TRABAJO DE GRADO
Opción Seminario-Diplomado.

IMPLEMENTACIÓN DEL SERVIDOR WEB YEFER. COM

Corporación Universitaria Remington.
Facultad de ingeniería de sistemas
Ingeniería de sistemas
(Seminario AWS)

Nombres de los estudiantes autores del trabajo de grado.

MARIA FERNANDA AGREDA REVELO
EDWARD YESID CARDENAS CARREÑO

Nombre del Tutor del trabajo de grado.

JUAN PABLO BERRIO LOPEZ

Opción de Trabajo de grado Seminario-Diplomado.
2024

Tabla de Contenidos

Resumen

En este proyecto, exploraremos los servicios que nos ofrece AWS mediante instancias y contenedores, con el objetivo de implementarlos en una plataforma de alta disponibilidad.

El propósito es crear un servidor web capaz de ejecutar aplicaciones dentro de servidores en instancias automáticas, utilizando contenedores Docker para la gestión y despliegue de las aplicaciones.

Para garantizar el entendimiento del paso a paso del proyecto, se estructurará el proyecto en varias etapas (entregas 1, 2, 3), detallando el proceso necesario para la implementación. Estas etapas incluirán desde la configuración inicial de las instancias EC2, la creación de contenedores Docker, la implementación de aplicaciones dentro de estos contenedores, hasta la configuración de escalabilidad automática y balanceo de carga para garantizar una infraestructura de alta disponibilidad y rendimiento.

El uso de instancias EC2 proporcionará la flexibilidad necesaria para ejecutar aplicaciones, mientras que Docker ofrecerá un entorno aislado y eficiente para las aplicaciones. Además, se explorarán otros servicios complementarios de AWS como Amazon Elastic Load Balancing (para distribuir el tráfico) y Amazon CloudWatch (para monitoreo), todo con el fin de asegurar que la plataforma mantenga una alta disponibilidad, escalabilidad y eficiencia operativa.

Palabras clave:

- **AWS**
- **INSTANCIA**
- **SERVIDOR**
- **DOCKET**
- **PROXY**

MARCO CONCEPTUAL Y CONTEXTUAL

La implementación del servidor web YEFER.COM, esta dividida en tres etapas que daremos a conocer luego de algunos conceptos principales con el fin de garantizar el entendimiento del proyecto:

AWS: Según (AMAZON, 2024) “es la nube más adoptada y completa en el mundo, que ofrece más de 200 servicios integrales de centros de datos a nivel global. Millones de clientes, incluso las empresas emergentes que crecen más rápido, las compañías más grandes y los organismos gubernamentales líderes, están usando AWS para reducir los costos, aumentar su agilidad e innovar de forma más rápida.”

VPC: Según (AMAZON WEB SERVICE, 2024) el Amazon Virtual Private Cloud “puede lanzar recursos de AWS en una red virtual aislada de manera lógica que haya definido. Esta red virtual es muy similar a la red tradicional que usaría en su propio centro de datos, pero con los beneficios que supone utilizar la infraestructura escalable de AWS.”

EC2: (AMAZON WEB SERVICE, 2024) nos aclara su concepto al decir que “Amazon Elastic Compute Cloud (Amazon EC2) proporciona capacidad de computación escalable bajo demanda en la nube de Amazon Web Services (AWS). El uso de Amazon EC2 reduce los costos de hardware para que pueda desarrollar e implementar aplicaciones con mayor rapidez. Puede usar Amazon EC2 para lanzar tantos servidores virtuales como necesite, configurar la seguridad y las redes, y administrar el almacenamiento. Puede agregar capacidad (escalar verticalmente) para gestionar tareas que requieren mucha computación, como los procesos mensuales o anuales, o los picos de tráfico del sitio web. Cuando el uso disminuye, puede volver a reducir la capacidad (reducir verticalmente).

Una instancia de EC2 es un servidor virtual en la nube de AWS. Cuando inicia una instancia de EC2, el tipo de instancia que especifica determina el hardware disponible para la instancia. Cada tipo de instancia ofrece una combinación diferente de recursos de computación, memoria, red y almacenamiento.”

LOAD BALANCER: Es aquel recurso que se encarga de distribuir las peticiones a las diferentes instancias dentro del servidor web, asegurando que el trafico de las mismas sea equilibrado.

SUBNETS: Son las redes alternas que se implementan ya sea de tipo publico o privado para el correcto funcionamiento del balanceador de carga de acuerdo al TARGET GROUP.

AUTOSCALING: Dado que (AMAZON SERVICE, 2024) define que es el AUTOSCALING, podemos decir que “es un servicio web para desarrolladores y administradores de sistemas que necesitan una solución para escalar automáticamente sus recursos escalables para AWS servicios individuales más allá de AmazonEC2.”

SECURITY GROUP: Es el recurso utilizado para darle las reglas de seguridad a nuestra maquina virtual y por el cual obtenemos las reglas de entrada y salida desde internet, además de otras configuraciones necesarias para el funcionamiento de las instancias.

PUTTY: Según (WORDPRESS, 2015) “PUTTY es un cliente SSH y TELNET con el que podemos conectarnos a servidores remotos iniciando una sesión en ellos que nos permite ejecutar comandos. El ejemplo más claro es cuando empleamos PUTTY para ejecutar comandos en un servidor VPS y así poder instalar algún programa o configurar alguna parte del servidor.”

PROXY REVERSE: Según la empresa (PROGRESS, s.f.) “se aplica normalmente a un servicio que se encuentra frente a uno o más servidores (como un servidor web) y acepta solicitudes de los clientes para recursos ubicados en el servidor o los servidores. Desde el punto de vista del cliente, el proxy inverso parece ser el servidor web y, por lo tanto, es totalmente transparente para el usuario remoto.”

DOCKET: Según (AMAZON WEB SERVICE, 2024) “Docket es un sistema operativo para contenedores. De manera similar a cómo una máquina virtual virtualiza (elimina la necesidad de administrar directamente) el hardware del servidor, los contenedores virtualizan el sistema operativo de un servidor.”

El contexto del informe técnico aborda la implementación de un servidor web en AWS utilizando Amazon Linux, para generar un entorno automatizado, escalable y funcional, integrando tecnologías modernas como lo es “Docket” el cual utilizamos para gestionar contenedores. Dicha implementación se realizó en tres entregas que tendremos en cuenta como tres **etapas** diferentes desarrolladas de la siguiente manera:

PRIMERA. En esta etapa se realizó la configuración de una instancia EC2 con un servidor web Apache, habilitando reglas de seguridad para permitir tráfico web y verificando su funcionalidad a través de la IP pública. Además, se explica el uso de recursos de AWS como VPC, EC2, AMI, SECURITY GROUPS, EBS y AUTOSCALING, que sustentan la infraestructura en la nube.

SEGUNDA. En esta etapa se realiza la incorporación de la escalabilidad mediante la creación de una AMI a partir de un snapshot, el lanzamiento de múltiples

instancias, y la configuración de un balanceador de carga y un grupo de AUTOSCALING para asegurar la creación automática de instancias según las necesidades de carga. Estas pruebas confirman la resiliencia y escalabilidad del sistema, garantizando que las aplicaciones permanezcan funcionales bajo diferentes condiciones.

TERCERA. En esta etapa se introduce Docket como una solución de contenedores que permite ejecutar múltiples entornos virtualizados en una misma máquina anfitriona. Se detalla la instalación y configuración de Docket, la creación de contenedores con imágenes de tipo httpd y el uso de Nginx como servidor web. Además, se destaca la importancia de gestionar puertos únicos para evitar conflictos, configurando reglas en los Grupos de Seguridad de AWS.

El proceso incluye la creación de carpetas en el sistema anfitrión para enlazarlas con los contenedores, permitiendo que estos apunten a recursos específicos en la máquina virtual. A partir de estas configuraciones, se generan nuevas AMI para replicar el entorno automáticamente. Asimismo, se detalla la implementación de un proxy reverso utilizando Nginx para manejar peticiones entrantes y redirigirlas a los contenedores correspondientes.
Finalmente, se valida el sistema pegando la IP pública en el navegador y confirmando que las rutas configuradas en los contenedores responden adecuadamente. Este informe demuestra cómo AWS y Docket, junto con herramientas como Nginx, pueden combinarse para construir una infraestructura moderna, escalable y eficiente que soporta múltiples aplicaciones y entornos virtualizados.

DESARROLLO E IMPLEMENTACIÓN DEL APRENDIZAJE

ENTREGA N°1

a. Amazon EC2 (Elastic Compute Cloud)

Amazon EC2 es un servicio principal que proporciona capacidad de computación en la nube. Permite a los usuarios lanzar y gestionar instancias virtuales (servidores) en la infraestructura de AWS.
Las características clave incluyen:

- **Instancias:** Son servidores virtuales que pueden ejecutarse con diferentes configuraciones de CPU, memoria y almacenamiento.

- **Imágenes de Máquina de Amazon (AMI):** Son plantillas que contienen el sistema operativo y las aplicaciones necesarias para lanzar una instancia.

