TRABAJO DE GRADO
Opción Seminario-Diplomado.

Cafetería las tres J
Seminario Amazon AWS

Corporación Universitaria Remington.
Ingeniería de sistemas

Jhonatan Pineda Carmona
Juan Pablo Ladino Betancur
Johan David Montoya Loaiza
Seminario de Grado
2024.

Tabla de Contenidos

Resumen

El presente trabajo tiene como objetivo implementar soluciones tecnológicas avanzadas utilizando servicios de Amazon Web Services (AWS) para optimizar la infraestructura de una aplicación web, simulando un entorno empresarial real en el contexto de una Cafetería Las Tres J. Se despliega una arquitectura que aprovecha instancias EC2 (Elastic Compute Cloud) y contenedores Docker, garantizando una plataforma con alta disponibilidad, escalabilidad y gestión eficiente del tráfico. A lo largo del documento, se detalla el proceso paso a paso para configurar y desplegar servicios en AWS.

Marco conceptual y contextual

El presente informe técnico se centra en la implementación de soluciones basadas en Amazon Web Services (AWS), específicamente en el despliegue de instancias virtuales y contenedores para garantizar una infraestructura escalable y de alta disponibilidad. Este trabajo se enmarca dentro de un seminario de grado sobre AWS, en el que se abordan conceptos fundamentales de computación en la nube, virtualización, y balanceo de carga. Los principales conceptos trabajados incluyen:

Instancias EC2 (Elastic Compute Cloud): Servidores virtuales en la nube para ejecutar aplicaciones.

Contenedores Docker: Herramientas de virtualización ligera que permiten ejecutar múltiples aplicaciones de forma aislada.

Balanceo de Carga (Elastic Load Balancer): Distribución automática del tráfico para evitar sobrecargas.

Escalado Automático (Auto Scaling): Ajuste dinámico de recursos según la demanda.

Redes Virtuales (Amazon VPC): Segmentación y aislamiento de redes en la nube.

El informe describe un ejercicio práctico de implementación en una simulación empresarial para la Cafetería Las Tres J, una empresa ficticia que busca optimizar su infraestructura tecnológica. El objetivo es asegurar que la plataforma web de la cafetería pueda manejar grandes volúmenes de usuarios sin interrupciones y con la mayor seguridad posible. Se aplican los conocimientos adquiridos en el seminario para configurar servidores, implementar balanceo de carga, y desplegar aplicaciones en contenedores, simulando una operación empresarial real.

En esta sección se detalla la implementación y configuración de servicios esenciales de AWS para el despliegue de aplicaciones. Se abordan los siguientes temas:

Recursos de AWS para ejecutar instancias

Amazon EC2: Creación y configuración de instancias.

Amazon AMI (Amazon Machine Images): Plantillas para lanzar instancias con configuraciones predefinidas.

Grupos de Seguridad: Reglas de firewall para proteger las instancias.

Pares de Claves SSH: Mecanismos de autenticación para acceso seguro.

Servicios Complementarios a EC2:

Amazon VPC: Creación de redes privadas para aislar instancias.

Elastic Load Balancing: Distribución del tráfico entre instancias.

Amazon EBS: Almacenamiento persistente para instancias.

Amazon S3: Almacenamiento de objetos para respaldos y archivos estáticos.

Se ofrece una descripción detallada de cómo estos recursos se configuran e integran para lograr una infraestructura eficiente y de alta disponibilidad, adecuada para soportar aplicaciones web en un entorno empresarial.

Servicios Complementarios a EC2 con Ejemplos

Amazon Virtual Private Cloud (VPC)

Ejemplo: Imagina que tienes una aplicación web y un servidor de base de datos. Puedes crear dos subredes dentro de tu VPC, una para la aplicación web y otra para la base de datos. Esto te permite segmentar tu red y controlar el acceso entre los diferentes componentes de tu aplicación.

Usos comunes

Aislamiento de redes: Crear redes privadas completamente separadas para diferentes equipos o aplicaciones.

Seguridad mejorada: Controlar el flujo de tráfico entre subredes y hacia internet.

Redes definidas por software: Configurar redes complejas de forma automatizada.

Elastic Load Balancing (ELB)

Ejemplo: Tienes una aplicación web muy popular que experimenta picos de tráfico. Puedes utilizar un Load Balancer para distribuir el tráfico entre múltiples instancias de tu aplicación. Esto te permite manejar un mayor volumen de solicitudes y mejorar la disponibilidad de tu aplicación.

Usos comunes

Distribución de carga: Equilibrar el tráfico entre múltiples instancia

Alta disponibilidad: Garantizar que tu aplicación esté disponible incluso si una instancia falla.

Balanceo de carga basado en el contenido: Dirigir las solicitudes a instancias específicas en función de criterios como el encabezado de host o el path de la URL.

Amazon Elastic Block Store (EBS)

Ejemplo: Necesitas almacenar datos persistentes para tu aplicación, como archivos de base de datos o imágenes de usuario. Puedes crear volúmenes EBS y adjuntarlos a tus instancias. Si una instancia falla, puedes iniciar una nueva instancia y adjuntar el volumen EBS existente para recuperar tus datos.

Usos comunes

Almacenamiento persistente para instancias: Almacenar datos que deben persistir más allá del ciclo de vida de una instancia.

Copias de seguridad: Crear instantáneas de volúmenes EBS para realizar copias de seguridad.

Snapshots: Crear imágenes de punto en el tiempo de un volumen EBS para realizar restauraciones.

Amazon Simple Storage Service (S3)

Ejemplo: Tienes una aplicación que genera grandes cantidades de archivos de registro. Puedes utilizar S3 para almacenar estos archivos de forma segura y duradera. S3 también te permite compartir estos archivos con otros usuarios o aplicaciones.

Usos comunes

Almacenamiento de objetos: Almacenar cualquier tipo de archivo, desde imágenes y videos hasta datos de aplicaciones.

Copias de seguridad: Almacenar copias de seguridad de datos de bases de datos, sistemas operativos y aplicaciones.

Archivos estáticos: Servir archivos estáticos para sitios web, como imágenes, CSS y JavaScript.

Otros Servicios Complementarios

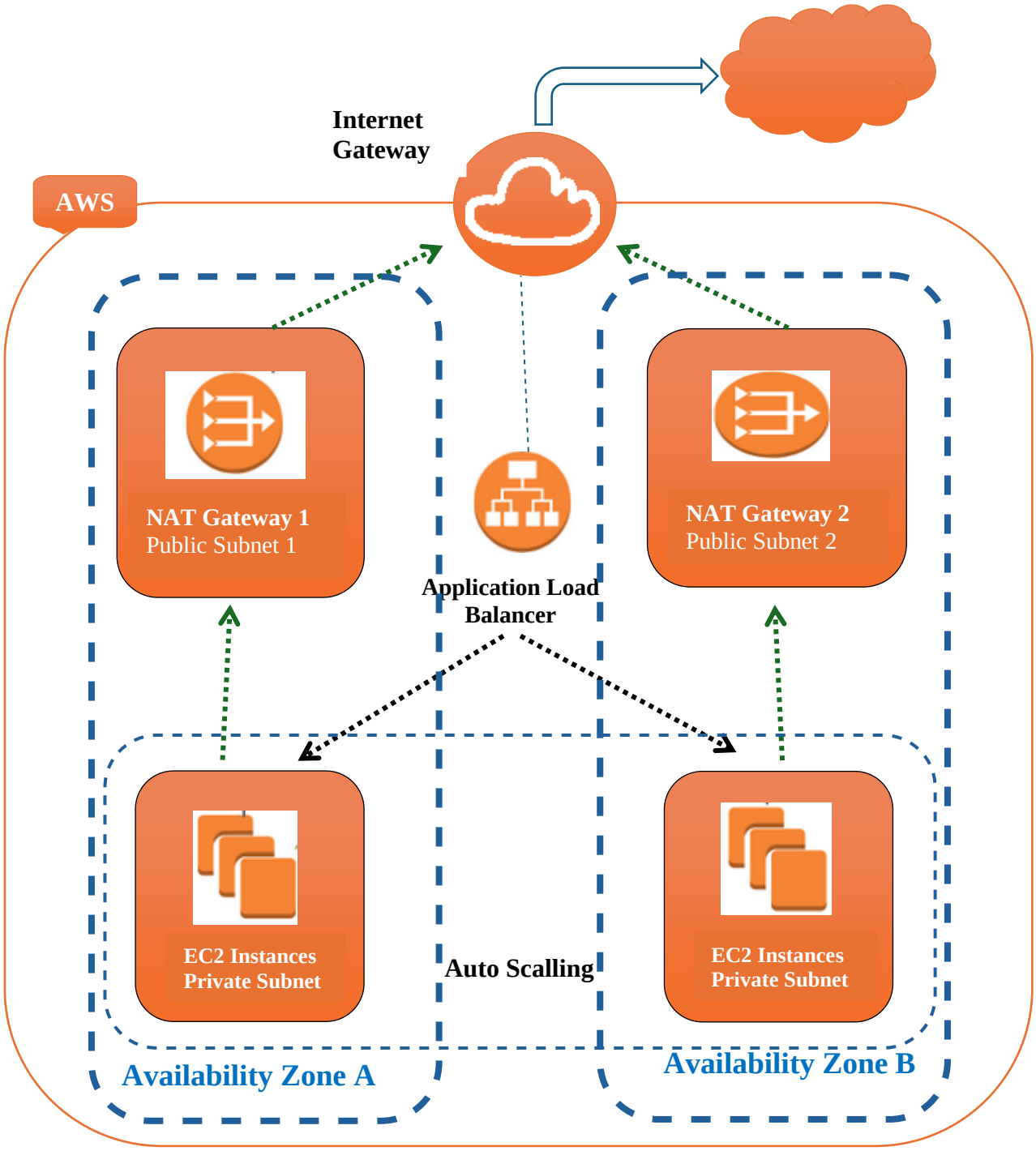
Auto Scaling: Escalar automáticamente el número de instancias en función de la demanda.

CloudWatch: Monitorear el rendimiento de tus instancias y aplicaciones.

Amazon RDS: Gestionar bases de datos relacionales.

Amazon ElastiCache: Implementar cachés in-memory para mejorar el rendimiento de tus aplicaciones.

Diagrama de flujo



Configuración servidor web en Amazon Linux

La imagen muestra el proceso de configuración inicial de una instancia EC2 en AWS. En esta configuración, se selecciona un par de claves para acceso seguro, se utiliza una VPC y subred predeterminadas, y se habilita la asignación automática de una IP pública, lo que permite que la instancia sea accesible desde Internet. Esta configuración es ideal para implementar servicios accesibles externamente, como un servidor web, y está optimizada para el nivel gratuito de AWS.

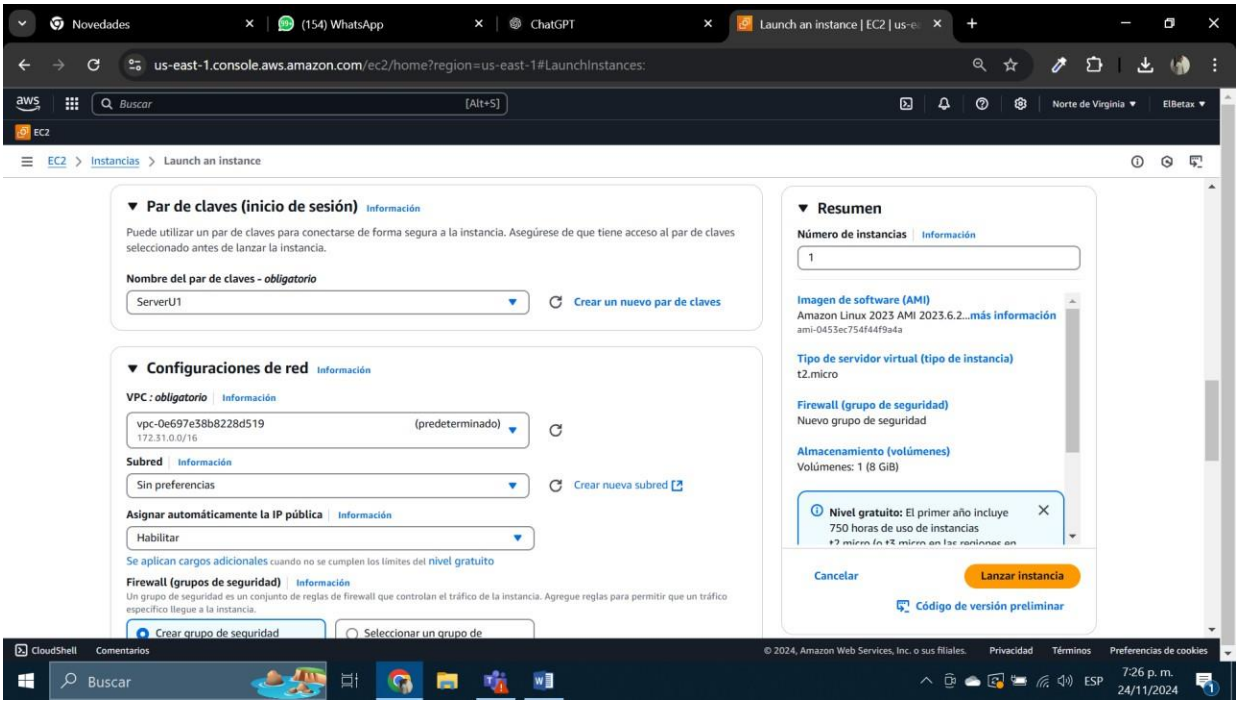


Figura 1 *Se asigna subred predeterminada y se habilita la ip pública*

La imagen ilustra los pasos finales en la configuración de una instancia EC2, destacando la definición de reglas de seguridad para el acceso remoto mediante SSH (puerto 22). Además, se valida el almacenamiento asignado y la configuración general antes de lanzar la instancia. Se enfatiza la importancia de confirmar que las reglas de seguridad sean adecuadas para proteger la instancia de accesos no autorizados.

