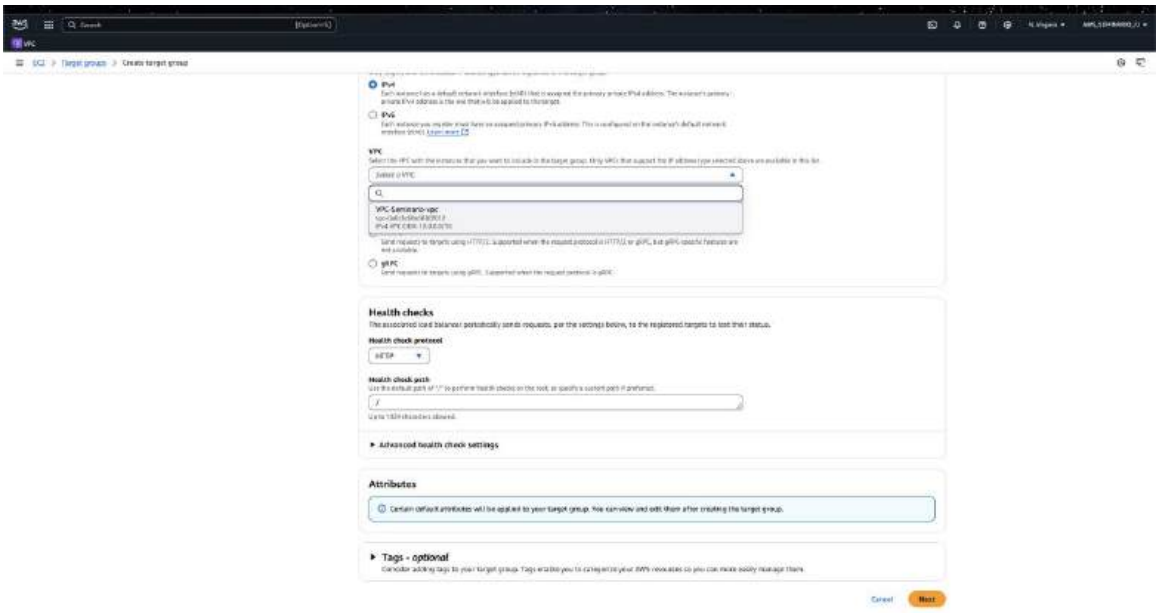


Figura 10: Ahora crearemos nuestro balanceador de carga desde la opción.

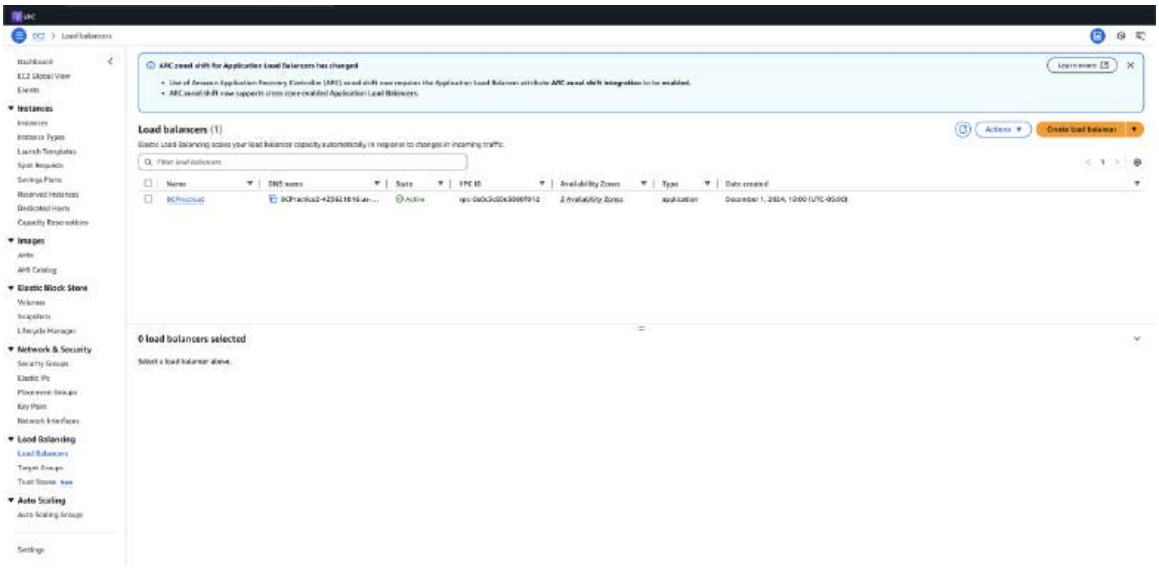


Figura 11: En este caso seleccionaremos la primera opción Aplication Load Balancer que aplicara para el tipo de peticiones que vamos a manejar.

Figura 12: Al momento de crear nuestro balanceador indicaremos la VPC en la cual trabajaran las instancias que ya existan y las que puedan existir.

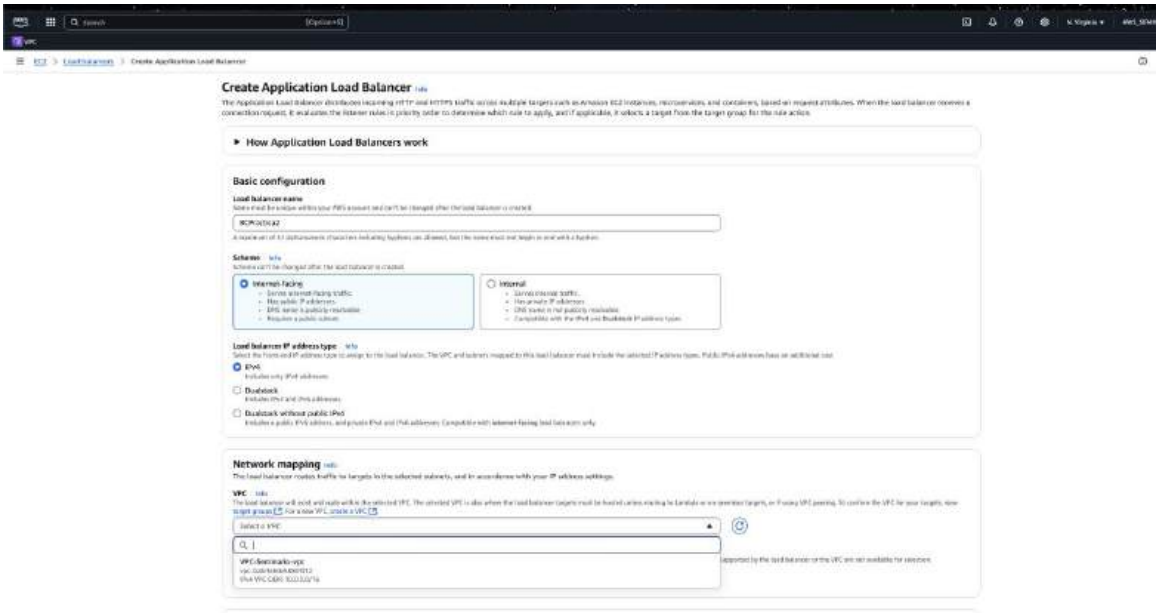


Figura 13: Indicaremos el Security Group al que pertenecerán las instancias, finalmente en Listeners and routing seleccionaremos el protocolo y el puertos al que le vamos a interceptar las peticiones, y seleccionaremos el target Group que creamos anteriormente.

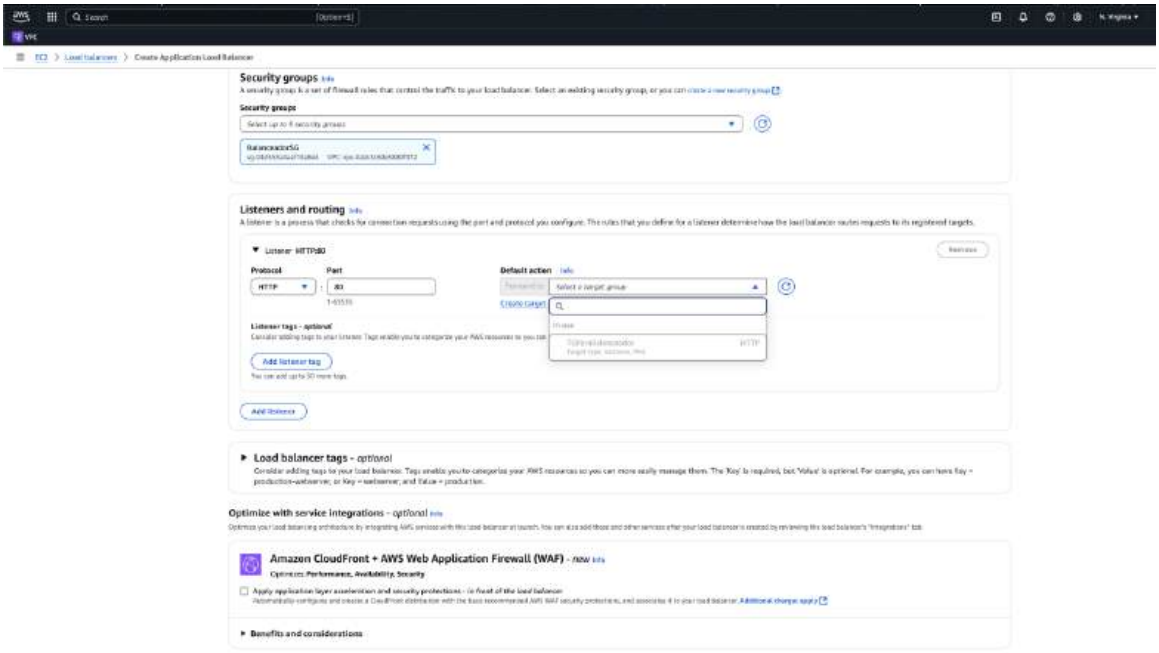


Figura 14: Una vez creado nuestro balanceador, veremos el listado de los balanceadores que existen, y si damos clic sobre alguno de estos veremos la información acerca de este y su configuración.