- **Tipos de Instancia:** Son diferentes configuraciones de hardware (t2.micro, m5.large, etc) que se adaptan a diversas necesidades de carga de trabajos.

b. Amazon Machine Imágenes (AMI)

Son imágenes preconfiguradas que contienen el sistema operativo y el software necesario para ejecutar aplicaciones. Puede usarse AMIs proporcionadas por AWS o crearlas personalizadas.

c. Grupos de Seguridad

Estos grupos actúan como un firewall virtual para controlar el tráfico entrante y saliente hacia las instancias. Define reglas especificas sobre que trafico es permitido, como SSH (puerto 22) para acceso remoto o HTP (puerto 80) para tráfico web.

d. Elastic Block store (EBS)

Proporciona almacenamiento para las instancias EC2 como discos duros para almacenar datos y aplicaciones. Los volúmenes EBS pueden ser conectados a instancias EC2 y pueden ser respaldados o replicados.

e. AWS Identity and Access Management (IAM)

Permite gestionar el acceso a los recursos de AWS, incluyendo quien puede lanzar o gestionar instancias EC2; puedes crear roles y políticas para otorgar permisos específicos a usuarios o servicios.

f. AWS Systems Manager

Permite la administración y automatización de tareas en las instancias EC2; se pueden ejecutar comandos remotos, aplicar parches y realizar configuraciones sin necesidad de acceso directo a las instancias.

g. Amazon VPC (Virtual Private cloud)

Permite crear una red privada en la nube de AWS donde se pueden lanzar las instancias EC2, se pueden definir subredes, direcciones IP y configuraciones de seguridad para controlar el acceso a los recursos.

Una vez que la instancia EC2 está en funcionamiento se pueden realizar varias tareas:

1. **Configuración del Sistema Operativo:** Utilizamos scripts de datos de usuario que permitan instalar un software automáticamente.

2. **Instalación de Aplicaciones:** se puede instalar aplicaciones y servicios necesarios para la carga de trabajo, como servidores web (Apache, Nginx), bases de datos (MySQL, PostgreSQL), etc.

3. **Gestión del Estado:** Puedes detener o reiniciar la instancia según sea necesario, así como escalar vertical u horizontalmente dependiendo de la demanda.

4. **Monitoreo:** Utilizando herramientas como Amazon CloudWatch, puedes monitorear el rendimiento y la salud de la instancia, estableciendo alarmas para notificar sobre problemas.

5. **Seguridad:** Implementar medidas de seguridad como actualizaciones del sistema operativo, configuración adecuada del firewall (grupos de seguridad), y gestión del acceso mediante IAM.

6. **Almacenamiento:** Los datos pueden ser almacenados en volúmenes EBS o en servicios como Amazon S3 para un acceso más flexible y duradero.

7. **Redes**: Configurar interfaces de red y reglas para controlar cómo se comunican las instancias entre sí y con otros servicios en la nube o fuera de ella.

EXPLICACIÓN DE LA IMPLEMENTACIÓN Y CREACIÓN DEL SERVIDOR WEB

Para la implementación de un servidor web en Amazon linux 2023 es necesario seguir una serie de pasos que incluyen la creación de una instancia EC2 la instalación de la configuración de apache y la configuración de las reglas de seguridad para permitir el tráfico web a continuación veremos los pasos

Cómo primer punto debemos crear una cuenta de AWS ligada a una tarjeta de crédito para poder obtener sus servicios

Después de crear nuestra cuenta debemos ingresar a la opción de VPC

El VPC es una agrupación de recursos ligados a una IP en donde se realiza la creación de un espacio grande para redes de tipo privadas y públicas

Para la creación de un VPC debemos tener en cuenta los siguientes pasos:

Ingresar así:

- VPC and more
- Nombre del VPC
- Ingresar la cantidad de redes privadas y públicas que queremos obtener según la necesidad, también tenemos la opción de configurar el servicio NAT en este caso utilizaremos la opción NONE.
- Luego tomamos la opción créate VPC
- Y hacemos Clic sobre view VPC.

Con este paso culminamos la creación de nuestro VPC y seguimos con la creación de la instancia o máquina virtual.

Cabe resaltar que una subred hace parte de la IP asignada por el IP y sólo toma una parte de ella, al interior de la subred se pueden ver los recursos, una subred privada no tiene salida ni entrada desde internet mientras que la subred publica permite la conexión desde internet y servidores para entradas.

Para la creación de la instancia damos Clic en la opción EC2 y luego en la opción launch instance

Luego de esto seleccionamos la opción Amazon Linux 2023 ami como imagen del sistema operativo teniendo en cuenta la opción Free Tier eligible

Luego elegimos el tipo de instancia para utilizar en este caso t2.Micro para utilizar una capa gratuita.

Dentro de la configuración que se realiza para lanzar la instancia EC2

Se realiza la configuración del key pair para descargar el archivo de certificado de seguridad que nos permite loguearnos.

Hay tipo. ppk y pem los cuales se pueden utilizar y convertir entre ellos según la necesidad a la hora de cargarlos para la contraseña (si este archivo se borra no hay forma de recuperar lo realizado)

Luego de esto se realiza la configuración del VPC y de la subred creada con anterioridad al crear dicho VPC pública o privada según el requerimiento se realiza la habilitación de la IP pública auto asignada pasamos a habilitar la opción create security group y se le asigna un nombre, luego pasamos a configurar las reglas de entrada del grupo de seguridad

SSH----TYPE----
PORT RANGE = 22

Los cuales vienen por defecto

En el siguiente paso veremos las medidas de almacenamiento de 8 gigas por defecto y se lanza la instancia esta nos muestra toda la descripción como lo es la dirección privada, pública, tipo de instancia, nombre del VPC, subredes, detalles de la imagen (ami: Amazon machine image) lo cual básicamente es el sistema operativo, las reglas de entrada y salida, etc.

El siguiente paso es conectar la instancia se selecciona el cliente SSH y allí encontramos el usuario que inicia por EC2, y termina en Amazon aws.com; lo pegamos en el cliente putty con el puerto 22.

Damos Clic en la pestaña SH luego en la opción AUTH, y luego en la opción credencial pasamos al área de private key, allí es donde cargamos el certificado

de seguridad y damos en open; con esto ya estamos dentro de la administración de Linux

Cuando nos encontramos dentro de la terminal de PUTTY sólo nos queda iniciar la instalación del paquete apache y su configuración para tener funcionando nuestro servidor web esto lo hacemos con los comandos

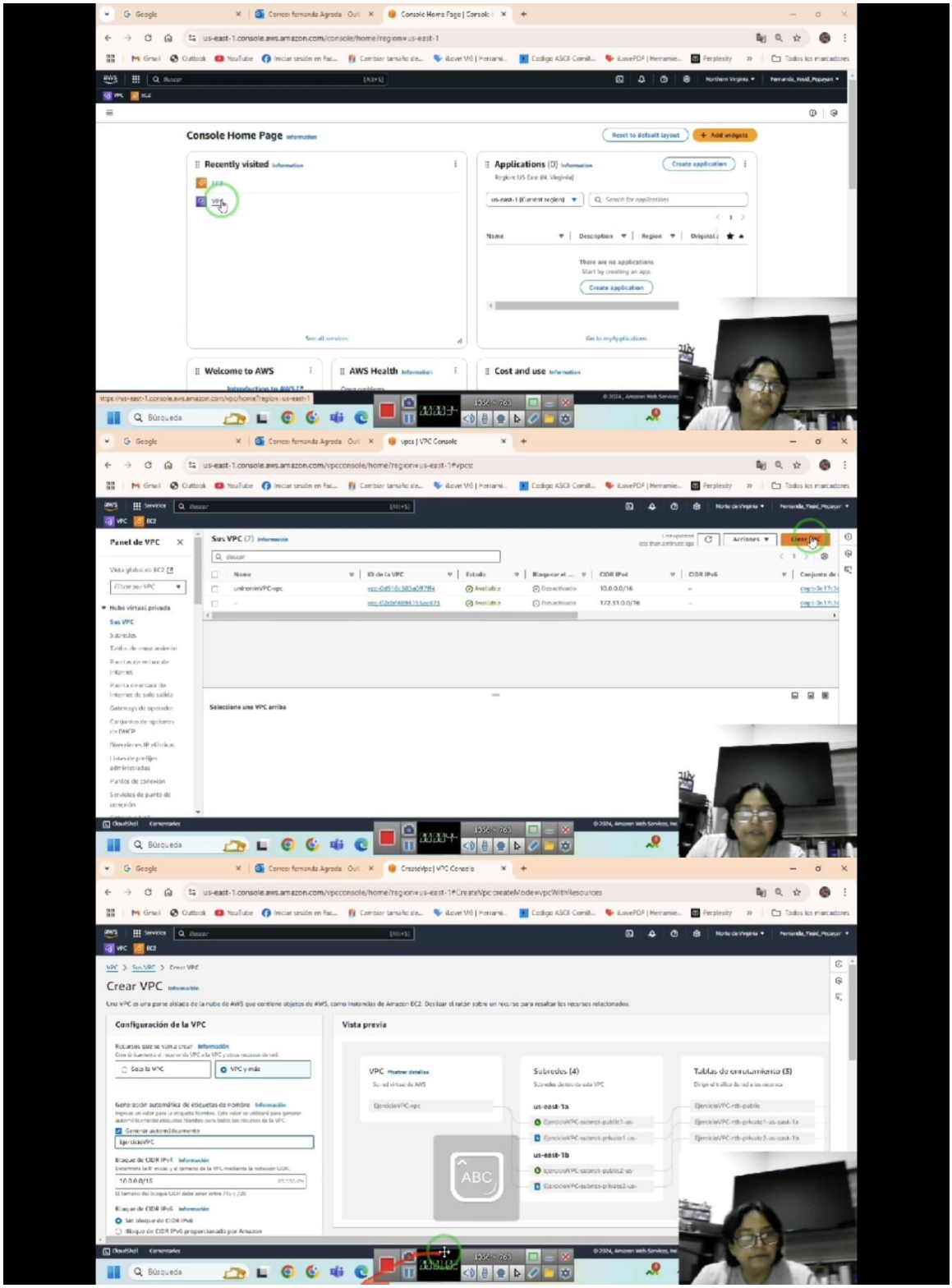
- yum install httpd
- systemctl status httpd
- iniciamos el servicio con systemctl start httpd
- revisamos en status de nuevo (systemctl status httpd).
- verificamos que se encuentre activo