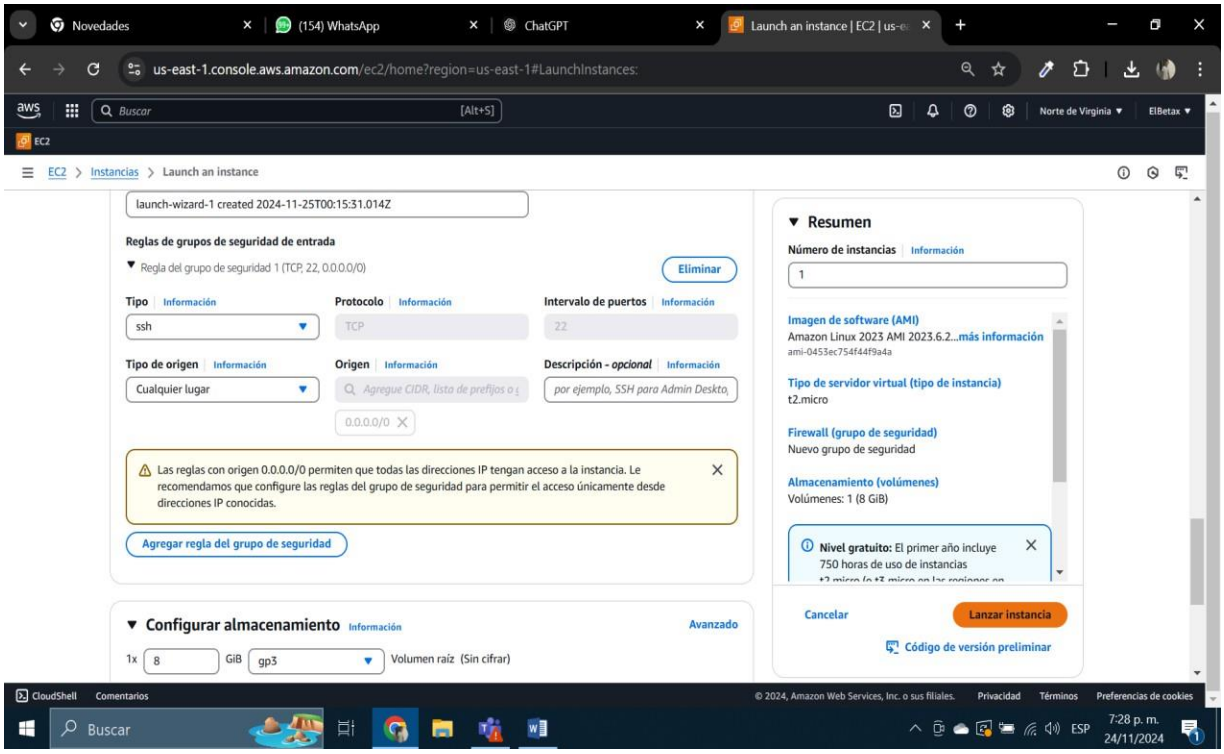


Figura 2 *Se lanza la instancia después de confirmar que la configuración de seguridad esté ok*

La imagen presenta los detalles completos de una instancia EC2 ya creada y en ejecución en AWS. Se destacan las direcciones IP asignadas (pública y privada), la red (VPC y subred) y el estado actual de la instancia. Esta información es clave para administrar y conectar la instancia a otros servicios o configuraciones específicas

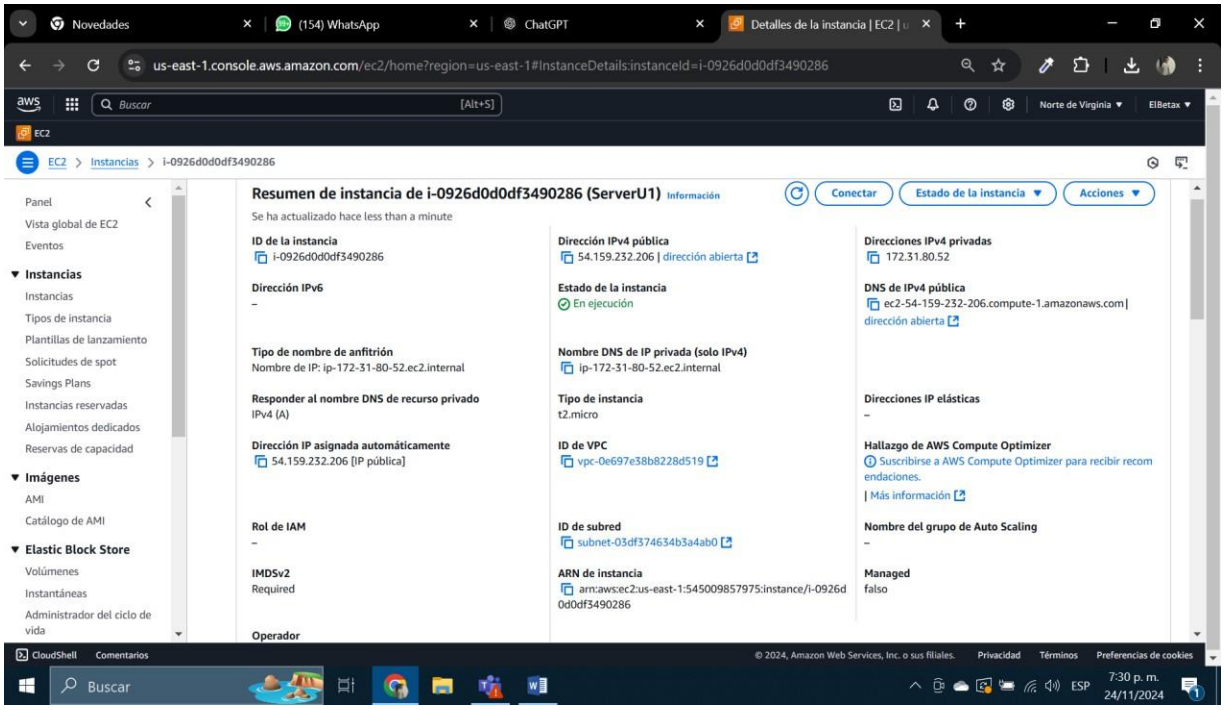


Figura 3 *Información de la instancia ya creada como ip privada y pública, subred, vpc, etc.*

La imagen captura una sección de la consola de gestión de AWS que muestra los detalles de seguridad de una instancia EC2. Se observa que la instancia está configurada con un grupo de seguridad que permite el acceso desde cualquier dirección IP a través de cualquier puerto. Esta configuración extremadamente permisiva expone la instancia a un riesgo significativo de seguridad, ya que prácticamente cualquier dispositivo en internet puede intentar establecer una conexión.

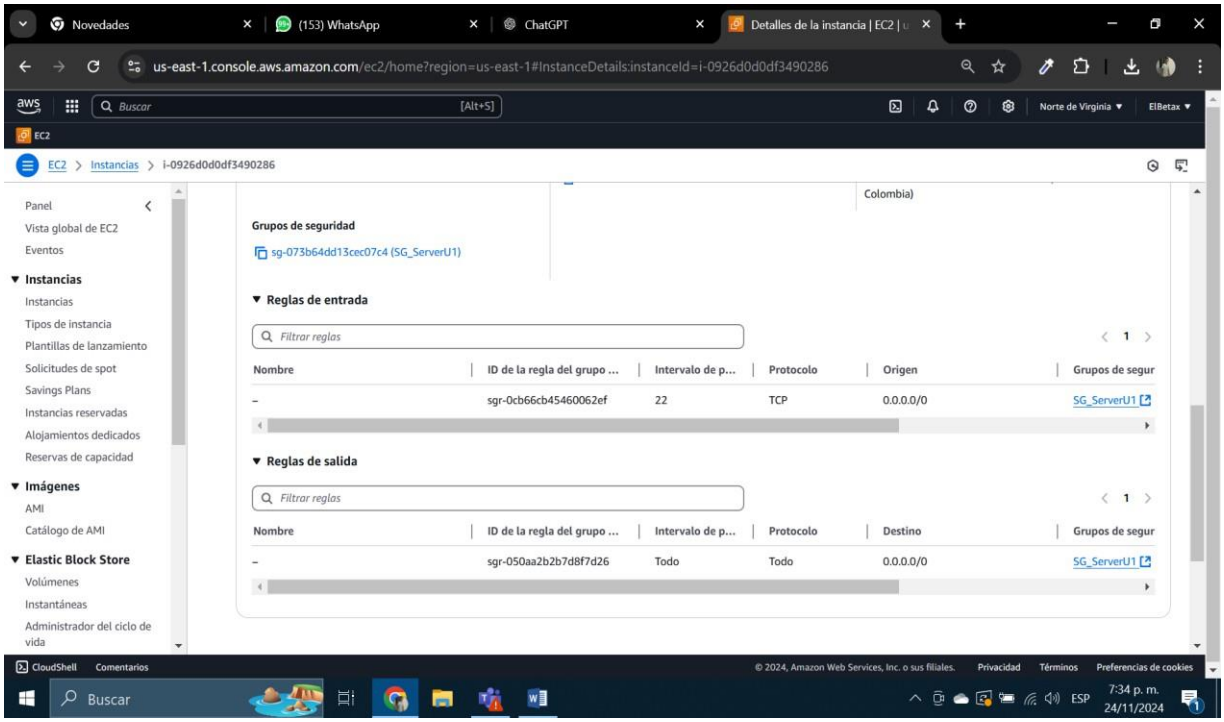


Figura 4 *Regla de entrada que permite el acceso para todos*

Esta imagen proporciona una vista general de las instancias EC2 creadas en un entorno específico de AWS. Permite a los administradores verificar el estado de las instancias, su configuración y realizar diversas acciones de gestión como iniciar, detener o terminar instancias.

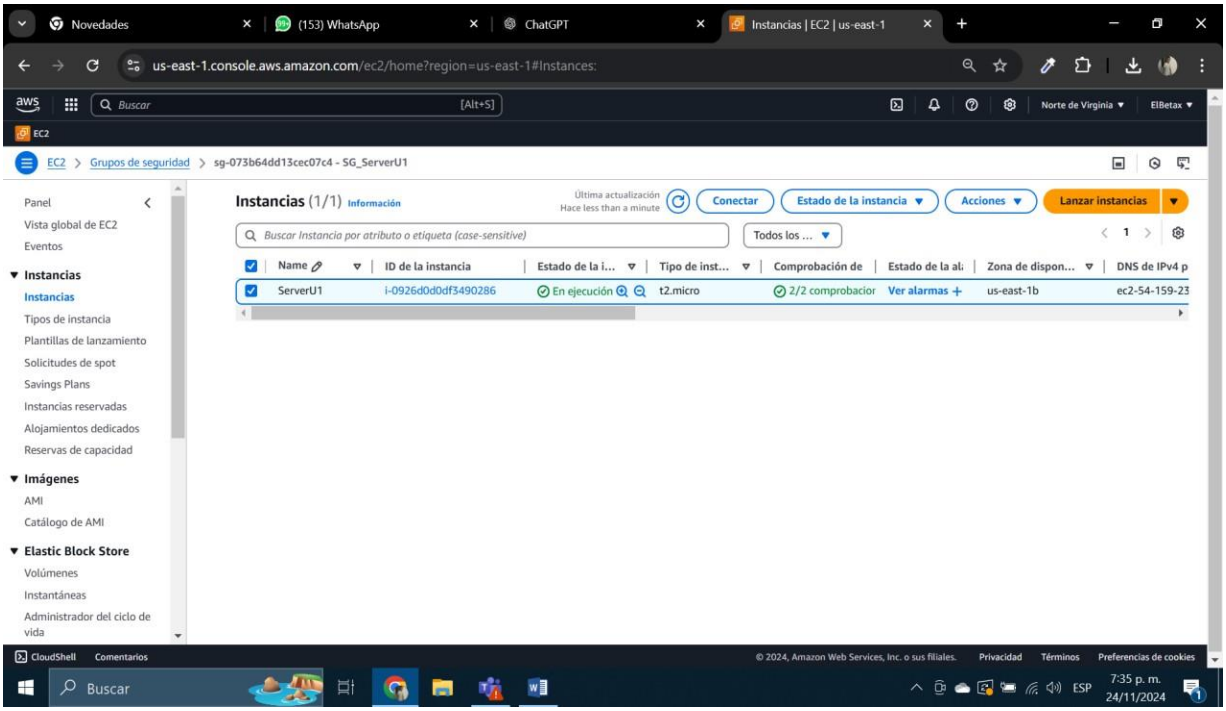


Figura 5 *Confirmamos que esté 2/2 en comprobación, lo que indica que ya finalizó la creación*

Esta imagen representa un paso fundamental en la gestión de una instancia EC2. Al conectar a la instancia a través de SSH, los administradores pueden ejecutar comandos directamente en la línea de comandos de la instancia, lo que les permite realizar una amplia variedad de tareas, como instalar software, configurar servicios, solucionar problemas, etc.