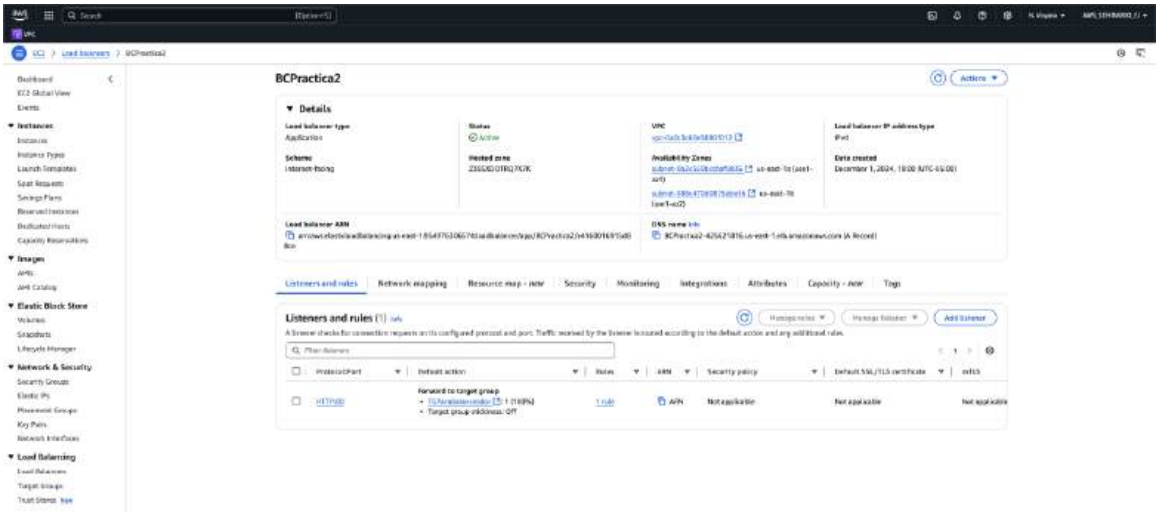


Figura 15: Para automatizar la creación de las instancias del balanceador de carga, vamos a crear un Launch Template desde la siguiente opción:

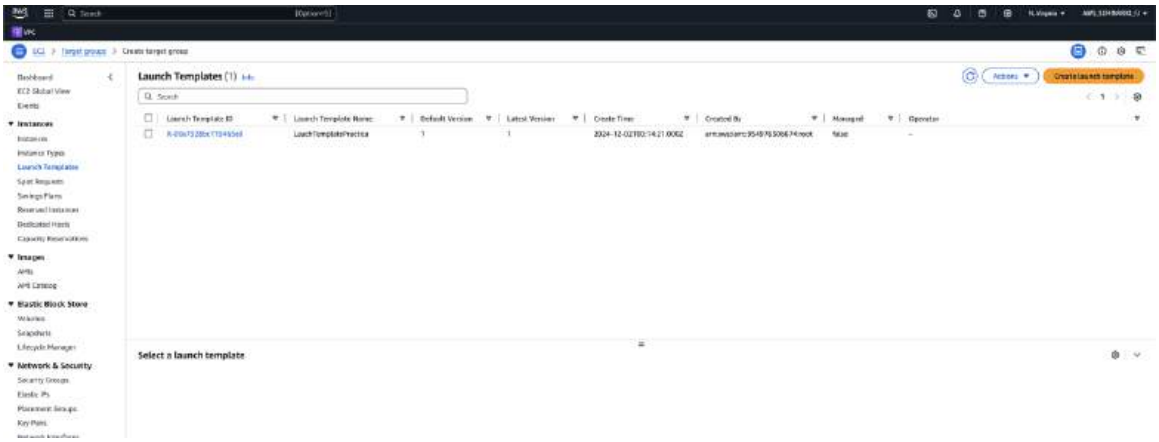


Figura 16: Crearemos una nueva asignaremos la información necesaria, y en Application and OS Images seleccionaremos la AMI que creamos inicialmente la cual contendrá la configuración necesaria para el balanceador de carga pueda crear las instancias de manera adecuada cuando sea necesario.

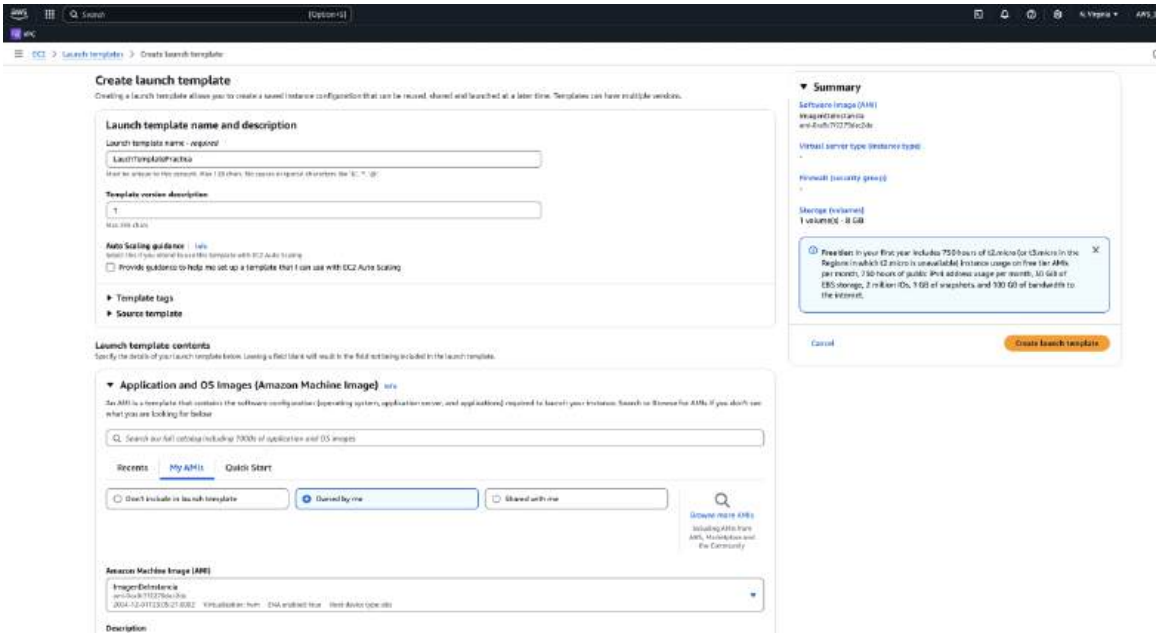


Figura 17: Continuando con la creación del Launch Template en Instance Type seleccionaremos la opción t2.micro, en key pair seleccionaremos una que ya exista o bien crearemos una nueva y seleccionaremos el Security Group que se aplicara.

El resto de la configura aplicara según las necesidades.

