

**TRABAJO DE GRADO**  
**Opción Seminario-Diplomado.**

**Título del trabajo**  
Amazon Web Services

Corporación Universitaria Remington.

Nombre de la facultad:

Ing. De sistemas

Nombre del programa académico:

Amazon Web Services

(AWS)

estudiante

Francisco Eduardo Zaac Andrade

Nombre del Tutor del trabajo de grado

Juan Pablo Berrio López

Opción de Trabajo de grado Seminario

2024

## Tabla de Contenidos

Tabla de Figuras.....	3
Resumen.....	6
Palabras clave.....	7
Marco conceptual y contextual .....	7
Desarrollo e implementación del aprendizaje.....	10
Ejercicio 2 .....	10
Ejercicio 3 .....	44
Ejercicio 4 .....	59
Conclusiones .....	100
Referencias.....	101

## Tabla de Figuras

Ilustración 1. Crear VPC.....	10
Ilustración 2. Información VPC.....	11
Ilustración 3. Configuración VPC.....	11
Ilustración 4. Configuración VPC.....	12
Ilustración 5. Crear Instancia .....	13
Ilustración 6. Información Instancia .....	14
Ilustración 7. Servidor Web Amazon Linux .....	15
Ilustración 8. Requerimientos Físicos De La EC2.....	15
Ilustración 9. Configuración De Red De Instancia .....	16
Ilustración 10. Lanzar Instancia.....	17
Ilustración 11. EC2 Linux2.....	18
Ilustración 12. Resumen Instancia .....	19
Ilustración 13. MobaXterm.....	20
Ilustración 14. MobaXterm SSH .....	21
Ilustración 15. Conexión SSH.....	22
Ilustración 16. Conexión Exitosa SSH.....	23
Ilustración 17. Comandos De Consola Linux .....	24
Ilustración 18. Comandos De Consola Linux2 .....	24
Ilustración 19. Comandos De Consola Linux3 .....	25
Ilustración 20. Comandos De Consola Linux4 .....	26
Ilustración 21. Creamos Ruta.....	27
Ilustración 22. Contendor.....	28
Ilustración 23. Configuración Grupo Seguridad.....	29
Ilustración 24. Reglas De Entrada .....	30
Ilustración 25. Agregar Regla.....	31
Ilustración 26. TCP Personalizado.....	32
Ilustración 27. Vista Nuevas Reglas .....	32
Ilustración 28. Resultado 8080 .....	33
Ilustración 29. Resultado 8081 .....	34
Ilustración 30. Nginx Proxy .....	35
Ilustración 31. Nginx Proxy Resultado.....	35
Ilustración 32. Filezilla .....	36
Ilustración 33. Filezilla SSH.....	37
Ilustración 34. Resultado Conexión.....	37
Ilustración 35. Nginx.Conf .....	38
Ilustración 36. Código Por Editar .....	39
Ilustración 37. Código Editado .....	40
Ilustración 38. Reinicio Nginx .....	41
Ilustración 39. Hosts De Windows .....	41
Ilustración 40. Reglas De Entradas Nuevas.....	42

	4
Ilustración 41. Página Pang1.Com.....	43
Ilustración 42. Página Pang2.Com.....	43
Ilustración 43. Ejercicio 3 Crear Bucket.....	44
Ilustración 44. Lista De Control Bucket .....	45
Ilustración 45. Lista De Control Bucket2 .....	45
Ilustración 46. Tipo De Encriptación.....	46
Ilustración 47. Crear Bucket .....	47
Ilustración 48. Cargamos El Arhivo Al Bucket .....	48
Ilustración 49. Vista Archivo.....	48
Ilustración 50. Creamos DBR.....	49
Ilustración 51. Metodo De Creación.....	50
Ilustración 52. Tipo De BDR .....	51
Ilustración 53. Especificaciones Tecnicas De DBR .....	52
Ilustración 54. Contraseña De BDR.....	53
Ilustración 55. Crear DB .....	54
Ilustración 56. Base De Datos Creada .....	54
Ilustración 57. Resumen DB .....	55
Ilustración 58. Paquete De Comandos Mysql.....	56
Ilustración 59. Comando Database DB.....	57
Ilustración 60. Vista De La Base De Datos .....	58
Ilustración 61. Ejercicio 4 Servidor Wed Windows .....	59
Ilustración 62. Tipo De Ami Windows.....	60
Ilustración 63. Hardware De Nuestra Instancia Windows.....	61
Ilustración 64. Configuración De Red De Inst Windows .....	62
Ilustración 65. Lanzar Instancia Windows.....	63
Ilustración 66. Pestaña Almacenamiento .....	64
Ilustración 67 . Crear Instantánea .....	65
Ilustración 68. Crear Instantánea .....	66
Ilustración 69. Creando Instantánea.....	66
Ilustración 70. Crear Imagen.....	67
Ilustración 71. Configuración De Imagen.....	67
Ilustración 72. Crear Imagen.....	68
Ilustración 73. Crear Balanceador De Carga .....	69
Ilustración 74. Tipos De Equilibradores De Carga.....	70
Ilustración 75. Crear.....	70
Ilustración 76. Configuraciones De Balanceador De Carga .....	71
Ilustración 77. Grupo De Seguridad Balanceador De Carga .....	72
Ilustración 78. Crear Grupo De Destino .....	73
Ilustración 79. Configuracion Basica.....	74
Ilustración 80. Nombre Del Grupo De Destino .....	75
Ilustración 81. Selección De La Vpc Para El Grupo De Trabajo .....	76
Ilustración 82. Selección De Las Instancias .....	76
Ilustración 83. Creamos Balanceador De Carga .....	77
Ilustración 84. Grupo De Auto Escalado .....	78

Ilustración 85. Configuración De Lanzamiento.....	79
Ilustración 86. Nombre Del Lanzamiento.....	79
Ilustración 87. Seleccionamos Nuestra Ami.....	80
Ilustración 88. Claves De Inicio De Sección .....	81
Ilustración 89. Subred .....	82
Ilustración 90. Configuraciones De Red Y Seguridad.....	83
Ilustración 91. Vista De Lanzamiento .....	84
Ilustración 92. Selección De Vpc E Ip Públicas .....	84
Ilustración 93. Asignar Balacedor De Cargar .....	85
Ilustración 94. Selección Del Balanceador De Carga.....	86
Ilustración 95. Limites De Implementación De Las Instancias .....	86
Ilustración 96. Política De Escalado.....	87
Ilustración 97. Matrices De Escalado De Uso De CPU.....	88
Ilustración 98. Escalado Final.....	89
Ilustración 99. Crear Grupo De Escalamiento .....	89
Ilustración 100. Escalado Automatico .....	90
Ilustración 101. Política Uso De CPU .....	91
Ilustración 102. Detalles De La Política De Uso De CPU.....	92
Ilustración 103. Crear Política .....	93
Ilustración 104. Primer Resultado.....	93
Ilustración 105. Nuestro Dns .....	94
Ilustración 106. Conexión A Escritorio Remoto.....	95
Ilustración 107. Siverbench .....	96
Ilustración 108. Segundo Resultado .....	97
Ilustración 109. Inicializando Las Instancias Creadas.....	97
Ilustración 110. Conectadas Las Instancias Creadas .....	98
Ilustración 111. Resultado Final .....	99

## **Resumen**

El objetivo central del seminario es conocer a fondo la colección de los servicios de computación en la nube de Amazon Web Services (AWS) lo que común mente en el área de informática se conoce como computación en la nube que es la disponibilidad a pedido de los recursos de procesamiento como los servicios por internet para que las empresas tengan un mejor rendimiento y administren mejor sus recursos ya que con esto pueden montar sus aplicaciones y solo pagan por lo que están utilizando lo cual también permite conectar y o acceder no solo de forma local desde cualquier dispositivo si no también desde el internet. Ya que los procesos de información y almacenamiento tienen lugar en servidores o centro de datos donde todos los que tengan acceso a sus aplicaciones la pueden configurar de forma local o vía internet.

A principios del siglo XXI cuando la palabra internet comenzó a expandirse por el mundo con más auge y con la llegada de los celulares inteligentes, tables, tv inteligentes etc. El uso de una red de servidores conectados a internet para almacenar, administrar y procesar datos en lugar de depender de un servicio físico instalado de forma local ya sea personal o empresarial, se conecta al acceder a una estructura donde tanto el software como el hardware son integral mente virtualizados.

**Palabras clave**

AWS

Internet

Computación

Nube

Servidores

**Marco conceptual y contextual**

La virtualización es una herramienta tecnológica que se utiliza para generar representaciones virtuales de servidores, almacenamiento, redes y otras máquinas virtuales. El software virtual es una imitación de las funciones del hardware físico para ejecutar varios equipos virtuales a la vez en una única máquina física, las empresas de hoy en día recurren a la digitalización para utilizar sus recursos de hardware para tener un mayor rendimiento satisfacer los objetivos estipulados y multiplicar la inversión plasmada. Tiene una mayor afinidad con los servicios de computación en la nube que ayudan a la organización y o administración de la infraestructura de manera más eficaz.

Se origina en los inicios de las grandes PC mainframe en la década de los 60, cuando cada una de estas super estructuras masivas de hardware solo podían ejecutar o funcionar con un proceso a la vez. Con el paso del tiempo los usuarios o clientes comenzaron a exigir que estas grandes e importantes infraestructuras pudieran admitir a más de un usuario o proceso a la vez. Esto llevo que a finales de la década de los 60 la empresa IBM desarrollara un sistema operativo CP-67 que se encargaba de ejecutar

varias máquinas virtuales en una única máquina física (HIPERVISOR) que introdujo la memoria virtual al grupo de servidores System 360 de la empresa IBM, también hubo varios avances de desarrollo para que múltiples usuarios trabajen en un solo servidor.

En esa misma década fue cuando un estadounidense informático Licklider tuvo una grandiosa idea de interconectar varios ordenadores entre sí. Así tras unos años más delante de desarrollo e investigación termino diseñando un sistema llamado ARPANET (red de agencia de proyectos de investigación avanzada) Licklider también conto con ayuda para el desarrollo que fueron Bob Taylor y Larry Roberts los cuales presentaron el proyecto en el año 1969 ARPANET.

Es conocida como la pionera del Internet por tanto su participación en la computación en la nube es fundamental. Ya que fue la primera red que permitió compartir información digital de un lugar a otro sin que estuviese en un mismo lugar. Licklider imagino todo un mundo conectado entre sí y gracias a esta visión y o idea fue que surgió el desarrollo de computador en la nube. Desde los años 1970 y 1990 surgieron varios avances tecnológicos en los que se refiere al internet y de la computación en la nube un caso más destacado fue de la empresa IBM que desarrollo su primera máquina virtual en 1972.

Ya en la década de 90 treinta años después cuando muchas empresas tenían una visión de mantenerse al día con las nuevas tecnologías TIC, donde varias empresas dedicadas a las telecomunicaciones las cuales desarrollaron sus propias versiones de redes privadas virtualizadas (VPN) y es esta década cuando se desarrolla la fecha clave de la historia de la computación en la nube. Se dieron cuenta de la necesidad y o virtudes

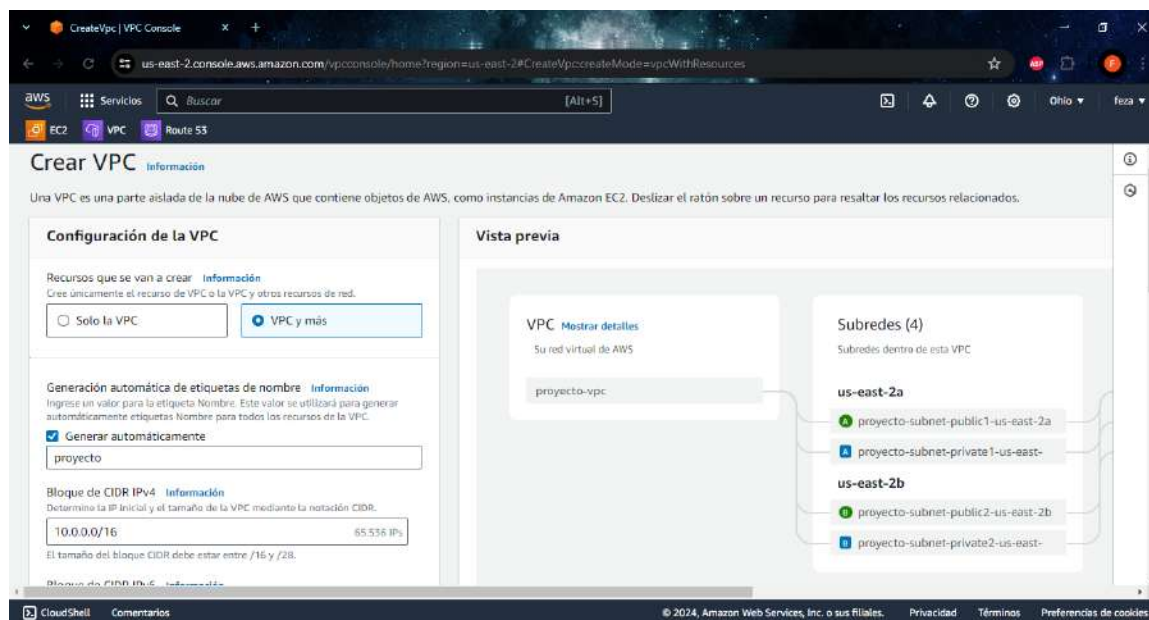
de usar mejor sus recursos de servidores que antes con mucha frecuencia no se utilizaban lo suficiente, no solo dividieron su infraestructura de servidores de manera más eficaz sino que ejecutaron sus aplicaciones obsoletas en diferentes tipos y versiones de Sistemas Operativos (O.S) gracias a una gran red compuesta por muchos tipos diferentes de pc utilizando diferentes Sistemas Operativos (O.S) ya que en esa época el internet creció de manera significativa lo cual ayudo a impulsar la adopción de la virtualización. Por este motivo se empezó a utilizar la virtualización con mayor frecuencia, redujo la dependencia de un proveedor para servicios de servidores y fue la base para el desarrollo de la computación en la nube.

La computación en la nube tubo un alza y crecimiento en el año 1996 principalmente para las empresas como para los sectores educativos como las instituciones y las universidades más que todo los que empleaban sistemas de almacenamiento remoto y que poseyeran una conexión entre sí. En los primeros años del siglo XXI se produjo un gran desarrollo en los sistemas relacionados con las telecomunicaciones gracias a esto se logró que sea mucho más eficaz la computación en la nube y que avanzara a lo que es hoy en día.

## Desarrollo e implementación del aprendizaje

### Ejercicio 2

#### Como crear una VPC



*Ilustración 1. crear VPC*

#### Información de la VPC que vamos a crear

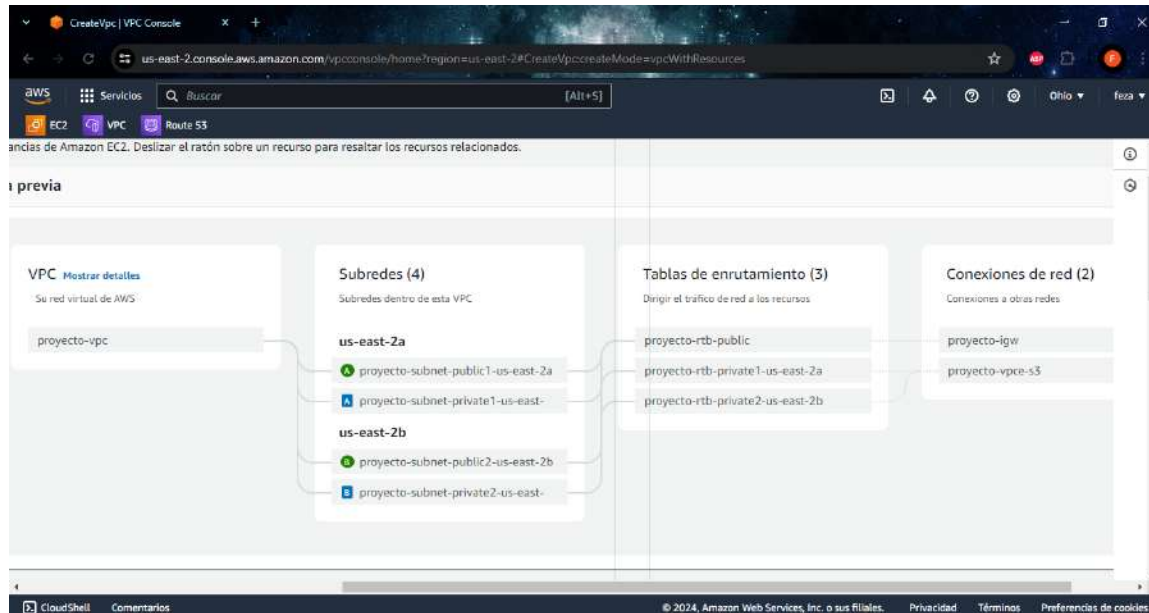


Ilustración 2. información VPC

## Configuración de la VCP

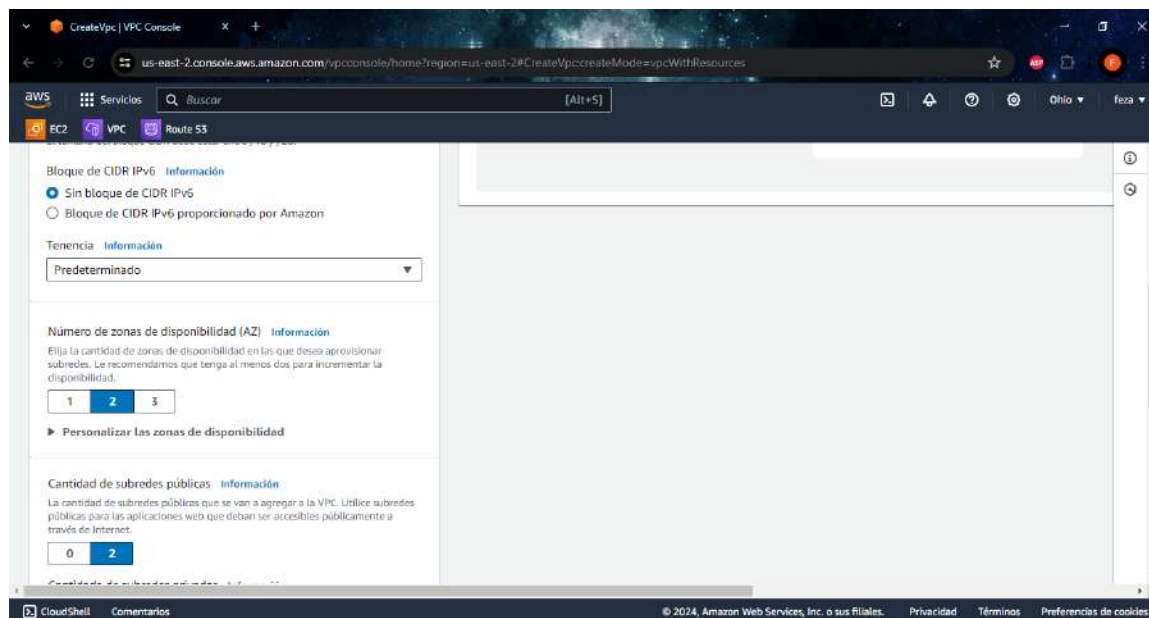
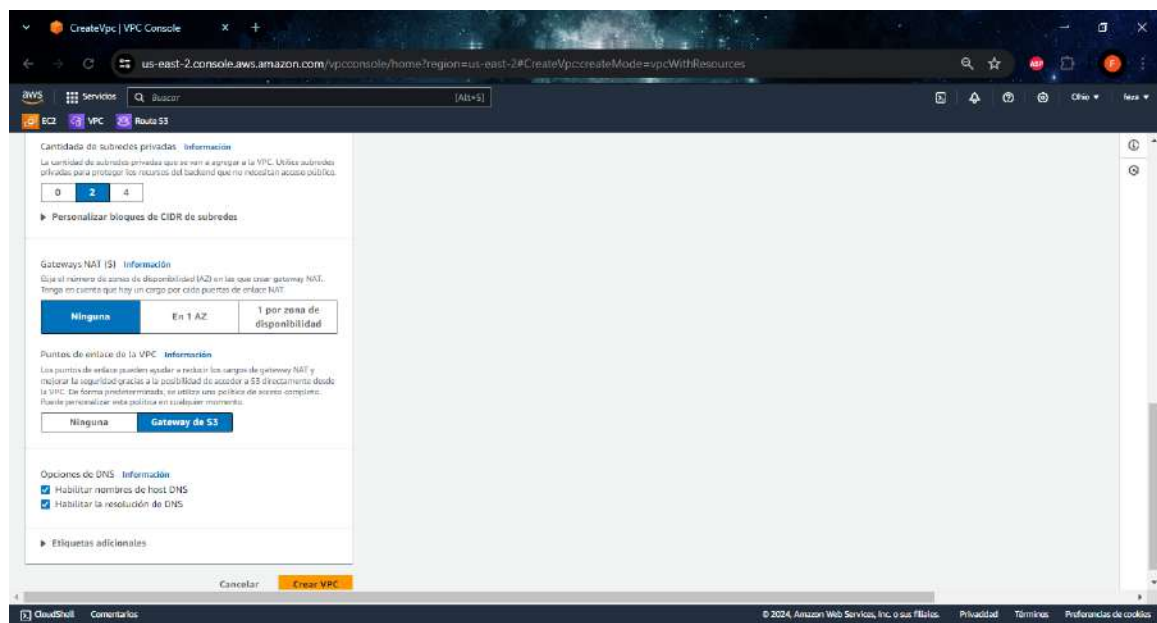
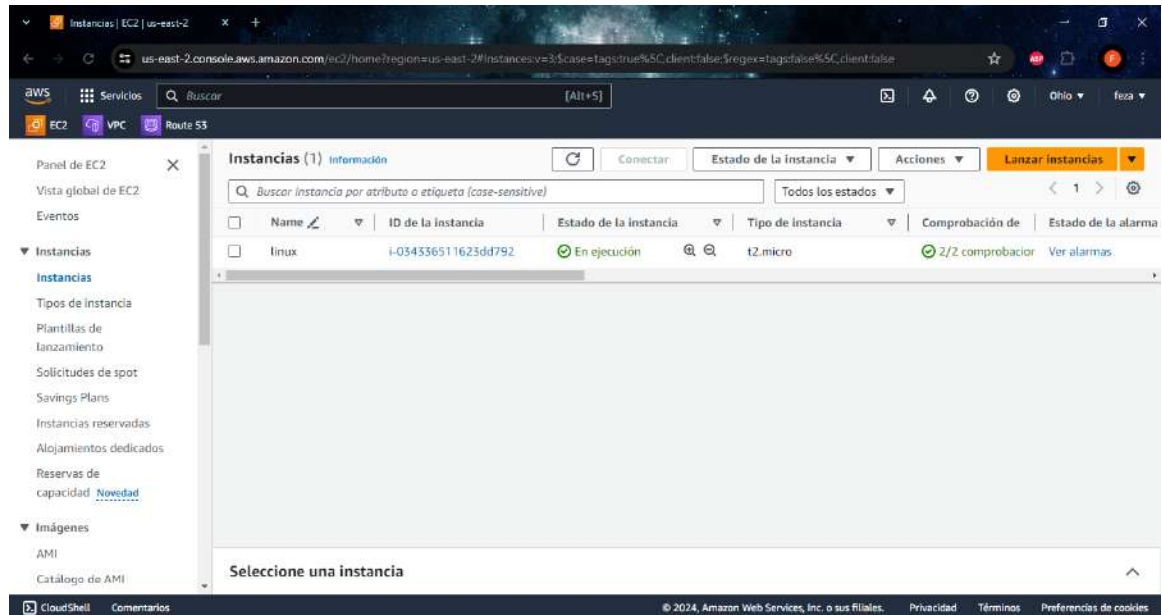