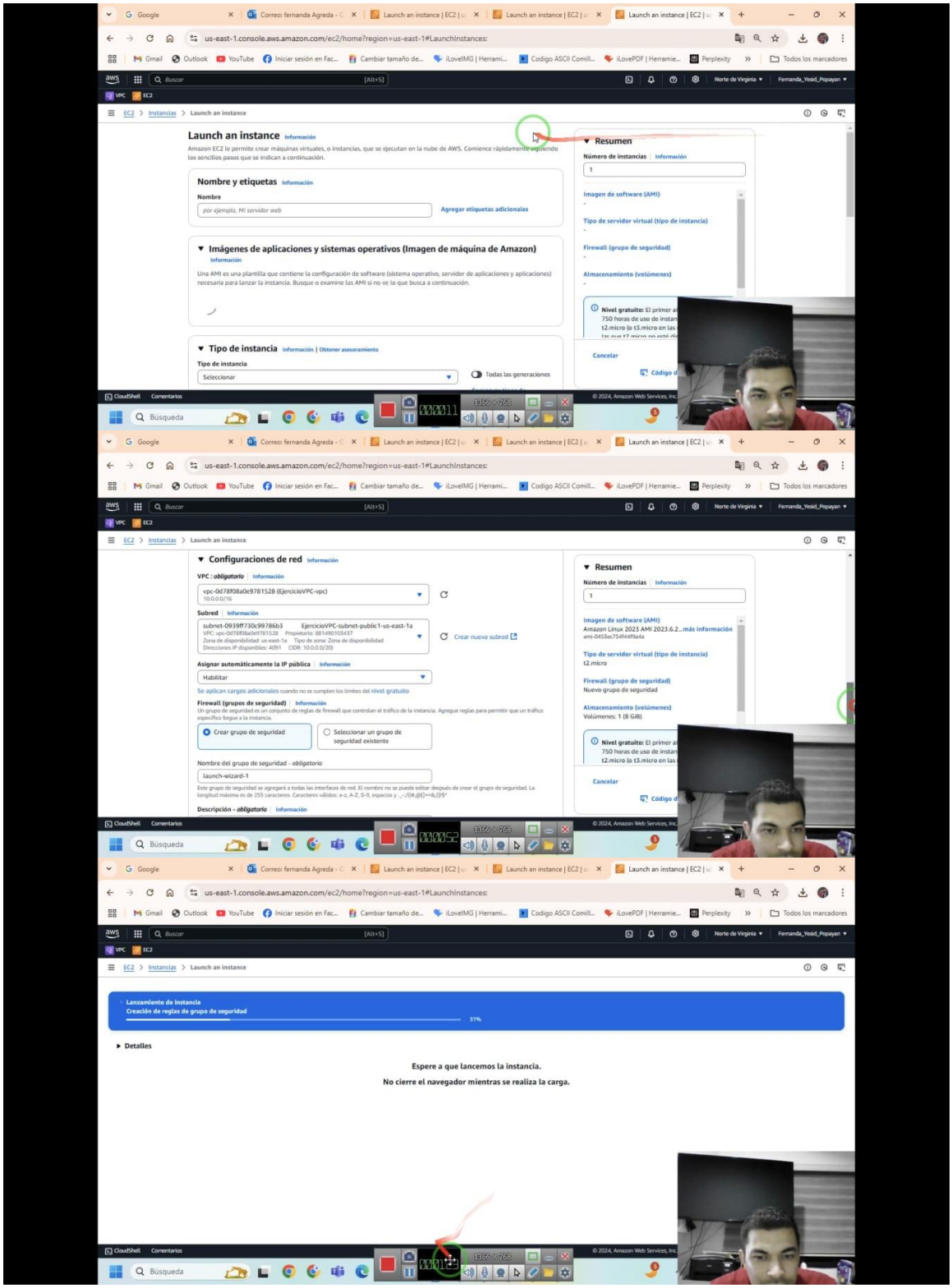
Cómo últimos pasos buscamos la dirección IP pública (public IPV4 address dentro de la instancia en AWS)

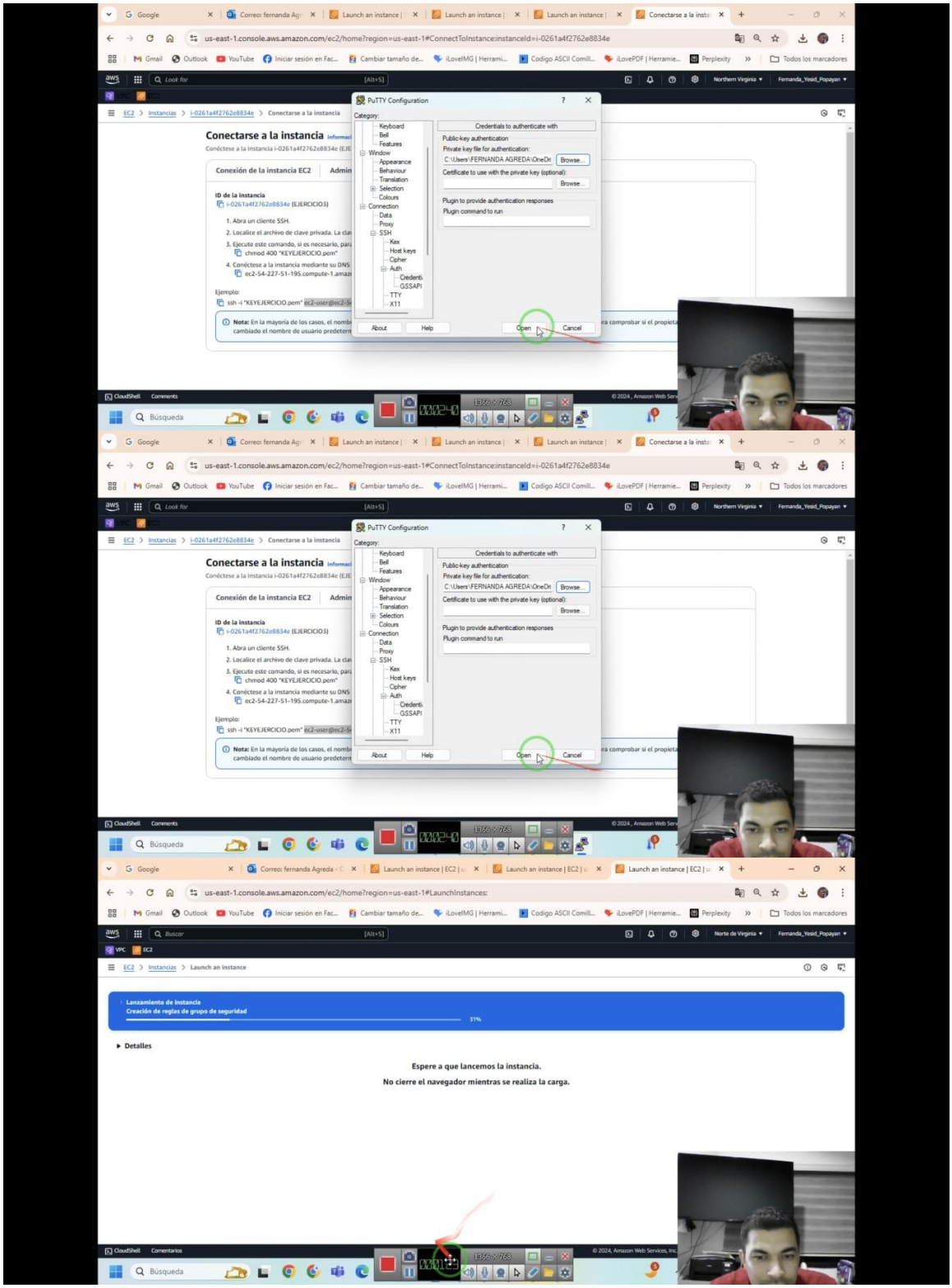
Copiamos esta IP y la pegamos en el navegador luego de hacer la modificación en el security group en donde vamos a dar permiso al puerto 80, asi:

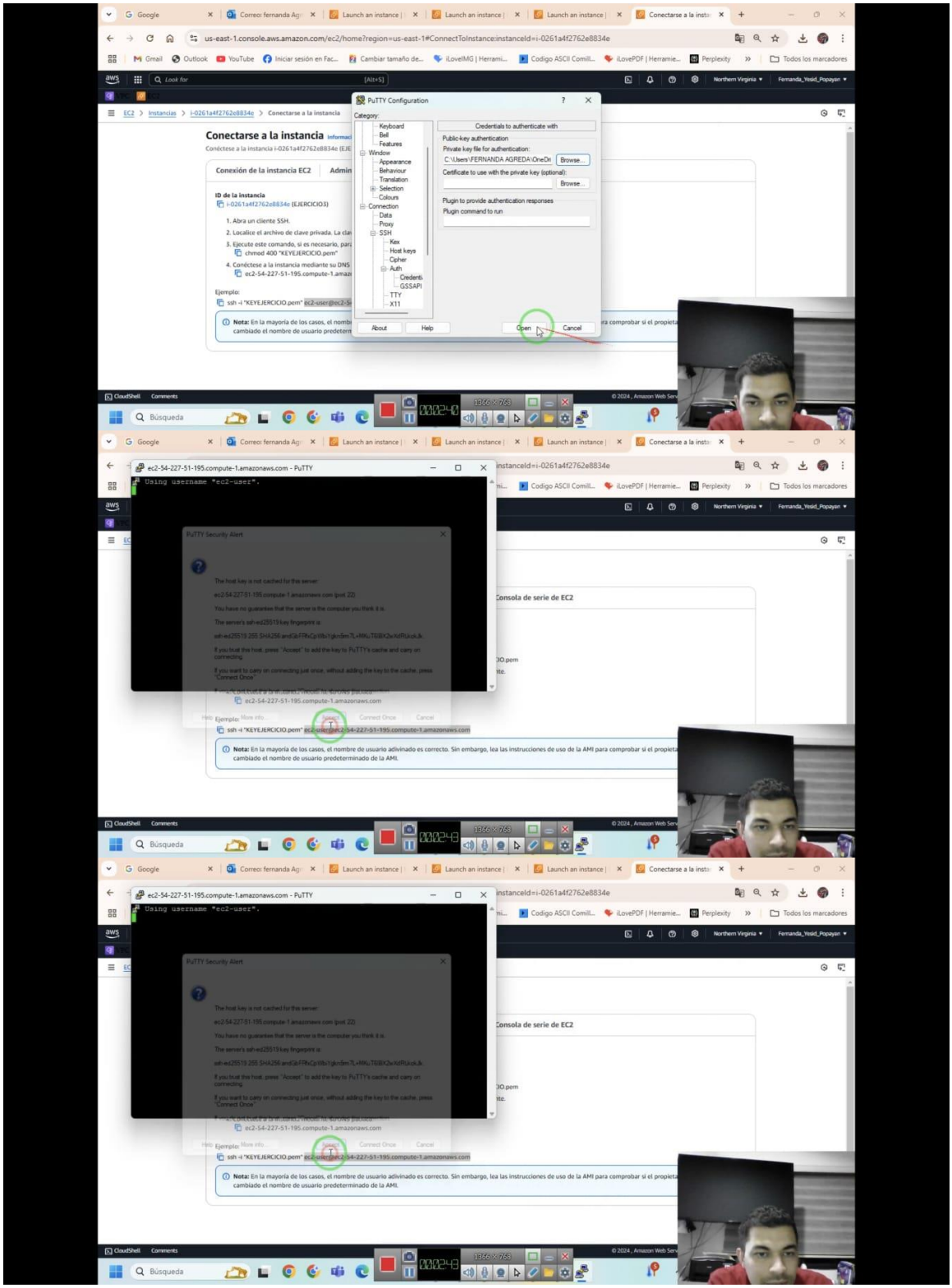
- security group
- inbound reles---editar---add rule
- Custom tcp---port range = 80
- Description = 0.0.0.0/0 (para pruebas iniciales)

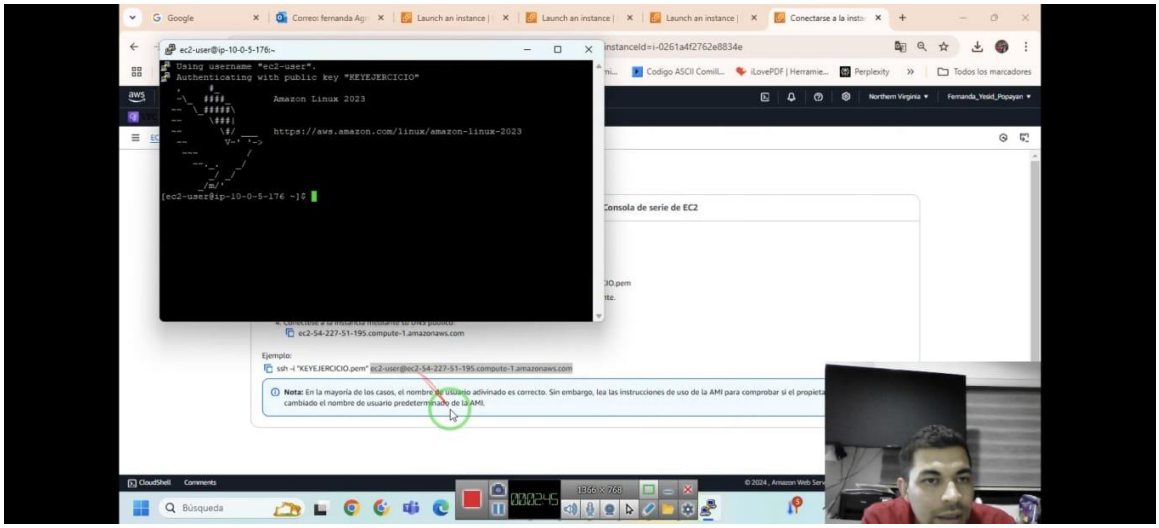
Pegamos nuestra IP pública en el navegador y obtenemos un anuncio en la página que dice: “it works” Con esto ya podemos verificar que nuestro servidor web está en correcto funcionamiento.

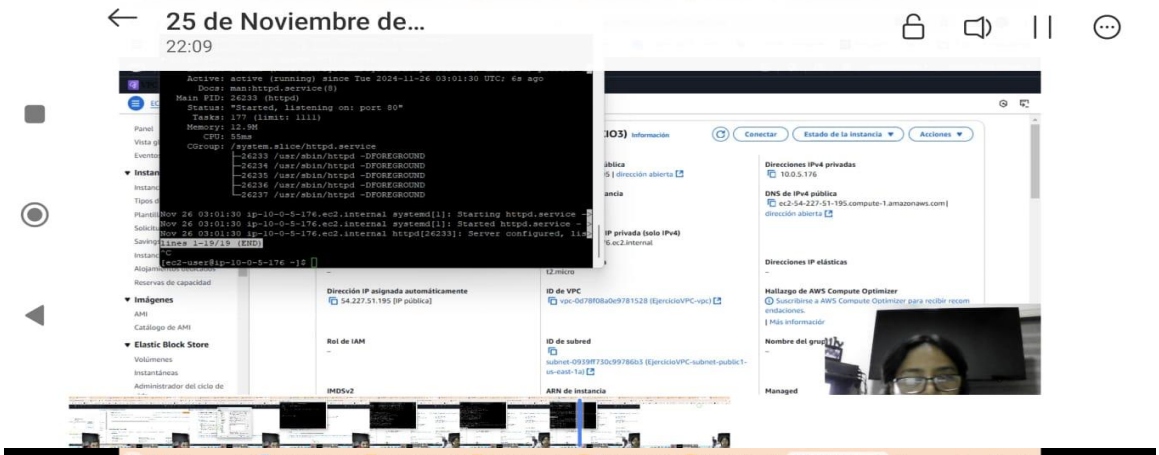

Console Home Page
Recently visited
Applications (0)
Welcome to AWS
AWS Health
Cost and use
Panel de VPC
Tus VPC (7)
Crear VPC
Configuración de la VPC
Vista previa

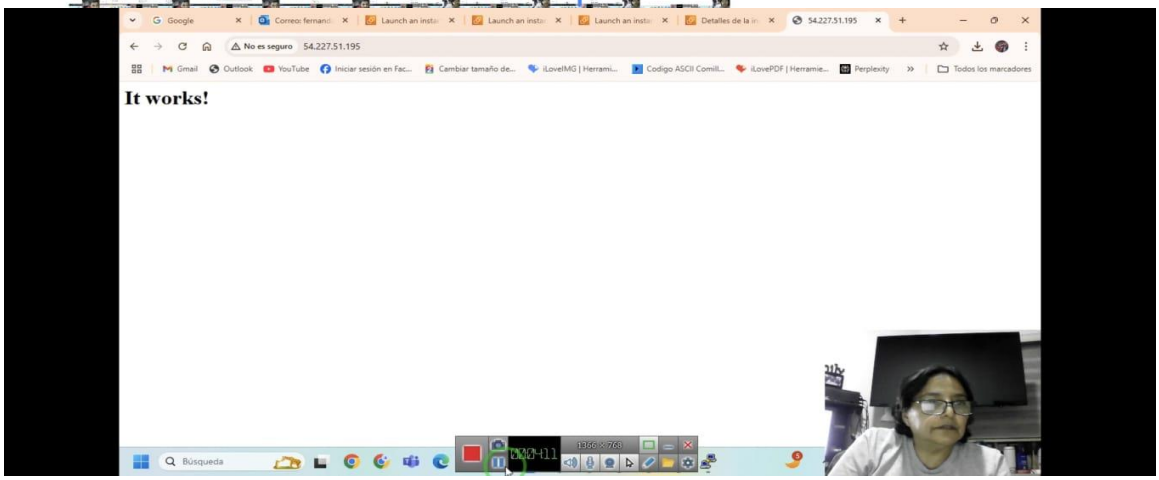

us-east-1.console.aws.amazon.com/ec2/home?region=us-east-1#LaunchInstances:
Launch an instance
Nombre y etiquetas
Imágenes de aplicaciones y sistemas operativos (Imagen de máquina de Amazon)
Tipo de instancia
Resumen
Configuraciones de red
Lanzamiento de instancia
Creación de reglas de grupo de seguridad
Detalles
Espere a que lancemos la instancia.
No cierre el navegador mientras se realiza la carga.