Hay que recordar que esta será la configuración que tomará cada una de las maquinas que se creen de forma automática por el balanceador de carga

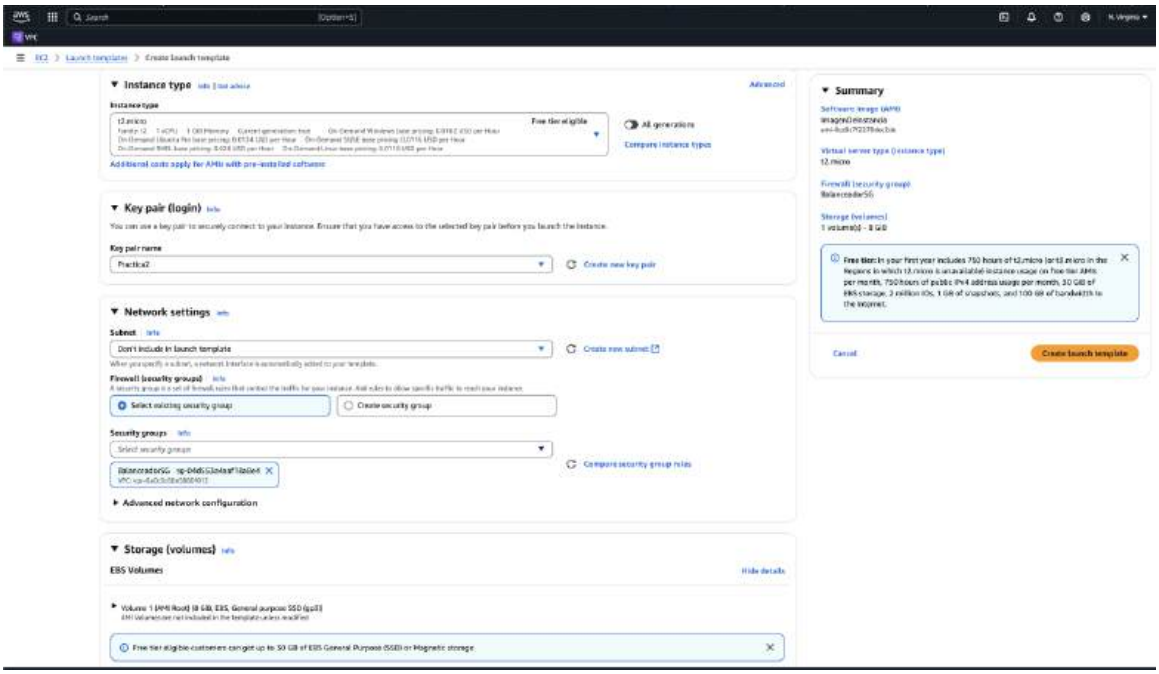


Figura 18: Ahora para finalizar la creación de nuestro balanceador de carga automático, crearemos un Auto Scaling Group el cual recopilara todos los datos necesarios para iniciar y expandir el balanceador, iremos a la siguiente opción para realizar este proceso.

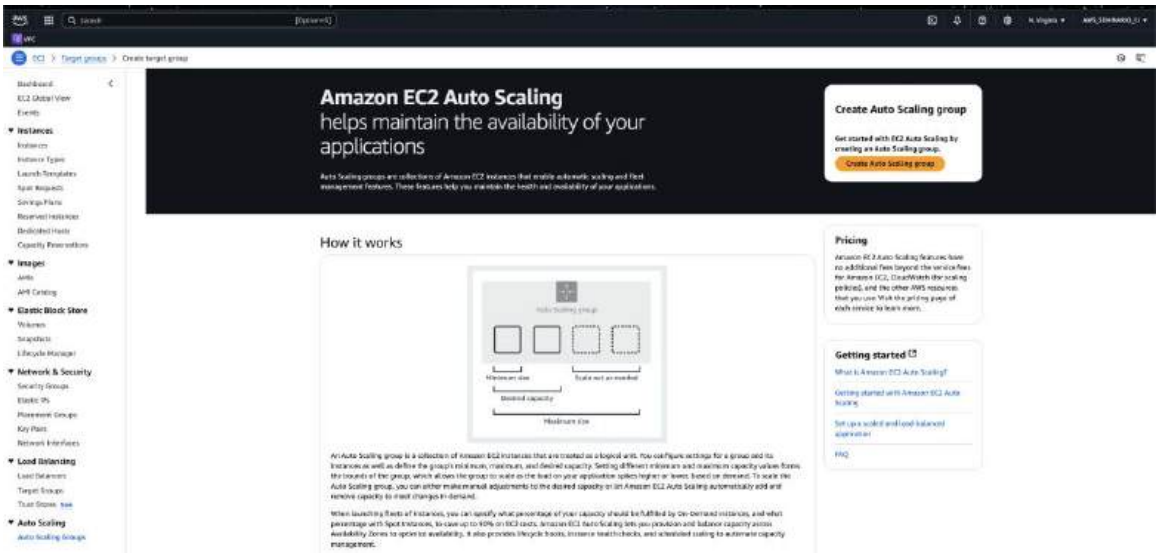


Figura 19: Le daremos un nombre, y seleccionaremos el Launch Template que creamos anteriormente, el cual tendrá la configuración necesarias para las instancias.

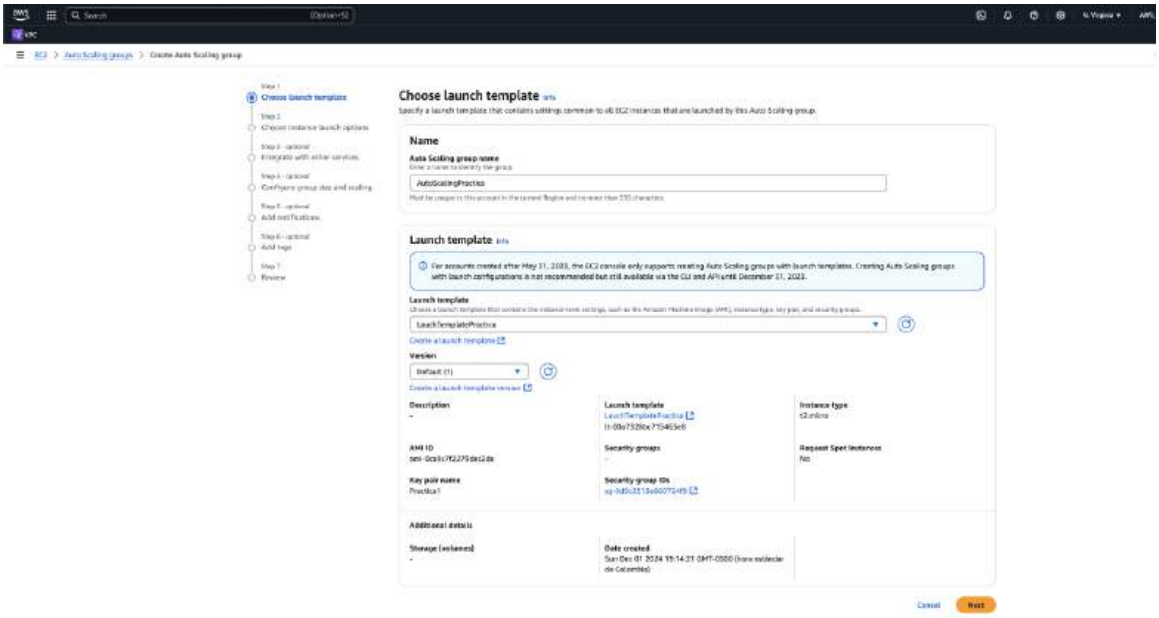


Figura 20: Seleccionaremos la VPC en la cual trabajaran las instancias y seleccionaremos las zonas y las subnets en las que se crearan las instancias, si recordamos al crear la VPC seleccionamos 2 zonas para la creación de esta, en este punto será de suma importancia esa configuración ya que en caso de haber problemas con una la otra podrá reemplazarla y continuar con el funcionamiento de las instancias.

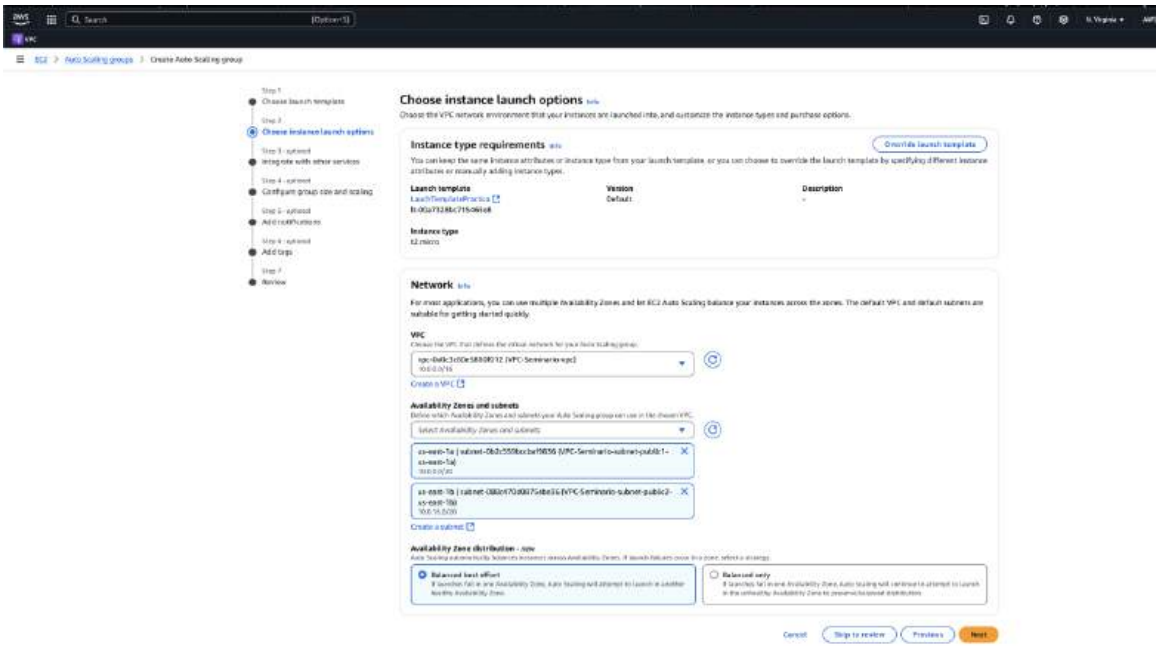


Figura 21: Continuando con la configuración, podemos dejarlo tal cual nos indica la configuración por defecto, y en la parte de los Health Checks indicaremos el tiempo en el que se hará la verificación de las instancias que se creen nuevas, en este caso el valor se asignará en segundos.