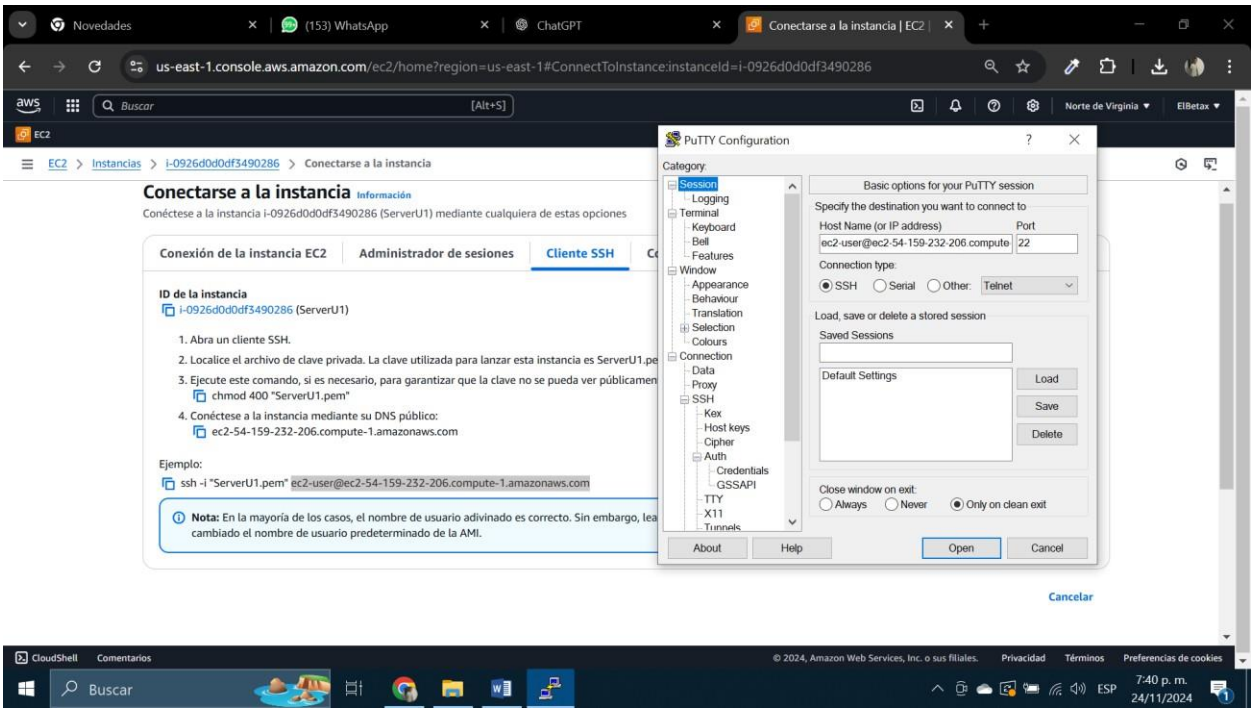


Figura 6 *Se ingresa a putty y se asigna la cadena de conexión*

Al configurar la autenticación basada en claves SSH, se está estableciendo un mecanismo de seguridad más robusto para acceder a la instancia EC2. Esto se debe a que las claves SSH son más difíciles de robar y crackear en comparación con las contraseñas. Además, las claves SSH permiten automatizar el proceso de conexión a la instancia.

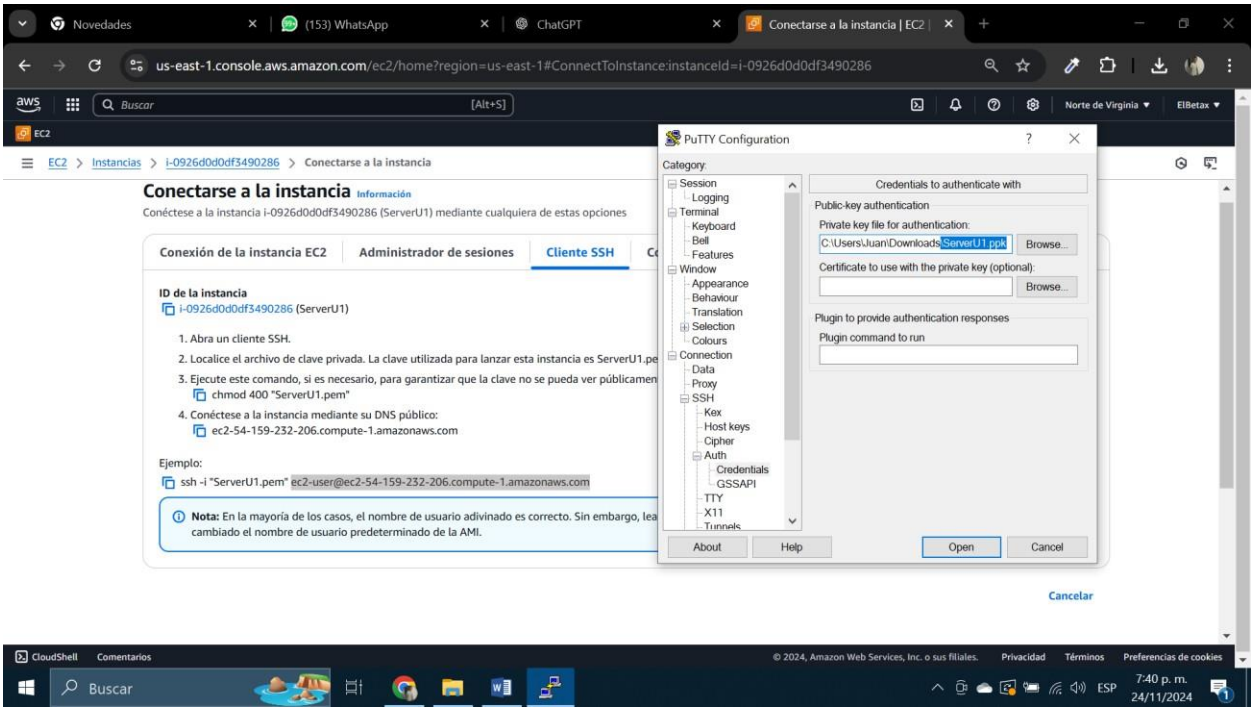


Figura 7 *Se carga la cadena de seguridad que se descargó previamente.*

Esta imagen representa el punto final del proceso de conexión a una instancia EC2. Al obtener acceso a la línea de comandos de la instancia, podemos realizar una amplia variedad de tareas de administración, como instalar software, configurar servicios, ejecutar scripts, etc.

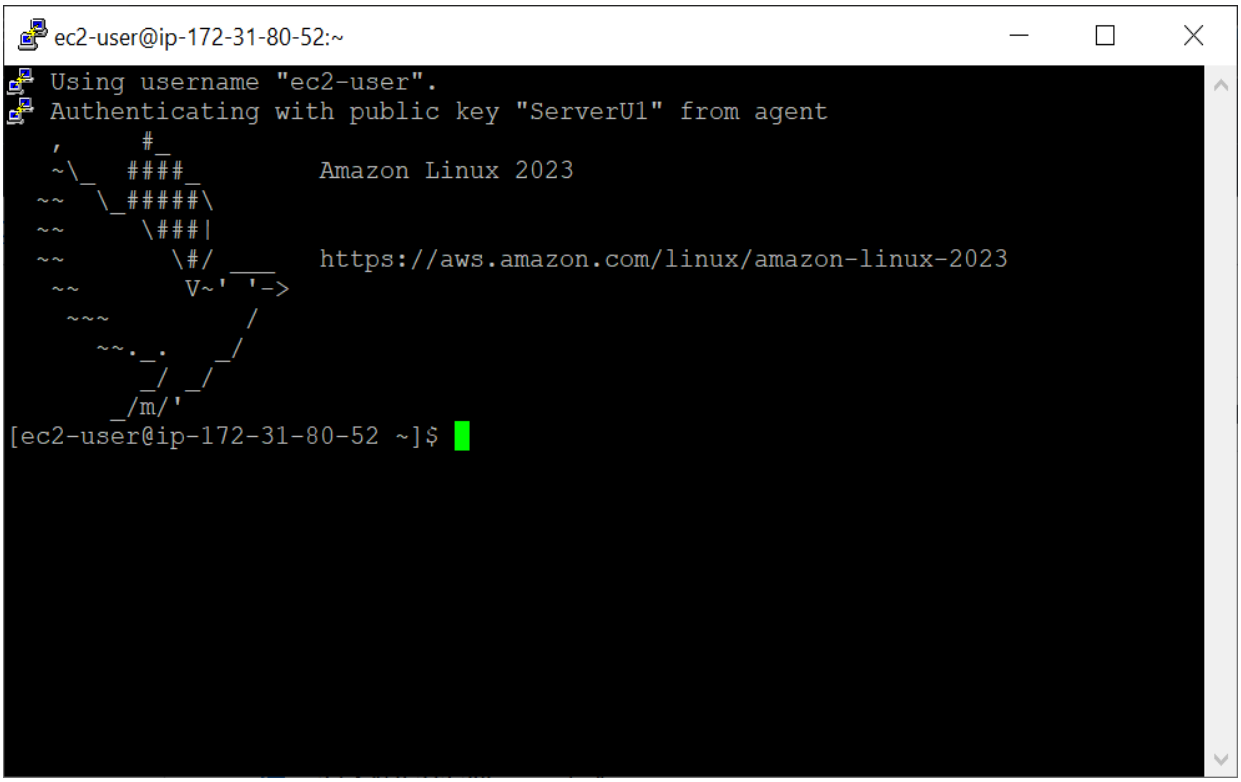


Figura 8 *Se crea acceso a la máquina virtual desde la consola.*

Esta imagen nos permite visualizar de forma clara el proceso de instalación de un software en una instancia EC2. En este caso, se está instalando el servidor web Apache, lo que permitirá a la instancia servir contenido web.

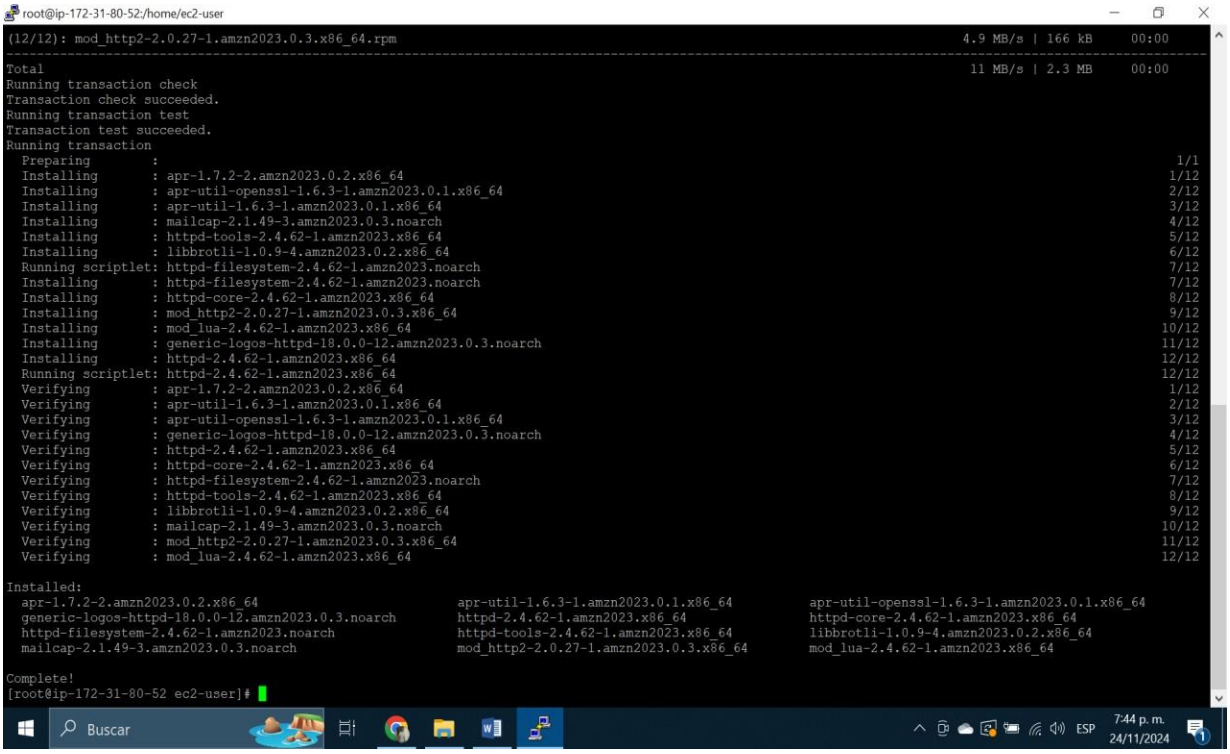


Figura 9 *Se instala servicio apache.*

Esta imagen nos muestra el paso final en la configuración de un servidor web Apache en una instancia EC2. Al verificar que el servicio Apache está en ejecución y escuchando en el puerto 80, podemos confirmar que nuestra instancia está lista para servir contenido web.