PuTTY Configuration
Conectarse a la instancia
Conexión de la instancia EC2
Lanzamiento de instancia
Creación de reglas de grupo de seguridad
Detalles
Espere a que lancemos la instancia.
No cierre el navegador mientras se realiza la carga.


PuTTY Configuration
Conectarse a la instancia
Conexión de la instancia EC2
PuTTY Security Alert
Consola de serie de EC2







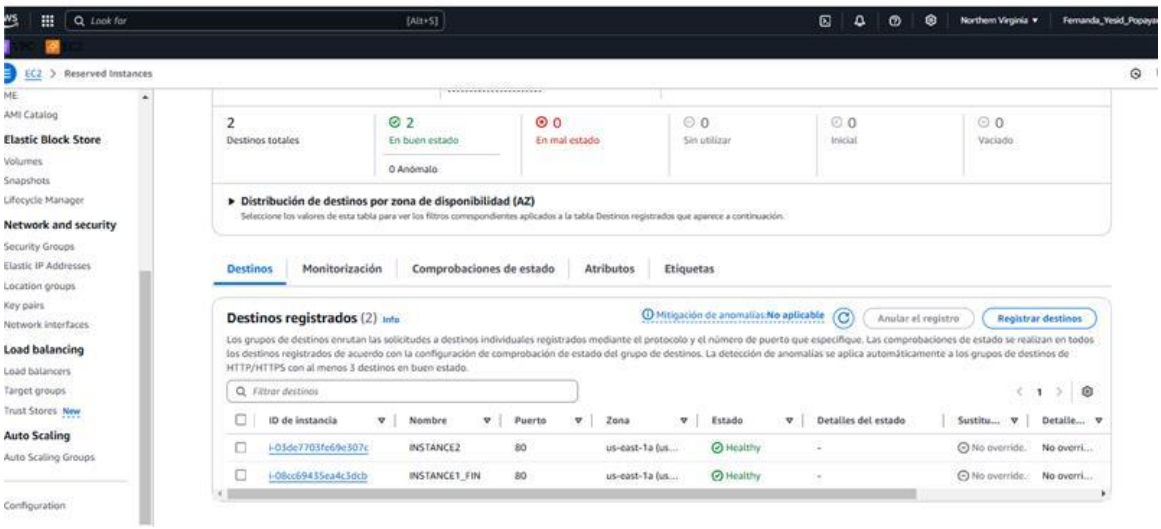
3.Link y pantallazos

https://youtu.be/sQ4pRhtaNdQ

ENTREGA N°2

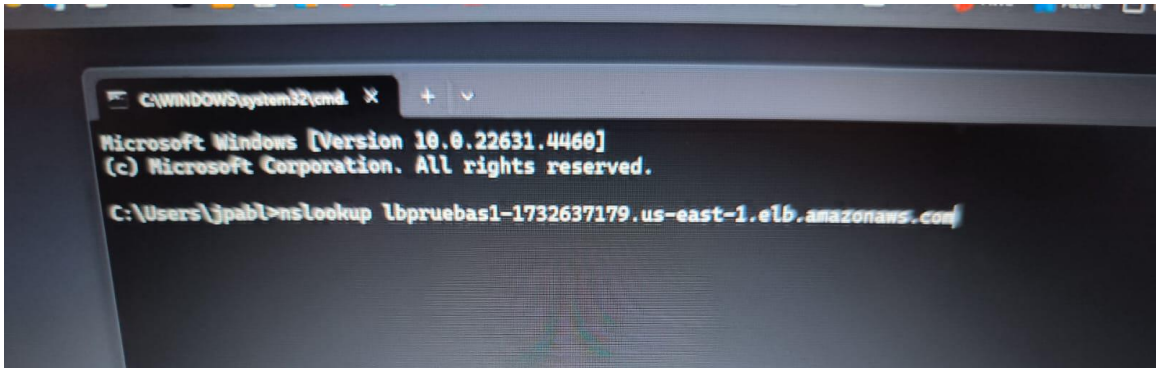
Como ya sabemos crear una **INSTANCIA**, verificar que funciona y correr el servidor web; vamos a ir al área de **VOLÚMENES** donde a partir de ese volumen vamos a crear un **SNAPSHOT** y a partir de este **SNAPSHOT** vamos a crear una **AMI** luego vamos a lanzar una segunda **INSTANCIA** utilizando la **AMI** que creamos y vamos a verificar su IP pública en el navegador.

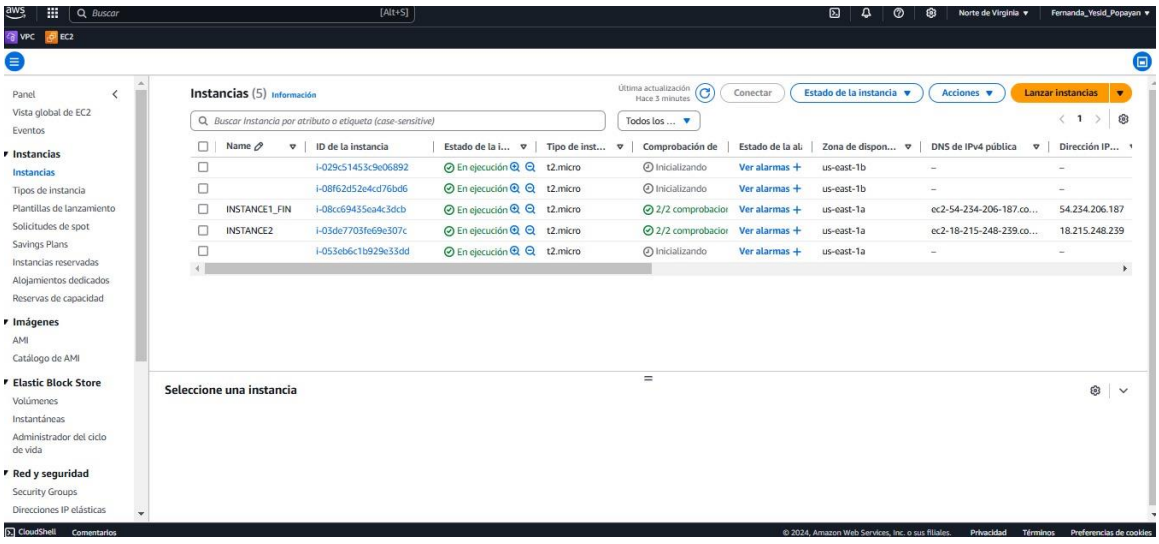
Luego de haber creado estas dos instancias vamos a ir a la configuración de **SUBREDES** y vamos a agregar una nueva subred de tipo pública, pasamos al área del **BALANCEADOR DE CARGA** donde vamos a darle una configuración necesaria y esta nos va a pedir la creación del **TARGETGROUP** qué es donde vamos a agrupar nuestras dos instancias, al ver que nuestro balanceador de carga se encuentra funcionando vamos a pasar al área de **AUTOSCALING** donde vamos a permitir que las instancias se creen automáticamente, al verificar que estas instancias se crean automáticamente vamos a eliminar las instancias que creamos inicialmente, luego de esto vamos a crear una instancia aparte que vamos a utilizar como servidor o administrador en la que vamos a ingresar a **PUTTY** como se hizo con la primera instancia cuando ya estemos dentro de la terminal vamos a utilizar el comando **ssh -y** ponemos el nombre de nuestro servidor y @ agregándole la IP pública o privada de una de las instancias que se está creando automáticamente; instalamos el paquete **h top** y verificamos que todo esté funcional con esto ya podemos saber que nuestra aplicación de creación de instancias automáticas se encuentra correctamente ejecutada.