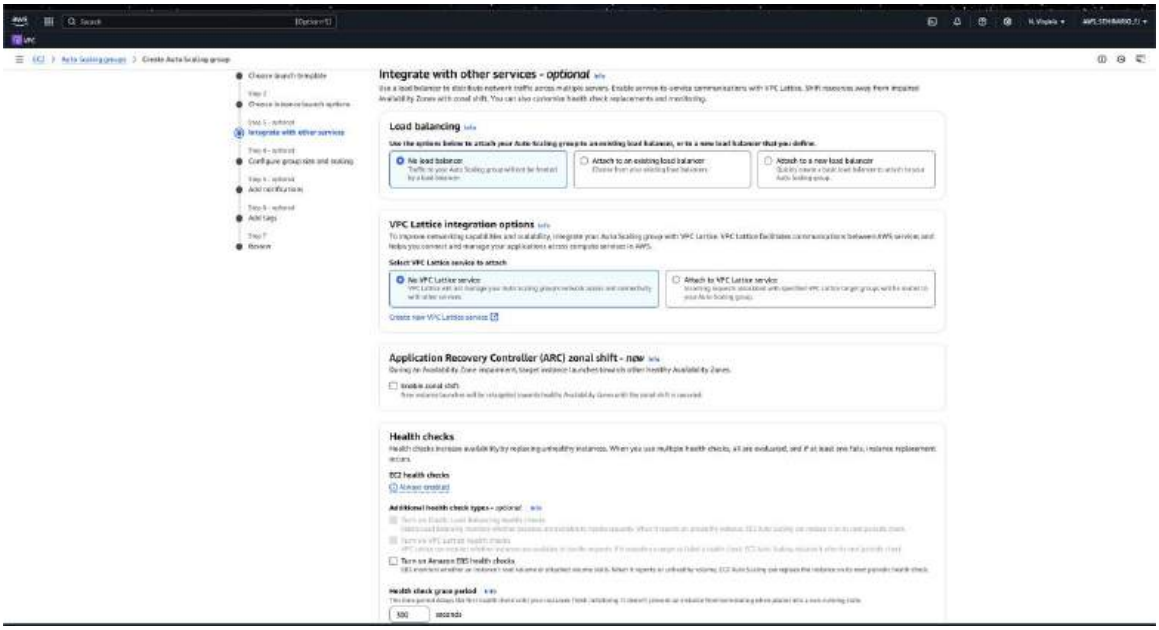


Figura 22: Finalmente para nuestra configuración, indicaremos el Group Size, aquí indicaremos la cantidad de instancias que puede tener nuestro balanceador tanto la cantidad de instancias deseadas, mínimas y máximas que pueden haber, en este caso configuramos los 3 valores con 2 el resto de la configuración la dejaremos tal cual esta por defecto.

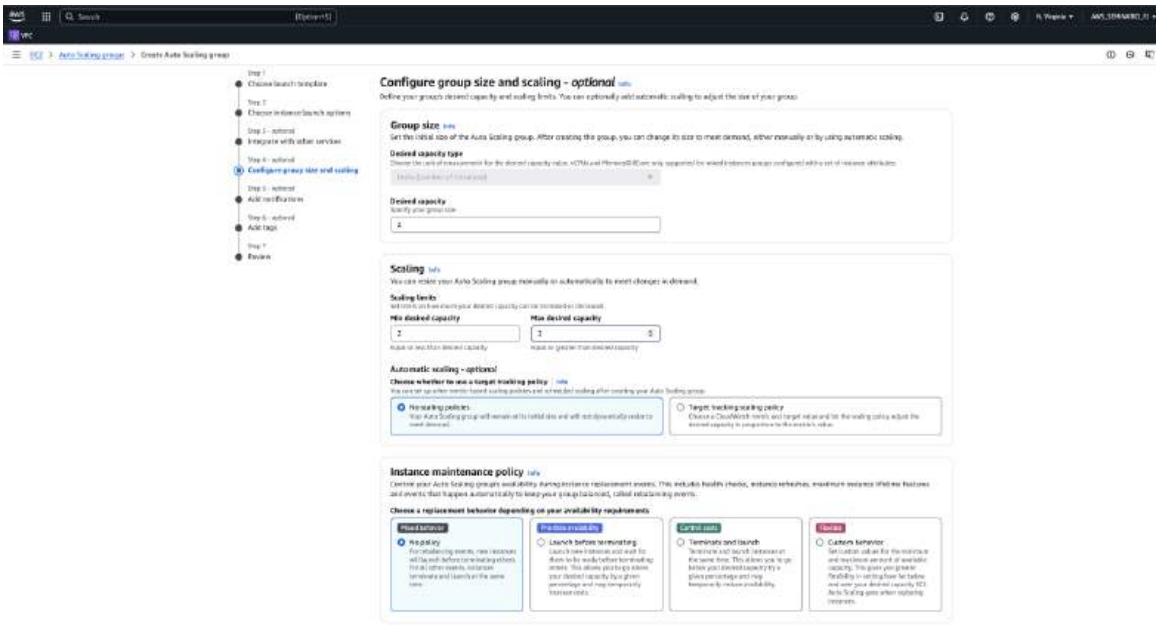


Figura 23 y 24: También tendremos otras opciones que nos permitirán crear notificaciones o tags, en este caso las pasaremos por alto ya que no son necesarias.

Iremos a la última opción Review en la cual tendremos un vistazo de como quedó nuestra configuración.

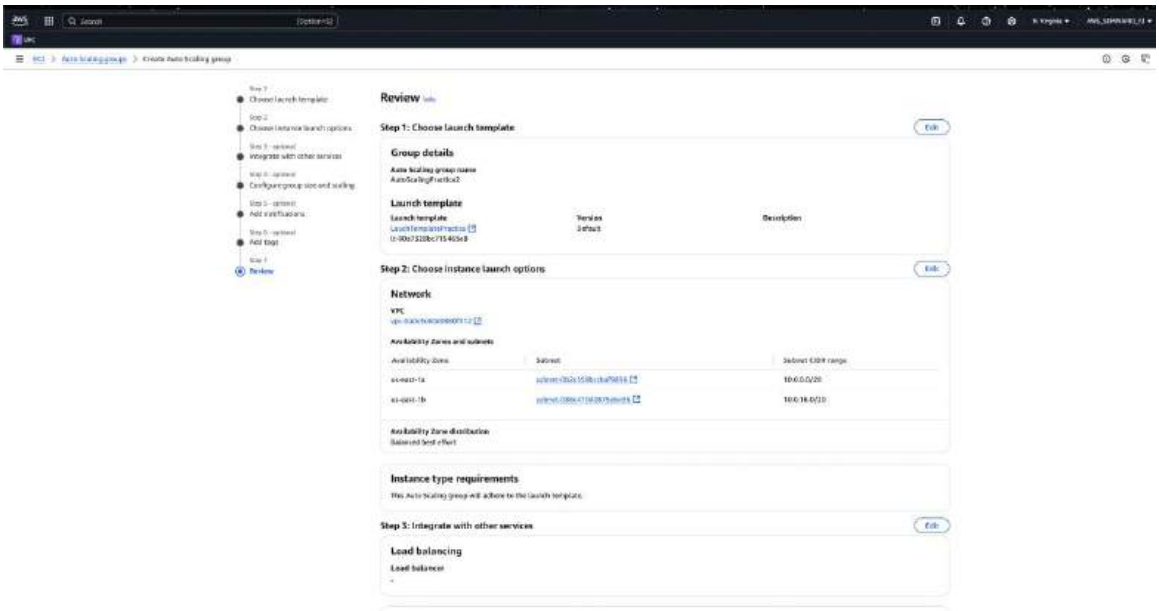


Figura 24:

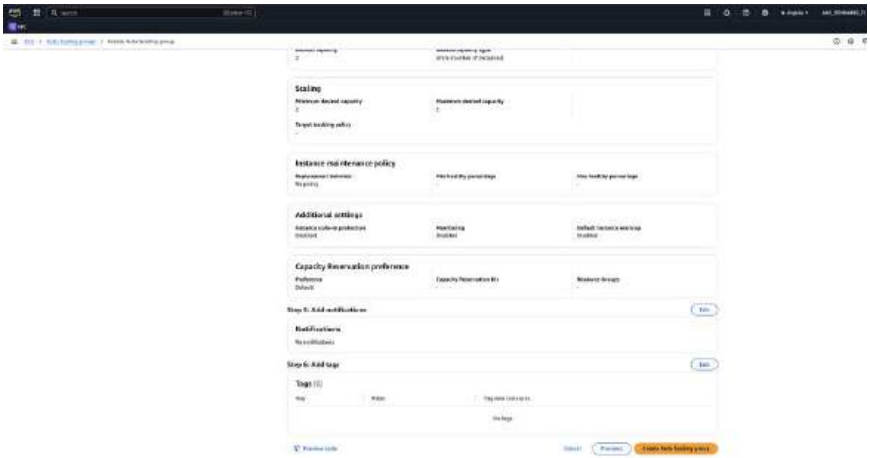


Figura 25: Con esto tendremos nuestro Auto Scaling Group creado y funcionando.

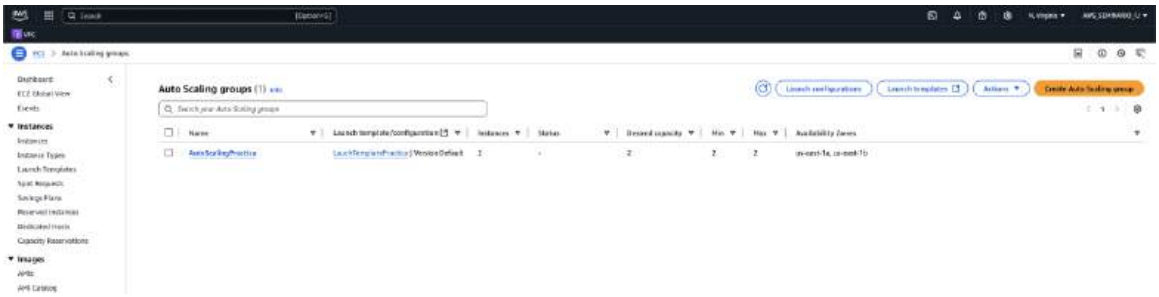


Figura 26: Si volvemos a la parte de Target Groups veremos que hay 2 instancias asociadas con son las que se crearon automáticamente de acuerdo con nuestra configuración. También podemos ver el estado de las maquinas en este caso ya que ambas se encuentran funcionando hay 2 en estado Healthy, en caso tal de que alguna falle aparecerá en Unhealty, incluso si se crean nuevas máquinas aquí mismo podremos seguir viendo el estado de estas.

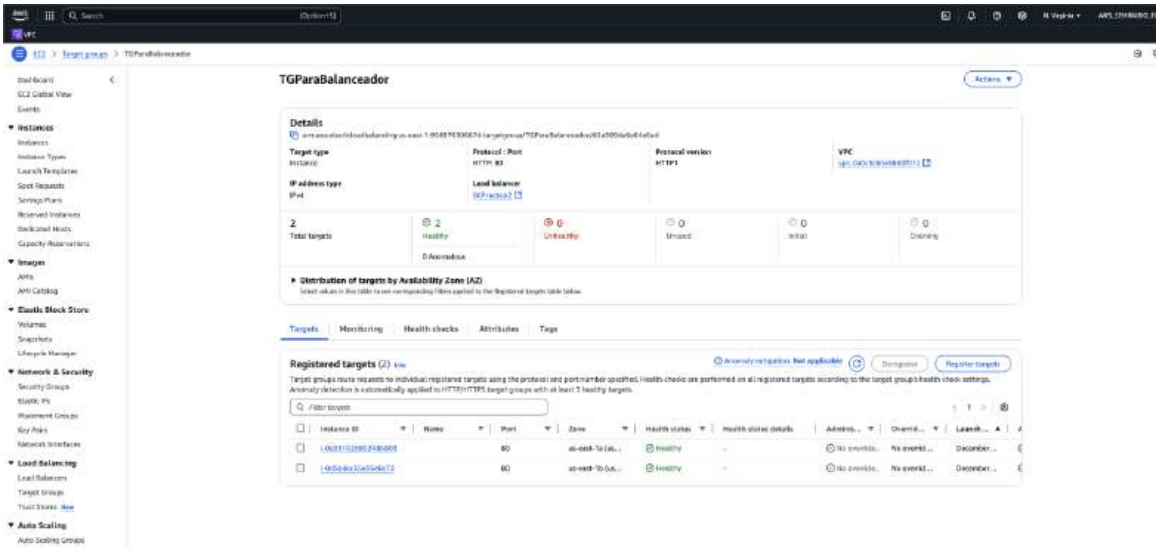


Figura 27: Volviendo a la parte de instancias veremos que las 2 se encuentran funcionando en este momento.

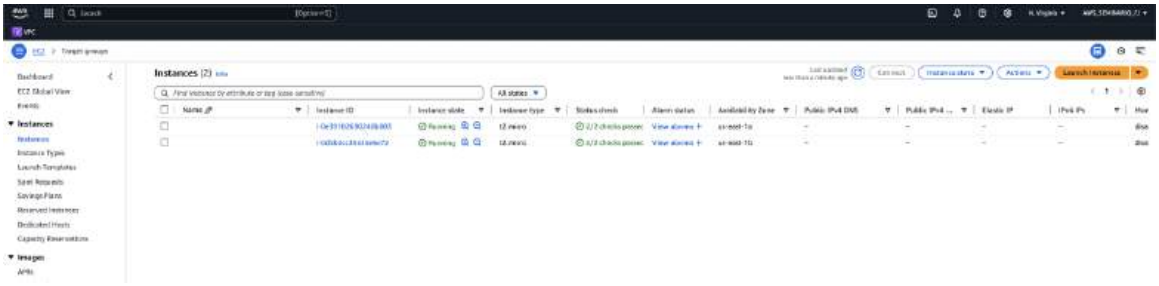


Figura 28: Ya que tenemos nuestras instancias si bien recordamos al inicio de la práctica, para acceder a nuestros servicios lo debíamos hacer haciendo uso de la IP de cada una.

Ahora que tenemos un balanceador de carga y el procesamiento de las peticiones lo puede hacer cualquier instancia, para acceder a nuestro servicio lo haremos a través del DNS que se generó cuando se realizó la creación de este, para eso daremos clic en cualquiera de las instancias tal cual se muestra en la imagen de abajo.