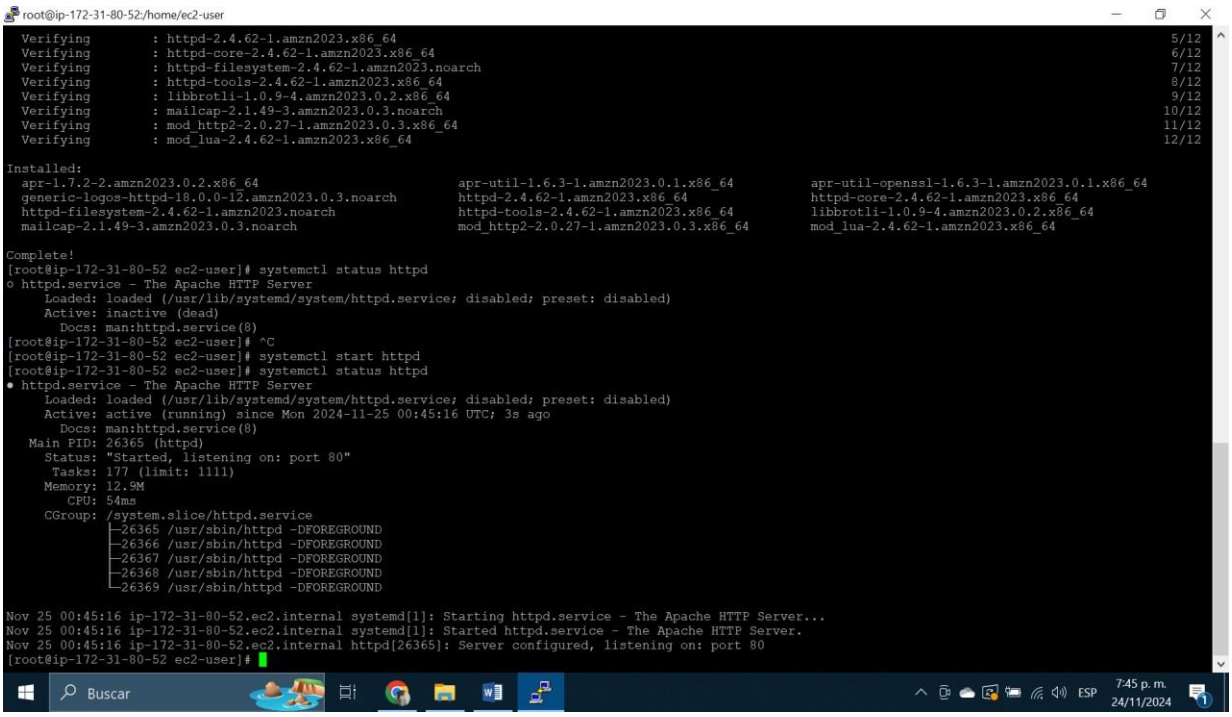


Figura 10 *Se corre servicio apache ya que se instala y queda apagado.*

Esta imagen muestra un paso crucial en la configuración de la seguridad de una instancia EC2. Al crear una regla de entrada específica, se está permitiendo el acceso a la instancia desde una dirección IP determinada, lo que garantiza que solo usuarios autorizados puedan acceder a ella.

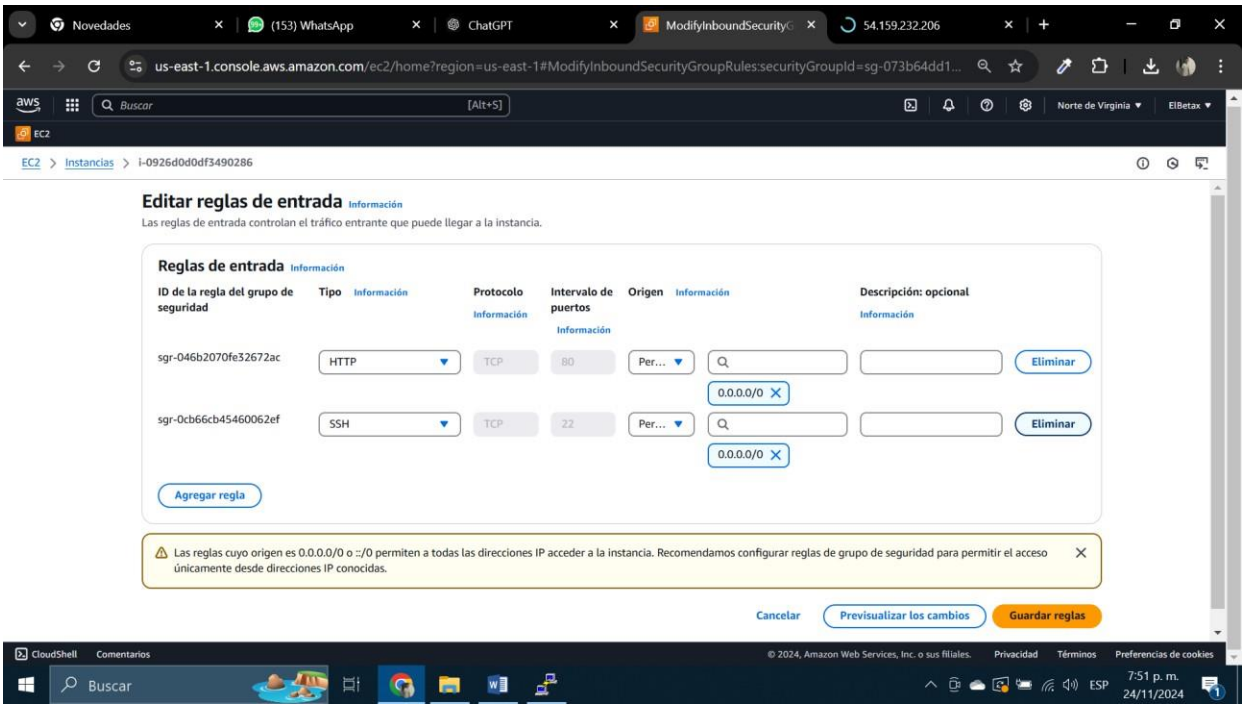


Figura 11 *Se crea nueva regla de entrada que permita acceder a la ip pública del servidor.*

Esta imagen es una confirmación visual de que una determinada tarea o configuración se ha realizado correctamente, una configuración y prueba exitosa.

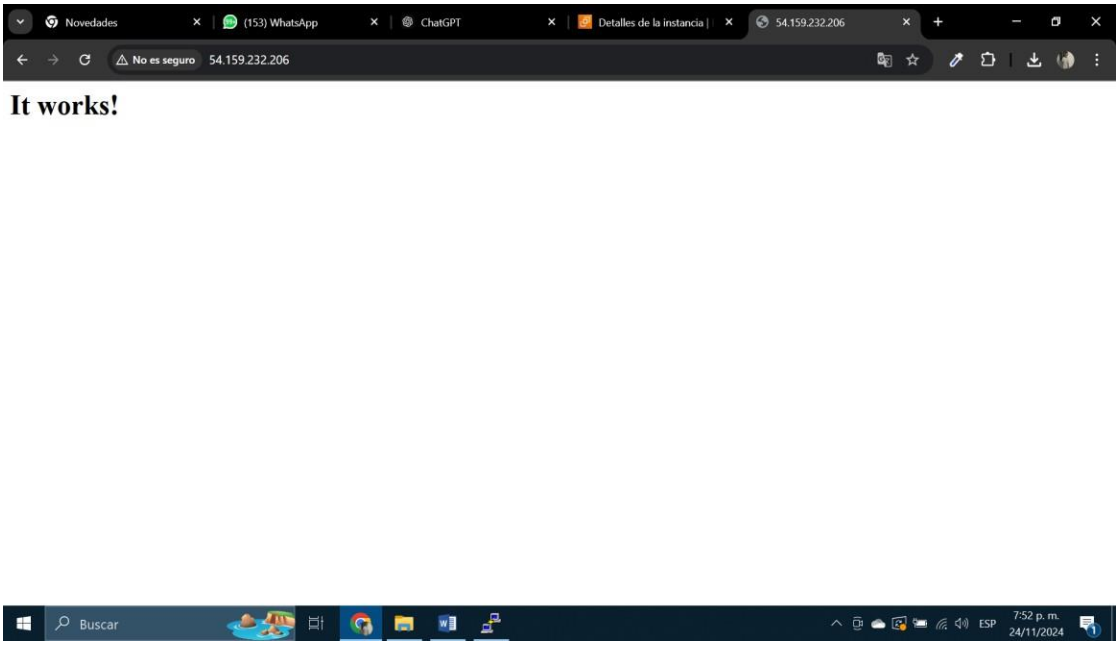


Figura 12 *Funcionamiento ok.*

Esta imagen muestra los pasos básicos para crear una página web simple en un servidor Apache. Al crear un archivo index.html y configurar los permisos correctamente, se puede servir contenido web desde la instancia EC2.

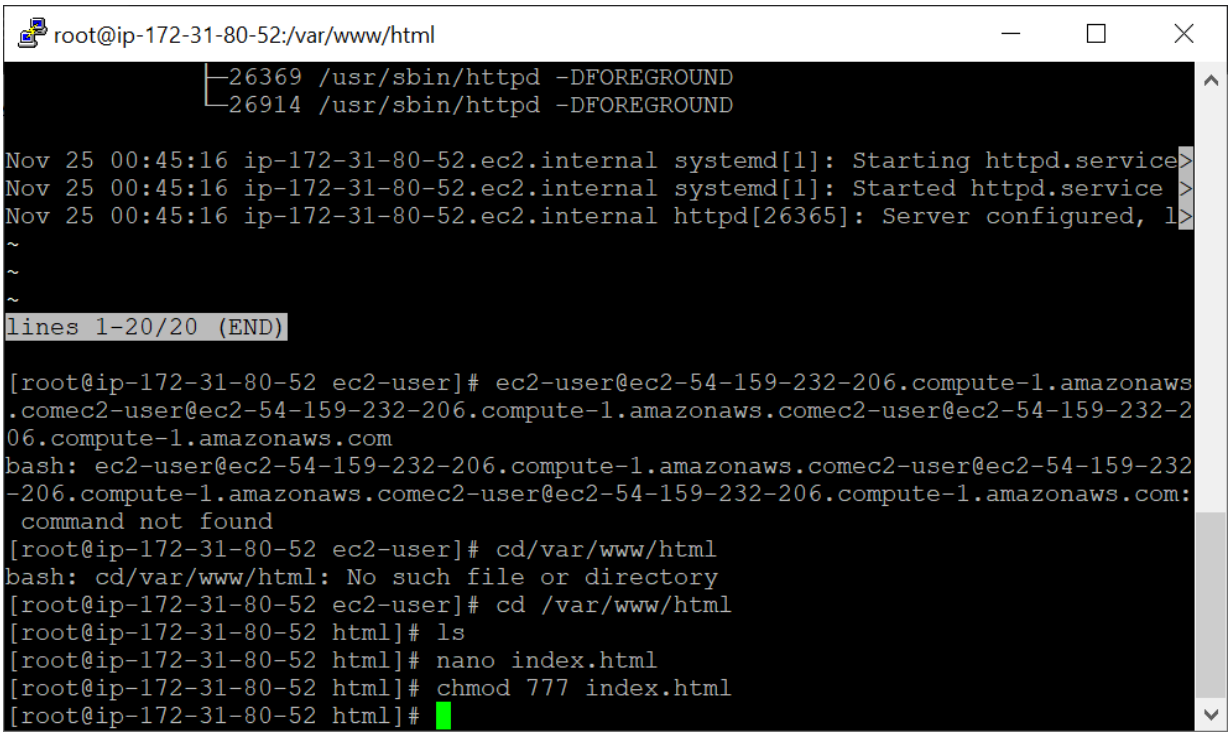


Figura 13 *Se crea archivo index.html para editar la pági*

Esta imagen representa un paso básico en la configuración de un servidor web. Al crear una página HTML simple y acceder a ella a través de un navegador, se verifica que el servidor web esté funcionando correctamente y que pueda servir contenido estático.

Figura 14 *Se confirma que el archivo html muestre el contenido el servidor web.*

Video explicativo Primer entrega:

[Video]. YouTube. **https://youtu.be/iQIzbvXArjE**

Crear y configurar el balanceador de carga.

La imagen captura el momento en que se está configurando un balanceador de carga de aplicaciones en AWS. El usuario está definiendo los parámetros básicos del balanceador, como su nombre, esquema de enrutamiento y tipo de dirección IP. Una vez configurado, este balanceador de carga podrá distribuir el tráfico entrante entre múltiples instancias EC2, mejorando la disponibilidad y el rendimiento de la aplicación.

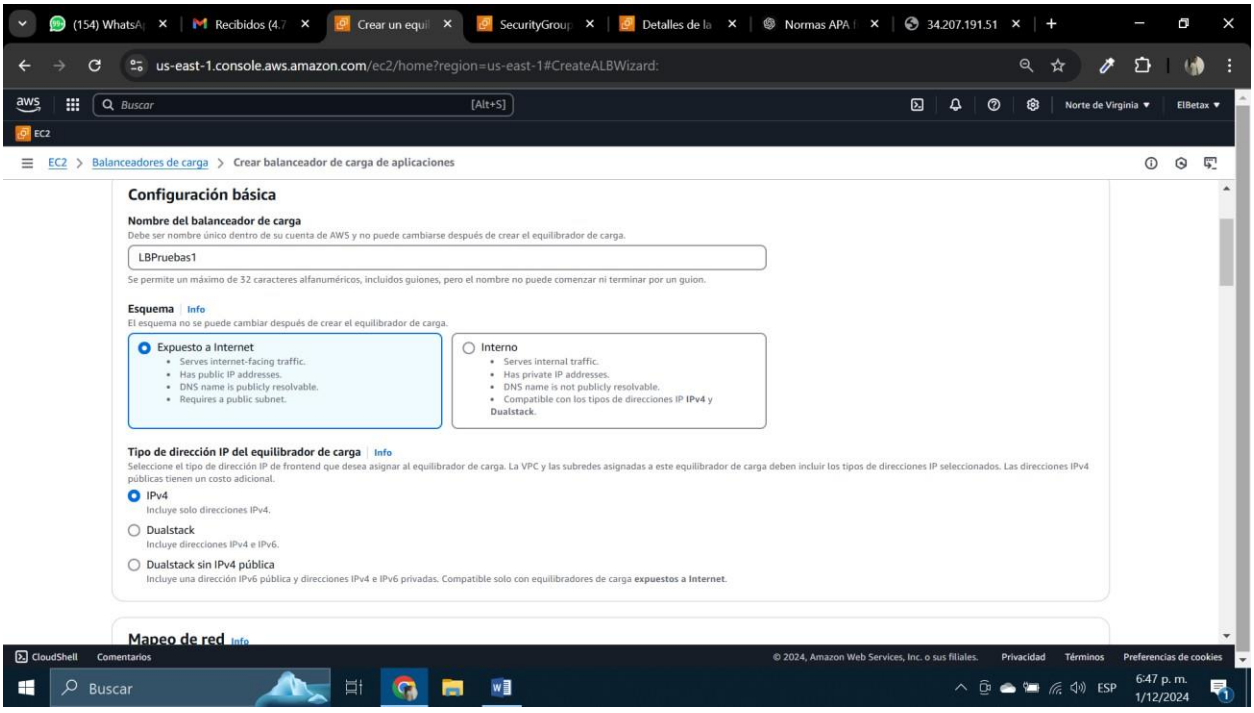


Figura 15 *Configuración de un balanceador de carga en AWS*

La configuración de la subred es un paso crucial en la creación de un balanceador de carga. Al seleccionar la subred adecuada, se garantiza que el balanceador pueda comunicarse con las instancias y que esté accesible desde Internet. Se debe crear en la misma VPC que la instancia, y además utilizar una subred pública.