Ilustración 3. configuración VPC



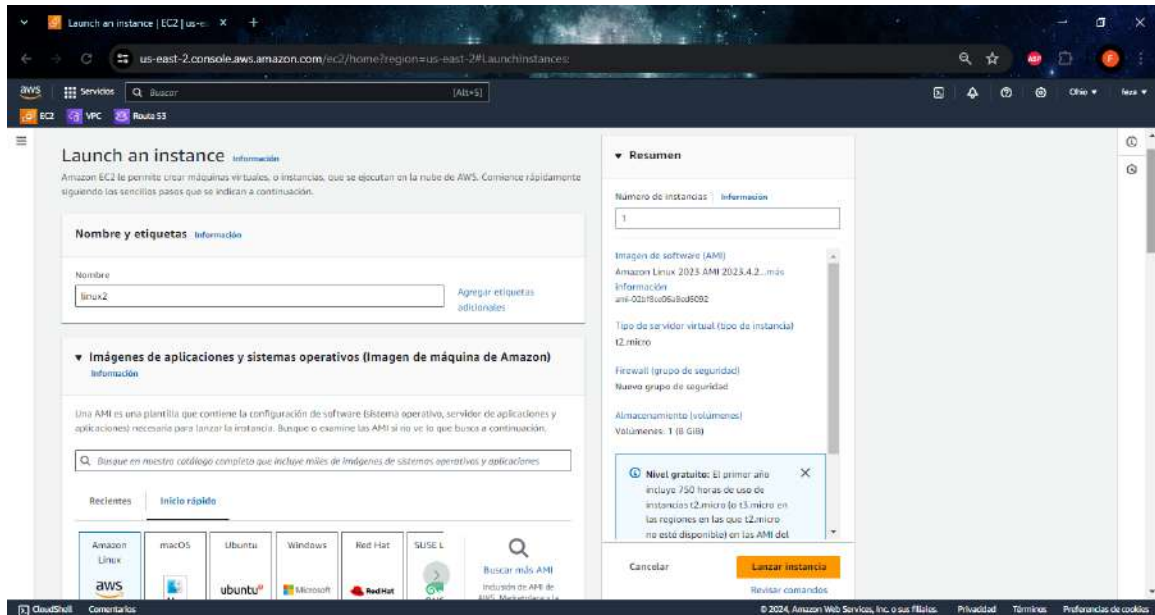
*Ilustración 4. configuración VPC*

Una vez creada la VPC crearemos una EC2 en lanzar instancia



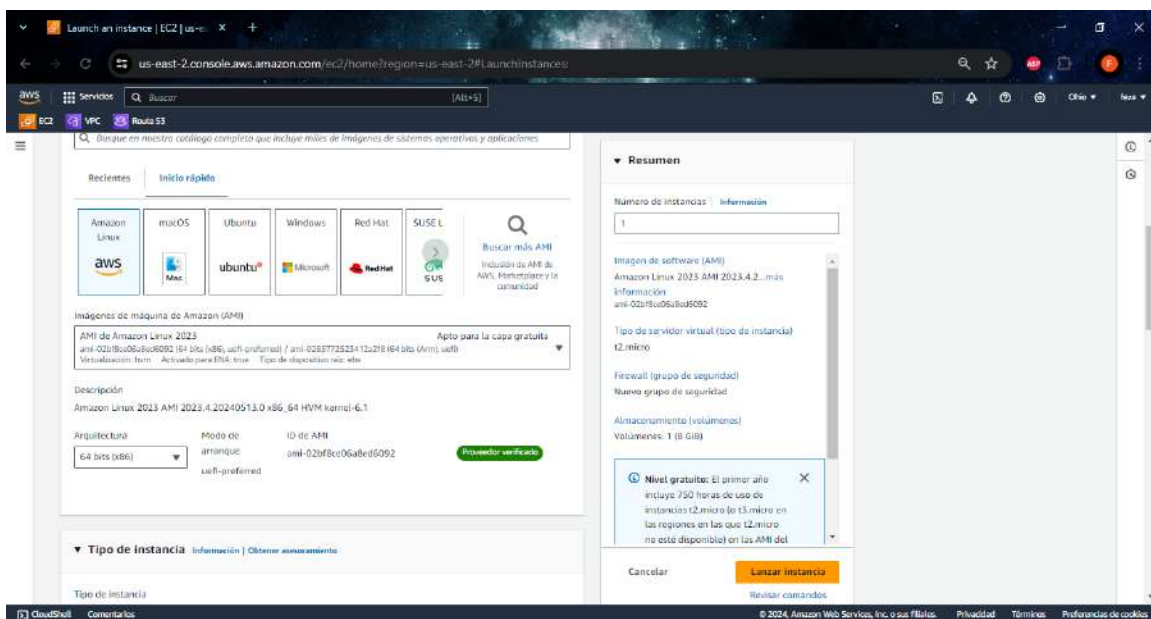
*Ilustración 5. crear Instancia*

Creamos una instancia con el nombre Linux 2 el nombre puede ser cualquiera que se desee si no que escogemos Linux para identificar que el sistema operativo que vamos a utilizar para nuestro servidor web es linux



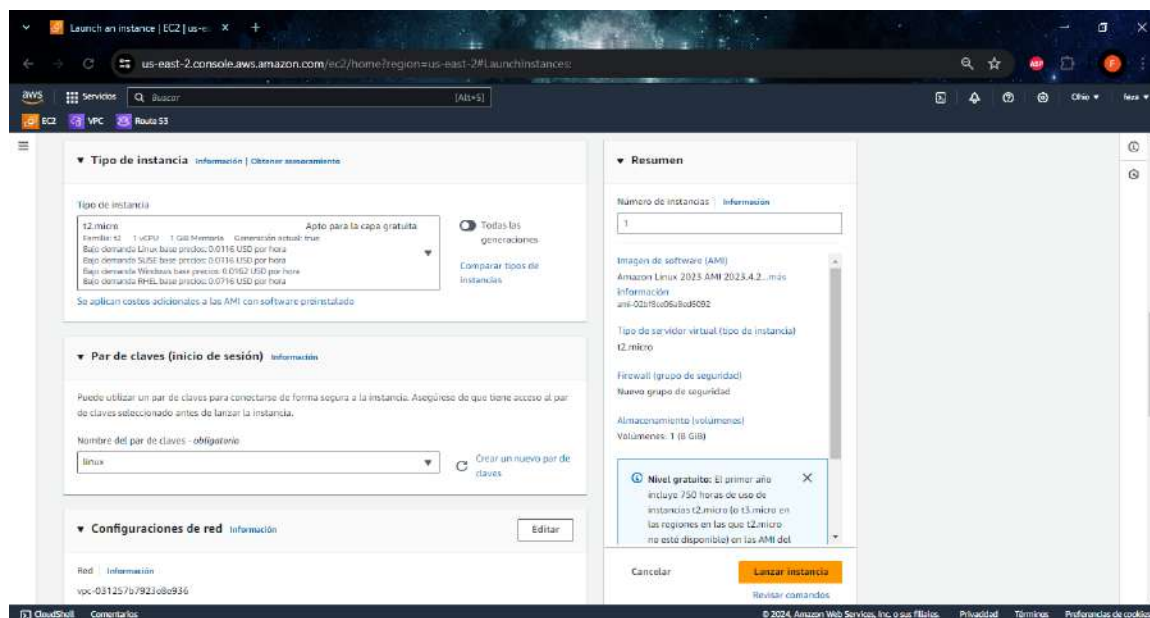
*Ilustración 6. información Instancia*

Seleccionamos el sistema que vamos a utilizar en este caso es Amazon Linux y también el tipo de imagen



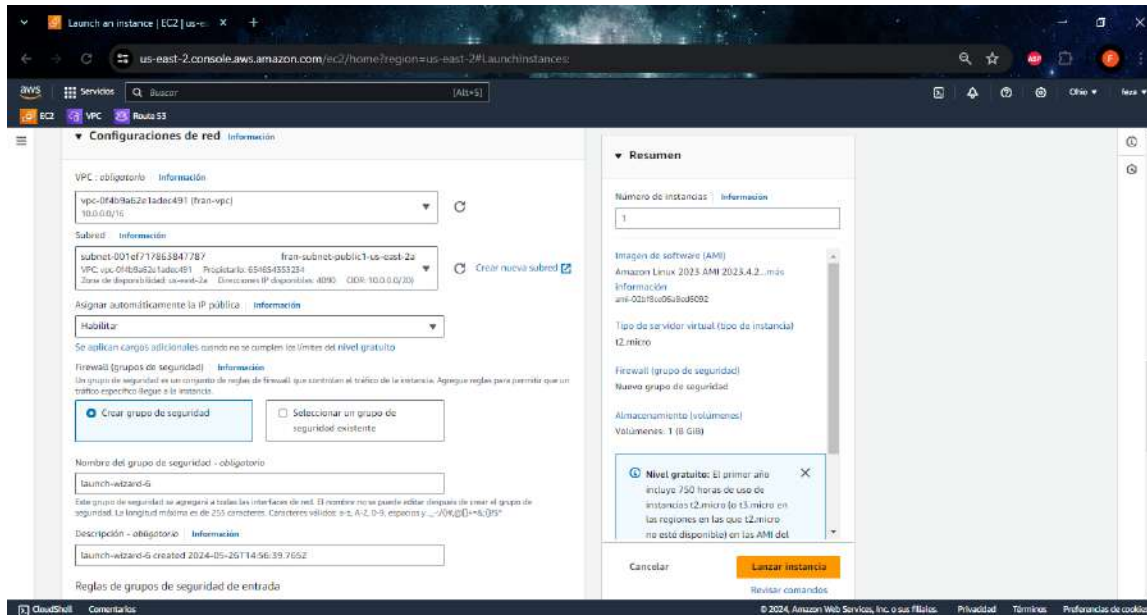
### Ilustración 7. servidor web amazon linux

El tipo de instancia que son los requisitos físicos de nuestra EC2 y la llave para poder iniciar sesión en la EC2



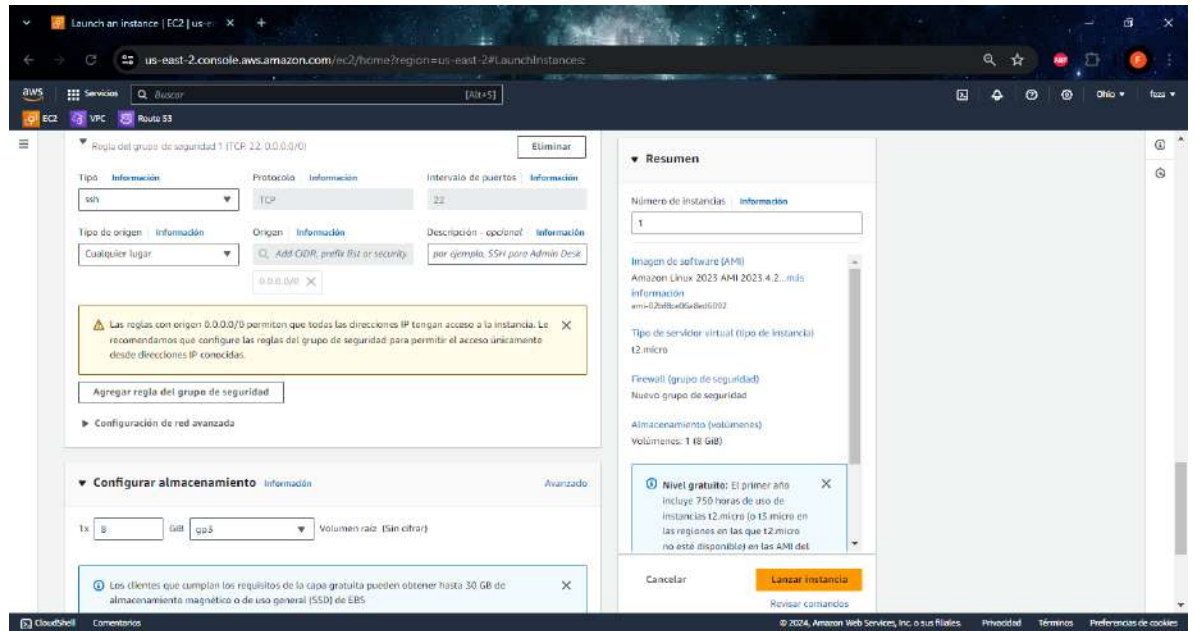
### Ilustración 8. requerimientos físicos de la EC2

Las configuraciones de red donde seleccionas nuestra VPC ya antes creada, nuestra subred habilitamos la IP pública para la conexión desde afuera y donde asignamos nombres de grupo de seguridad



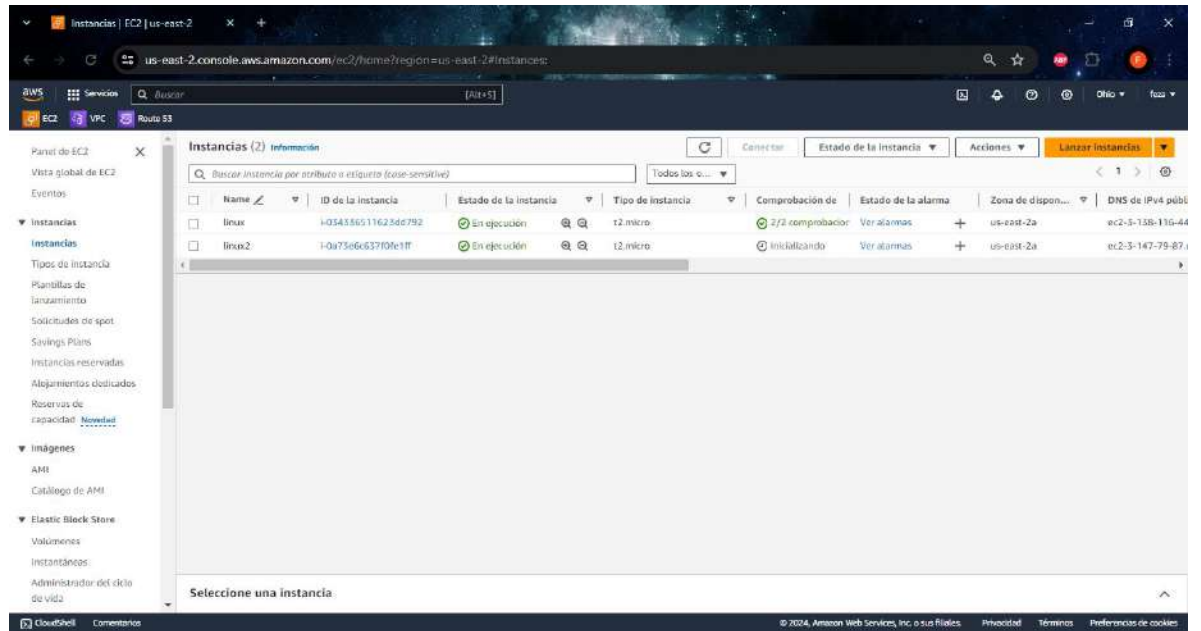
*Ilustración 9. configuración de red de instancia*

Las reglas para poder conectarnos y configuración de almacenamiento y le damos en lanzar instancia para crearla



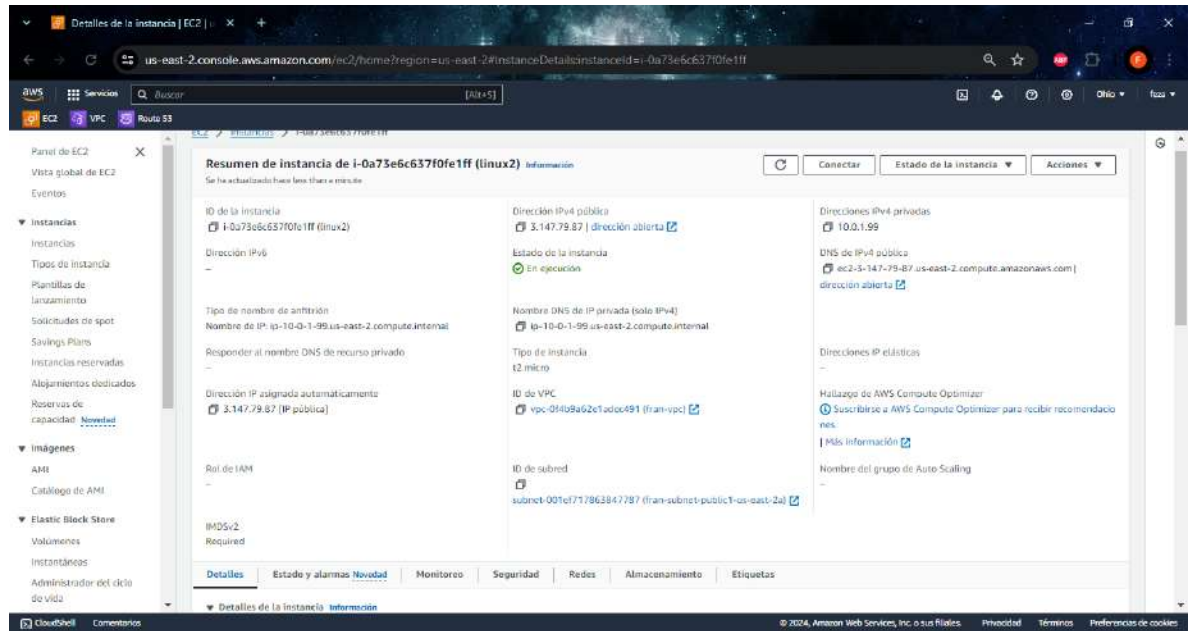
*Ilustración 10. lanzar Instancia*

Hay nos aparece la de Linux2 que fue la que creamos anterior mente y le damos en el id de la instancia de Linux2



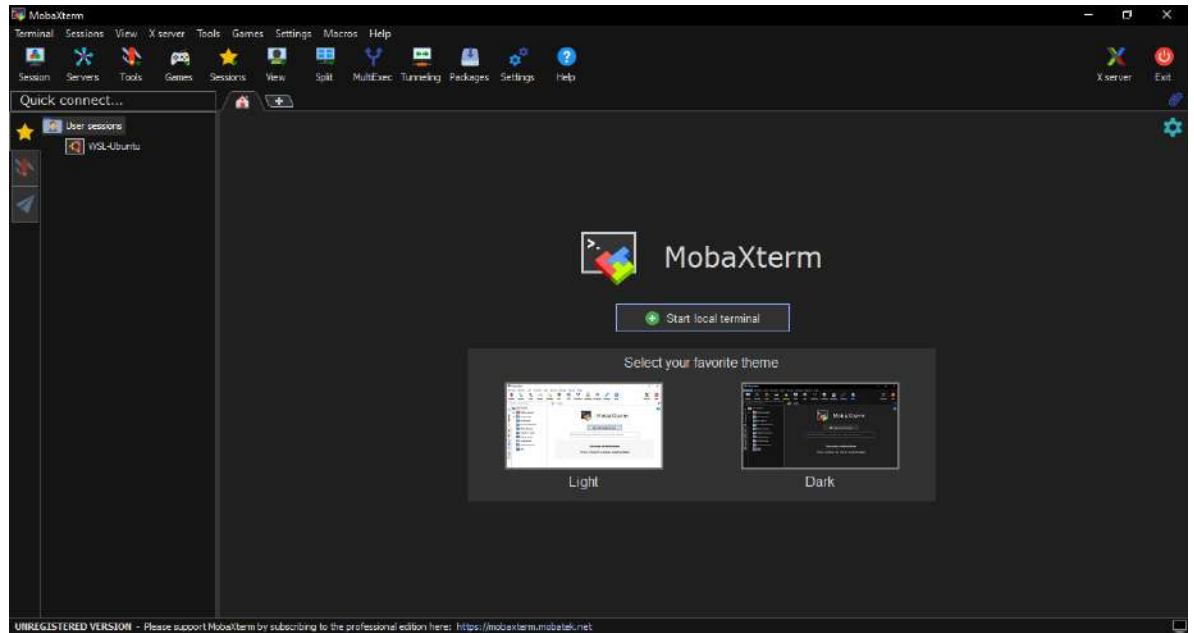
*Ilustración 11. EC2 linux2*

Donde nos muestra sus datos más relevantes y le damos en el apartado de conectar para acceder a la ec2 para crear nuestros contenedores



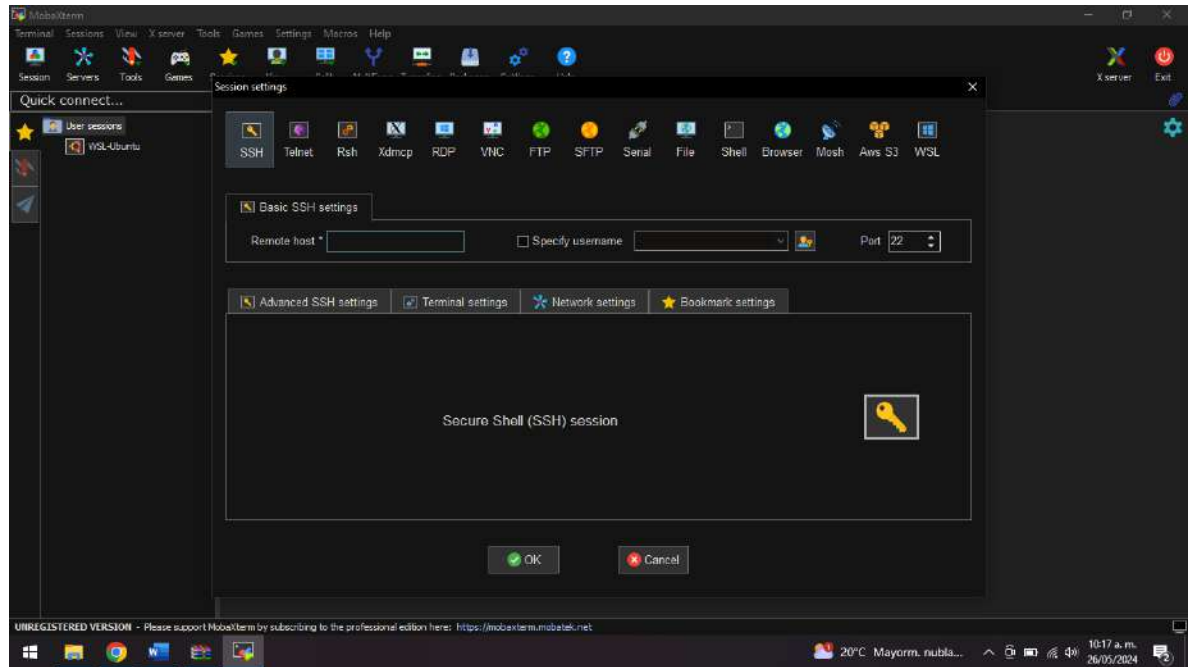
*Ilustración 12. resumen Instancia*

Para la conexión a la ec2 por medio de ssh utilizaremos el programa mobaxterm  
nos dirigimos al icono de sección



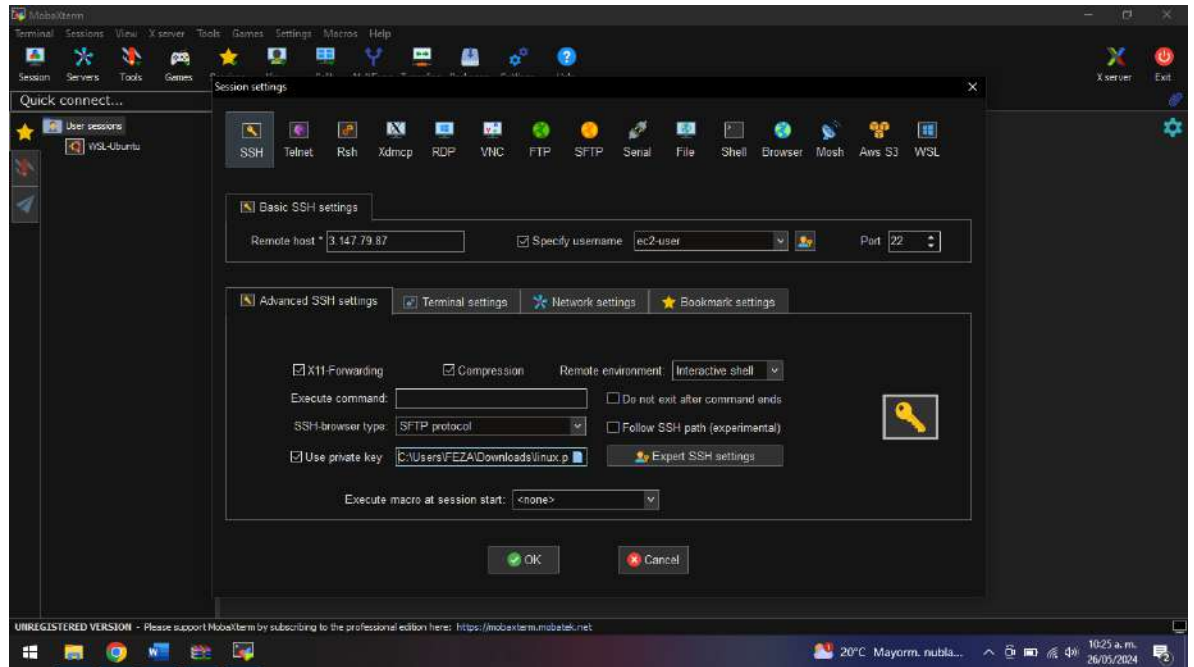
*Ilustración 13. Mobaxterm*

Y en el apartado de ssh es donde nos vamos a enfocar para hacer la respectiva conexión ingresamos el host que es nuestra Ip y en username nuestro usuario de ec2 y para ingresar nuestra clave “llave” le damos en la pestaña de ssh settings y donde dice use private key es donde importamos nuestra llave ya anterior mente creada en AWS



*Ilustración 14. mobaxterm SSH*

Así es como de quedar los campos ya llenos con los valores correspondientes a nuestro ec2 y le damos conectar