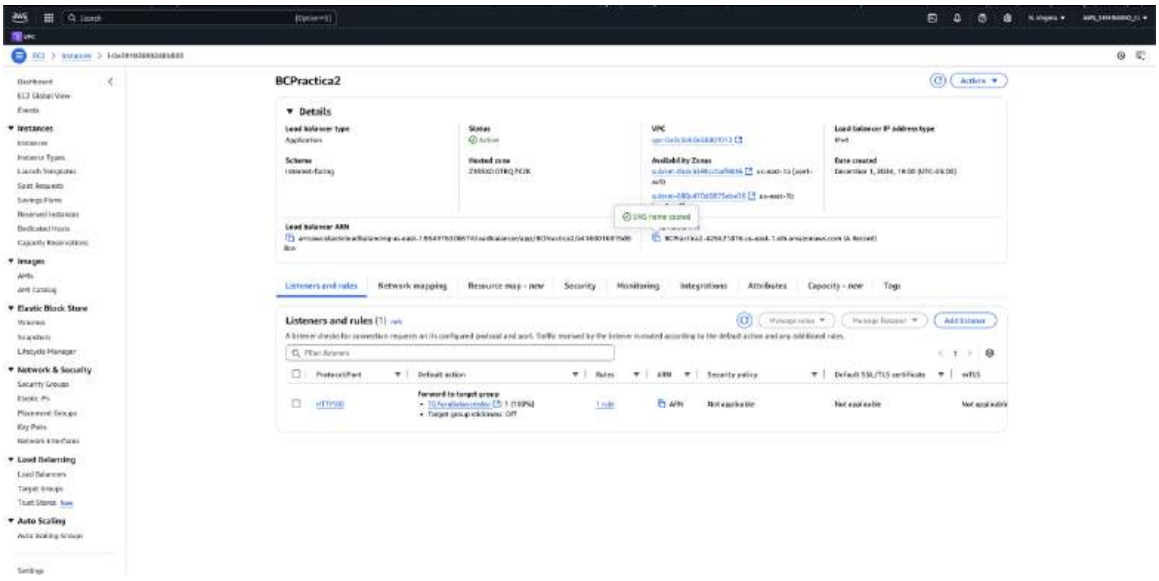
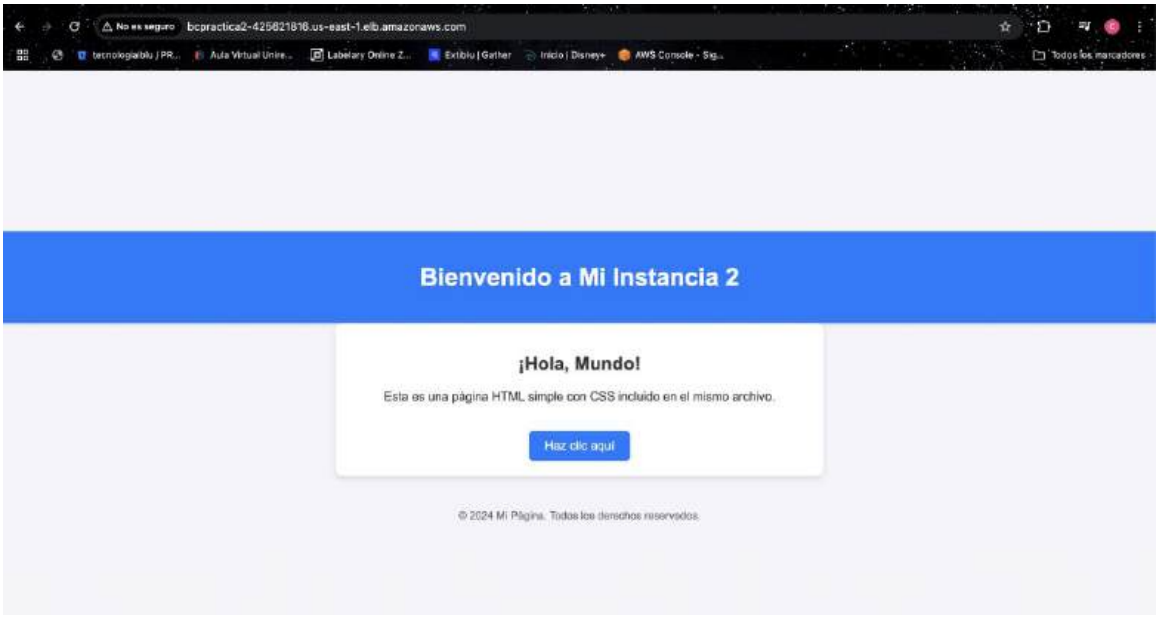


Figura 29: Finalmente si accedemos al DNS indicado podremos ver nuestro servicio en funcionamiento.



Entrega 3

Entrega 3
figura 3.1.

Figura 30: Se instala el servidor web Nginx para poder manejar balanceador de carga como proxy inverso.

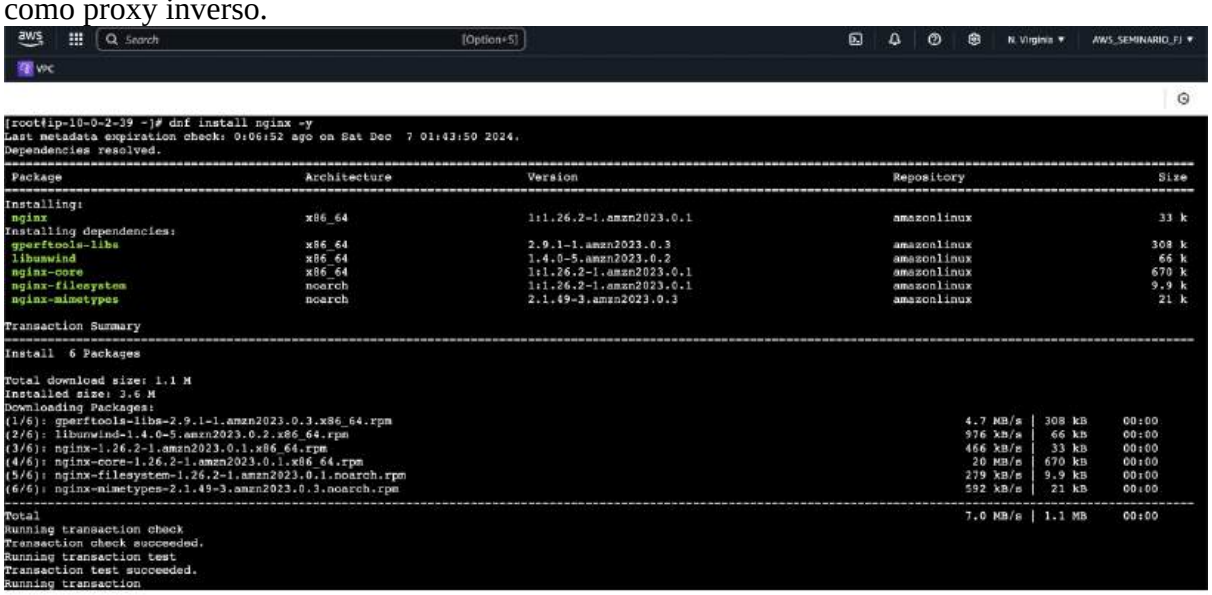


Figura 31: acá se define los puertos de las maquinas de Docker y redireccionarlo al puerto 80 de la maquina anfitrión.

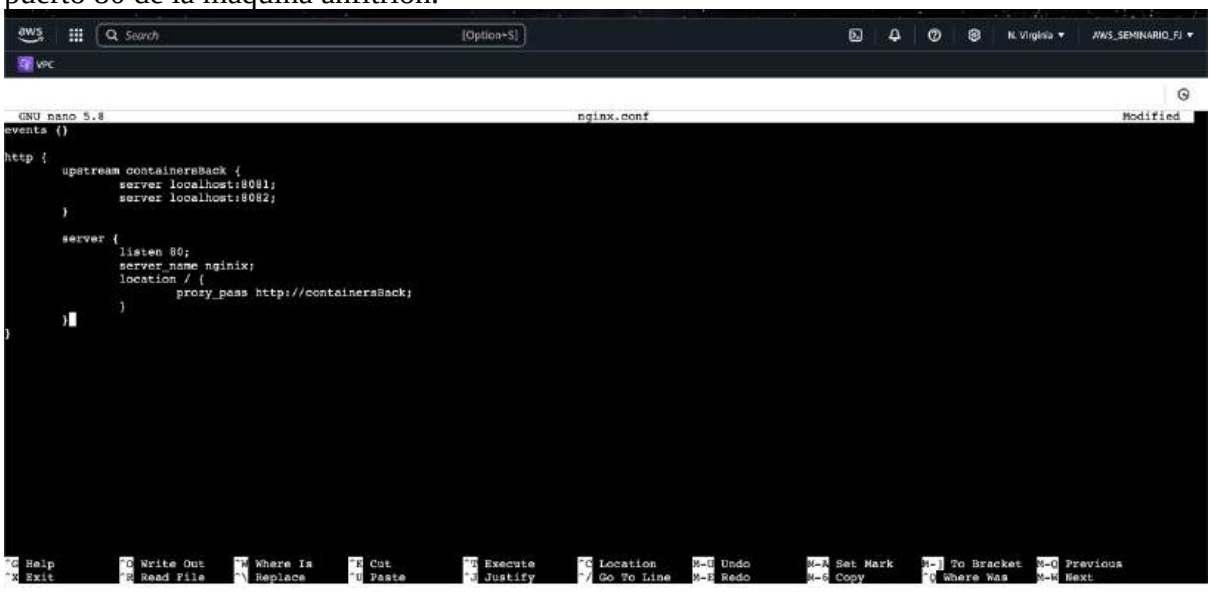


Figura 32: Acá se instala el Docker para ejecutar contenedores y sus dependencia necesarias a través del administrador de paquetes