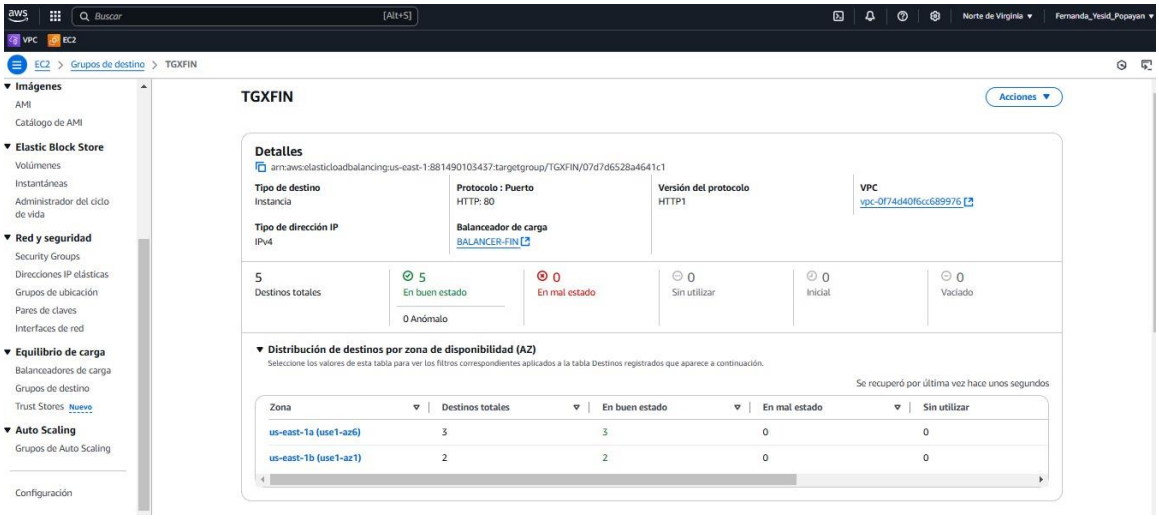


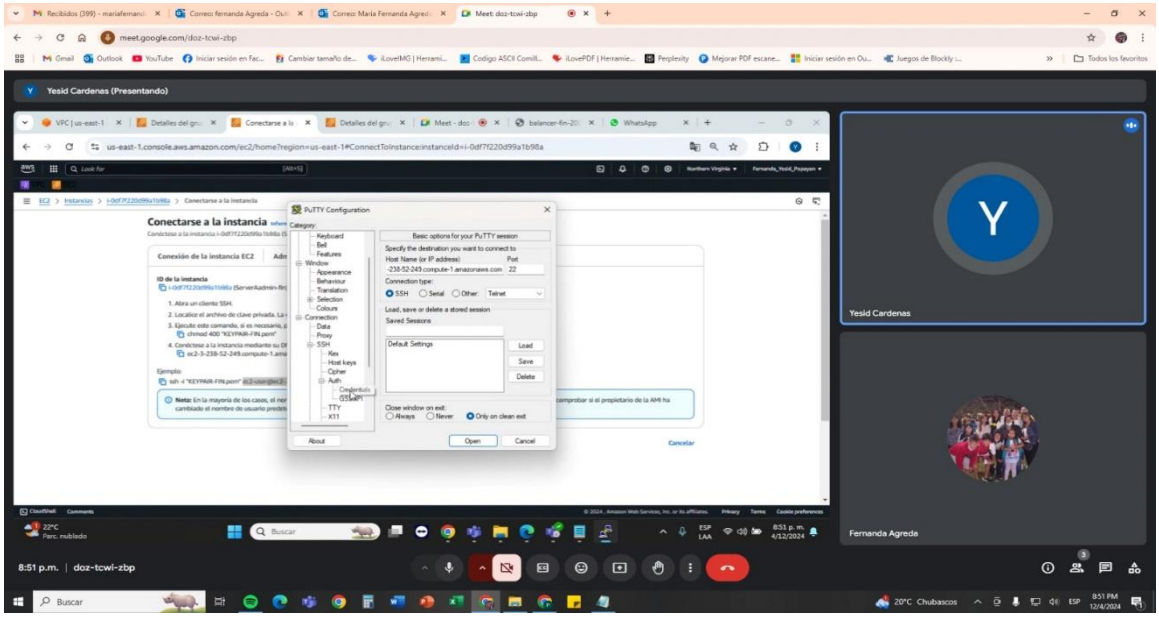

No es seguro
balancer-fin-2006180160.us-east-1.elb.amazonaws.com
NO TENGAS MIEDO DE FALLAR, TEN MIEDO DE INTENTARLO!!!

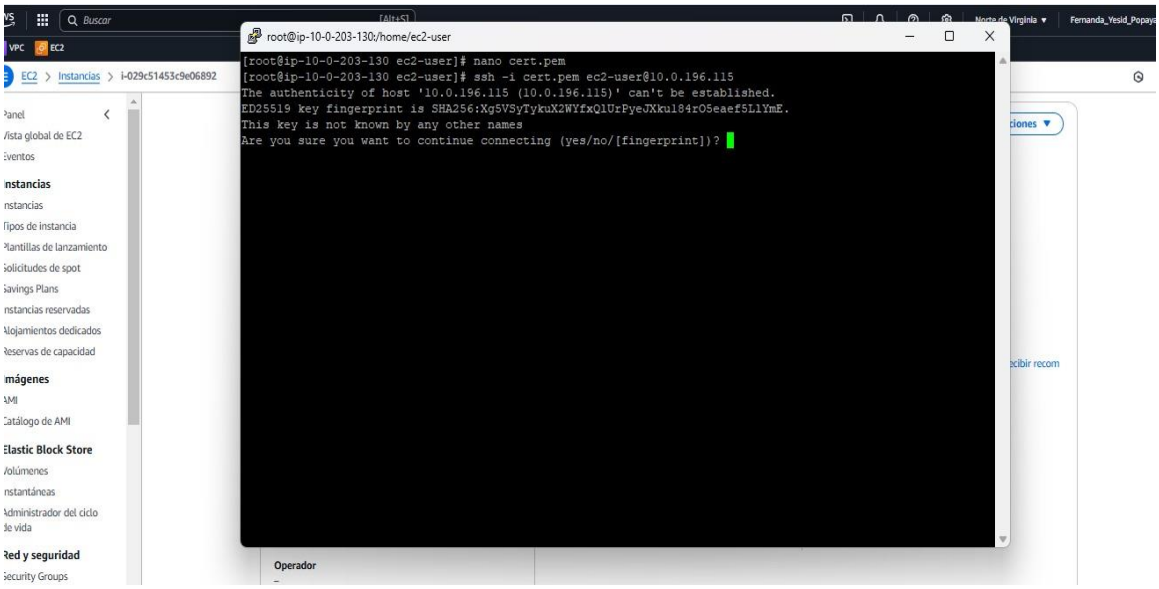

balancer-fin-2006180160.us-east-1.elb.amazonaws.com
http://balancer-fin-2006180160.us-east-1.elb.amazonaws.com/
balancer-fin-2006180160.us-east-1.elb.amazonaws.com
http://balancer-fin-2006180160.us-east-1.elb.amazonaws.com/
5:07 p. m.

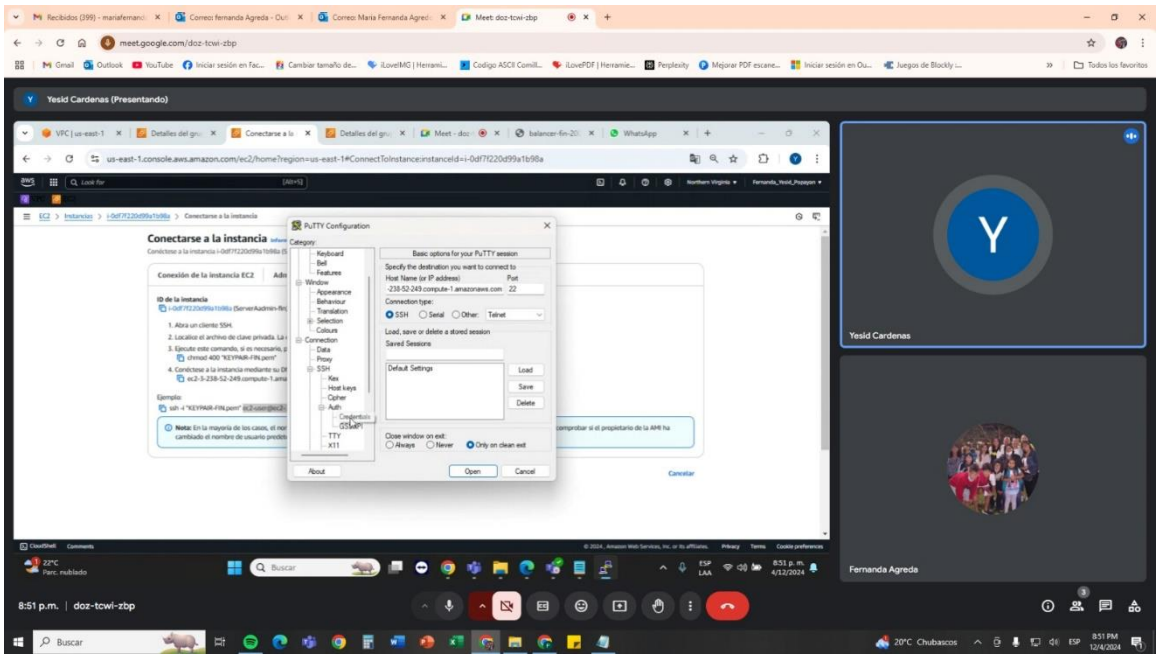

C:\WINDOWS\system32\cmd.exe
Microsoft Windows [Version 10.0.22631.4460]
(c) Microsoft Corporation. All rights reserved.
C:\Users\jpabl>nslookup lbpruebas1-1732637179.us-east-1.elb.amazonaws.com