*Ilustración 15. conexión SSH*

Aquí ya aparecemos conectados a la ec2 por medio de consola de comandos donde comenzaremos a crear nuestros contenedores



```

[ec2-user@ip-10-0-1-99 ~]$
[ec2-user@ip-10-0-1-99 ~]$
[ec2-user@ip-10-0-1-99 ~]$ sudo su
[root@ip-10-0-1-99 ec2-user]# yum install -y docker
Last metadata expiration check: 0:24:44 ago on Sun May 26 15:08:43 2024.
Dependencies resolved.

Package                Architecture    Verstion        Repository        Size
Installing:
docker                 x86_64         25.0.3-1.amzn2023.0.1  amazon linux     44 M
Installing dependencies:
containerd             x86_64         1.7.11-1.amzn2023.0.1  amazon linux     35 M
iptables-libs         x86_64         1.8.8-3.amzn2023.0.2  amazon linux     401 k
iptables-nft          x86_64         1.8.8-3.amzn2023.0.2  amazon linux     183 k
libcgroup              x86_64         3.0-1.amzn2023.0.1    amazon linux     75 k
libnetfilter_conntrack x86_64         1.0.8-2.amzn2023.0.2  amazon linux     58 k
libnftnl               x86_64         1.0.1-19.amzn2023.0.2 amazon linux     30 k
libnftnl               x86_64         1.2.2-2.amzn2023.0.2  amazon linux     84 k
pigz                   x86_64         2.5-1.amzn2023.0.3    amazon linux     83 k
runc                   x86_64         1.1.11-1.amzn2023.0.1  amazon linux     3.0 M

Transaction Summary
Install 10 Packages

Total download size: 83 M
Installed size: 313 M
Downloading Packages:
(1/10): iptables-libs-1.8.8-3.amzn2023.0.2.x86_64.rpm 5.8 MB/s | 401 kB 00:00
(2/10): iptables-nft-1.8.8-3.amzn2023.0.2.x86_64.rpm 7.0 MB/s | 183 kB 00:00
(3/10): libcgroup-3.0-1.amzn2023.0.1.x86_64.rpm      3.7 MB/s | 75 kB 00:00
(4/10): libnetfilter_conntrack-1.0.8-2.amzn2023.0.2.x86_64.rpm 2.8 MB/s | 58 kB 00:00
(5/10): libnftnl-1.0.1-19.amzn2023.0.2.x86_64.rpm   1.1 MB/s | 30 kB 00:00
(6/10): libnftnl-1.2.2-2.amzn2023.0.2.x86_64.rpm    2.3 MB/s | 84 kB 00:00
(7/10): pigz-2.5-1.amzn2023.0.3.x86_64.rpm          3.8 MB/s | 83 kB 00:00
(8/10): runc-1.1.11-1.amzn2023.0.1.x86_64.rpm       17 MB/s | 3.0 MB 00:00
(9/10): containerd-1.7.11-1.amzn2023.0.1.x86_64.rpm 59 MB/s | 35 MB 00:00

```

Ilustración 17. comandos de consola linux

```

Installing : runc-1.1.11-1.amzn2023.0.1.x86_64 1/10
Installing : containerd-1.7.11-1.amzn2023.0.1.x86_64 2/10
Running scriptlet: containerd-1.7.11-1.amzn2023.0.1.x86_64 2/10
Installing : pigz-2.5-1.amzn2023.0.3.x86_64 3/10
Installing : libnftnl-1.2.2-2.amzn2023.0.2.x86_64 4/10
Installing : libnftnl-1.0.1-19.amzn2023.0.2.x86_64 5/10
Installing : libnetfilter_conntrack-1.0.8-2.amzn2023.0.2.x86_64 6/10
Installing : iptables-libs-1.8.8-3.amzn2023.0.2.x86_64 7/10
Installing : iptables-nft-1.8.8-3.amzn2023.0.2.x86_64 8/10
Running scriptlet: iptables-nft-1.8.8-3.amzn2023.0.2.x86_64 8/10
Installing : libcgroup-3.0-1.amzn2023.0.1.x86_64 9/10
Running scriptlet: docker-25.0.3-1.amzn2023.0.1.x86_64 9/10
Installing : docker-25.0.3-1.amzn2023.0.1.x86_64 10/10
Running scriptlet: docker-25.0.3-1.amzn2023.0.1.x86_64 10/10
Created symlink /etc/systemd/system/sockets.target.wants/docker.socket + /usr/lib/systemd/system/docker.socket.

Verifying : containerd-1.7.11-1.amzn2023.0.1.x86_64 1/10
Verifying : docker-25.0.3-1.amzn2023.0.1.x86_64 2/10
Verifying : iptables-libs-1.8.8-3.amzn2023.0.2.x86_64 3/10
Verifying : iptables-nft-1.8.8-3.amzn2023.0.2.x86_64 4/10
Verifying : libcgroup-3.0-1.amzn2023.0.1.x86_64 5/10
Verifying : libnetfilter_conntrack-1.0.8-2.amzn2023.0.2.x86_64 6/10
Verifying : libnftnl-1.0.1-19.amzn2023.0.2.x86_64 7/10
Verifying : libnftnl-1.2.2-2.amzn2023.0.2.x86_64 8/10
Verifying : pigz-2.5-1.amzn2023.0.3.x86_64 9/10
Verifying : runc-1.1.11-1.amzn2023.0.1.x86_64 10/10

Installed:
containerd-1.7.11-1.amzn2023.0.1.x86_64  docker-25.0.3-1.amzn2023.0.1.x86_64  iptables-libs-1.8.8-3.amzn2023.0.2.x86_64
iptables-nft-1.8.8-3.amzn2023.0.2.x86_64  libcgroup-3.0-1.amzn2023.0.1.x86_64  libnetfilter_conntrack-1.0.8-2.amzn2023.0.2.x86_64
libnftnl-1.0.1-19.amzn2023.0.2.x86_64  libnftnl-1.2.2-2.amzn2023.0.2.x86_64  pigz-2.5-1.amzn2023.0.3.x86_64
runc-1.1.11-1.amzn2023.0.1.x86_64

Complete!
[root@ip-10-0-1-99 ec2-user]#
[root@ip-10-0-1-99 ec2-user]# systemctl start docker
[root@ip-10-0-1-99 ec2-user]#
[root@ip-10-0-1-99 ec2-user]#

```

Ilustración 18. comandos de consola linux2

Para saber si ya está activado o no utilizamos el siguiente comando `systemctl status Docker`

```

3.147.79.87 (ec2-user)
Terminal Sessions View X server Tools Games Settings Macros Help
Session Servers Tools Games Sessions View Split MultiExec Tuning Packages Settings Help
Quick connect...
/home/ec2-user/
Name
ssh
.bash_logout
.bash_profile
.bashrc
Verifying : libnftnl-1.2.2-2.amzn2023.0.2.x86_64 8/10
Verifying : pigz-2.5-1.amzn2023.0.3.x86_64 9/10
Verifying : runc-1.1.11-1.amzn2023.0.1.x86_64 10/10
Installed:
containerd-1.7.11-1.amzn2023.0.1.x86_64 docker-25.0.3-1.amzn2023.0.1.x86_64 iptables-libs-1.8.8-3.amzn2023.0.2.x86_64
iptables-nft-1.0.8-3.amzn2023.0.2.x86_64 libcgrouprp-3.0-1.amzn2023.0.1.x86_64 libnetfilter_conntrack-1.0.8-2.amzn2023.0.2.x86_64
libnftnl-1.0.1-19.amzn2023.0.2.x86_64 libnftnl-1.2.2-2.amzn2023.0.2.x86_64 libnftnl-1.2.2-2.amzn2023.0.2.x86_64
runc-1.1.11-1.amzn2023.0.1.x86_64 pigz-2.5-1.amzn2023.0.3.x86_64
Complete!
[root@ip-10-0-1-99 ec2-user]#
[root@ip-10-0-1-99 ec2-user]# systemctl start docker
[root@ip-10-0-1-99 ec2-user]#
[root@ip-10-0-1-99 ec2-user]# systemctl status docker
● docker.service - Docker Application Container Engine
   Loaded: loaded (/usr/lib/systemd/system/docker.service; disabled; preset: disabled)
   Active: active (running) since Sun 2024-05-26 15:38:20 UTC; 3min 16s ago
   TriggeredBy: ● docker.socket
   Docs: https://docs.docker.com
   Process: 27196 ExecStartPre=/bin/mkdir -p /run/docker (code=exited, status=0/SUCCESS)
   Process: 27197 ExecStartPre=/usr/libexec/docker/docker-setup-runtimes.sh (code=exited, status=0/SUCCESS)
   Main PID: 27198 (dockerd)
   Tasks: 0
   Memory: 39.4M
   CPU: 334ms
   CGroup: /system.slice/docker.service
           └─27198 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd.sock --default-ulimit nfile=32768:65536
May 26 15:38:19 ip-10-0-1-99.us-east-2.compute.internal systemd[1]: Starting docker.service - Docker Application Container Engine...
May 26 15:38:20 ip-10-0-1-99.us-east-2.compute.internal dockerd[27198]: time="2024-05-26T15:38:20.840762562Z" level=info msg="Starting u
May 26 15:38:20 ip-10-0-1-99.us-east-2.compute.internal dockerd[27198]: time="2024-05-26T15:38:20.111889265Z" level=info msg="Loading co
May 26 15:38:20 ip-10-0-1-99.us-east-2.compute.internal dockerd[27198]: time="2024-05-26T15:38:20.472664136Z" level=info msg="Loading co
May 26 15:38:20 ip-10-0-1-99.us-east-2.compute.internal dockerd[27198]: time="2024-05-26T15:38:20.496050492Z" level=info msg="Docker dae
May 26 15:38:20 ip-10-0-1-99.us-east-2.compute.internal dockerd[27198]: time="2024-05-26T15:38:20.498318203Z" level=info msg="Daemon has
May 26 15:38:20 ip-10-0-1-99.us-east-2.compute.internal dockerd[27198]: time="2024-05-26T15:38:20.535808625Z" level=info msg="API Listen
May 26 15:38:20 ip-10-0-1-99.us-east-2.compute.internal systemd[1]: Started docker.service - Docker Application Container Engine.
[root@ip-10-0-1-99 ec2-user]#
  
```

Ilustración 19. comandos de consola linux3

Ahora lo que instalaremos es una imagen para mostrar en ese contenedor para eso vamos a escribir el siguiente comando `Docker pull httpd` y una vez ya haiga instala la mostramos con el comando `Docker images` donde nos muestra su tamaño y otras especificaciones

```

1.147.79.87 (ec2-user)
Terminal Sessions View X server Tools Games Settings Macros Help
Session Servers Tools Games Sessions View Split MultiExec Tunneling Packages Settings Help
Quick connect...
/home/ec2-user/
Name
- .ssh
- .ssh_logout
- .ssh_profile
- .bashrc
TriggeredBy: ● docker.socket
Date: https://docs.docker.com
Process: 27196 ExecStartPre=/bin/mkdir -p /run/docker [code=exited, status=0/SUCCESS]
Process: 27197 ExecStartPre=/usr/libexec/docker/docker-setup-runtimes.sh [code=exited, status=0/SUCCESS]
Main PID: 27198 (dockerd)
Tasks: 8
Memory: 39.1M
CPU: 334ms
CGROUP: /system.slice/docker.service
└─27198 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd.sock --default-ulimit nofile=32768:65536

May 26 15:38:19 ip-10-0-1-99.us-east-2.compute.internal systemd[1]: Starting docker.service - Docker Application Container Engine...
May 26 15:38:20 ip-10-0-1-99.us-east-2.compute.internal dockerd[27198]: time="2024-05-26T15:38:20.849761256Z" level=info msg="Starting u...
May 26 15:38:20 ip-10-0-1-99.us-east-2.compute.internal dockerd[27198]: time="2024-05-26T15:38:20.111889265Z" level=info msg="Loading co...
May 26 15:38:20 ip-10-0-1-99.us-east-2.compute.internal dockerd[27198]: time="2024-05-26T15:38:20.472664136Z" level=info msg="Loading co...
May 26 15:38:20 ip-10-0-1-99.us-east-2.compute.internal dockerd[27198]: time="2024-05-26T15:38:20.496090492Z" level=info msg="Docker dae...
May 26 15:38:20 ip-10-0-1-99.us-east-2.compute.internal dockerd[27198]: time="2024-05-26T15:38:20.496310293Z" level=info msg="Daemon has...
May 26 15:38:20 ip-10-0-1-99.us-east-2.compute.internal dockerd[27198]: time="2024-05-26T15:38:20.535088625Z" level=info msg="APT listen...
May 26 15:38:20 ip-10-0-1-99.us-east-2.compute.internal systemd[1]: Started docker.service - Docker Application Container Engine.

[root@ip-10-0-1-99 ec2-user]#
[root@ip-10-0-1-99 ec2-user]#
[root@ip-10-0-1-99 ec2-user]#
[root@ip-10-0-1-99 ec2-user]# docker pull httpd
Using default tag: latest
latest: Pulling from library/httpd
09f376ebb190: Pull complete
d4b5584bfc3: Pull complete
4f4fb780ef54: Pull complete
1a6d0283f224: Pull complete
1abf0118528c: Pull complete
7bacb8f85f3a: Pull complete
Digest: sha256:43c7061a3243c84b0955c81ac694ea13a1d8a1e53c15023a7b3cd9e8bb25de3c
Status: Downloaded newer image for httpd:latest
docker.io/library/httpd:latest
[root@ip-10-0-1-99 ec2-user]# docker images
REPOSITORY TAG IMAGE ID CREATED SIZE
httpd latest 356125da0595 7 weeks ago 147MB
[root@ip-10-0-1-99 ec2-user]#

```

*Ilustración 20. comandos de consola linux4*

Ahora vamos a crear una ruta donde vamos a correr nuestras imágenes para los contenedores con los comandos `cd..` para salir, ls para mostrar `cd` y el nombre de la carpeta para ingresar en ella y con `mkdir` para crear carpeta en este caso vamos a crear una carpeta contenedor en la carpeta raíz `tmp` e ingresamos en esa carpeta contenedor

```

1.147.79.87 (ec2-user)
Terminal Sessions View X server Tools Games Settings Macros Help
Session Servers Tools Games Sessions View Split Multitex Tunneling Packages Settings Help
Quick connect...
L-27198 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd.sock --default-ulimit nofile=32768:65536
May 26 15:38:19 ip-10-0-1-99.us-east-2.compute.internal systemd[1]: Starting docker.service - Docker Application Container Engine...
May 26 15:38:20 ip-10-0-1-99.us-east-2.compute.internal dockerd[27198]: time="2024-05-26T15:38:20.949761256Z" level=info msg="Starting u
May 26 15:38:20 ip-10-0-1-99.us-east-2.compute.internal dockerd[27198]: time="2024-05-26T15:38:20.472664130Z" level=info msg="Loading co
May 26 15:38:20 ip-10-0-1-99.us-east-2.compute.internal dockerd[27198]: time="2024-05-26T15:38:20.496090422Z" level=info msg="Docker dae
May 26 15:38:20 ip-10-0-1-99.us-east-2.compute.internal dockerd[27198]: time="2024-05-26T15:38:20.496319203Z" level=info msg="Daemon has
May 26 15:38:20 ip-10-0-1-99.us-east-2.compute.internal systemd[1]: Started docker.service - Docker Application Container Engine.
[root@ip-10-0-1-99 ec2-user]#
[root@ip-10-0-1-99 ec2-user]#
[root@ip-10-0-1-99 ec2-user]# docker pull httpd
Using default tag: latest
latest: Pulling from library/httpd
69f736ebb190: Pull complete
dab5b4abfc3: Pull complete
4f4b70nef54: Pull complete
1a6d0283f224: Pull complete
1abf9119528c: Pull complete
7baeb0f85fa: Pull complete
Digest: sha256:43c766193242c64b0955c81ac094ea13a1d8a1e53c15023a7b3cd5e8bb25de3c
Status: Downloaded newer image for httpd:latest
docker.io/library/httpd:latest
[root@ip-10-0-1-99 ec2-user]# docker images
REPOSITORY          TAG         IMAGE ID      CREATED      SIZE
httpd                latest     356125da0595   7 weeks ago  147MB
[root@ip-10-0-1-99 ec2-user]# ls
[root@ip-10-0-1-99 home]# cd ..
[root@ip-10-0-1-99 /]# ls
bin boot dev etc home lib lib64 local media mnt opt proc root run sbin srv sys tmp usr var
[root@ip-10-0-1-99 /]# cd tmp/
[root@ip-10-0-1-99 tmp]# mkdir contenedor
[root@ip-10-0-1-99 tmp]# cd contenedor/
[root@ip-10-0-1-99 contenedor]#

```

*Ilustración 21. creamos ruta*

Ya una vez hallamos creado la ruta vamos a asignar nuestro httpd ya antes descargado con el siguiente comando al contenedor `docker run -dit --name contenedor -p 8080:80 -v /tmp/contenedor:/usr/local/apache2/htdocs/ httpd` donde asignamos el nombre los puertos la ruta de la carpeta contenedor y la ruta de nuestro httpd y cómo vamos utilizar 2 creamos el otro pero hay que tener en cuenta que hay que cambiarles los campos de nombre y del puerto para que no genere un error y ya con `Docker ps` nuestro los contenedores creados y otras especificaciones

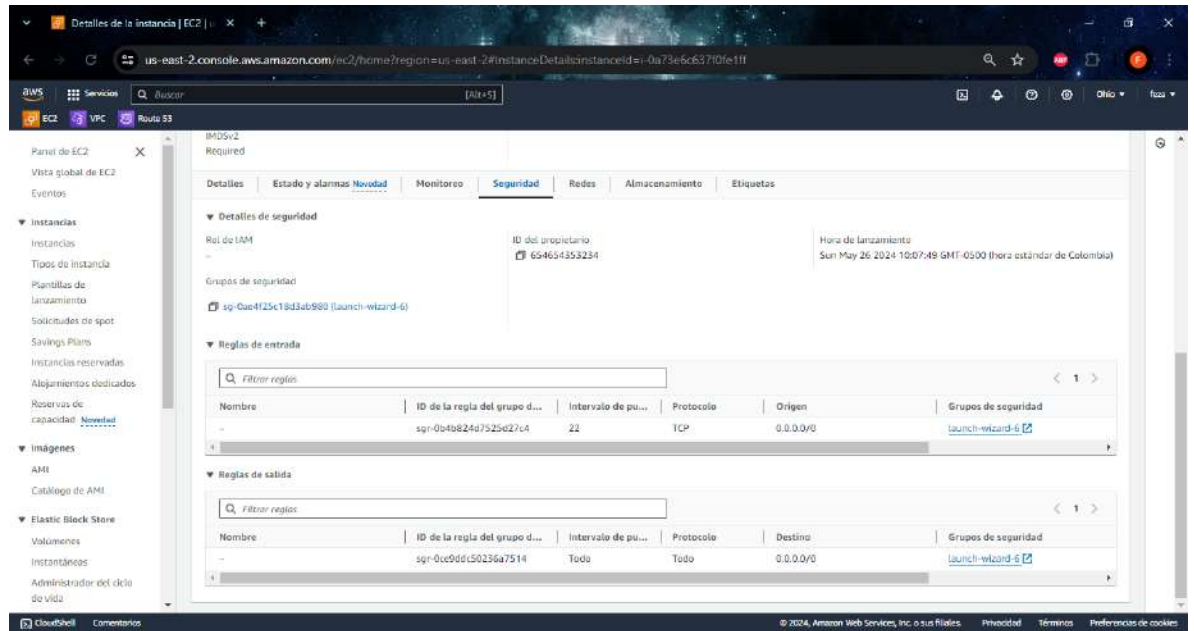
```

[ec2-user@ip-10-0-1-99 ~]$ docker pull httpd
Using default tag: latest
latest: Pulling from library/httpd
09f376ebb190: Pull complete
da05b4abfc3: Pull complete
4f4fb790ef54: Pull complete
1a6d0283f224: Pull complete
1abf9110528c: Pull complete
7bacb8f85f3a: Pull complete
Digest: sha256:43c7661a3243c04b9955c81ac94ea13a1d8a1e53c15023a7b3cd5e8bb25deac
Status: Downloaded newer image for httpd:latest
docker.io/library/httpd:latest
[ec2-user@ip-10-0-1-99 ~]$ docker images
REPOSITORY TAG IMAGE ID CREATED SIZE
httpd latest 356125da0595 7 weeks ago 147MB
[ec2-user@ip-10-0-1-99 ~]$ cd /tmp/
[ec2-user@ip-10-0-1-99 tmp]$ mkdir contenedor
[ec2-user@ip-10-0-1-99 tmp]$ cd contenedor/
[ec2-user@ip-10-0-1-99 contenedor]$ docker run -dit --name contenedor -p 8080:80 -v /tmp/contenedor:/usr/local/apache2/htdocs/ httpd
652f3639260c7d9142e3ad4137d02c252b8d63d3bd1b15dcca0ee1b04556
[ec2-user@ip-10-0-1-99 contenedor]$ docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
652f3639260c httpd "httpd-foreground" 58 seconds ago Up 57 seconds 0.0.0.0:8080->80/tcp, :::8080->80/tcp contenedor
[ec2-user@ip-10-0-1-99 contenedor]$ docker run -dit --name contenedor1 -p 8081:80 -v /tmp/contenedor:/usr/local/apache2/htdocs/ httpd
acc9e87f2813a40ae199d877fe167b95898081bc901455be967bb2b92d3c522
[ec2-user@ip-10-0-1-99 contenedor]$ docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
acc9e87f28f3 httpd "httpd-foreground" 4 seconds ago Up 3 seconds 0.0.0.0:8081->80/tcp, :::8081->80/tcp contenedor1
652f3639260c httpd "httpd-foreground" 2 minutes ago Up 2 minutes 0.0.0.0:8080->80/tcp, :::8080->80/tcp contenedor
[ec2-user@ip-10-0-1-99 contenedor]$

```

*Ilustración 22. contendor*

Ahora nos vamos AWS para crear 2 nuevas puertas de entrada para nuestros 2 contenedores creados anterior mente le damos en ip de la instancia Linux2 como en las imágenes anteriores y en la parte inferior hay una pestaña de seguridad y le damos en grupo de seguridad



*Ilustración 23. configuración grupo seguridad*

Una vez hay nos salen las otras pestañas de reglas tanto de entrada como salida vamos a utilizar la de entrada y le damos en editar regla de entrada

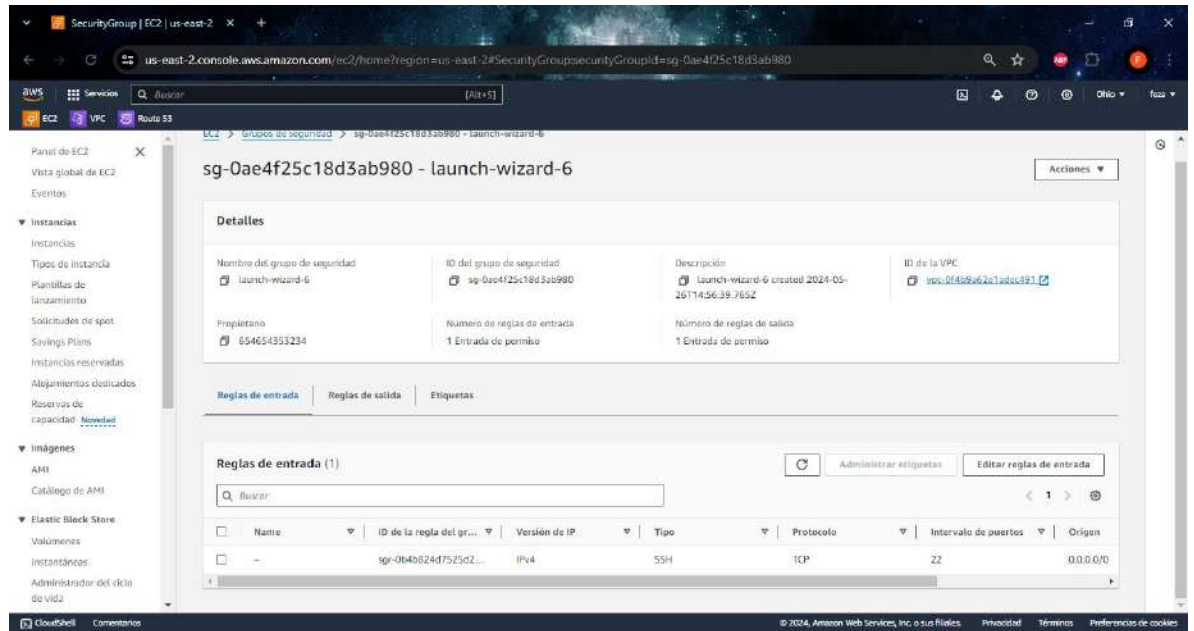
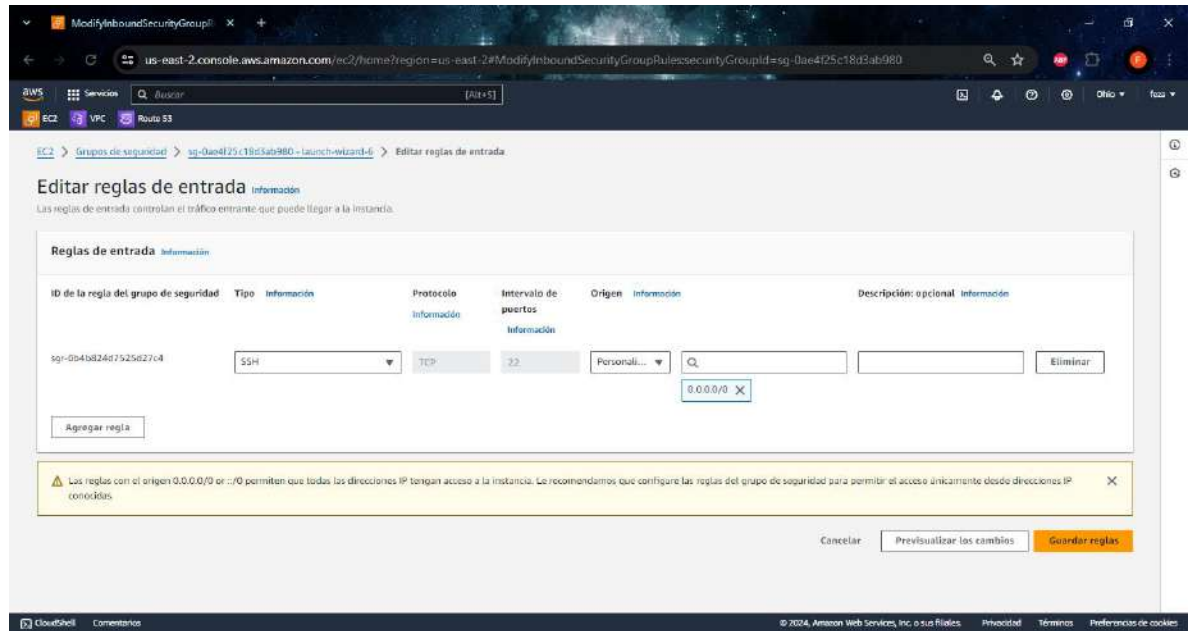