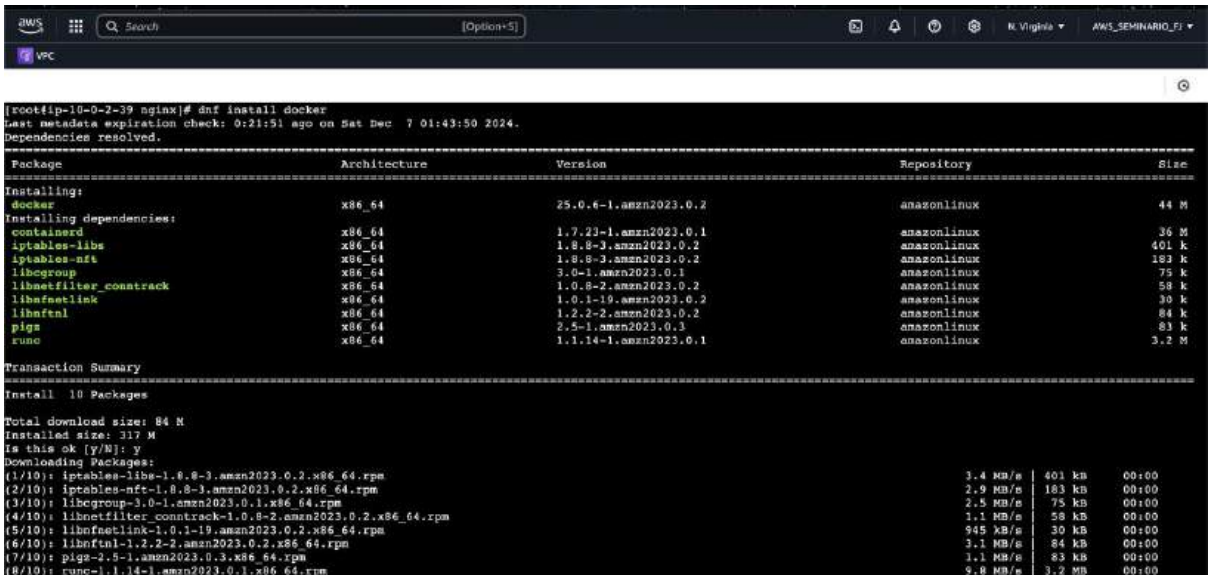


Figura 33: Acá se realiza acciones para la configuración y poder activar el servicio de Docker.

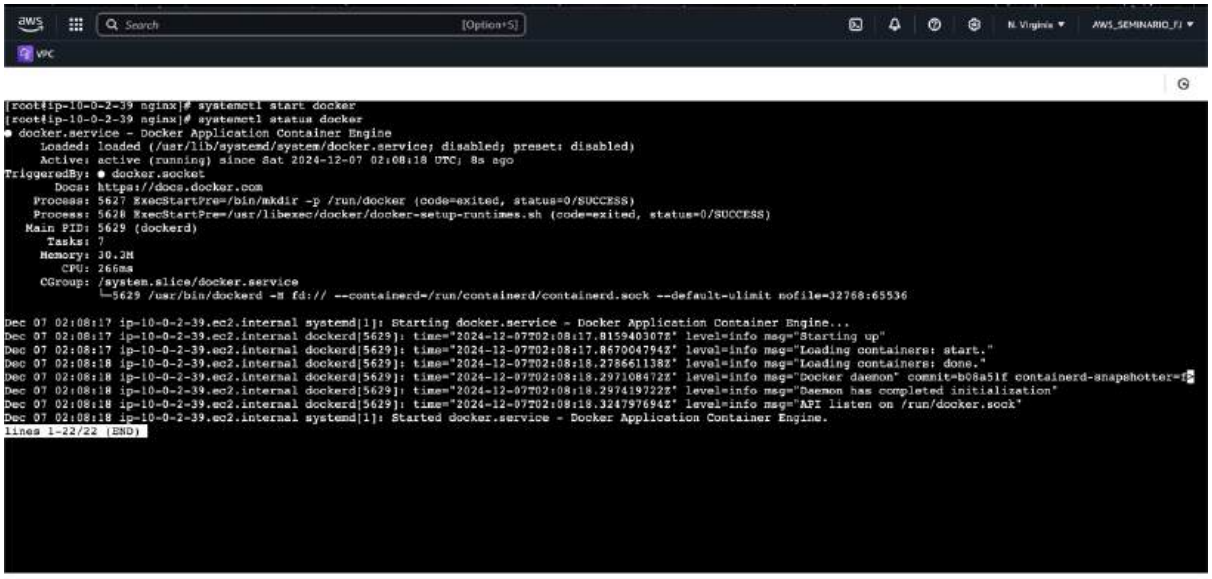


Figura 34: Acá nos está indicando que se esta descargando la imagen Docker para el servidor Apache HTTP

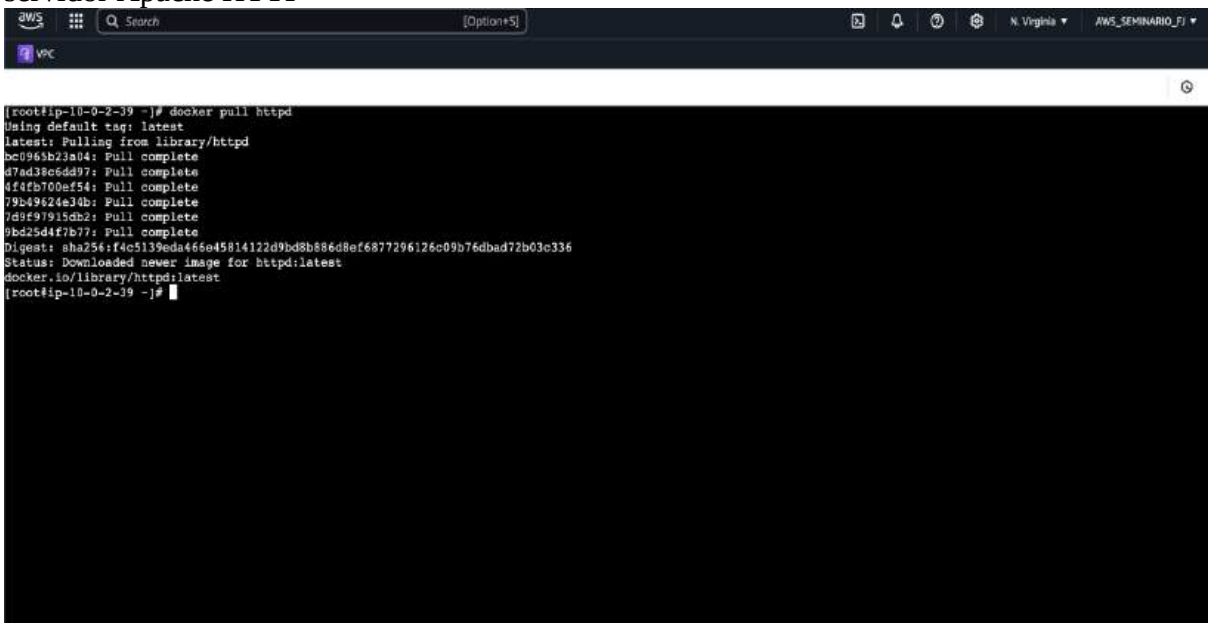


Figura 35: Acá se esta ejecutando para iniciar un contenedor de Docker con la imagen del servidor HTTP

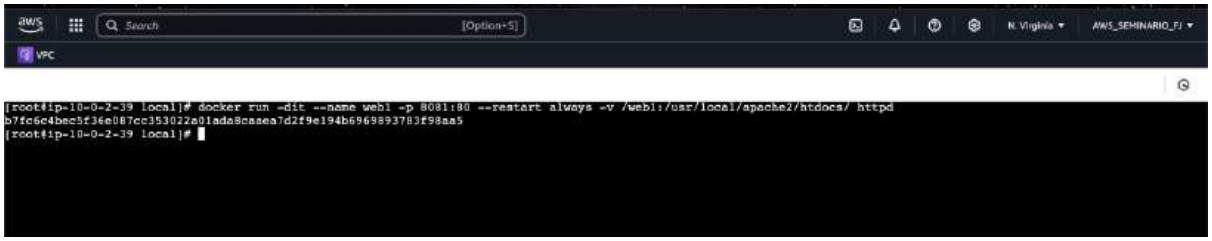


Figura 36: Acá se tienen dos servidores Apache corriendo 8081 y 8082

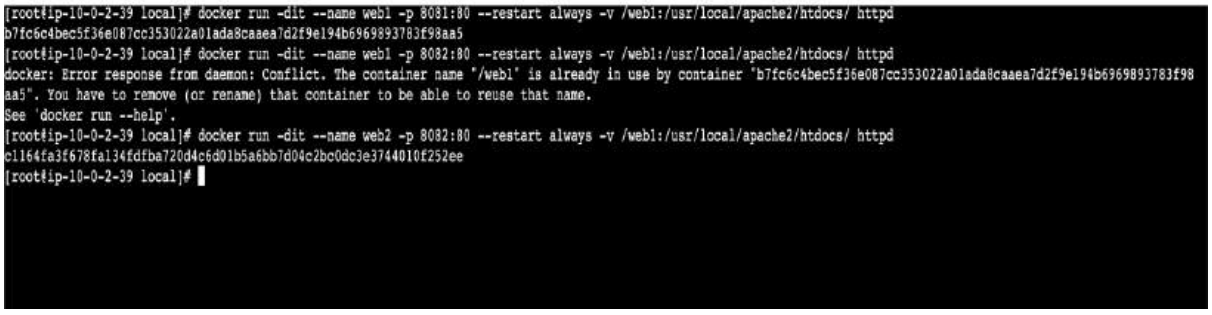


Figura 37: Los contenedores están corriendo y listo para manejar trafico HTTP

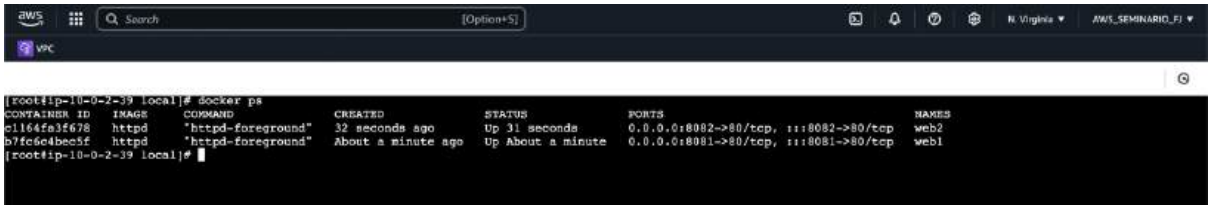


Figura 38: Acá se realiza la salida del comando Docker, y se Permite administrar tareas