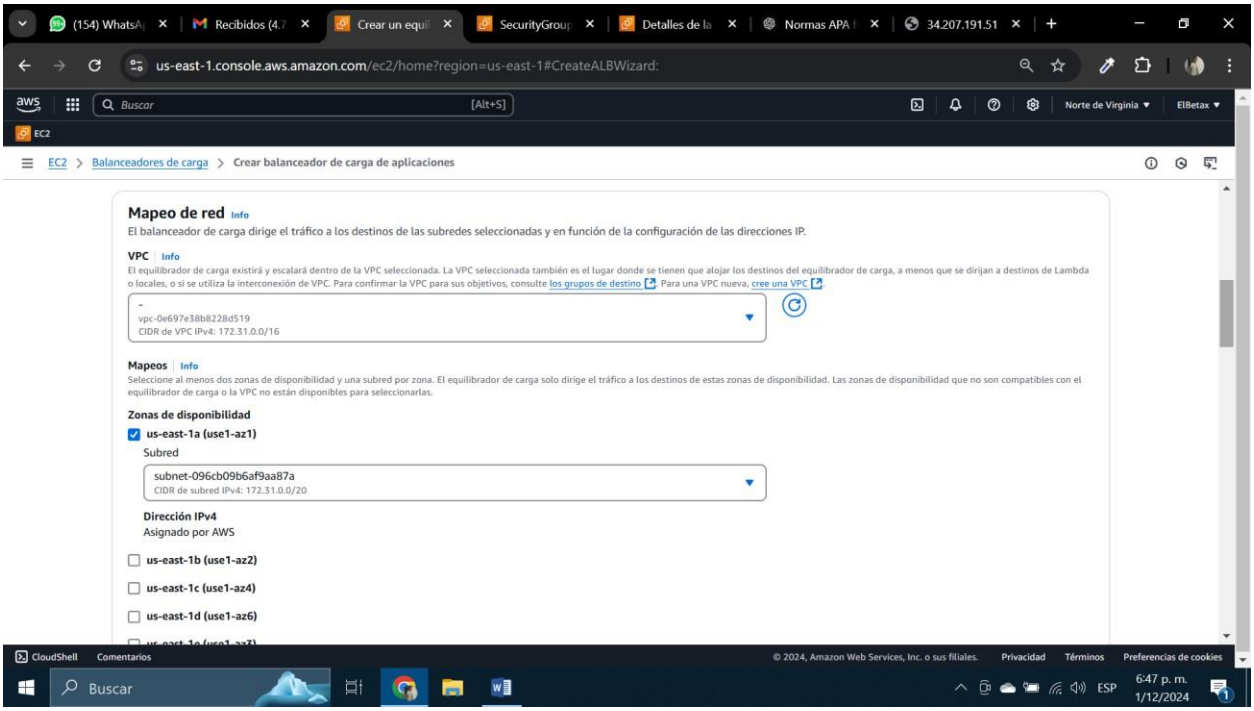


Figura 16 *Configuración de la subred para el balanceador de carga*

La configuración del grupo de seguridad es esencial para garantizar la seguridad del balanceador de carga. Al permitir solo el tráfico necesario y bloquear el tráfico no deseado, se reduce el riesgo de ataques y se protege la infraestructura.

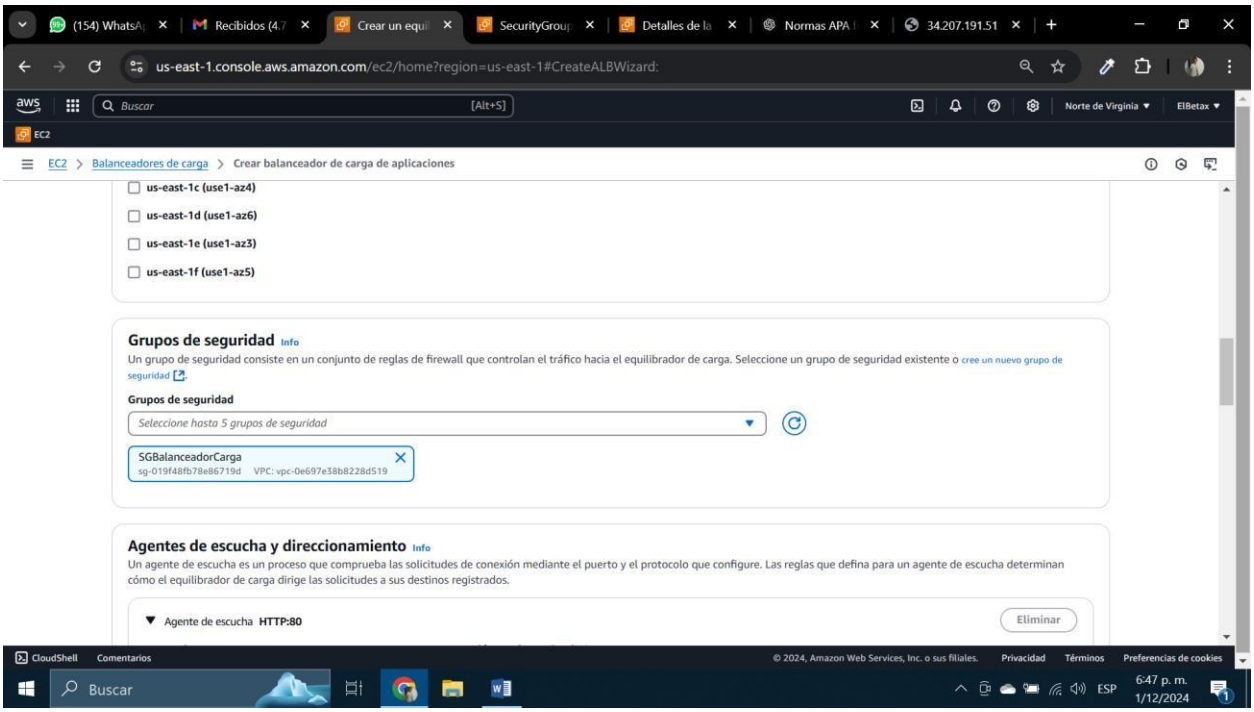


Figura 17 *Configuración del grupo de seguridad para el balanceador de carga*

El grupo de destino es un componente esencial de un balanceador de carga. Define los recursos a los que se dirigirá el tráfico y permite una gestión flexible y escalable de la aplicación.

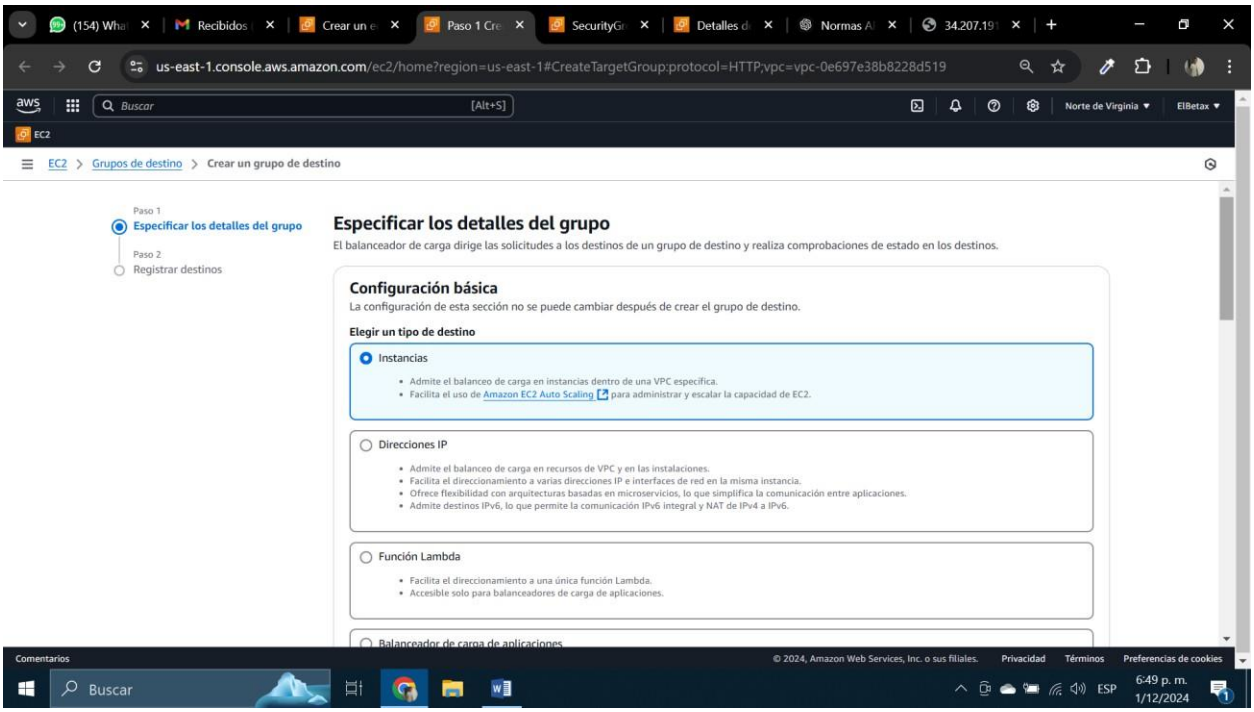


Figura 18 *Configuración del grupo de destino para el balanceador de carga*

La configuración del protocolo y las comprobaciones de estado es fundamental para garantizar que el balanceador de carga funcione correctamente y distribuya el tráfico de manera eficiente entre las instancias.

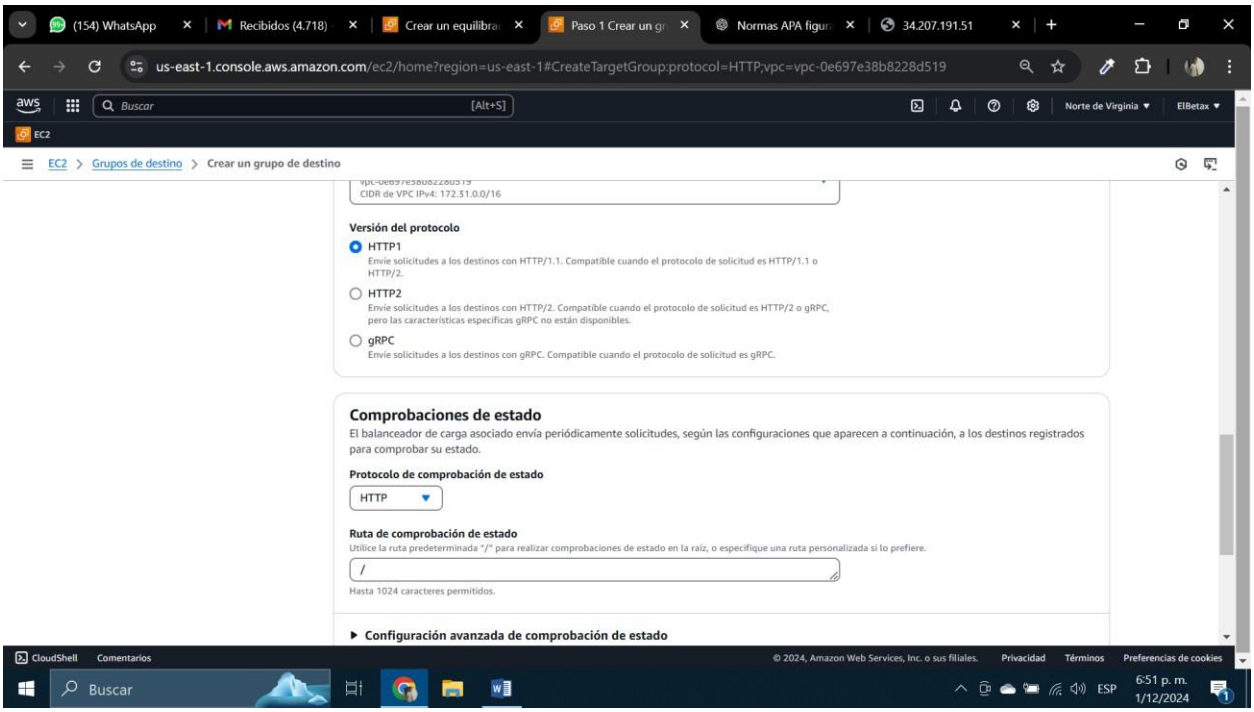


Figura 19 *Configuración del protocolo y comprobaciones de estado*

La captura de pantalla muestra que el grupo de destino se ha creado correctamente y que el balanceador de carga está listo para distribuir el tráfico entre las instancias especificadas.