Ilustración 24. reglas de entrada

Vemos que nos sale solo una regla de entrada que es la de ssh por la cual nos conectamos por medio de consola para poder crear los contenedores



*Ilustración 25. agregar regla*

Le damos en agregar regla para ingresar las 2 nuevas reglas de nuestros contenedores hay que tener en cuenta que si modificamos esa regla de ssh no tendremos conexión con nuestro ec2 y le asignamos los puertos que asignamos en nuestros contenedores por ssh y el tipo de información que es tcp personalizado y que cualquier ip con cualquier mascara se puedo conectar a ellos y le damos en guardar

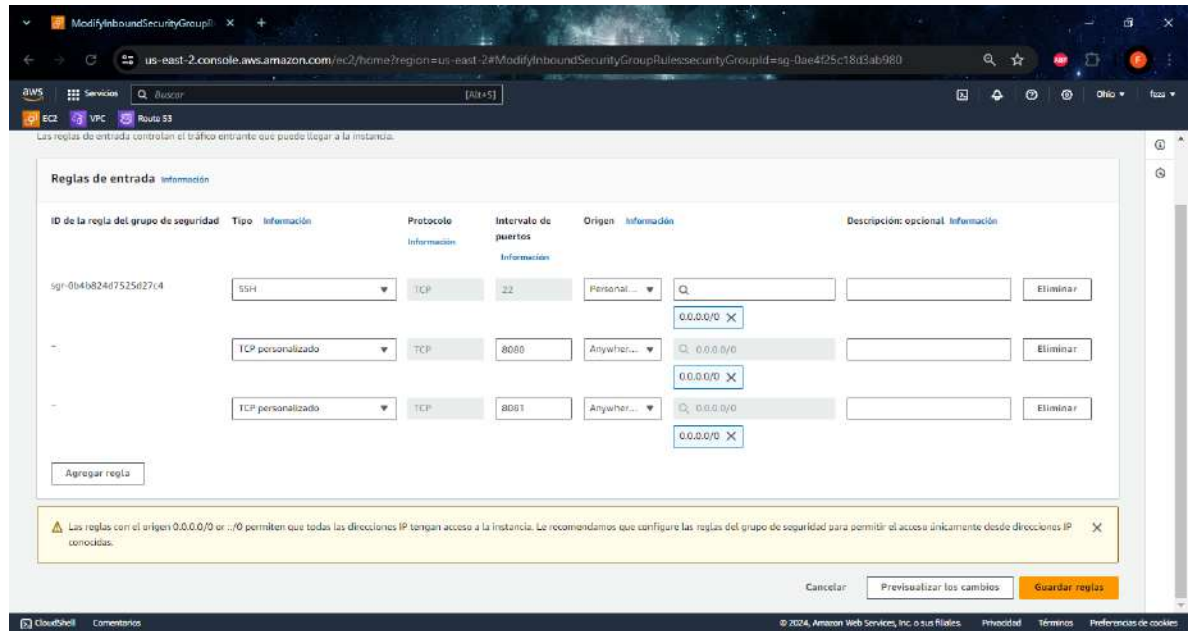


Ilustración 26. TCP personalizado

Nos quedara así

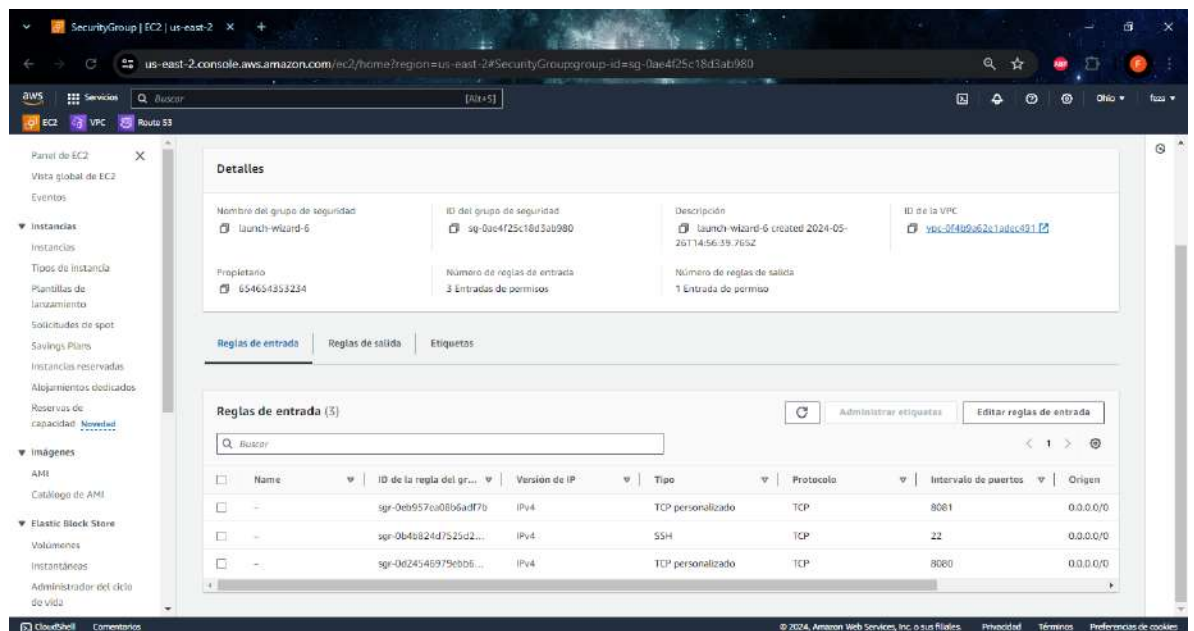


Ilustración 27. Vista nuevas reglas

Ahora probamos nuestro acceso escribiendo el direccionamiento ip publica de nuestra instancia linux2 con su respectivo puerto 8080 y 8081 para saber si no hay errores



*Ilustración 28. resultado 8080*



*Ilustración 29. resultado 8081*

Ahora procederemos a instalar nginx para hacer un proxy de dns para poder conectar por medio de nombres de dominio con el comando `yum install nginx` y el automáticamente instalara sus paquetes

```

1.147.79.87 (ec2-user)
Terminal Sessions View X server Tools Games Settings Macros Help
Session Servers Tools Games Sessions View Split MultiExec Tunneling Packages Settings Help
Quick connect...
/home/ec2-user/
Name
ash
bash_logout
bash_profile
bashrc
Remote monitoring
Follow terminal folder
UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: https://mobaxterm.mobatek.net

Last login: Sun May 26 16:08:29 2024 from 186.84.88.142
[ec2-user@ip-10-0-1-99 ~]$
[ec2-user@ip-10-0-1-99 ~]$ sudo su
[root@ip-10-0-1-99 ec2-user]# sudo yum install nginx
Last metadata expiration check: 2:05:34 ago on Sun May 26 15:08:43 2024.
Dependencies resolved.

Package Architecture Version Repository Size
-----
Installing:
nginx x86_64 1:1.24.0-1.amzn2023.0.2 amazonlinux 32 k
Installing dependencies:
generic-logs-httpd noarch 18.0-0-12.amzn2023.0.3 amazonlinux 19 k
gperftools-libs x86_64 2.9.1-1.amzn2023.0.3 amazonlinux 368 k
libunwind x86_64 1.4.0-5.amzn2023.0.2 amazonlinux 66 k
nginx-core x86_64 1:1.24.0-1.amzn2023.0.2 amazonlinux 586 k
nginx-filessystem noarch 1:1.24.0-1.amzn2023.0.2 amazonlinux 9.1 k
nginx-mimetypes noarch 2.1.49-3.amzn2023.0.3 amazonlinux 21 k

Transaction Summary
-----
Install 7 Packages

Total download size: 1.0 M
Installed size: 3.4 M
Is this ok [y/N]: y
Downloading Packages:
(3/7): libunwind-1.4.0-5.amzn2023.0.2.x86_64.rpm 1.1 MB/s | 65 kB 00:00
(2/7): generic-logs-httpd-18.0-0-12.amzn2023.0.3.noarch.rpm 293 kB/s | 19 kB 00:00
(3/7): gperftools-libs-2.9.1-1.amzn2023.0.3.x86_64.rpm 3.9 MB/s | 368 kB 00:00
(4/7): nginx-1.24.0-1.amzn2023.0.2.x86_64.rpm 1.6 MB/s | 32 kB 00:00
(5/7): nginx-core-1.24.0-1.amzn2023.0.2.x86_64.rpm 19 MB/s | 586 kB 00:00
(6/7): nginx-filessystem-1.24.0-1.amzn2023.0.2.noarch.rpm 483 kB/s | 9.1 kB 00:00
(7/7): nginx-mimetypes-2.1.49-3.amzn2023.0.3.noarch.rpm 1.1 MB/s | 21 kB 00:00
-----
Total 6.4 MB/s | 1.0 MB 00:00
Running transaction check
Transaction check succeeded.

```

Ilustración 30. nginx proxy

```

1.147.79.87 (ec2-user)
Terminal Sessions View X server Tools Games Settings Macros Help
Session Servers Tools Games Sessions View Split MultiExec Tunneling Packages Settings Help
Quick connect...
/home/ec2-user/
Name
ash
bash_logout
bash_profile
bashrc
Remote monitoring
Follow terminal folder
UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: https://mobaxterm.mobatek.net

(3/7): gperftools-libs-2.9.1-1.amzn2023.0.3.x86_64.rpm 3.9 MB/s | 368 kB 00:00
(4/7): nginx-1.24.0-1.amzn2023.0.2.x86_64.rpm 1.6 MB/s | 32 kB 00:00
(5/7): nginx-core-1.24.0-1.amzn2023.0.2.x86_64.rpm 19 MB/s | 586 kB 00:00
(6/7): nginx-filessystem-1.24.0-1.amzn2023.0.2.noarch.rpm 483 kB/s | 9.1 kB 00:00
(7/7): nginx-mimetypes-2.1.49-3.amzn2023.0.3.noarch.rpm 1.1 MB/s | 21 kB 00:00
-----
Total 6.4 MB/s | 1.0 MB 00:00
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
Preparing :
Running scriptlet: nginx-filessystem-1:1.24.0-1.amzn2023.0.2.noarch 1/1
Installing : nginx-filessystem-1:1.24.0-1.amzn2023.0.2.noarch 1/1
Installing : nginx-mimetypes-2.1.49-3.amzn2023.0.3.noarch 2/1
Installing : libunwind-1.4.0-5.amzn2023.0.2.x86_64 3/1
Installing : gperftools-libs-2.9.1-1.amzn2023.0.3.x86_64 4/1
Installing : nginx-core-1:1.24.0-1.amzn2023.0.2.x86_64 5/1
Installing : generic-logs-httpd-18.0-0-12.amzn2023.0.3.noarch 6/1
Installing : nginx-1:1.24.0-1.amzn2023.0.2.x86_64 7/1
Running scriptlet: nginx-1:1.24.0-1.amzn2023.0.2.x86_64 7/1
Verifying : generic-logs-httpd-18.0-0-12.amzn2023.0.3.noarch 1/1
Verifying : gperftools-libs-2.9.1-1.amzn2023.0.3.x86_64 2/1
Verifying : libunwind-1.4.0-5.amzn2023.0.2.x86_64 3/1
Verifying : nginx-1:1.24.0-1.amzn2023.0.2.x86_64 4/1
Verifying : nginx-core-1:1.24.0-1.amzn2023.0.2.x86_64 5/1
Verifying : nginx-filessystem-1:1.24.0-1.amzn2023.0.2.noarch 6/1
Verifying : nginx-mimetypes-2.1.49-3.amzn2023.0.3.noarch 7/1

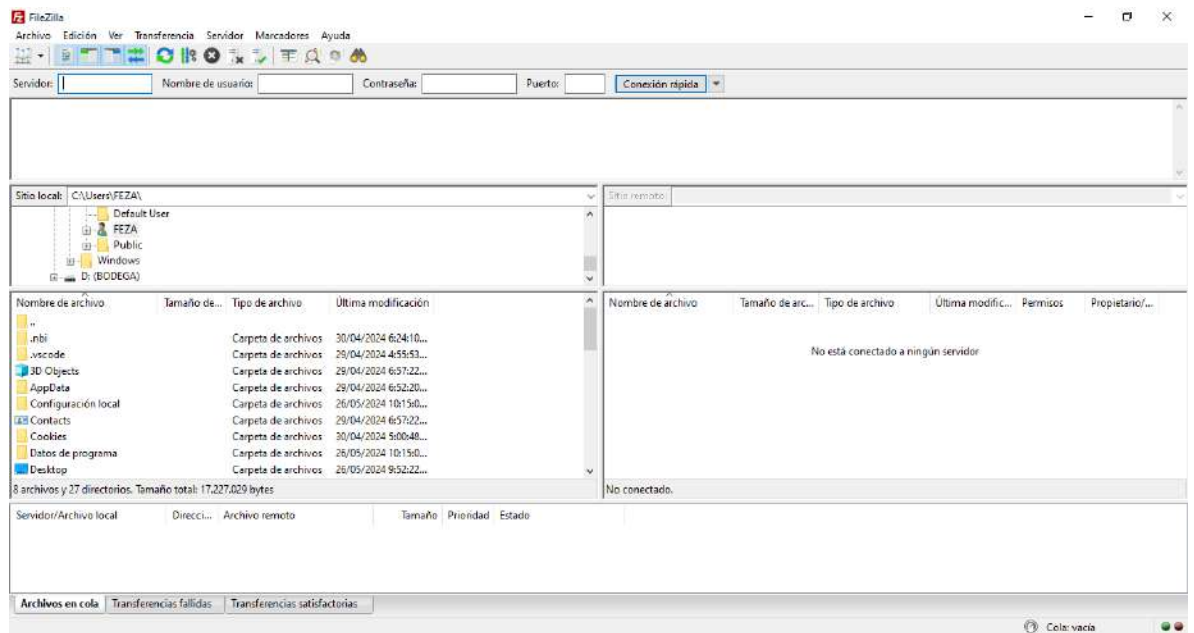
Installed:
generic-logs-httpd-18.0-0-12.amzn2023.0.3.noarch gperftools-libs-2.9.1-1.amzn2023.0.3.x86_64
libunwind-1.4.0-5.amzn2023.0.2.x86_64 nginx-1:1.24.0-1.amzn2023.0.2.x86_64
nginx-core-1:1.24.0-1.amzn2023.0.2.x86_64 nginx-filessystem-1:1.24.0-1.amzn2023.0.2.noarch
nginx-mimetypes-2.1.49-3.amzn2023.0.3.noarch

Complete!
[root@ip-10-0-1-99 ec2-user]#

```

Ilustración 31. nginx proxy resultado

Ahora utilizaremos el programa filezilla con el cual por medio de un entorno grafico veremos nuestras carpetas de la ec2 con la que estamos trabajando



*Ilustración 32. filezilla*

Nos vamos a la pestaña de archivo y le damos en gestor de sitios donde vamos a suministrar los datos de conexión de nuestra ec2 como lo hicimos con el programa mobaxterm y le damos conectar

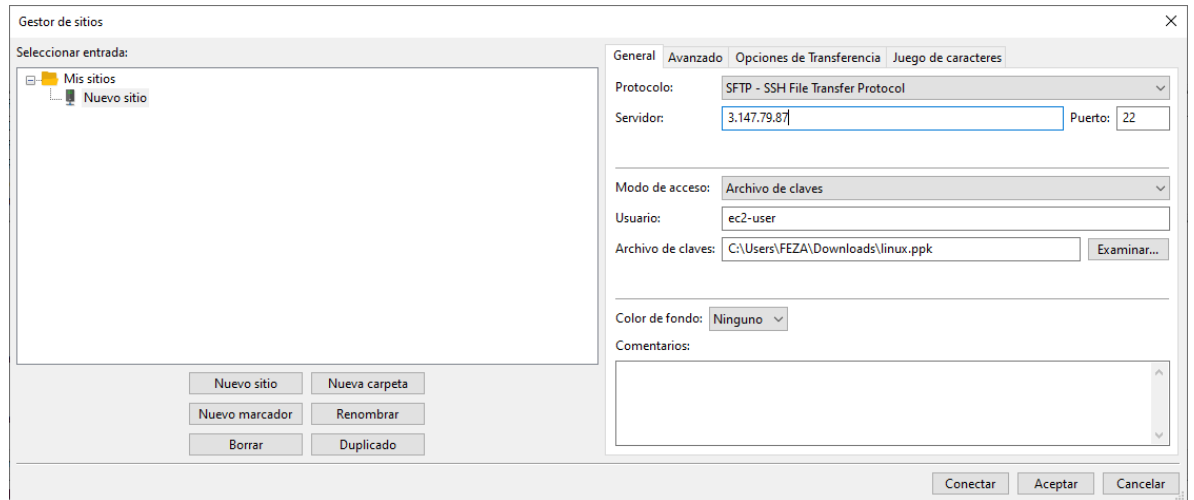


Ilustración 33. filezilla SSH

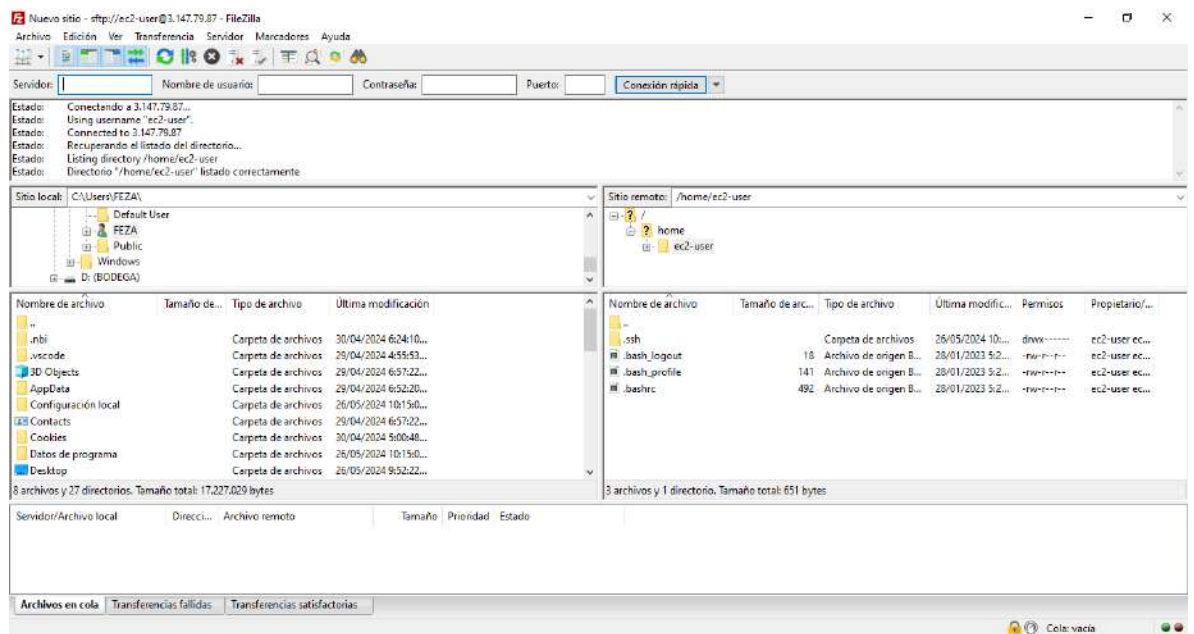


Ilustración 34. resultado conexión

Una vez hay buscamos esta dirección /etc/nginx y en esa carpeta de nginx buscamos el siguiente archivo nginx.conf donde le vamos a dar a editar con nuestro editor de texto de confianza para darle los parámetros para nuestro proxy de dns

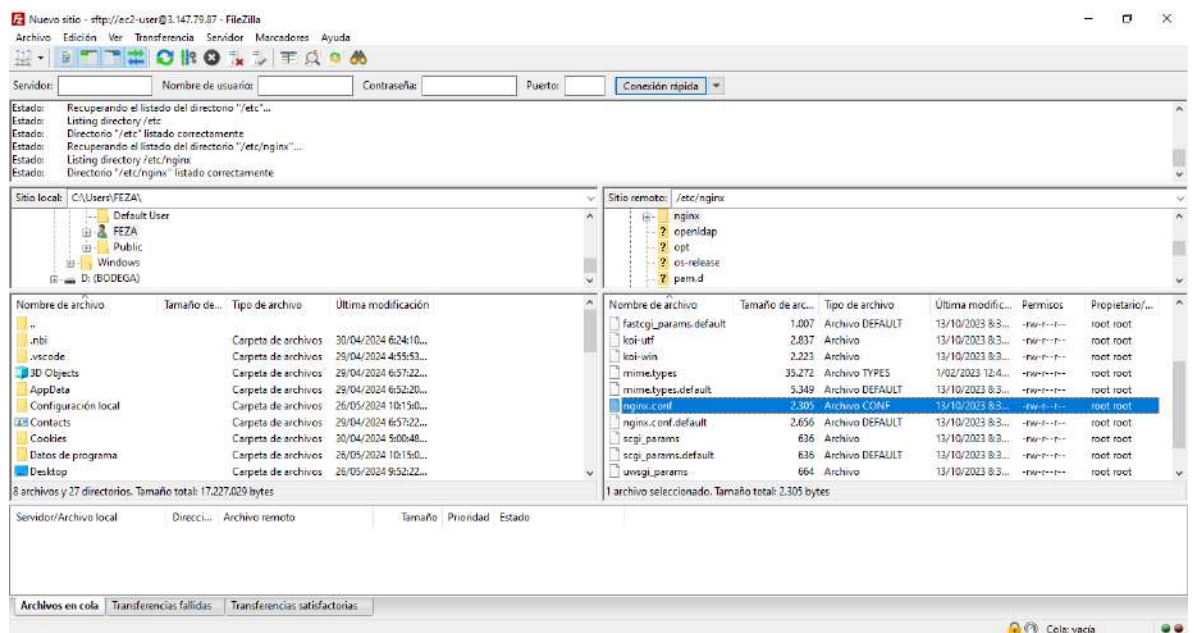
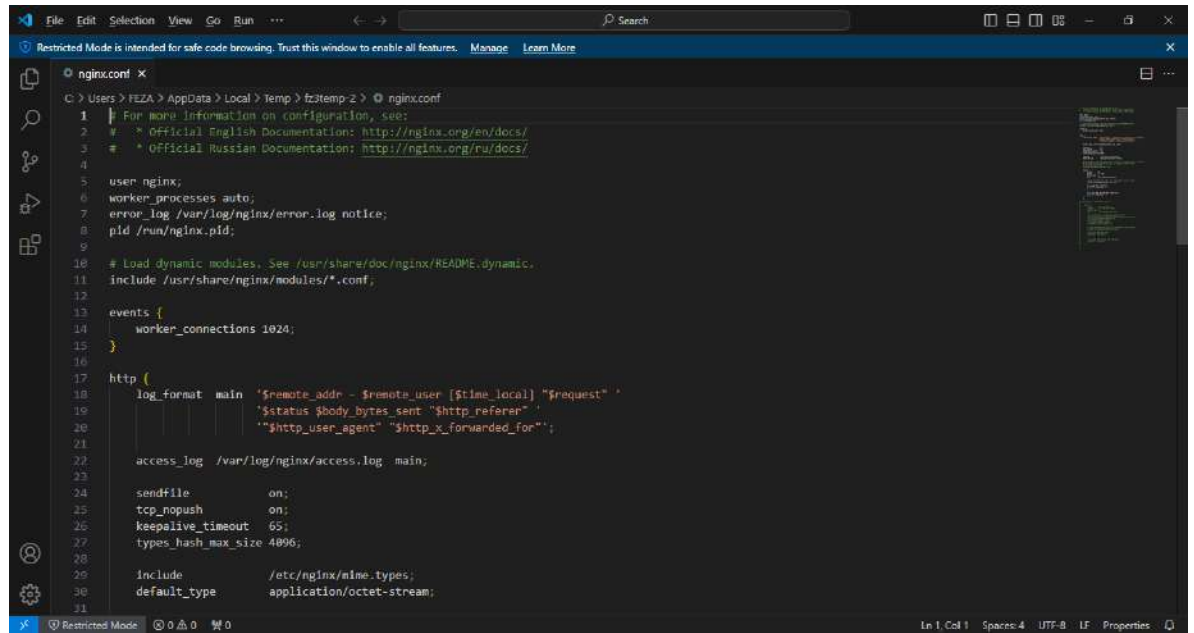