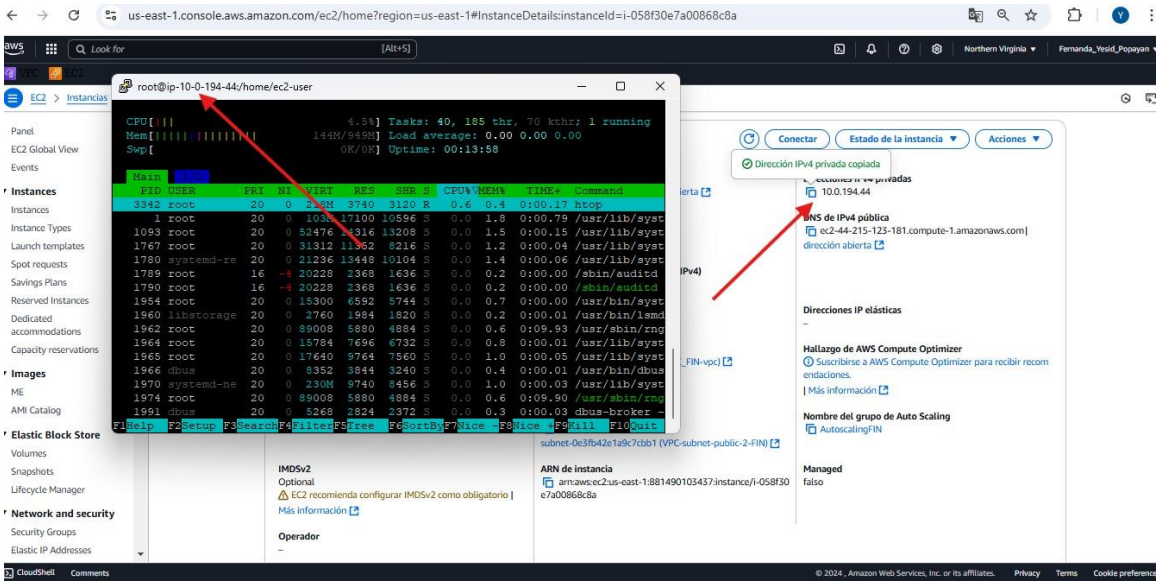

Instancias (5)
Seleccione una instancia


TGXFIN
Acciones
Detalles
arn:aws:elasticloadbalancing:us-east-1:881490103437:targetgroup/TGXFIN/07d7d6528a4641c1
Tipo de destino
Instancia
Protocolo : Puerto
HTTP: 80
Versión del protocolo
HTTP1
VPC
vpc-0f74d40f6cc689976
Tipo de dirección IP
IPv4
Balanceador de carga
BALANCER-FIN
5
Destinos totales
5
En buen estado
0 Anómalo
0
En mal estado
0
Sin utilizar
0
Inicial
0
Vaciado
Distribución de destinos por zona de disponibilidad (AZ)
Seleccione los valores de esta tabla para ver los filtros correspondientes aplicados a la tabla Destinos registrados que aparece a continuación.
Se recuperó por última vez hace unos segundos
Zona
Destinos totales
En buen estado
En mal estado
Sin utilizar
us-east-1a (use1-az6)
3
3
0
0
us-east-1b (use1-az1)
2
2
0
0


Yesid Cardenes (Presentando)
Conectarse a la instancia
PuTTY Configuration
Yesid Cardenas
Fernanda Agreda
8:51 p.m. | doz-tcwi-zbp


root@ip-10-0-203-130:/home/ec2-user
[root@ip-10-0-203-130 ec2-user]# nano cert.pem
[root@ip-10-0-203-130 ec2-user]# ssh -i cert.pem ec2-user@10.0.196.115
The authenticity of host '10.0.196.115 (10.0.196.115)' can't be established.
ED25519 key fingerprint is SHA256:Xg5VSyTykuX2WYfxQlUrPyeJXkul84rO5eaef5L1YmE.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])?
Operador


Yesid Cardenas (Presentando)
Conectarse a la instancia
PuTTY Configuration
Yesid Cardenas
Fernanda Agreda
8:51 p.m. | doz-tcwi-zbp



Enlace de youtube:

https://youtu.be/OA8MKAWh3To?si=7Sz-mbzND_kkb-rX

ENTREGA N°3

Primeramente debemos verificar la creación automática de las instancias con la imagen que ya teníamos establecida en el servidor web ejecutándose nos conectamos a ella y vamos al concepto de lo que es el docket que es una implementación a partir de la virtualización de los sistemas operativos

El servicio de docket nos permite tener varias máquinas virtuales al interior y cada máquina virtual puede tener un sistema operativo diferente con aplicaciones diferentes dentro de un contenedor

Docket es una marca o nombre de empresa donde se creo el concepto de contenedores

La idea principal de la creación e implementación del docket es para que los contenedores funcionen como un programa que se ejecuta dentro del sistema operativo funciona en cualquier Puerto o cualquier recurso que necesite comunicarse ya que tiene un puerto de red entonces se puede elegir el puerto con el que queramos trabajar teniendo en cuenta que los contenedores que están dentro de una misma máquina no pueden tener el mismo puerto

Dicho Puerto es por donde ingresan las peticiones el cual está definido en el security group o firewall de aws que es gestionado en la máquina virtual allí establecemos las peticiones y las reglas de entrada para poder habilitar el puerto que le dimos a dicha máquina virtual

En este caso instalamos nginx luego de detener apache para ver que se cumpla la función de servidor web en donde utilizamos los comandos

- Dnf install nginx -y
- Systemctl Status nginx
- Systemctl Start nginx
- Systemctl enable nginx

Luego pasamos a instalar docket con el comando:

- Yum install docket -y
- Systemctl Start docket
- Systemctl enable docket

- Systemctl Status docket

Allí podemos ver qué ya se encuentra ejecutándose y pasamos a crear el contenedor con una imagen de docket

Instalamos la imagen de acuerdo al comando docket pulll y el nombre de la imagen que en este caso es httpd

Creamos las diferentes carpetas con el comando mkdir y las utilizamos para darle los permisos con chmod

Con esto vamos a generar una ruta donde se puede apuntar a buscar los archivos interiormente dentro del contenedor esta va a estar apuntando internamente a la máquina anfitriona a la carpeta que acabamos de crear al caminar todo este proceso a partir de la instancia que creamos en donde esté corriendo nuestro servicio web de manera automática y donde instalamos el servicio de engine x que es otro tipo de servidor web ya podremos tomar una nueva instantánea para que a partir de ella podamos crear una ami y dar el arranque hasta este punto

Luego de verificar la habilitación de las diferentes ips en enex y en las carpetas correspondientes creadas en la máquina virtual vamos a detener el servicio de engine y utilizar el de docket para este ejercicio debemos de tener el servicio de engine ex para no causar un conflicto en el puerto 80

Dentro de la configuración y creación de las carpetas a partir de docket podemos encontrar un ID container este ID es una cadena de caracteres que podemos identificar al utilizar el comando cualesquiera El comando Ls

Con esto podemos detener el el contenedor docket stop y el ID container que nos generó nuevamente activamos los permisos para todos los archivos identificamos que sea funcional entonces cuando se ingresan las peticiones la función del proxy reverso es llegar a la petición y que esa petición venga con una dirección de origen

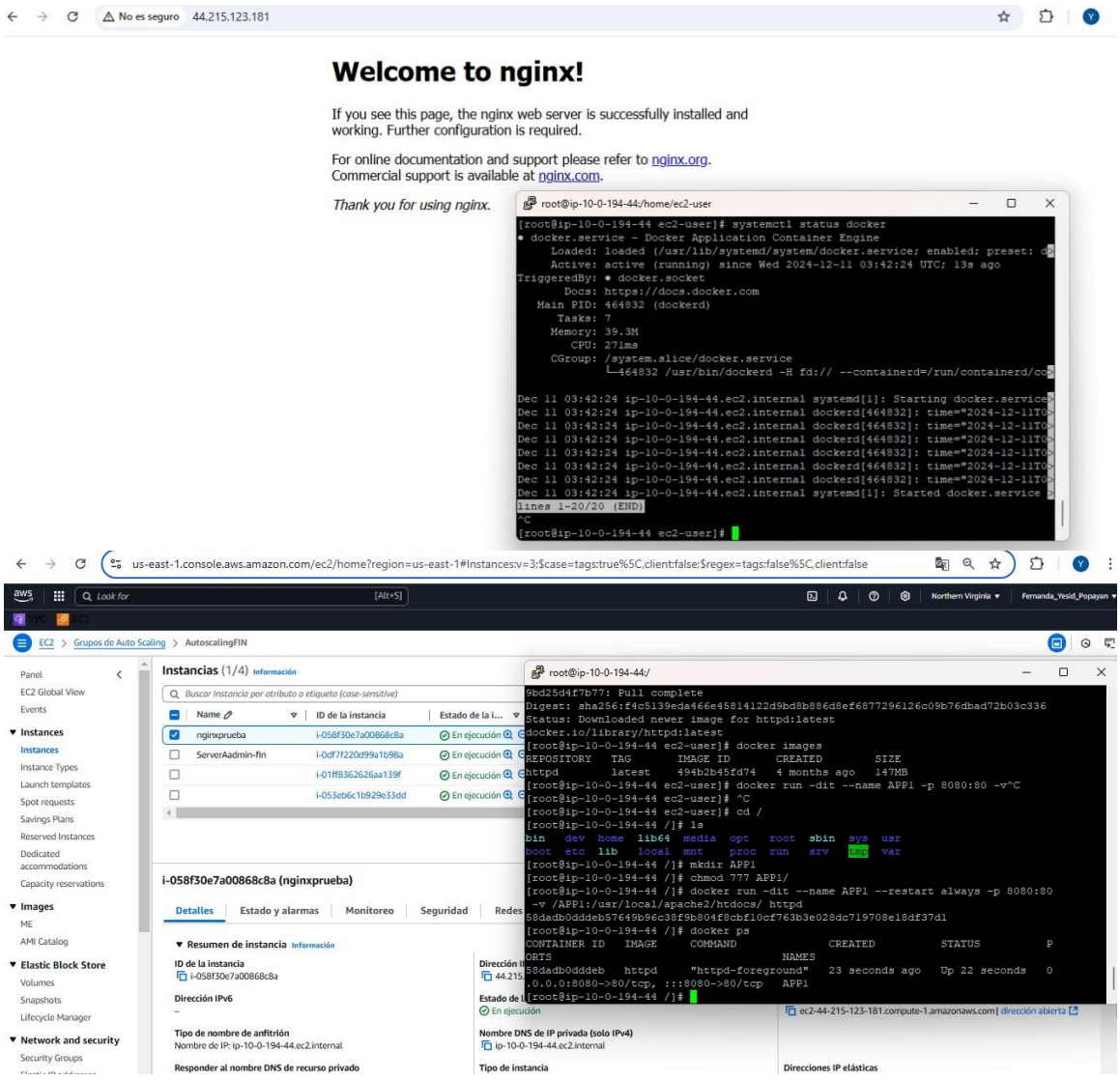
Luego de esto pasamos a la sección del offspring