Figura 39: Acá podemos ver la salida en un entorno de Docker, Este directorio está destinado a servir archivos web a un entorno de Apache

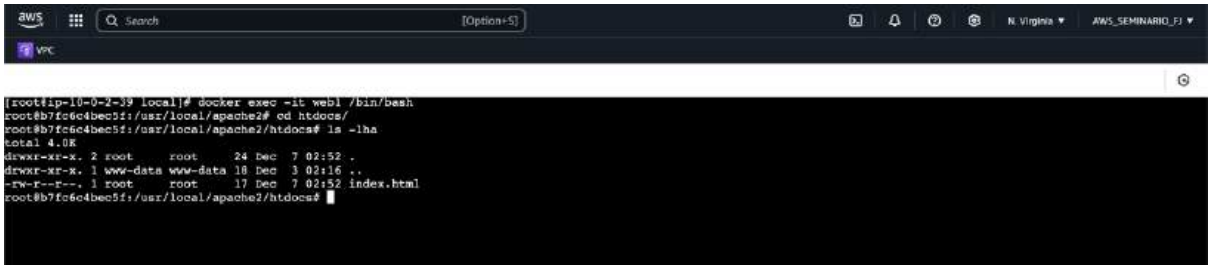


Figura 40: Acá podemos ver la salida de un archivo index.html ambos con permisos específicos

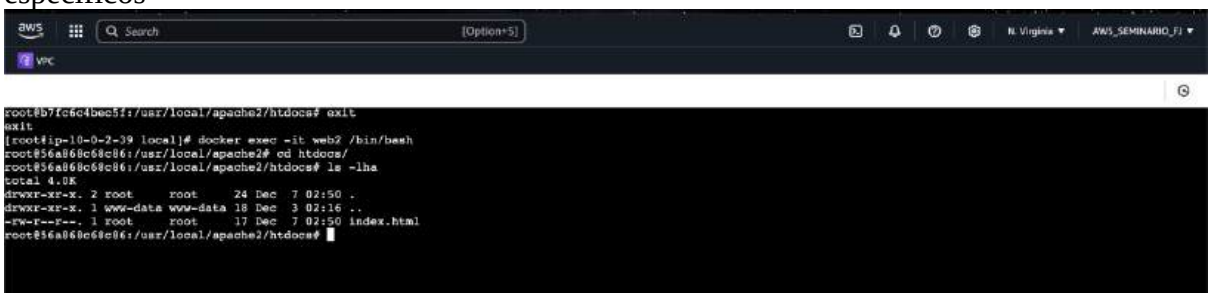
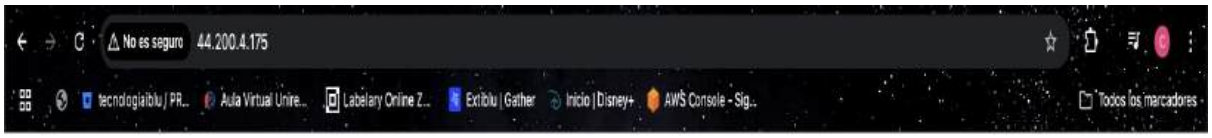


Figura 41: Acá el navegador web se esta conectando al servidor web, podemos observar que el contenedor esta funcionando y sirviendo a la página web

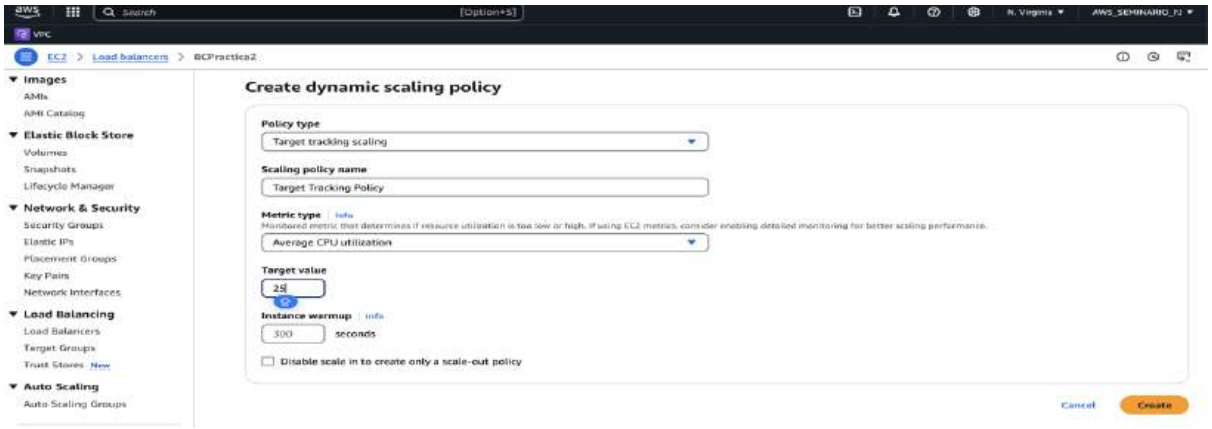
Contenedor web 1

Figura 42: En este segundo contenedor se puede ver que el servidor esta funcionando y sirviendo al contenido web



Contenedor web 2

Figura 43: Se crea una política que es un objetivo de un promedio de uso de CPU y se estable el valor de 25% acá podemos asegurar que la aplicación pueda manejar cargas de trabajo de manera eficiente



Conclusiones

Entender y aplicar el funcionamiento de una aplicación usando AWS es esencial e importante para las empresas actualmente ya que mejora en gran medida la escalabilidad de las aplicaciones, la estabilidad y el funcionamiento continuo de estas, asi mismo brinda gran cantidad de funcionalidades y servicios que permiten cumplir con las politicas de seguridad necesarias para la protección de la informacion.

La posibilidad de integrar contenedores de Docker dentro de una instancia, poder configurar un proxy reverso y que a su vez esta esté gestionada automáticamente por un AutoScaling Group y Load Balancers, demuestra la gran flexibilidad que tiene trabajar con la plataforma de AWS, permitiendo un nivel muy alto de autonomía para los profesionales en las TIC y facilitando la migración de sistemas tradicionales hacia la nube.

Referencias

W3Schools. (n.d.). *AWS First Recap*. Retrieved December 9, 2024, from https://www.w3schools.com/aws/aws_cloudessentials_awsfirstrecap.php

AWS Training and Certification. (2019, August 9). *AWS Virtual Private Cloud (VPC) Deep Dive* [Video]. YouTube. https://www.youtube.com/watch?v=L6U1DB72B3U

Google. (n.d.). *Que es VPC en redes*. Retrieved December 9, 2024, from https://www.google.com/search?q=que+es+vpc+en+redes

Amazon Web Services. (n.d.). *Launching and using instances*. Retrieved December 9, 2024, from https://docs.aws.amazon.com/es_es/AWSEC2/latest/UserGuide/LaunchingAndUsingInstances.html

Documentación oficial de NGINX
F5 Inc. (n.d.). *Reverse proxy*. Retrieved December 9, 2024, from https://docs.nginx.com/nginx/admin-guide/web-server/reverse-proxy/

Amazon Web Services. (n.d.). AWS Management Console. Retrieved December 9, 2024, from https://us-east-2.console.aws.amazon.com/console/home

Amazon Web Services. (n.d.). Install Docker. Retrieved December 9, 2024, from https://docs.aws.amazon.com/es_es/serverless-application-model/latest/developerguide/install-docker.html

Docker. (n.d.). *httpd*. Retrieved December 9, 2024, from https://hub.docker.com/_/httpd