Ilustración 35. nginx.conf

ese archivo trae por defecto unos valores predeterminados los borramos todos e insertamos lo siguiente



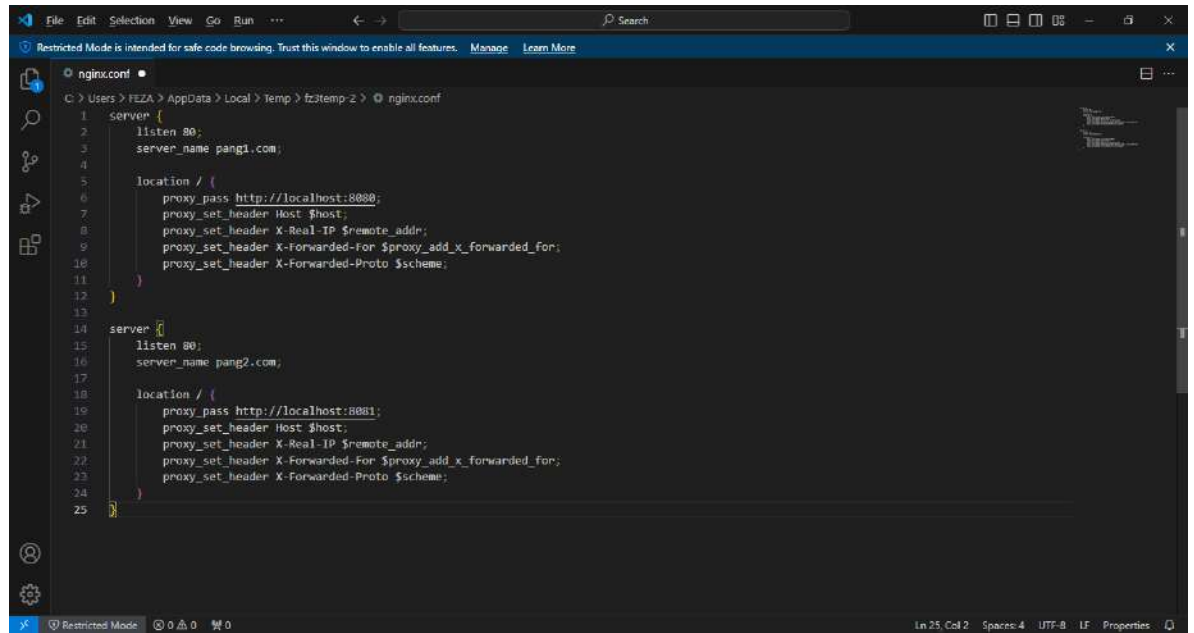
```

1 | For more information on configuration, see:
2 | * Official English Documentation: http://nginx.org/en/docs/
3 | * Official Russian Documentation: http://nginx.org/ru/docs/
4 |
5 | user nginx;
6 | worker_processes auto;
7 | error_log /var/log/nginx/error.log notice;
8 | pid /run/nginx.pid;
9 |
10 |
11 | # Load dynamic modules. See /usr/share/doc/nginx/README.dynamic.
12 | include /usr/share/nginx/modules/*.conf;
13 |
14 | events {
15 |     worker_connections 1024;
16 | }
17 |
18 | http {
19 |     log_format main '$remote_addr - $remote_user [$time_local] "$request" '
20 |                    '$status $body_bytes_sent "$http_referer" '
21 |                    '"$http_user_agent" "$http_x_forwarded_for"';
22 |
23 |     access_log /var/log/nginx/access.log main;
24 |
25 |     sendfile        on;
26 |     tcp_nopush     on;
27 |     keepalive_timeout 65;
28 |     types_hash_max_size 4096;
29 |
30 |     include        /etc/nginx/mime.types;
31 |     default_type   application/octet-stream;

```

### *Ilustración 36. código por editar*

Con esto le estamos amarrando los puertos 8080 y 8081 a 2 nombre de dominio que son pang1.com y pang2.com le damos en guardar y listo hay que tener presente que al guardar en filezilla nos va a pedir sobrescribir el archivo y le damos que sí y confirmamos que si haiga guardo abriendo otra vez



```
1 server {
2     listen 80;
3     server_name pang1.com;
4
5     location / {
6         proxy_pass http://localhost:8080;
7         proxy_set_header Host $host;
8         proxy_set_header X-Real-IP $remote_addr;
9         proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
10        proxy_set_header X-Forwarded-Proto $scheme;
11    }
12 }
13
14 server {
15     listen 80;
16     server_name pang2.com;
17
18     location / {
19         proxy_pass http://localhost:8081;
20         proxy_set_header Host $host;
21         proxy_set_header X-Real-IP $remote_addr;
22         proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
23         proxy_set_header X-Forwarded-Proto $scheme;
24    }
25 }
```

*Ilustración 37. código editado*

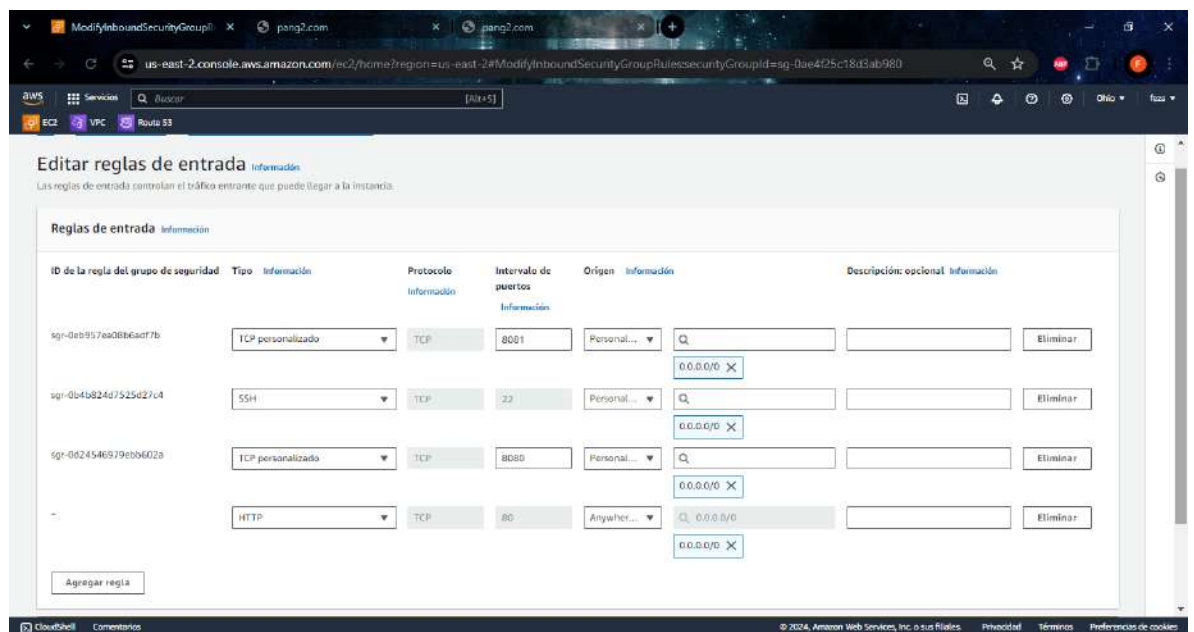
Ya una vez comprobado la información guarda vamos de nuevo a mobaxterm para resetear el nginx con este comando `sudo systemctl restart nginx`



Le agregamos estas 2 líneas de comandos donde asignamos la ip publica de ec2 a esos nombres de dominio 3.147.79.87 pang1.com 3.147.79.87 pang2.com

127.0.0.1 localhost

Ahora vamos a nuestro ec2 en el apartado de seguridad para agregarle otra regla de entra en este caso es http puerto 80 y le damos guardar



*Ilustración 40. reglas de entradas nuevas*

Ya con esto probamos los nombres de dominio pang1.com y pang2.com y listo ya con eso terminamos



## ¡Bienvenido a nginx!

Si ve esta página, el servidor web nginx se instaló correctamente y funciona. Se requiere configuración adicional.

Para obtener documentación y soporte en línea, consulte [nginx.org](http://nginx.org) .  
El soporte comercial está disponible en [nginx.com](http://nginx.com) .

*Gracias por usar nginx.*

*Ilustración 41. página pang1.com*



## ¡Bienvenido a nginx!

Si ve esta página, el servidor web nginx se instaló correctamente y funciona. Se requiere configuración adicional.

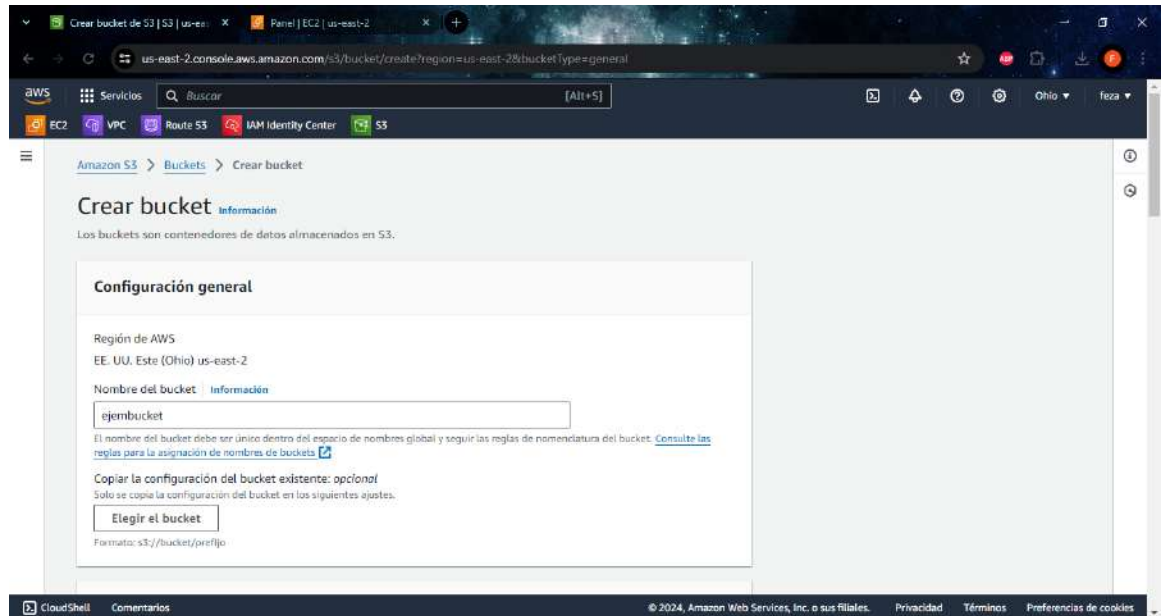
Para obtener documentación y soporte en línea, consulte [nginx.org](http://nginx.org) .  
El soporte comercial está disponible en [nginx.com](http://nginx.com) .

*Gracias por usar nginx.*

*Ilustración 42. página pang2.com*

### Ejercicio 3

Como crear un bucket nos Vamos a s3 y en crear bucket



*Ilustración 43. ejercicio 3 crear bucket*

Asignamos las listas de control de acceso del bucket

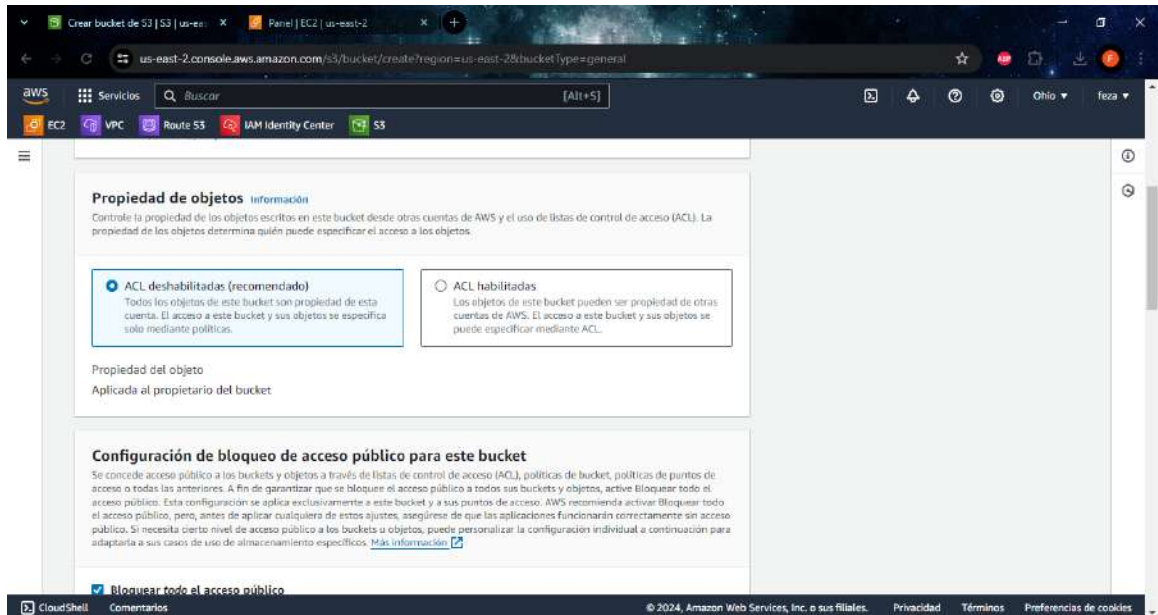


Ilustración 44. lista de control bucket

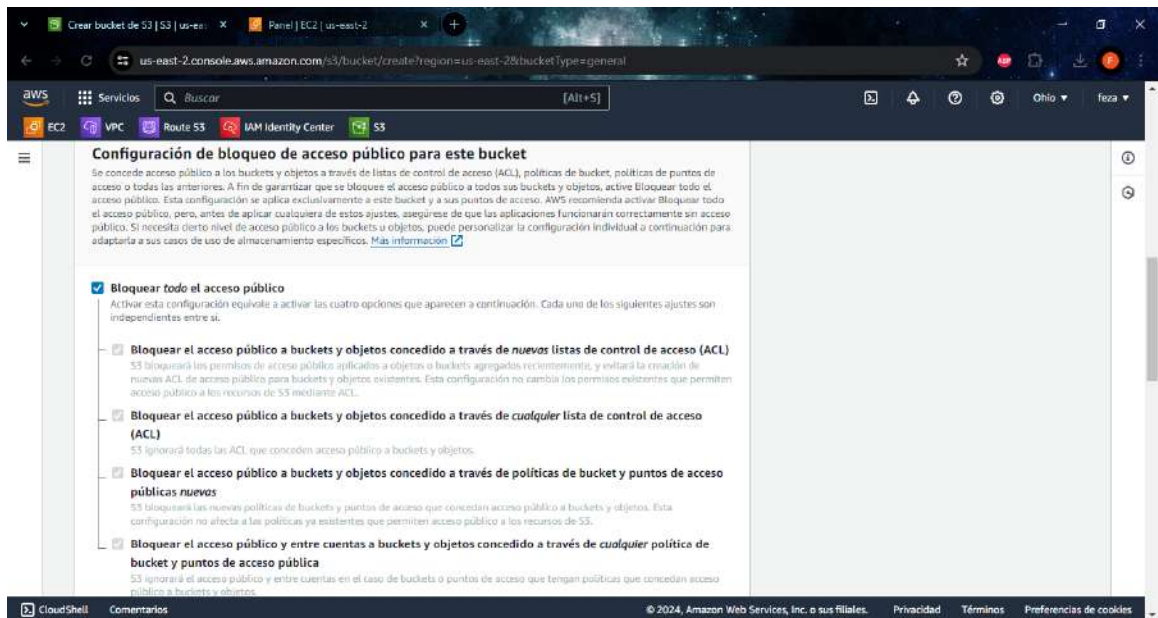


Ilustración 45. lista de control bucket2

Cogemos el tipo de cifrado para mayor seguridad en caso tal de que descarguen la información ella va a estar encriptada no la podrán ver

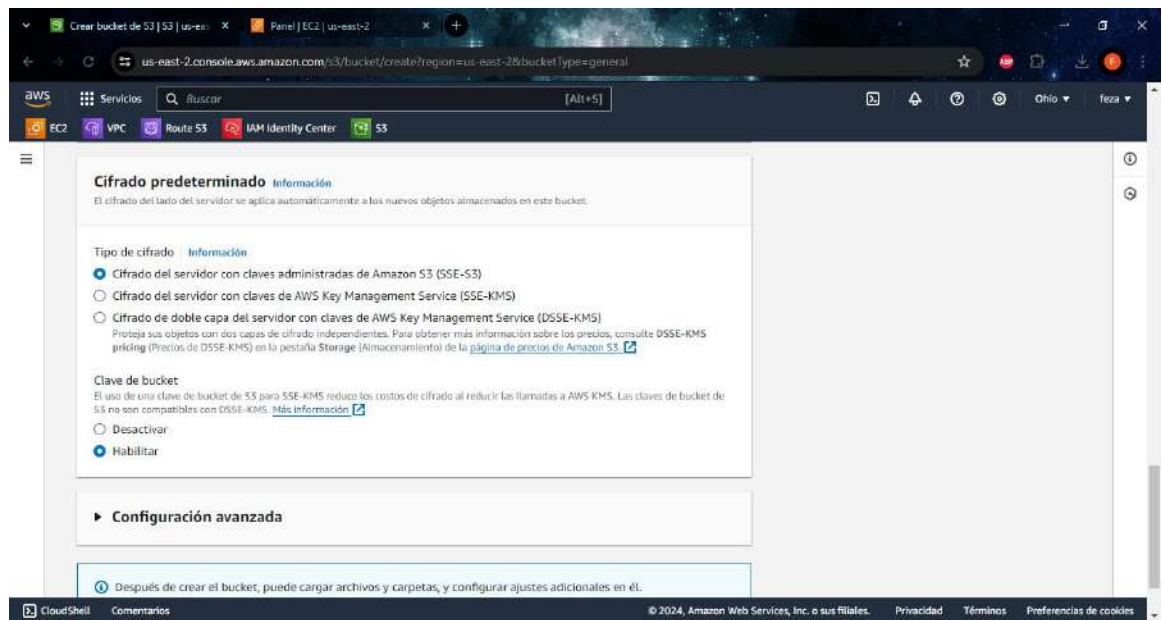
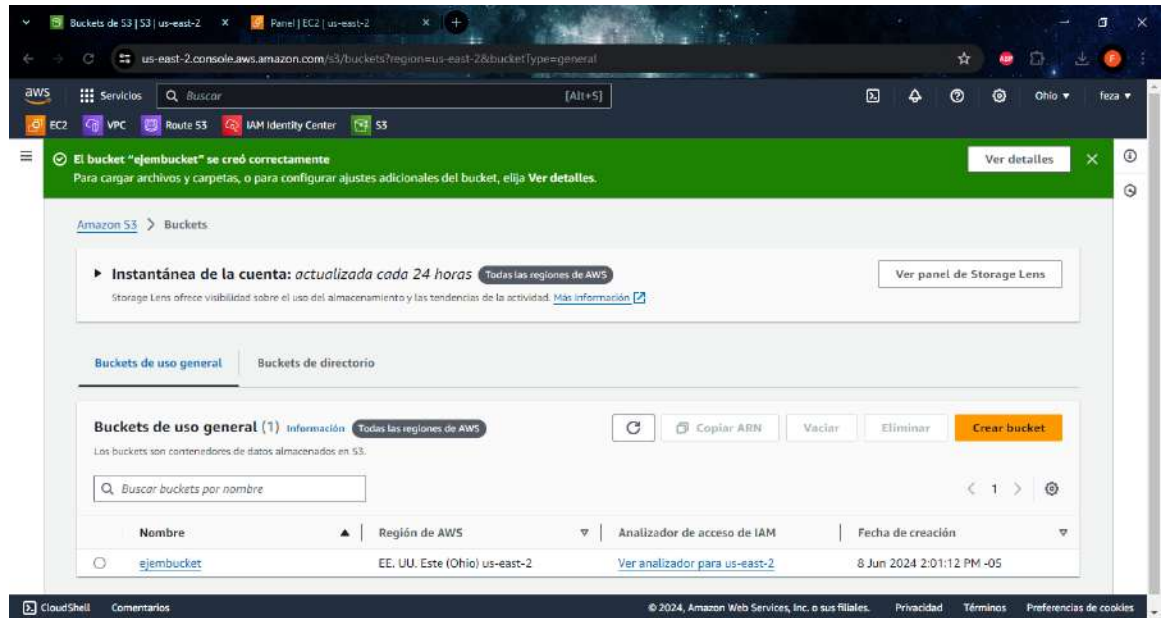