Seleccionamos las instancias, las agregamos ambas, y el sistema nos confirma creación exitosa.

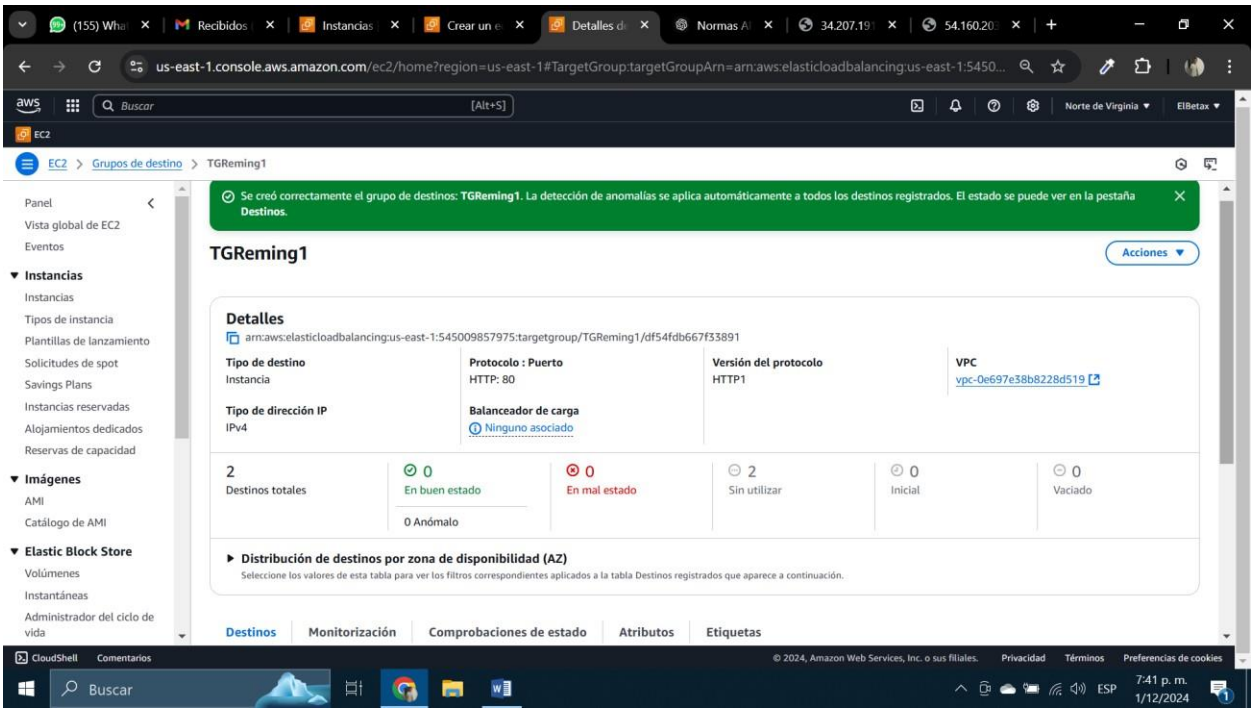


Figura 20 *Confirmación de la creación del grupo de destino*

Se asigna un escalado automático del 30%. Asignar un escalado automático del 30% significa que el sistema ajustará automáticamente el número de instancias en función de la utilización de la CPU, asegurando que siempre haya suficientes recursos disponibles para atender la demanda y optimizando los costos.

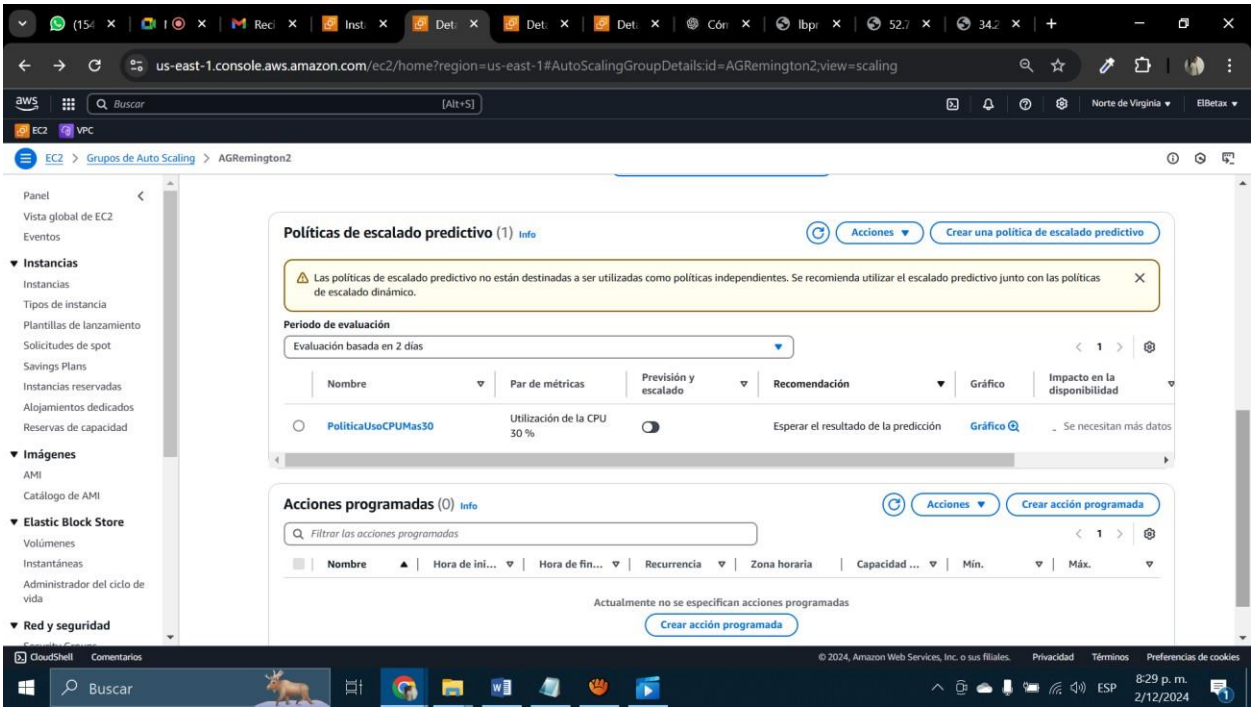


Figura 21 *Escalado automático en AWS*

En el apartado de Actividad del grupo de Auto Scaling, se puede ver el historial de creación de instancias. El historial de actividad de un grupo de Auto Scaling es una herramienta esencial para comprender el comportamiento de tu aplicación y garantizar que siempre tenga los recursos necesarios para satisfacer la demanda. Al analizar este historial, puedes identificar problemas, optimizar el rendimiento y tomar decisiones informadas sobre la configuración de tu grupo de Auto Scaling.

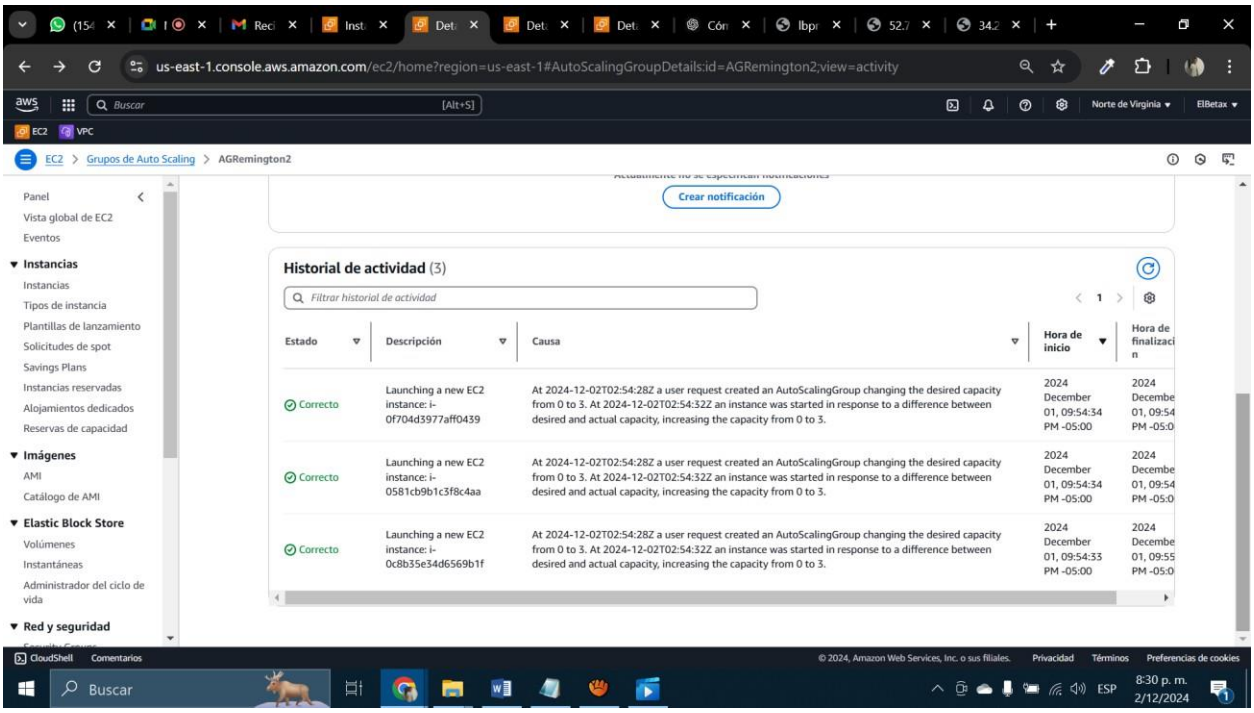


Figura 22 *Actividad del grupo de Auto Scaling*

Video explicativo Segunda entrega:

[Video]. YouTube. **https://youtu.be/ze5rz5I3XiA?si=HOuAyRaEn0FtHzQy**

Instalación de nginx

la captura de pantalla muestra el proceso de instalación de Nginx y sus dependencias en un servidor Linux. Una vez completada la instalación, el servidor estará listo para servir contenido web.

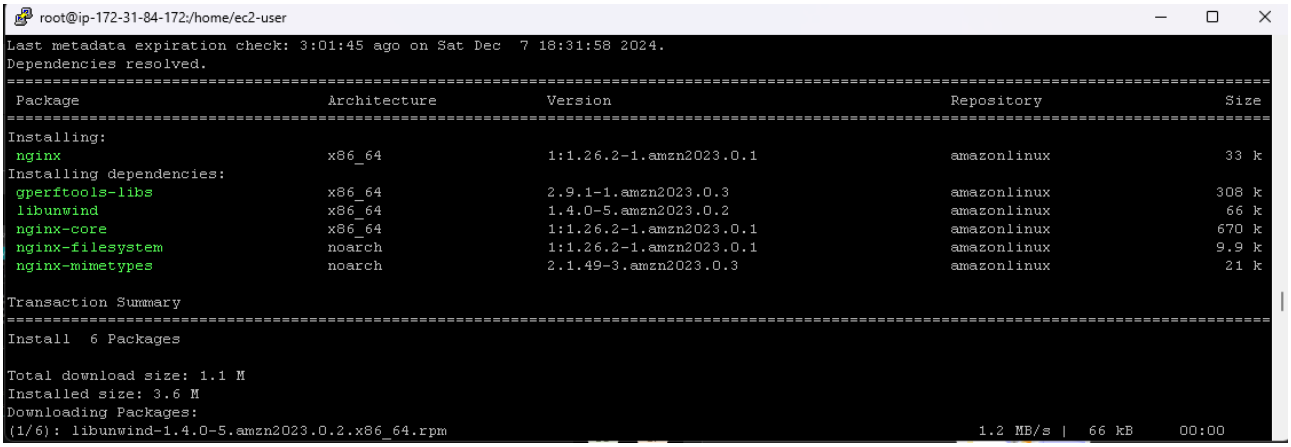


Figura 23 *Instalación de nginx*

La imagen captura el momento justo antes y durante la instalación de Docker en un servidor donde ya estaba funcionando un servicio HTTP. Esta acción amplía las capacidades del servidor al permitir ejecutar múltiples aplicaciones de forma aislada y eficiente dentro de contenedores Docker.

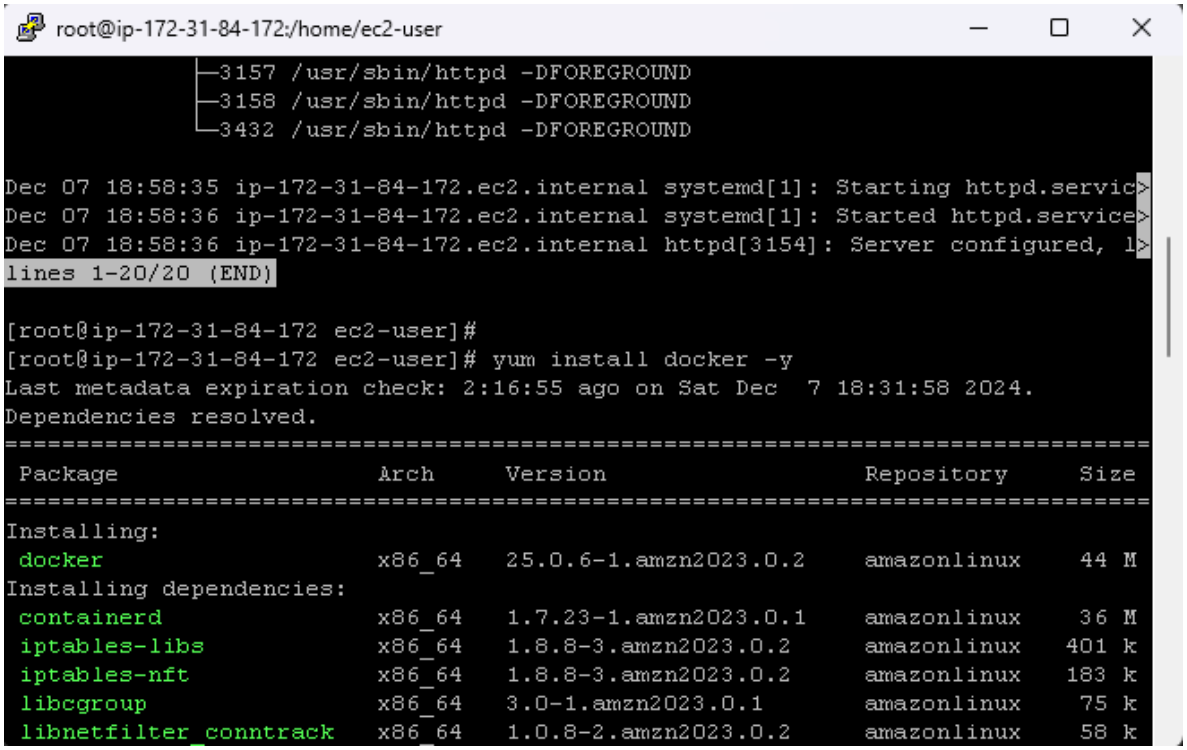


Figura 24 Instalación de Docker en la instancia

La imagen muestra el proceso de instalación de Docker y su configuración para que se inicie automáticamente al arrancar el sistema.