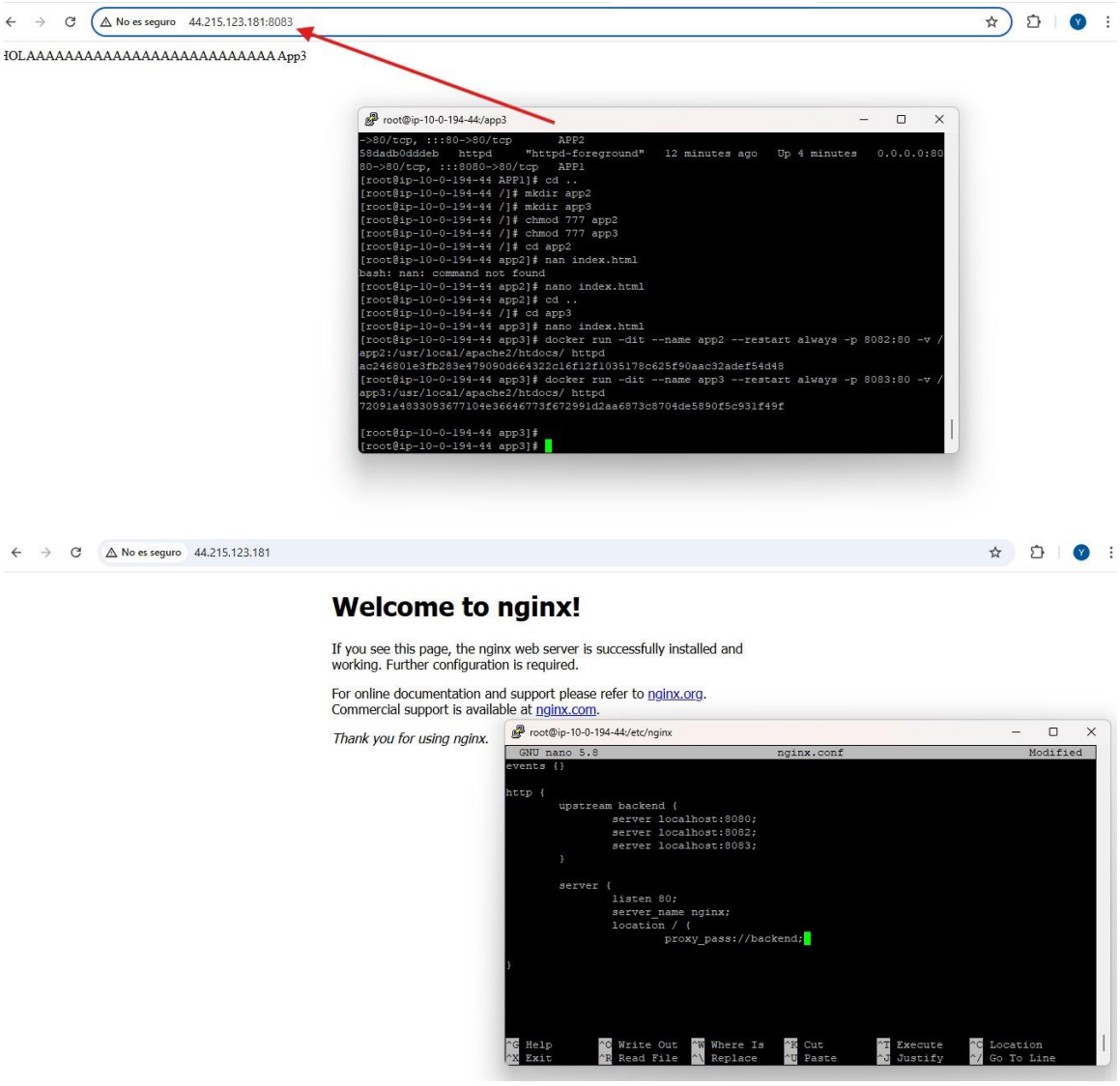
Donde se le indica a la configuración de Linux cuáles son los contenedores que vamos a tener y el nombre que se le da a este backen o servidor

Utilizamos los términos predeterminados para el diligenciamiento por medio del comando nano, entonces se utiliza server donde se está ejecutando el contenedor

en el local host y agregamos los contenedores que necesitamos según los puertos establecidos

Luego de la culminación de este proceso pasamos a comprobar la conexión del funcionamiento de El proxy reverso donde pegamos la IP pública en el navegador y con solo recargar la página obtenemos los nombres o el texto de prueba que pusimos en las diferentes rutas





Enlace de youtube:

https://youtu.be/UlUeAOQiQXQ

Enlace de youtube entrega final:

https://youtu.be/65l03onOONc

CONCLUSIONES

Este proyecto nos permitió adquirir una comprender mucho más acerca de la implementación y gestión de instancias en la nube, particularmente con Amazon Web Services (AWS). A través de la configuración y manejo de recursos como Amazon EC2, EBS, VPC aprendimos a construir una infraestructura escalable y flexible para ejecutar aplicaciones en la nube de manera eficiente. Además, la automatización de la creación de instancias fue un aspecto clave, ya que nos permitió optimizar recursos y reducir costos operativos al adaptar los servicios a la demanda sin intervención manual.

Otro componente fundamental fue el uso de Docker. Aprendimos a utilizar contenedores para desarrollar y probar aplicaciones de forma más ágil y eficiente, lo que simplifica la creación de entornos aislados y mejora la rapidez en el desarrollo de nuevas soluciones.

La combinación de estos servicios en la nube con tecnologías de virtualización y automatización nos permitió crear una infraestructura robusta, lista para adaptarse a un entorno digital en constante evolución. Este proyecto no solo nos proporcionó habilidades técnicas, sino que también nos preparó para enfrentar desafíos más complejos en el futuro, sentando las bases para el desarrollo de soluciones tecnológicas más avanzadas.

Referencias

AMAZON. (2024). *AWS.AMAZON*. Obtenido de AWS.AMAZON: https://aws.amazon.com/es/what-is-aws/

AMAZON SERVICE. (2024). *AWS*. Obtenido de AWS: https://docs.aws.amazon.com/es_es/AUTOSCALING/application/userguide/what-is-application-auto-scaling.html

AMAZON WEB SERVICE. (2024). *AWS*. Obtenido de AWS: https://docs.aws.amazon.com/es_es/vpc/latest/userguide/what-is-amazon-vpc.html

AMAZON WEB SERVICE. (2024). *AWS*. Obtenido de AWS: https://docs.aws.amazon.com/es_es/AWSEC2/latest/UserGuide/concepts.html

AMAZON WEB SERVICE. (2024). *AWS*. Obtenido de AWS: https://aws.amazon.com/es/docket/#:~:text=Docket%20es%20un%20sistema%20operativo,sistema%20operativo%20de%20un%20servidor.

PROGRESS. (s.f.). *PROGRESS*. Obtenido de PROGRESS: https://kemptechnologies.com/reverse-proxy?utm_source=google&utm_medium=cpc&utm_campaign=kemp-nb-adc-rp-ssl-ao-en-CALA&utm_term=reverse%20proxy&utm_id=&adgroupid=&network=&ad_copy=&utm_content=&gad_source=1&gclid=CjwKCAiAjeW6BhBAEiwAdKltMi7XRhV30sCNuLg4vE

WORDPRESS. (2015). *WORDPRESS*. Obtenido de WORDPRESS: https://brayangp.wordpress.com/wp-content/uploads/2015/02/que-es-putty.pdf