Ilustración 46. tipo de encriptación

Ya le damos en crear y finalizamos la creación del bucket



*Ilustración 47. crear bucket*

Subimos un archive a nuestro bucket

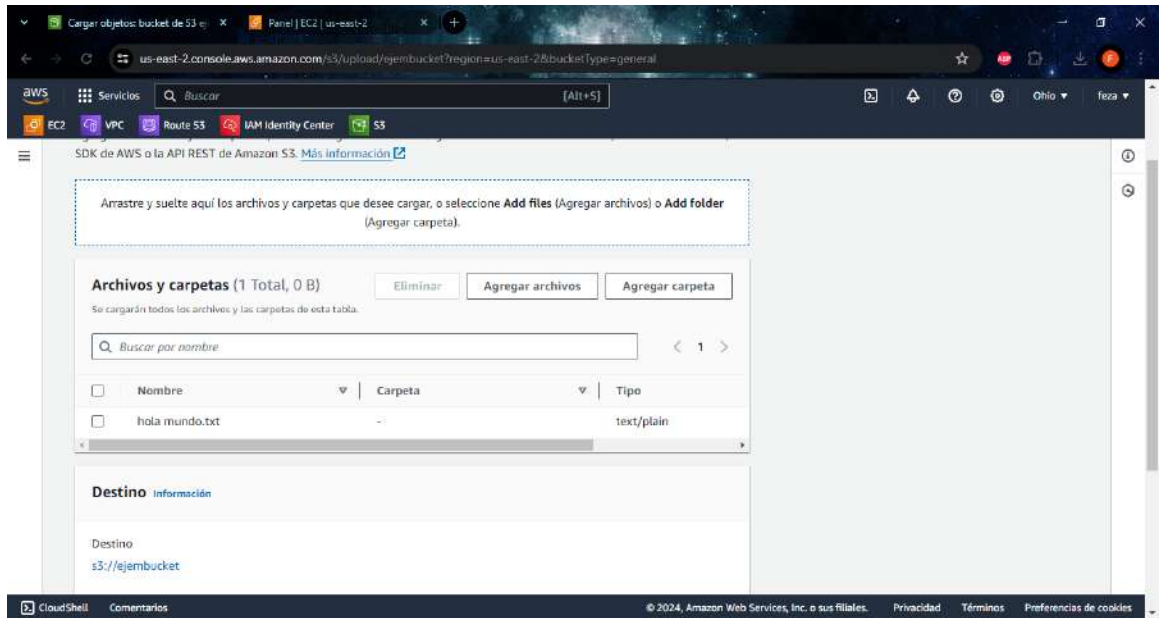


Ilustración 48. cargamos el archivo al bucket

Hay vemos nuestro archivo

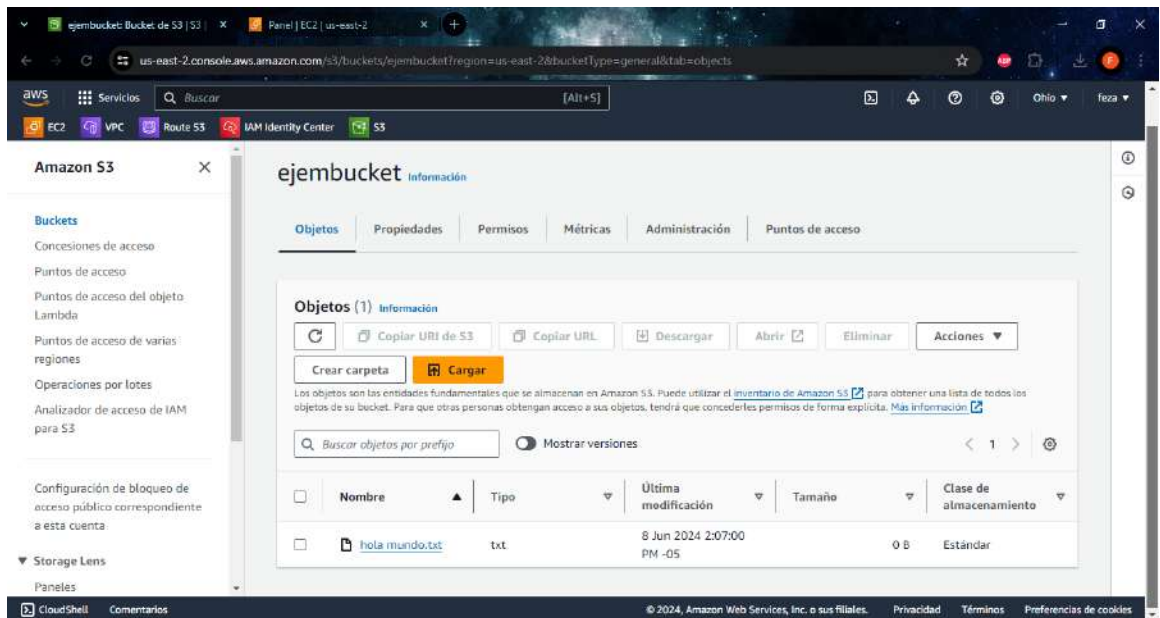
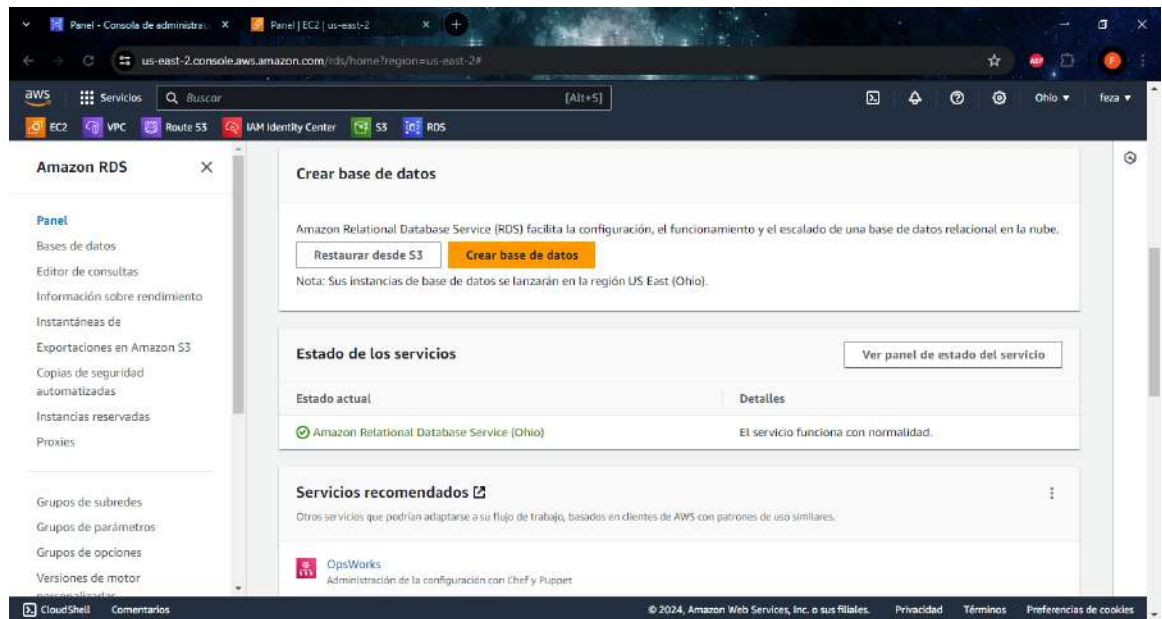


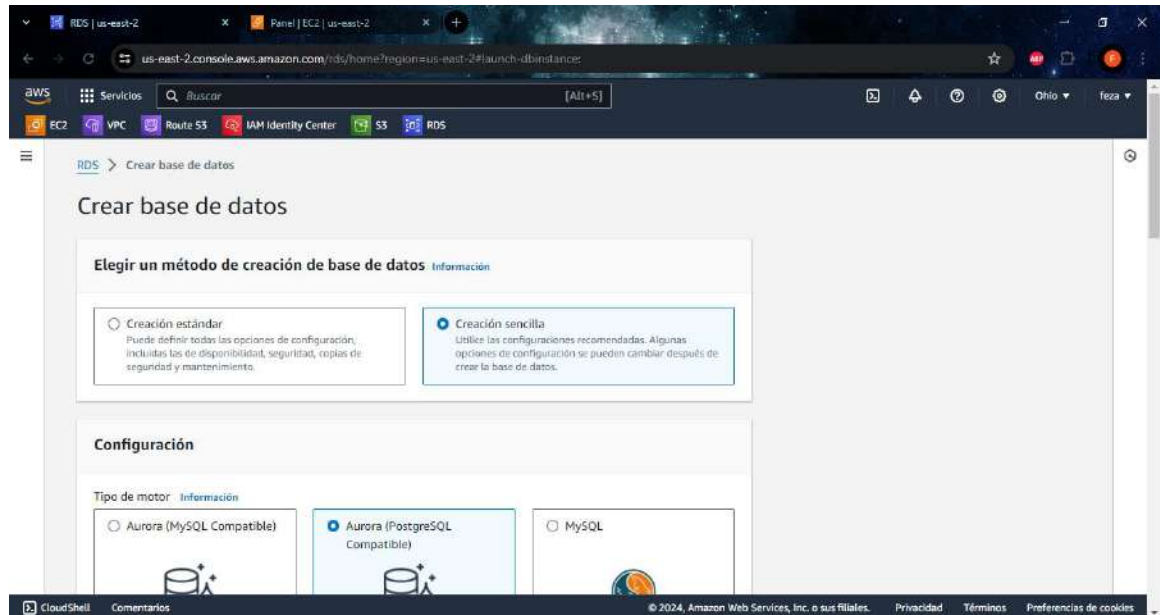
Ilustración 49. vista archivo

Ahora vamos a crear una rds que es un servicio de bases de datos relacionales



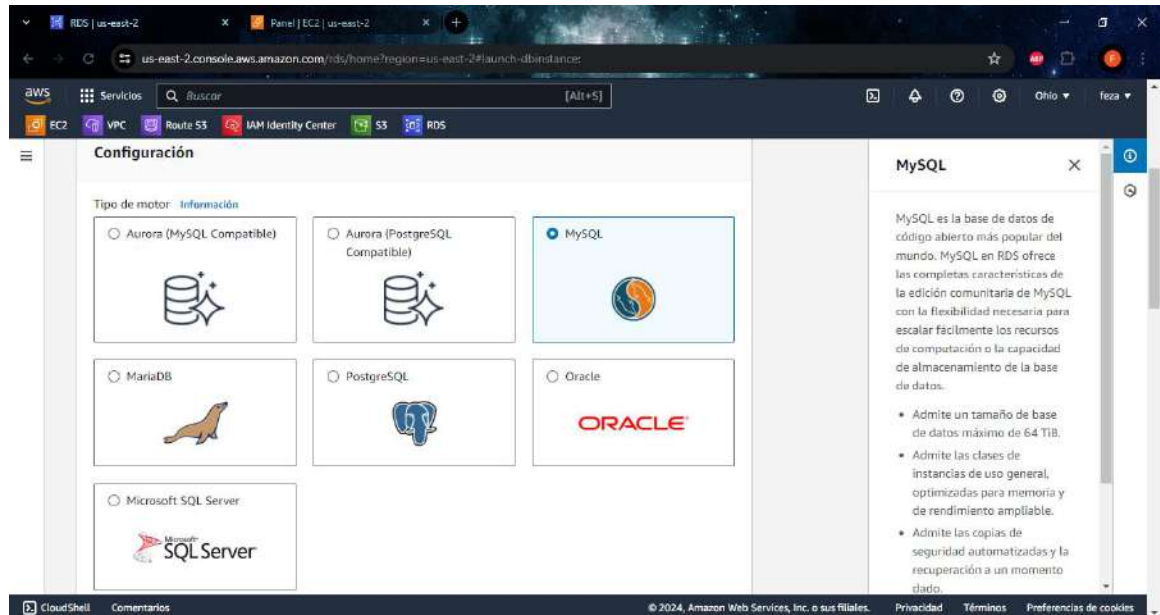
*Ilustración 50. creamos DBR*

Elegimos que tipo de creación queremos implementar para nuestra base de datos



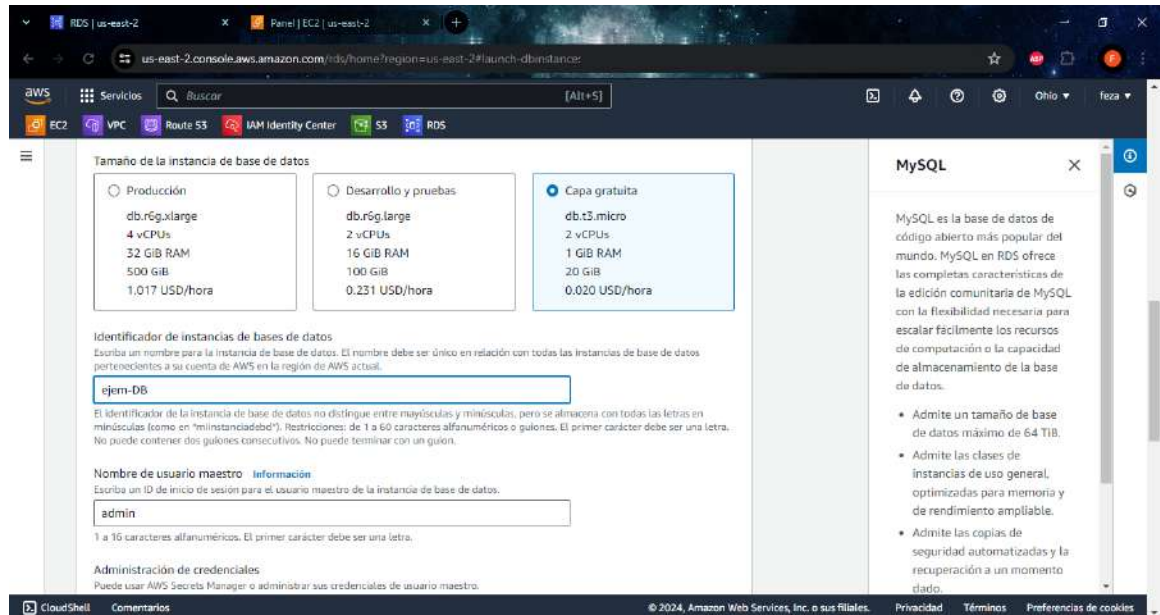
*Ilustración 51. metodo de creación*

Y nuestro motor de base de datos el cual cogemos a MYSQL



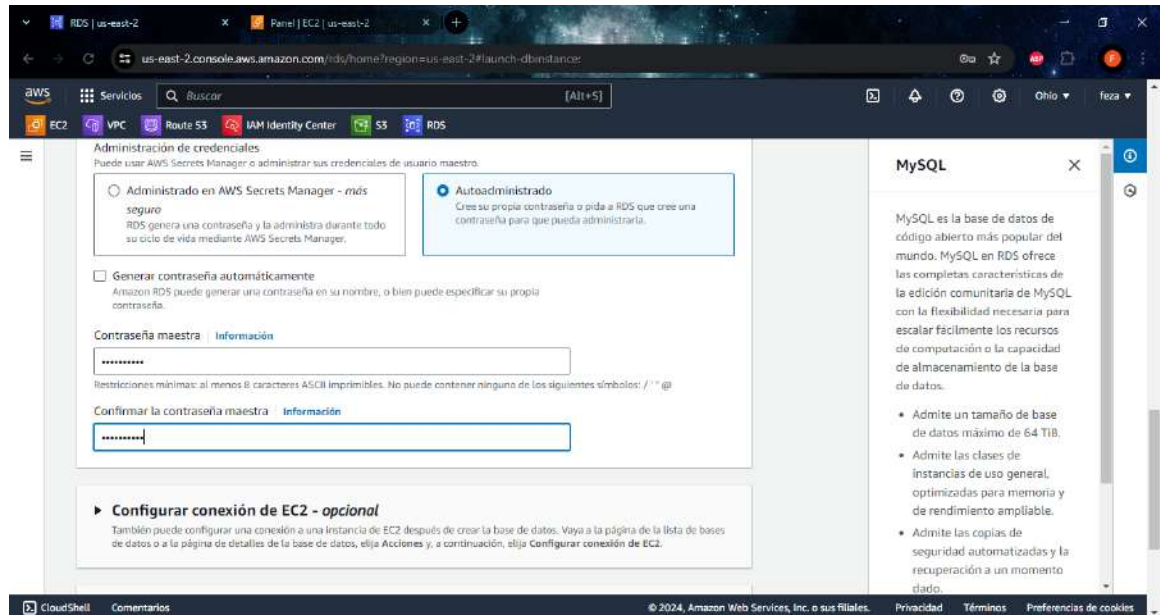
*Ilustración 52. tipo de BDR*

Cogemos el tipo de requerimientos técnicos de nuestra DB configuramos un nombre y asignamos el nombre de usuario maestro



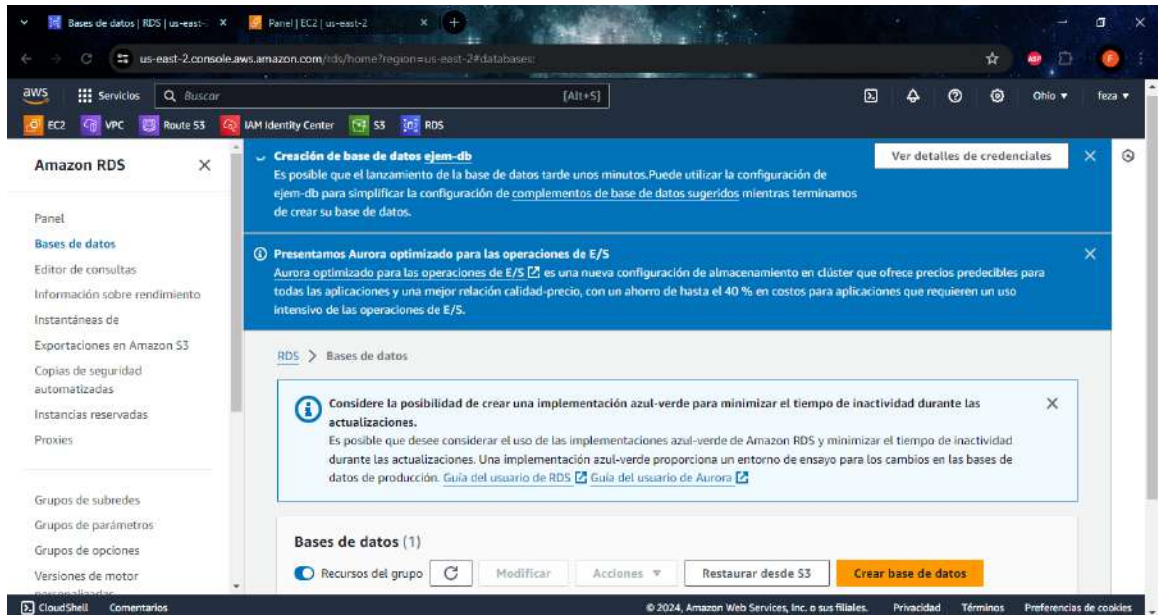
*Ilustración 53. especificaciones técnicas de DBR*

Asignamos la contraseña con la cual vamos a ingresar a nuestra DB en la opción de autoadministrado



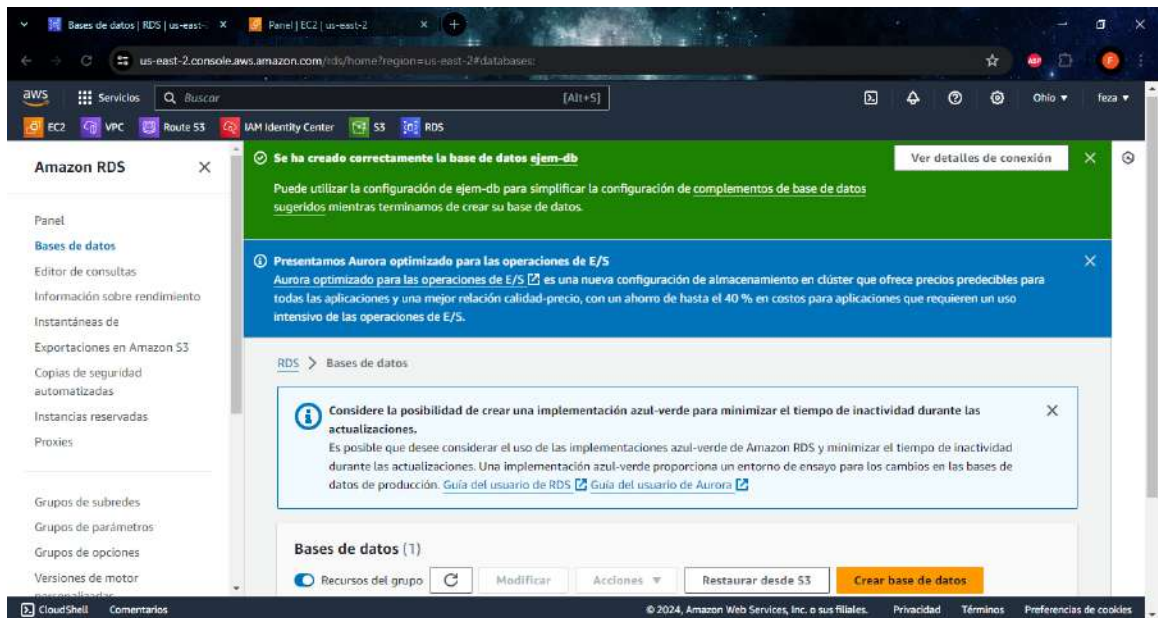
*Ilustración 54. contraseña de BDR*

Le damos en crear base de datos y esperamos a que la cree



*Ilustración 55. crear DB*

Listo ya Podemos trabajar con ella



*Ilustración 56. base de datos creada*

The screenshot displays the Amazon RDS console interface. The browser address bar shows the URL: `us-east-2.console.aws.amazon.com/rds/home?region=us-east-2&databaseid=ejem-db&cluster=false`. The page title is "Amazon RDS" and the instance name is "ejem-db".

**Resumen**

Identificador de base de datos	Estado	Rol	Motor	Recomendaciones
ejem-db	Disponible	Instancia	MySQL Community	
CPU: 3.81%	Clase: db.t3.micro	Actividad actual: 0	Región y AZ: us-east-2b	
		Conexiones		

Navigation tabs: [Conectividad y seguridad](#) | [Supervisión](#) | [Registros y eventos](#) | [Configuración](#) | [Integraciones sin extracción, trans](#)

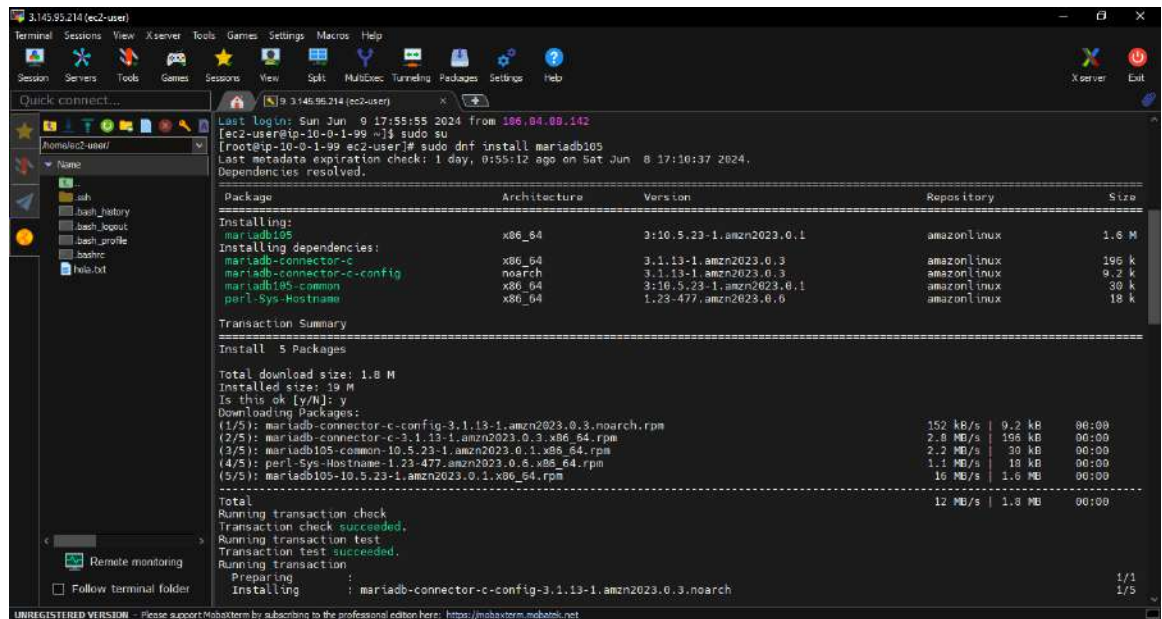
**Conectividad y seguridad**

Punto de enlace y puerto	Redes	Seguridad

Footer: © 2024, Amazon Web Services, Inc. o sus filiales. [Privacidad](#) | [Términos](#) | [Preferencias de cookies](#)

Ilustración 57. resumen DB

Instalamos el paquete de mysql con el comando `sudo dnf install mariadb105` y esperamos a que termine de instalar



```

3.145.95.214 (ec2-user)
Terminal Sessions View Xserver Tools Games Settings Macros Help
Session Servers Tools Games Sessions View Split Multitrac Tunneling Packages Settings Help
Quick connect...
Home/ec2-user/
Name
bash
bash_history
bash_logout
bash_profile
bashrc
hvia.txt
Remote monitoring
Follow terminal folder
Last login: Sun Jun  9 17:55:55 2024 from 106.04.09.142
[ec2-user@ip-10-0-1-99 ~]$ sudo su
[root@ip-10-0-1-99 ec2-user]# sudo dnf install mariadb105
Last metadata expiration check: 1 day, 0:55:12 ago on Sat Jun  8 17:10:37 2024.
Dependencies resolved.
-----
Package                               Architecture  Version                               Repository  Size
-----
Installing:
mariadb105                             x86_64       3:10.5.23-1.amzn2023.0.1             amazonlinux 1.6 M
Installing dependencies:
mariadb-connector-c                     x86_64       3.1.13-1.amzn2023.0.3                 amazonlinux 196 k
mariadb-connector-c-config              noarch      3.1.13-1.amzn2023.0.3                 amazonlinux 9.2 k
mariadb105-common                       x86_64      3:10.5.23-1.amzn2023.0.1             amazonlinux 30 k
perl-Sys-Hostname                        x86_64      1.23-477.amzn2023.0.6                 amazonlinux 18 k
-----
Transaction Summary
-----
Install 5 Packages

Total download size: 1.8 M
Installed size: 19 M
Is this ok [y/N]: y
Downloading Packages:
(1/5): mariadb-connector-c-config-3.1.13-1.amzn2023.0.3.noarch.rpm
(2/5): mariadb-connector-c-3.1.13-1.amzn2023.0.3.x86_64.rpm
(3/5): mariadb105-common-10.5.23-1.amzn2023.0.1.x86_64.rpm
(4/5): perl-Sys-Hostname-1.23-477.amzn2023.0.6.x86_64.rpm
(5/5): mariadb105-10.5.23-1.amzn2023.0.1.x86_64.rpm
-----
Total
-----
12 MB/s | 1.8 MB  00:00
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
Preparing      : mariadb-connector-c-config-3.1.13-1.amzn2023.0.3.noarch
Installing    : mariadb-connector-c-config-3.1.13-1.amzn2023.0.3.noarch
-----
1/1
1/5
UNREGISTERED VERSION - Please support Mobaxterm by subscribing to the professional edition here: https://mobaxterm.mobatek.net