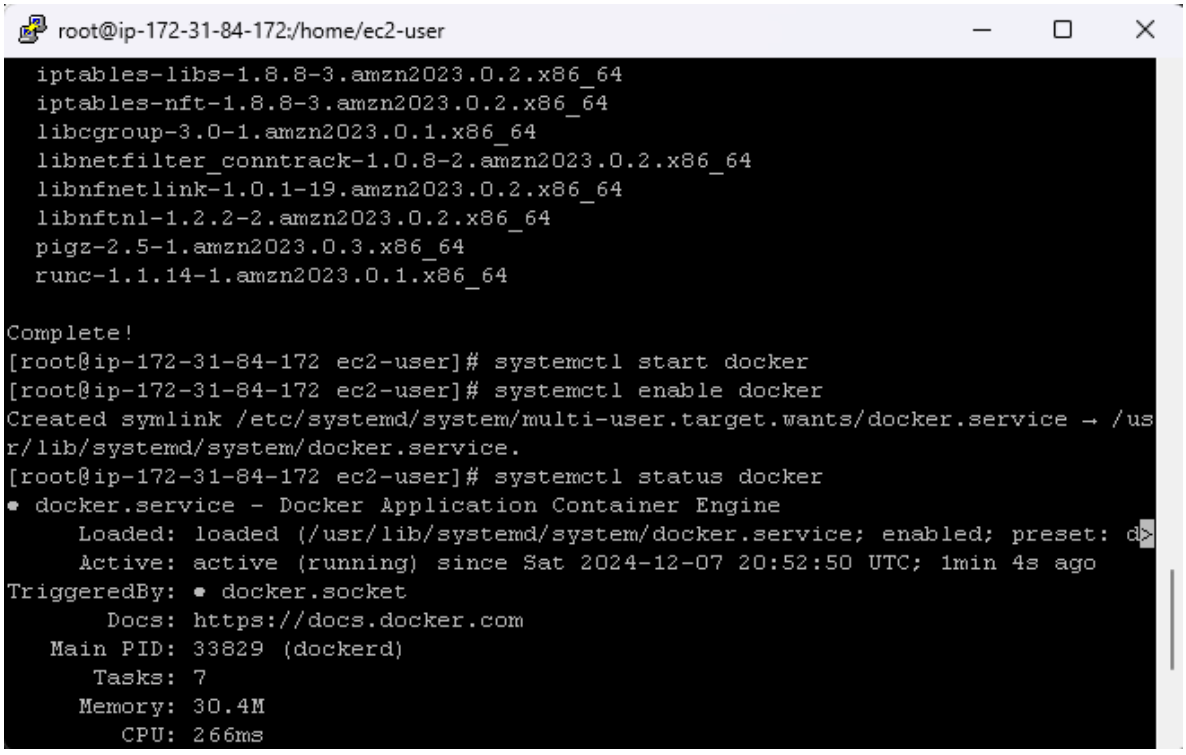


Figura 25 *Se inicia y se configura para que inicie el servicio al arrancar la instancia*

Se ha comprobado que Docker está funcionando correctamente en el sistema y se ha descargado una plantilla preconstruida (la imagen de httpd) que se puede utilizar para crear un servidor web.

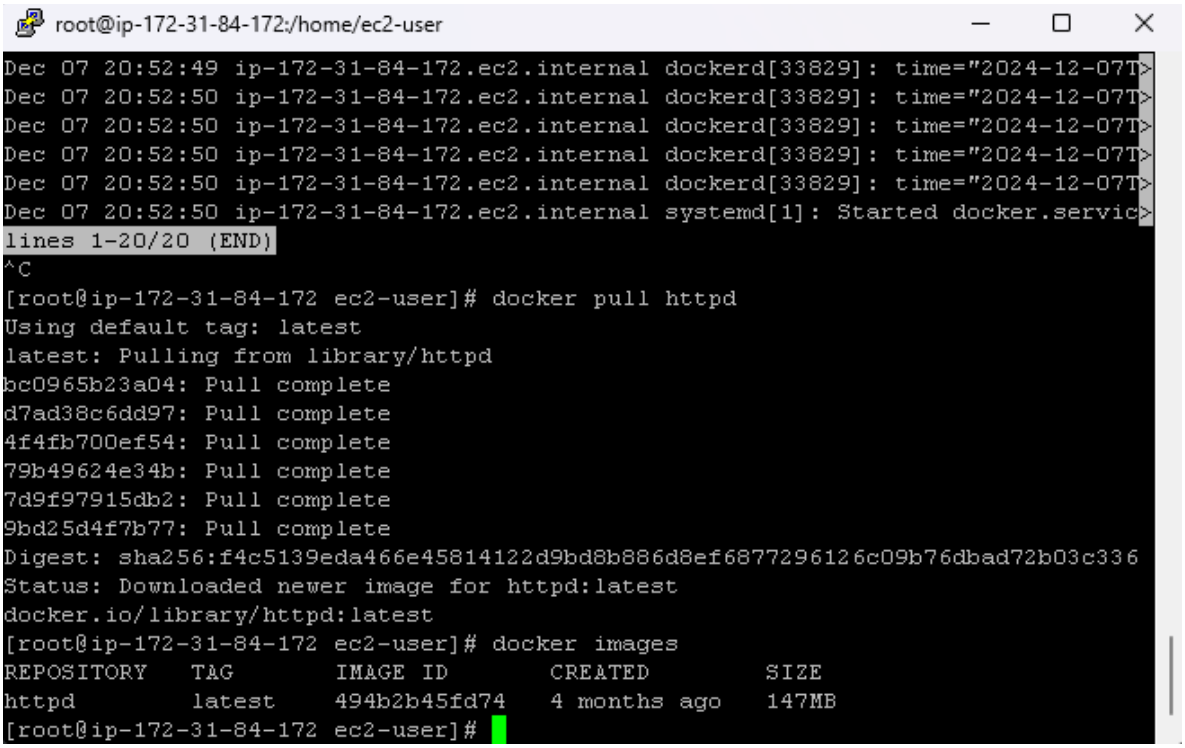


Figura 26 *Se descarga httpd*

La imagen muestra el proceso de creación de un contenedor Docker para ejecutar un servidor web Apache. Se configura el contenedor para que se ejecute en segundo plano, se le asigna un nombre, se mapean los puertos y se monta un volumen para compartir archivos entre el host y el contenedor. Este proceso se repite para crear contenedores adicionales con nombres "appl2" y "appl3", con configuraciones similares.

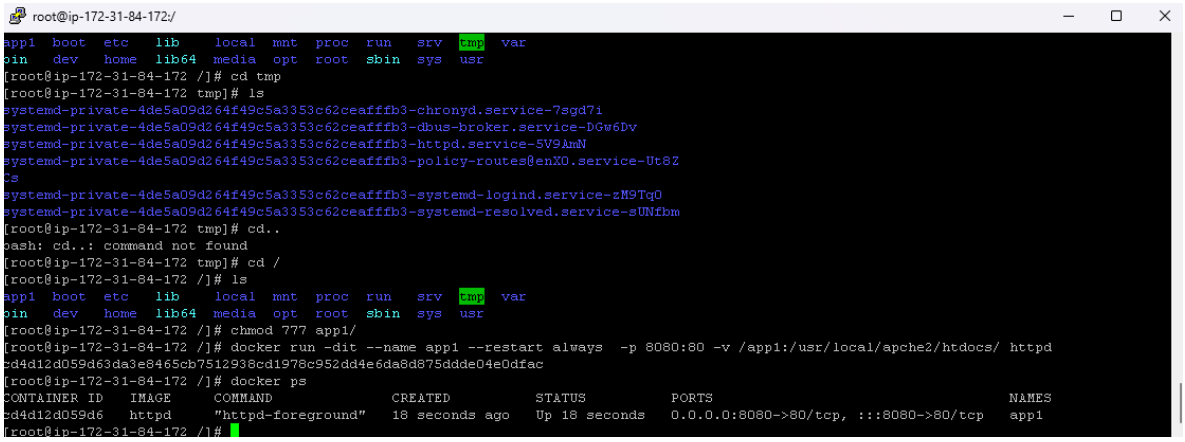


Figura 27 *Creación y configuración de la carpeta app1 y contenedor del httpd, esto mismo se realiza para app2 y app3.*

La imagen muestra la ejecución de un comando Docker para iniciar un contenedor en modo desacoplado (en segundo plano). El contenedor se basa en la imagen my-image y expone el puerto 80 del contenedor al puerto 80 del host. Esto significa que se está iniciando un servicio (un servidor web) que estará escuchando en el puerto 80 de la máquina host.

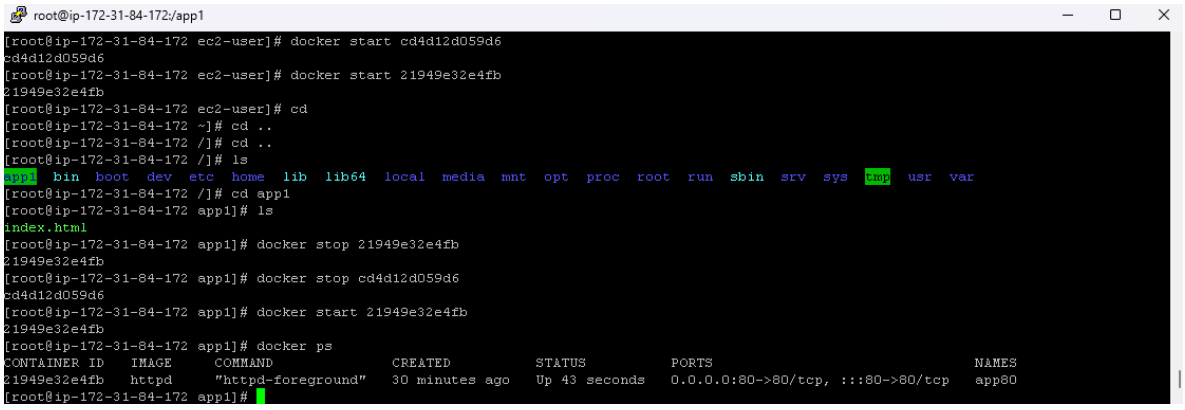


Figura 28 *Contenedor creado y funcional*

La imagen muestra que se han creado y están en ejecución tres contenedores Docker, cada uno de los cuales actúa como un servidor web independiente. Estos contenedores se han configurado para servir contenido estático desde un archivo index.html ubicado en una carpeta específica dentro del contenedor. La configuración de los puertos permite acceder a cada servidor web a través de una dirección IP y puerto diferentes.

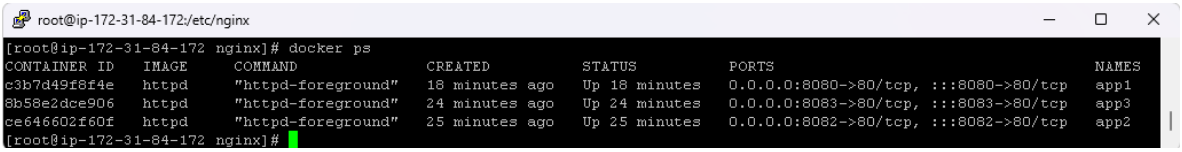


Figura 29 *Se crean los tres contenedores en el archivo index.html con su respectiva ruta*

La configuración de Nginx que se muestra en la imagen ha sido diseñada para proporcionar un equilibrio de carga y alta disponibilidad al distribuir el tráfico entre múltiples servidores. Los cambios realizados permiten una gestión más flexible y automatizada del entorno, adaptándose a las dinámicas de un entorno de contenedores.

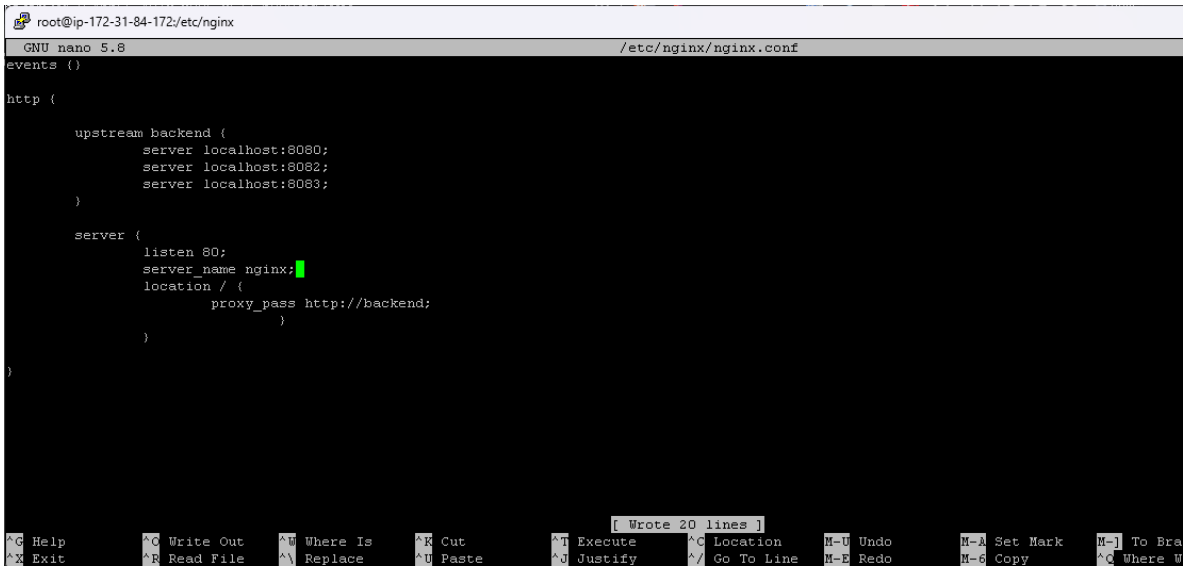


Figura 30 *Configuración de nginx.conf para que haga automáticamente los cambios entre los contenedores.*

La imagen muestra el resultado final de la configuración de Nginx: la capacidad de servir múltiples aplicaciones desde una única dirección IP. Esto se logra gracias al equilibrio de carga y la configuración de múltiples ubicaciones en el archivo de configuración de Nginx. Se realizan las 3 pruebas.

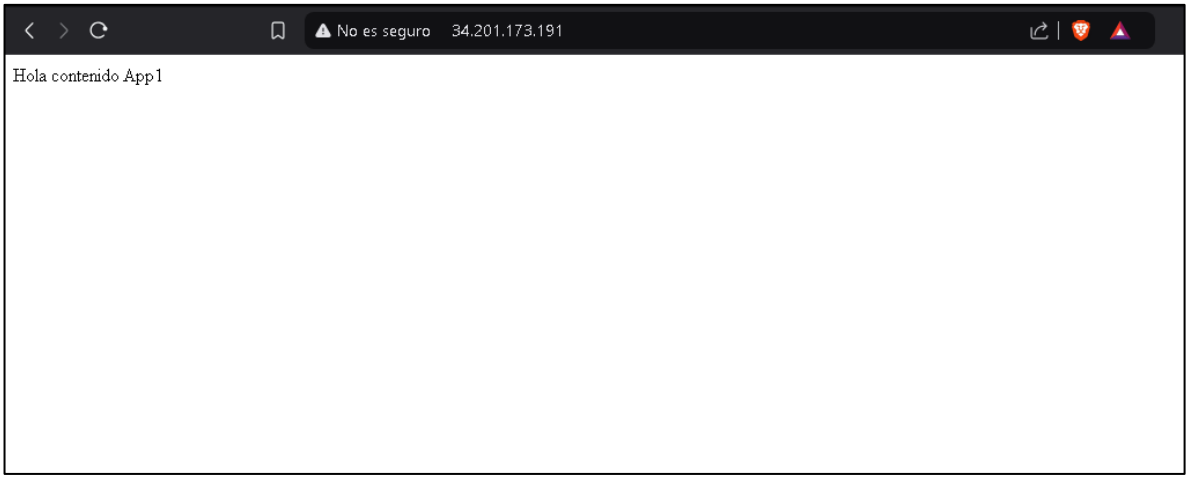


Figura 31 *Prueba App1*

Figura 32 *Prueba App2*

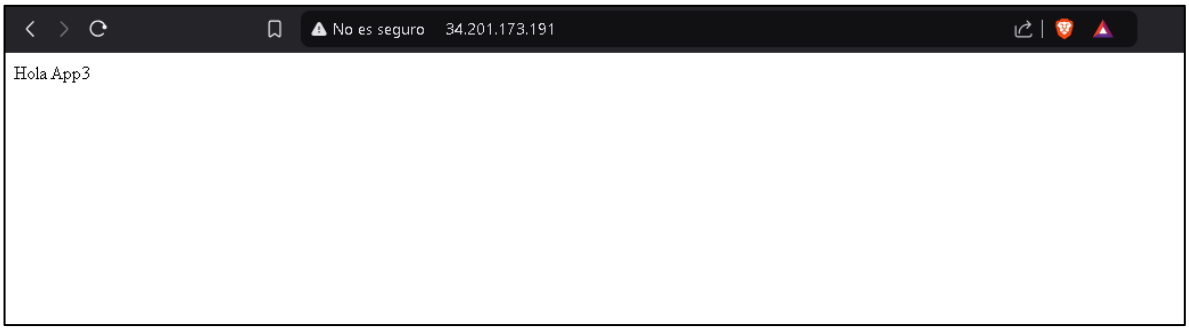


Figura 33 *Prueba App3*

El código HTML muestra una forma sencilla pero efectiva de personalizar el contenido servido por diferentes instancias de una aplicación, lo que facilita la gestión y el monitoreo de un entorno de microservicios. Ya con los contenedores funcionales re realiza modificación del archivo index.html para así realizar una simulación de una página de negocio; dentro de la página se dejó un texto que indica desde que instancia se está cargando la página y también desde que contenedor.

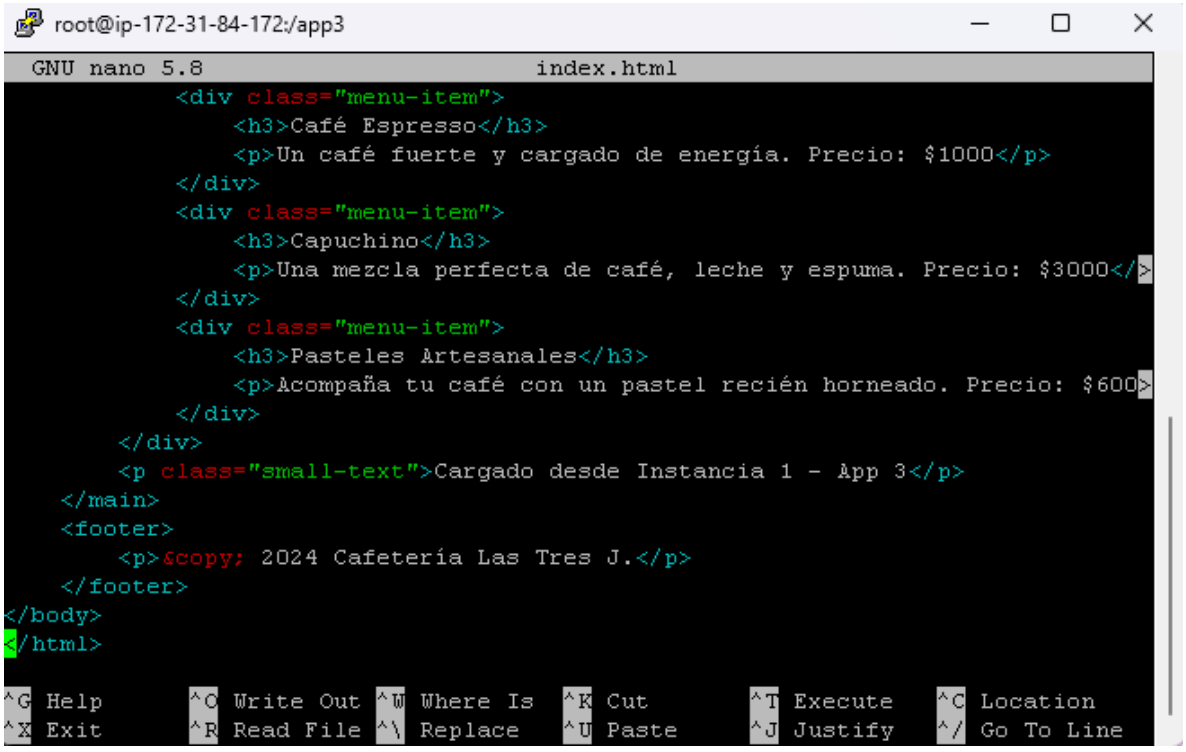


Figura 34 *Código HTML para visualizar la página web organizada*

La imagen ilustra de manera clara cómo la combinación de Nginx y contenedores permite crear aplicaciones web escalables, flexibles y personalizables.

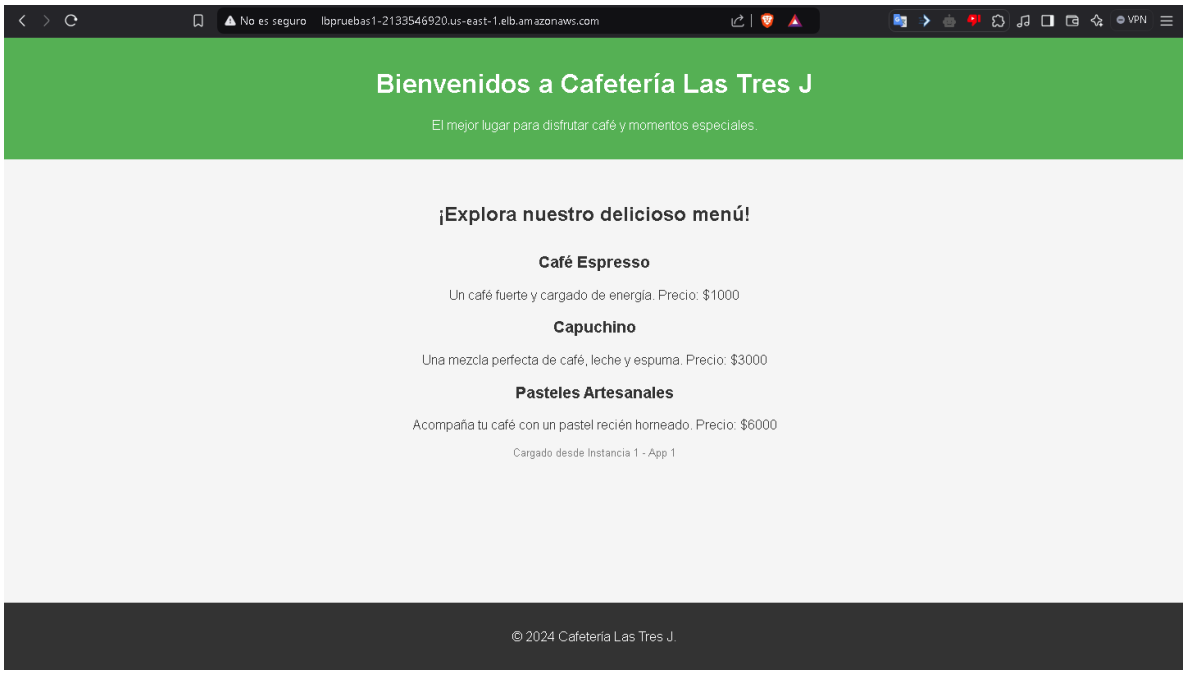


Figura 35: *Resultado de la página web cargando e indicando desde instancia 1- App 1.*

Nota: *Cabe aclarar que toda la configuración se realizó en las dos instancias.*

Video explicativo entrega final:

[Video]. YouTube. **https://youtu.be/gABNNlSsbco?si=AnJOHCFU-SVHasoD**

Conclusiones

El trabajo realizado sobre la implementación de servicios de Amazon Web Services (AWS) tiene como objetivo principal demostrar cómo las tecnologías de computación en la nube pueden optimizar la infraestructura tecnológica de una empresa, asegurando una operación continua y eficiente. Se concluye que AWS es una herramienta fundamental para implementar servicios de alta disponibilidad y redundancia, garantizando que, si una instancia falla, otras instancias pueden continuar operando sin interrupciones. De este modo, se asegura que la aplicación siga funcionando y los servicios se mantengan activos, evitando caídas que puedan afectar a los usuarios.

La combinación de instancias EC2, balanceadores de carga (ELB) y políticas de Auto Scaling permite distribuir el tráfico entre múltiples servidores, adaptándose automáticamente a los picos de demanda y reemplazando instancias dañadas de manera transparente. La implementación de contenedores Docker y el balanceo de carga con Nginx facilita además el despliegue y mantenimiento de múltiples aplicaciones independientes dentro de una misma infraestructura, mejorando la eficiencia y flexibilidad operativa.

Esta solución es especialmente útil para empresas que necesitan garantizar la continuidad del servicio y reducir el riesgo de fallos en sus sistemas. Además, elimina la dependencia de una sola infraestructura física, reduciendo costos operativos y permitiendo escalar recursos de manera dinámica según las necesidades del negocio.

En conclusión, el trabajo sirve como una guía práctica para implementar servicios en la nube que aseguran una infraestructura redundante, escalable y de alta disponibilidad, lo que permite a las empresas, independientemente de su tamaño, ofrecer servicios digitales robustos y confiables sin interrupciones.

Referencias

OpenAI. (2024). *ChatGPT: Un modelo de lenguaje de inteligencia artificial*. https://openai.com/chatgpt

Google. (2024). *Gemini: Una plataforma para modelos de IA y generativas*. https://gemini.google.com/app?hl=es

Amazon Web Services, Inc. (2024). *Servicios de computación en la nube y soluciones*. https://aws.amazon.com