```

*Ilustración 58. paquete de comandos MYSQL*

Una vez termine de instalar nos conectamos a nuestra instancia de db de nuestro Amazon y comenzamos a crear nuestra base de datos y nuestra tabla con su respectiva información

```

3.145.95.214 (ec2-user)
Terminal Sessions View Xserver Tools Games Settings Macros Help
Session Servers Tools Games Sessions View Split Multitab Tunneling Packages Settings Help
Quick connect...
/home/ec2-user/
Name
..
.csh
.csh_history
.csh_logout
.csh_profile
.cshrc
file.txt
WARNING:
A newer release of "Amazon Linux" is available.
Available Versions:
Version 2023.4.20240528:
Run the following command to upgrade to 2023.4.20240528:
dnf upgrade --releasever=2023.4.20240528
Release notes:
https://docs.aws.amazon.com/linux/al2023/release-notes/relnotes-2023.4.20240528.html
-----
Installed:
mariadb-connector-c-3.1.13-1.amzn2023.0.3.x86_64      mariadb-connector-c-config-3.1.13-1.amzn2023.0.3.noarch
mariadb105-3:10.5.23-1.amzn2023.0.1.x86_64          mariadb105-common-3:10.5.23-1.amzn2023.0.1.x86_64
perl-Sys-Hostname-1.23-477.amzn2023.0.6.x86_64
Complete!
[root@ip-10-0-1-99 ec2-user]# mysql -h database-1.cfwb9042veee.us-east-2.rds.amazonaws.com -u admin -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MySQL connection id is 30
Server version: 0.0.35 source distribution
Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.
MySQL [(none)]> CREATE DATABASE ejemplo;
Query OK, 1 row affected (0.007 sec)
UNREGISTERED VERSION - Please support Mobaxterm by subscribing to the professional edition here: https://mobaxterm.mobatec.net

```

*Ilustración 59. comando database DB*

Ya creamos nuestra base de datos con `CREATE DATABASE` procedemos a crear nuestra tabla con `CREATE TABLE` mostramos nuestra tabla con `DESCRIBE` e insertamos datos con `INSERT INTO VALUES` y los datos correspondientes a nuestra table a llenar cabe resaltar que hay que identificar el tipo de tabla que se va a llenar en mi caso la tabla se llama datos y ya por último mostramos los datos de nuestra tabla con `SELECT * FROM` y ya tenemos los datos de nuestra DB ejemplo y la table datos

```

3.145.95.214 (ec2-user)
Terminal Sessions View Xserver Tools Games Settings Macros Help
Session Servers Tools Games Sessions View Split Multitab Tunneling Packages Settings Help
Quick connect...
/home/ec2-user/
Name
..
.ssh
.ssh_history
.ssh_logout
.ssh_profile
.sshrc
.rviz.txt
Remote monitoring
Follow terminal folder

ntax to use near 'DATABASE' at line 1
MySQL [(none)]> SHOW database;
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right sy
ntax to use near 'database' at line 1
MySQL [(none)]> USE ejemplo;
Database changed
MySQL [ejemplo]> CREATE TABLE datos (nombre VARCHAR(30), edad INT);
Query OK, 0 rows affected (0.027 sec)

MySQL [ejemplo]> DESCRIBE datos;
+-----+
| Field | Type          | Null | Key | Default | Extra |
+-----+
| nombre | varchar(30)  | YES  |     | NULL    |      |
| edad   | int          | YES  |     | NULL    |      |
+-----+
2 rows in set (0.003 sec)

MySQL [ejemplo]> INSERT INTO datos VALUES ('Jose Sanchez', 22);
Query OK, 1 row affected (0.005 sec)

MySQL [ejemplo]> INSERT INTO datos VALUES ('Maria Perez', 18);
Query OK, 1 row affected (0.004 sec)

MySQL [ejemplo]> INSERT INTO datos VALUES ('Johana Gomez', 30);
Query OK, 1 row affected (0.003 sec)

MySQL [ejemplo]> SELECT * FROM datos;
+-----+
| nombre | edad |
+-----+
| Jose Sanchez | 22 |
| Maria Perez | 18 |
| Johana Gomez | 30 |
+-----+
3 rows in set (0.001 sec)

MySQL [ejemplo]>

```

UNREGISTERED VERSION - Please support Mobaxterm by subscribing to the professional edition here: <https://mobaxterm.mobatek.net>

*Ilustración 60. vista de la base de datos*

## Ejercicio 4

Creamos nuestra instancia y le damos un nombre en este caso le pondré Windows para especificar que va a ser instalado sistema operativo Windows para nuestro servidor

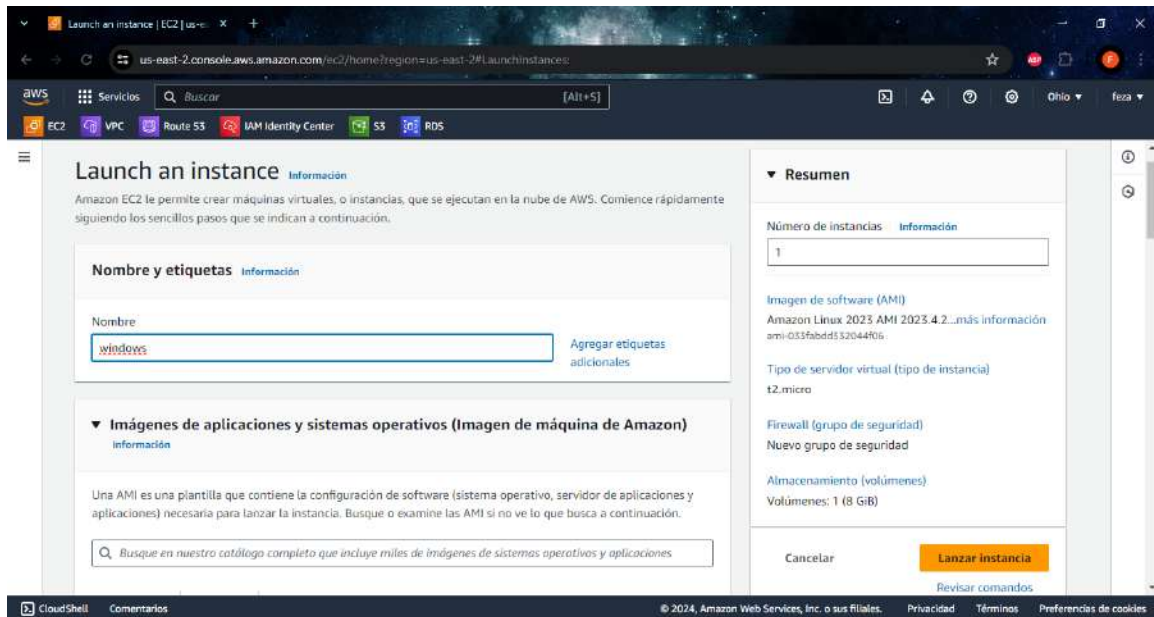
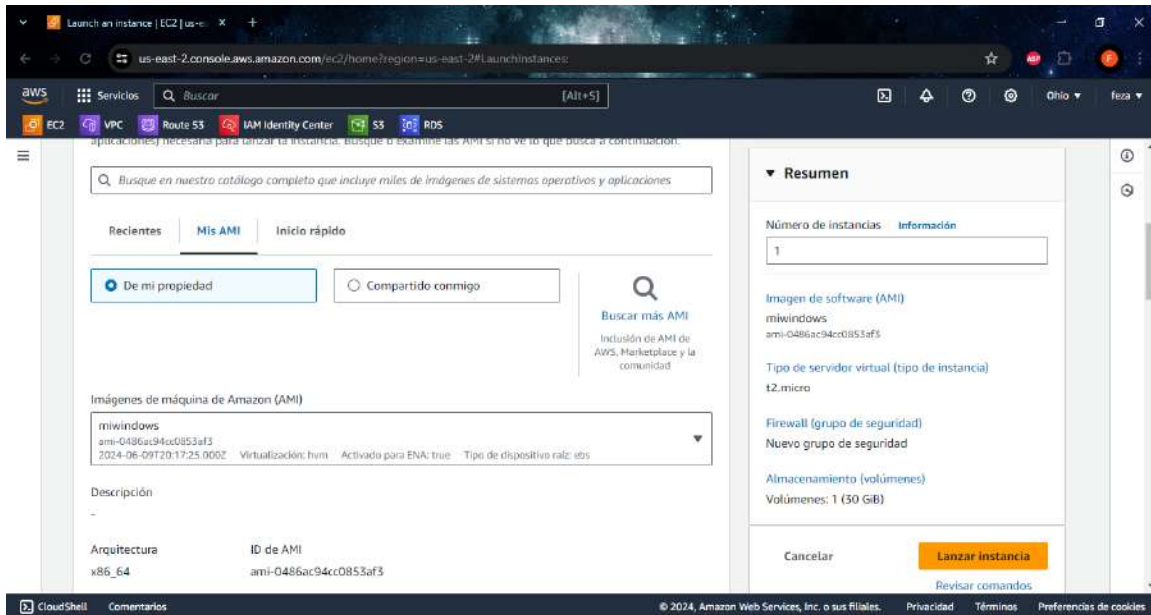


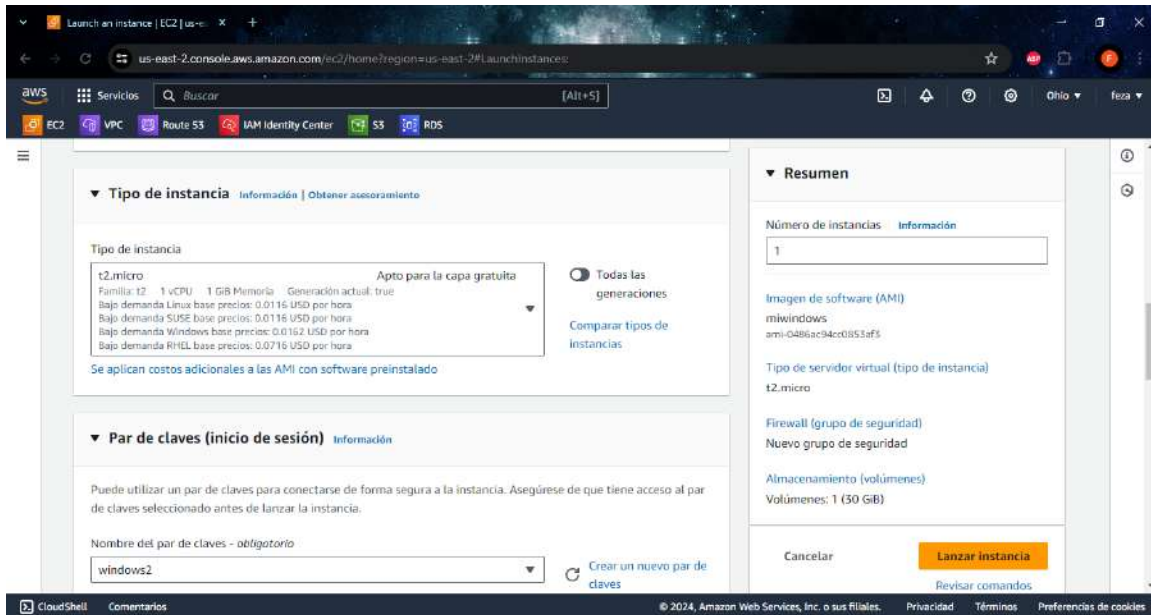
Ilustración 61. ejercicio 4 servidor wed windows

Seleccionamos el sistema operativo en mi caso como ya tenía creada un ami con unas aplicaciones ya instaladas y configurado la seleccionamos



*Ilustración 62. tipo de ami windows*

Cogemos los requerimientos de hardware que queramos instalar en nuestro servidor y asignamos el tipo de clave para ingresar en ella



*Ilustración 63. hardware de nuestra instancia windows*

En configuración de red cogemos nuestra vpc y habilitamos la ip publica y ponemos el nombre de grupo de seguridad

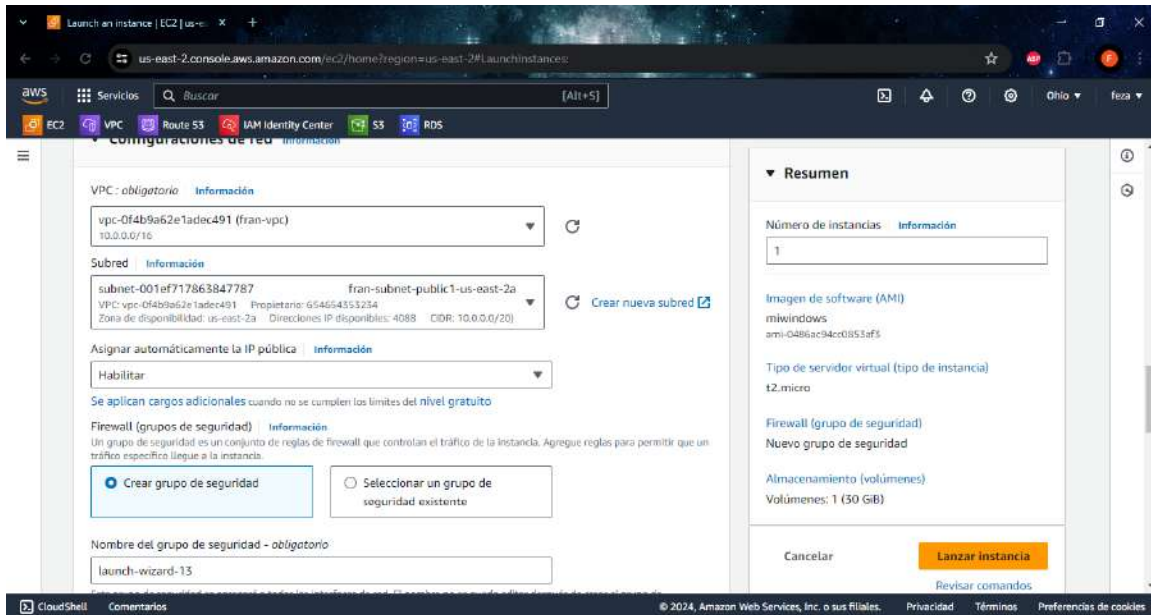


Ilustración 64. configuración de red de inst windows

Asignamos el tamaño de nuestro disco duro y le damos en lanzar instancia

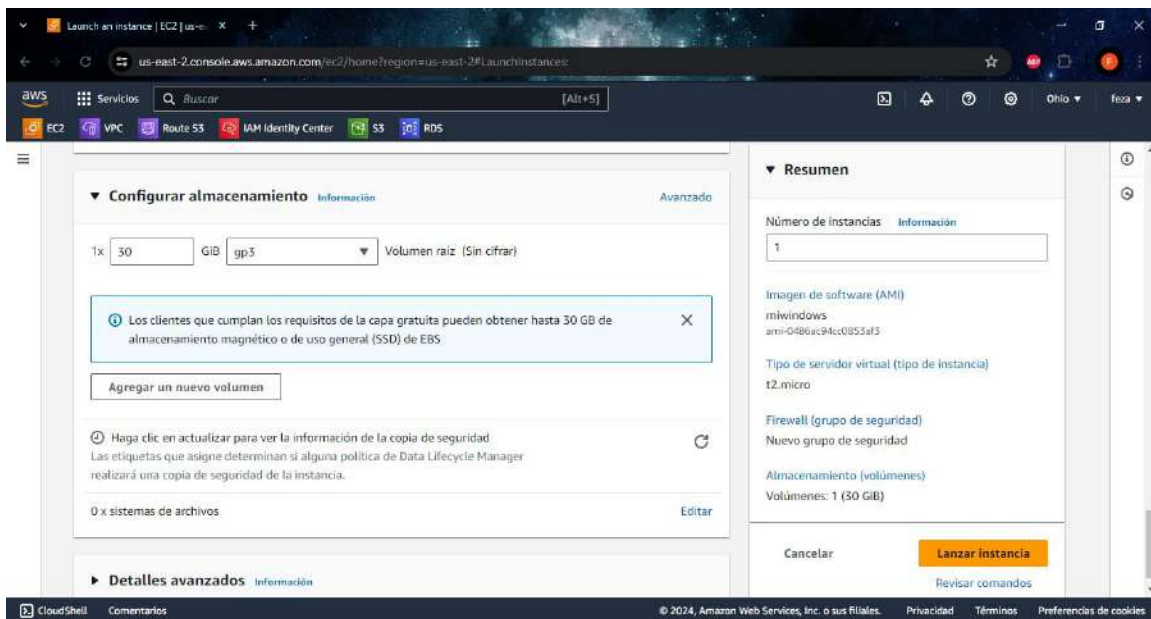
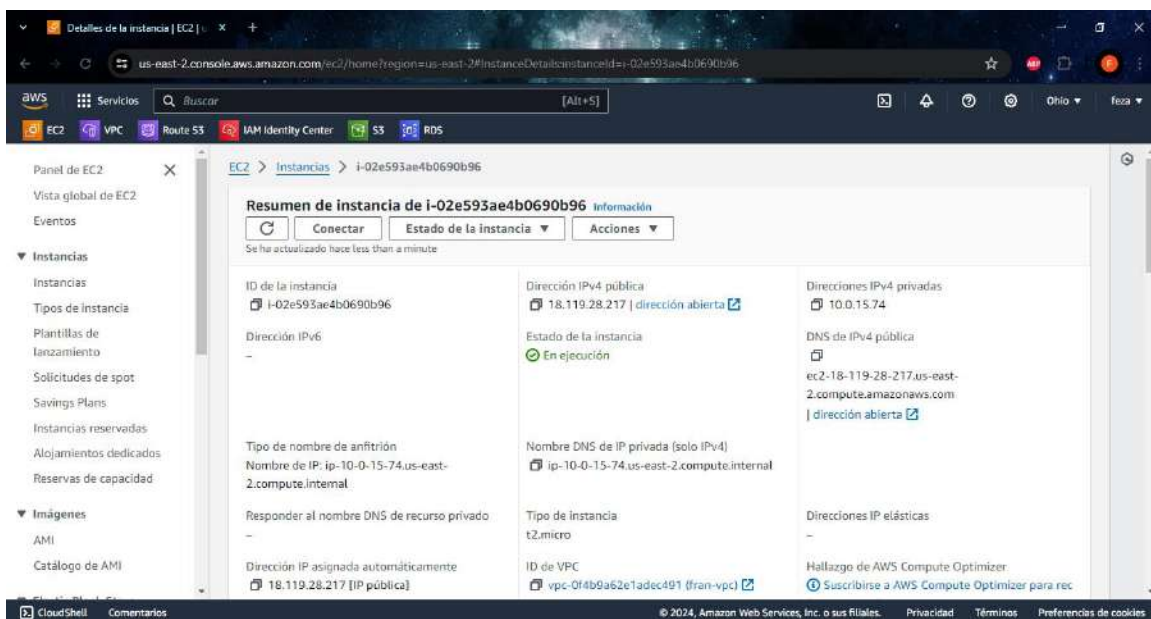
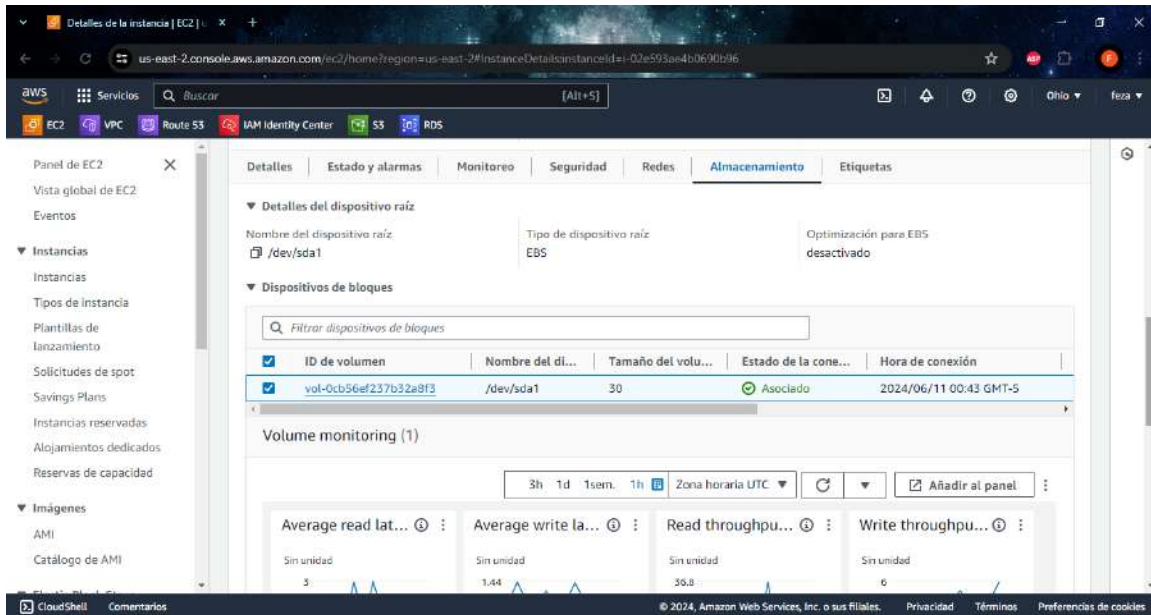


Ilustración 65. lanzar instancia windows

Ahora crearemos una imagen ami ingresamos a una instancia con la configuración que deseamos instalar en nuestras otras instancias



Y vamos a la parte inferior e ingresamos en la pestaña de almacenamiento



The screenshot displays the AWS Management Console interface for an EC2 instance. The 'Almacenamiento' (Storage) tab is selected, showing details for the root device and associated block devices. The root device is an EBS volume named '/dev/sda1' with a size of 30 GB and a state of 'Asociado' (Associated). Below this, a table lists the block devices, and a 'Volume monitoring' section is visible at the bottom.

**Detalles del dispositivo raíz**

Nombre del dispositivo raíz	Tipo de dispositivo raíz	Optimización para EBS
/dev/sda1	EBS	desactivado

**Dispositivos de bloques**

ID de volumen	Nombre del di...	Tamaño del volu...	Estado de la cone...	Hora de conexión
vol-0cb56ef237b32a8f3	/dev/sda1	30	Asociado	2024/06/11 00:43 GMT-5

**Volume monitoring (1)**

Average read lat...	Average write la...	Read throughpu...	Write throughpu...
Sin unidad	Sin unidad	Sin unidad	Sin unidad
3	1.44	35.3	6

*Ilustración 66. pestaña almacenamiento*

Le damos clic en él y lo seleccionamos y en la parte superior derecha le damos en crear instantánea

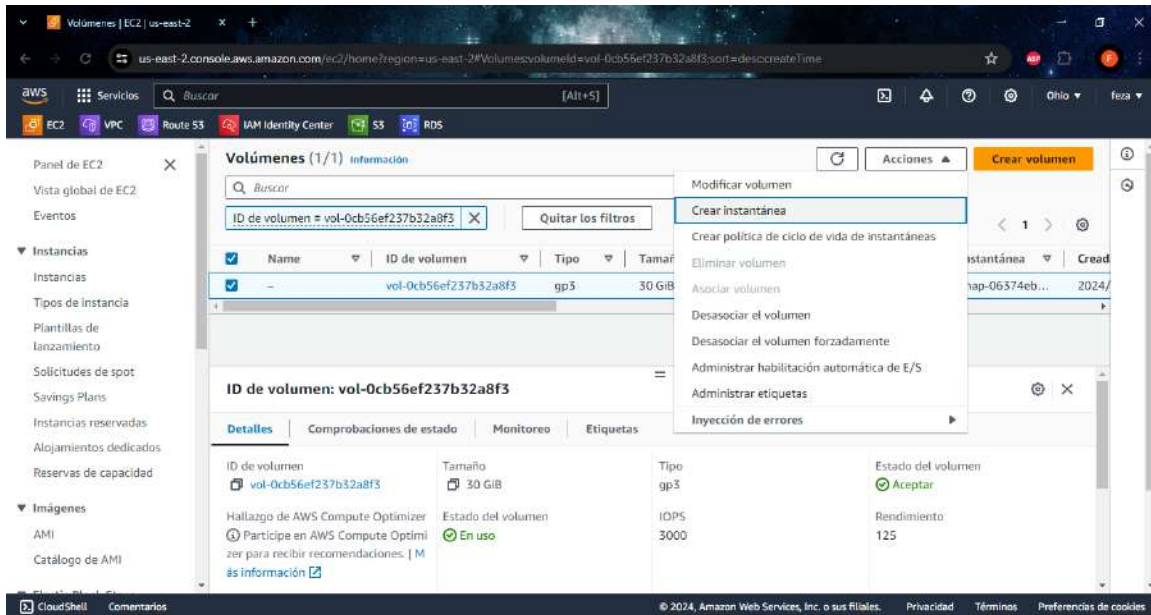


Ilustración 67. crear instantánea

Asignamos un nombre y le damos en crear

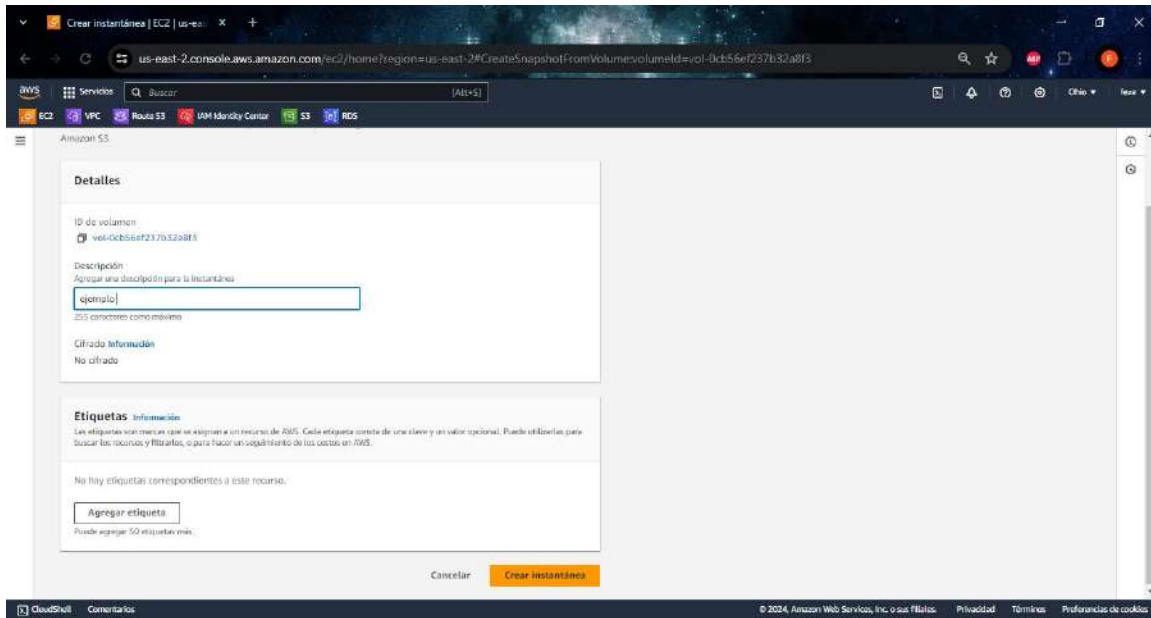


Ilustración 68. crear instantánea

Y esperamos a que esté listo para usar la instantánea que acabamos de crear

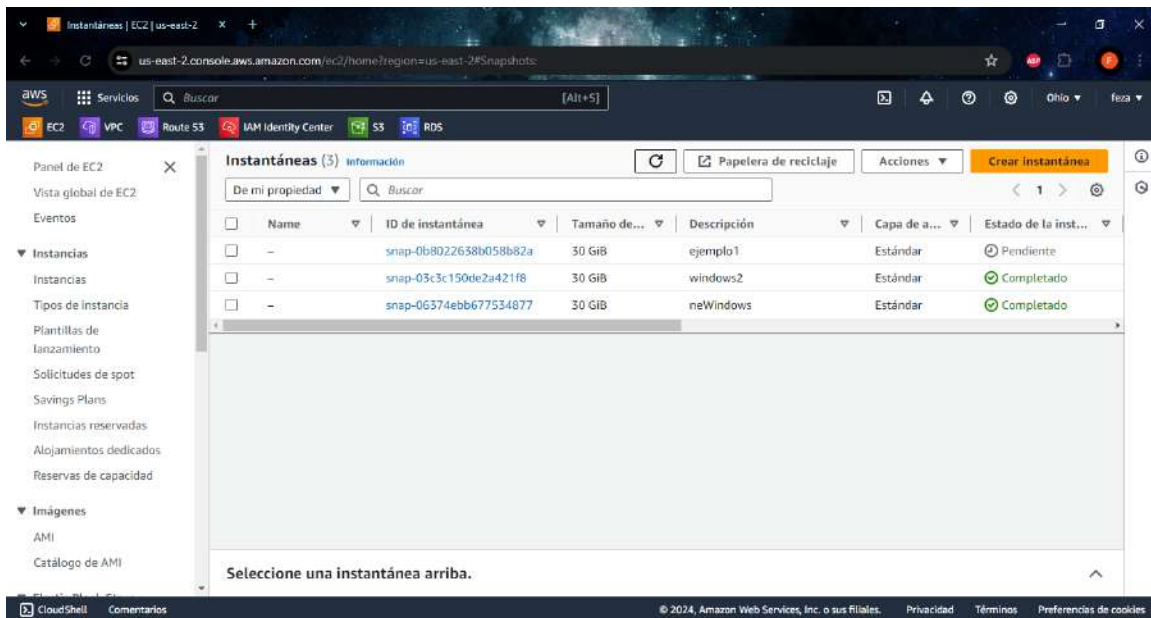


Ilustración 69. creando instantánea

La seleccionamos y le damos en crear imagen a partir de una instantánea

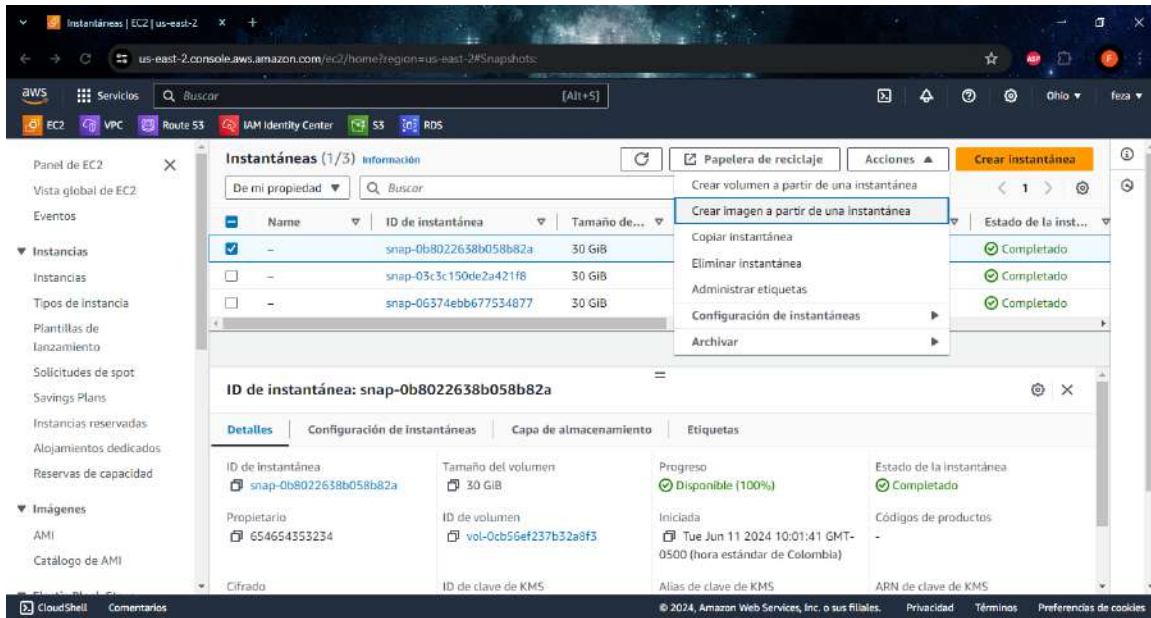


Ilustración 70. crear imagen

Asignamos un nombre

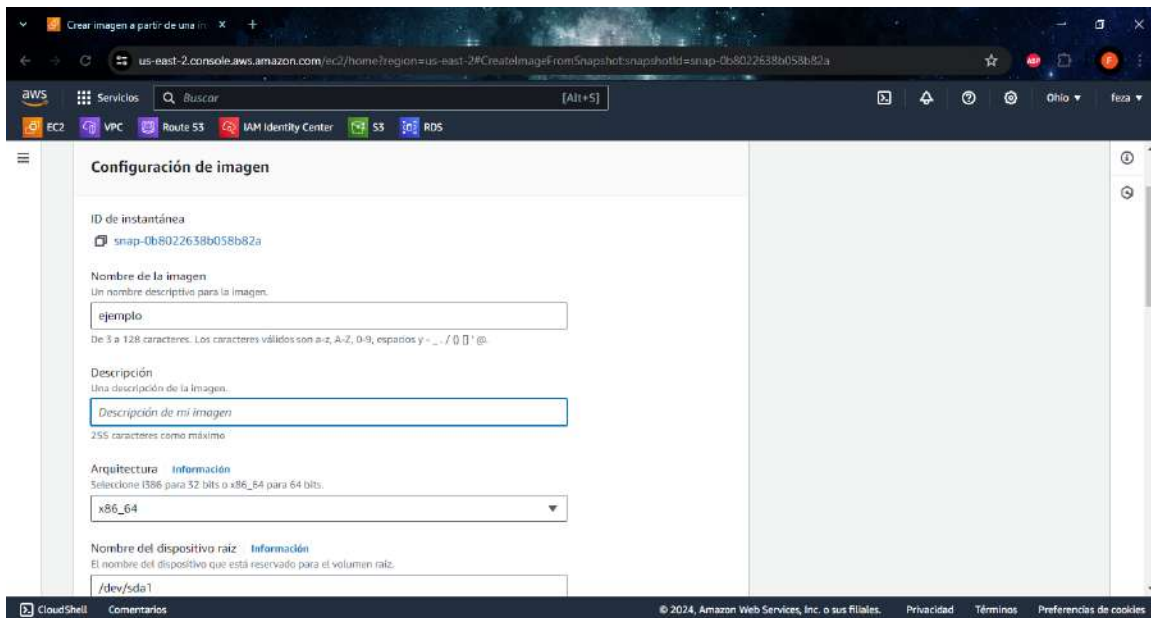


Ilustración 71. configuración de imagen

Y también el valor de la capacidad del disco duro y le damos en crear imagen cabe resaltar que como ya tiene datos con respecto a la instancia que modificamos no tenemos que modificar nada al menos que quieras cambiar algún valor específico

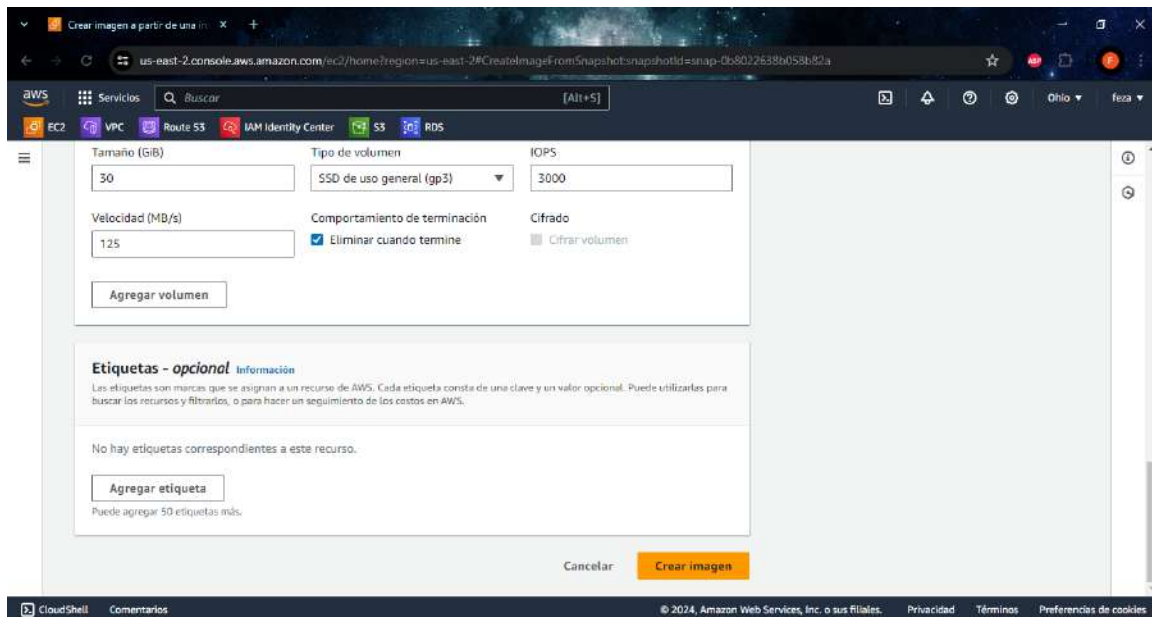
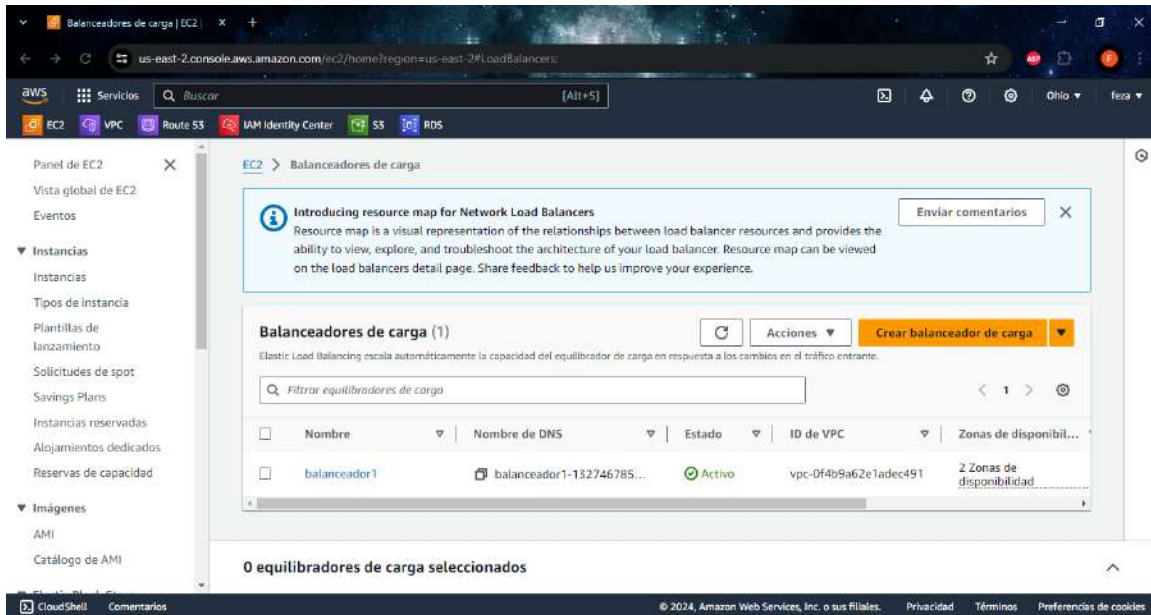


Ilustración 72. crear imagen

Una vez ya creado la imagen procederemos a crear nuestro balanceador de carga que es el que encargará de equilibrar las peticiones de nuestro servidor web



*Ilustración 73. crear balanceador de carga*

Seleccionamos el que es balanceador de carga de aplicaciones dándole en crear que está en la parte inferior izquierda



Ilustración 74. tipos de equilibradores de carga

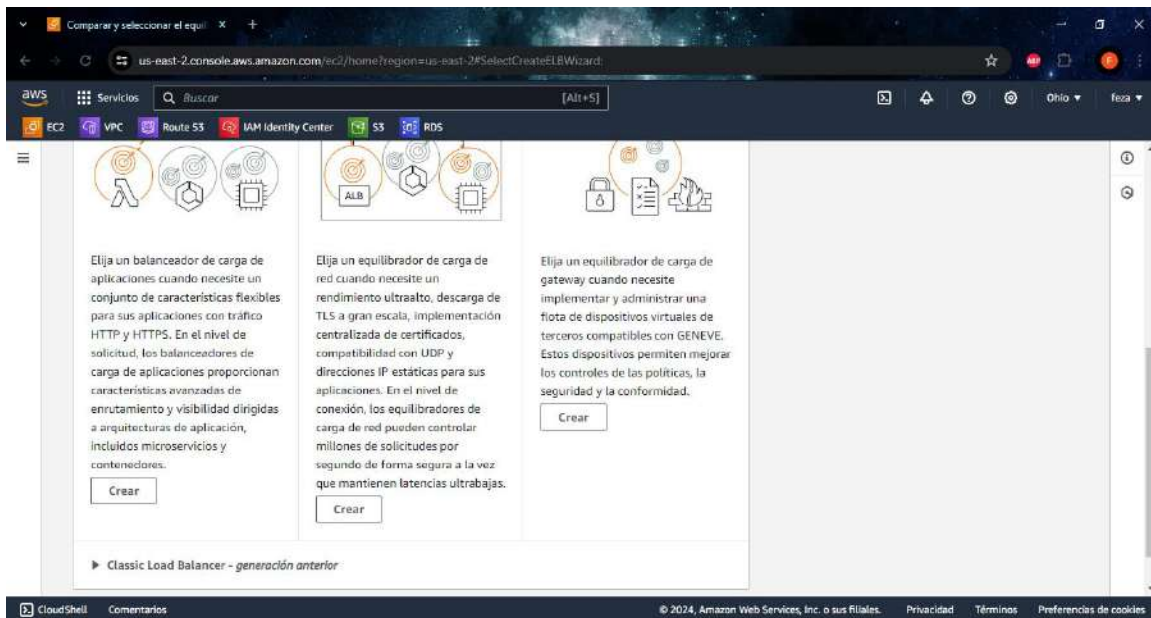
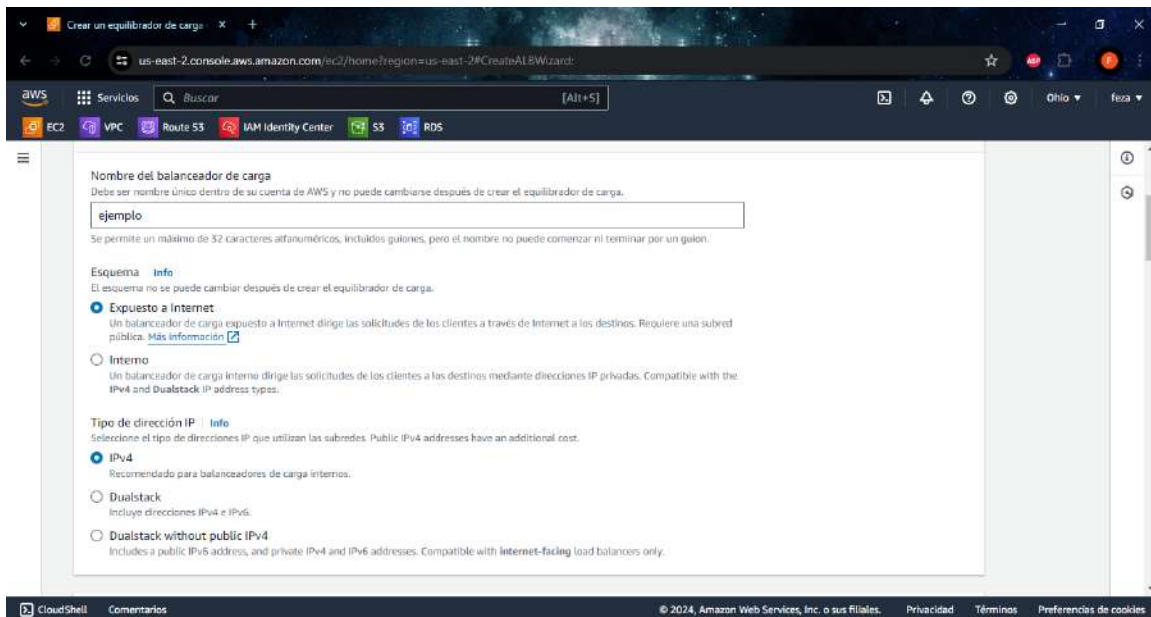


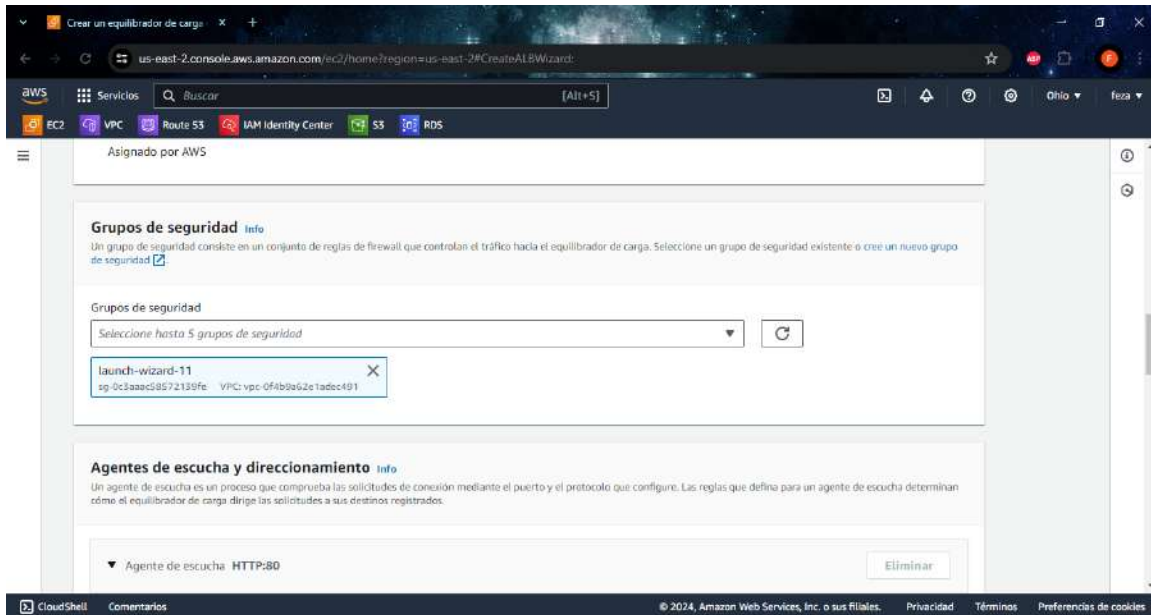
Ilustración 75. crear

Asignamos un nombre y seleccionamos nuestra vpc junto las subredes que deseamos asignarles



*Ilustración 76. configuraciones de balanceador de carga*

Seleccionamos nuestro grupo de trabajo cabe resaltar que tiene que ser el mismo que se le asigno a nuestras instancias antes creadas



*Ilustración 77. grupo de seguridad balanceador de carga*

## Creamos un grupo de destino para dirigir las solicitudes al grupo especificado

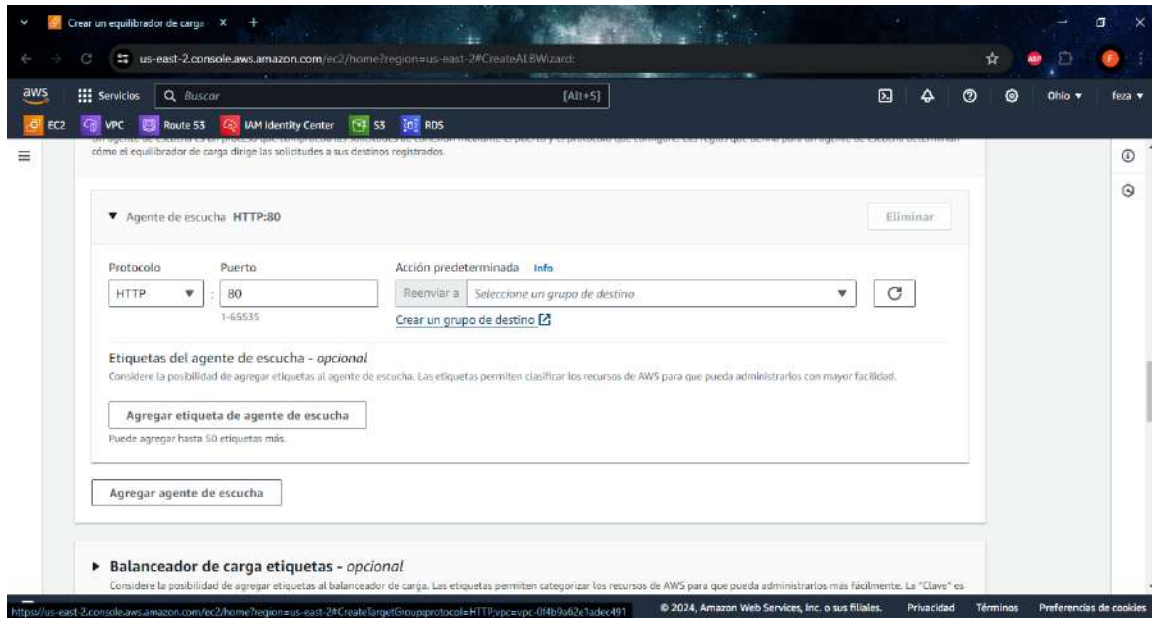
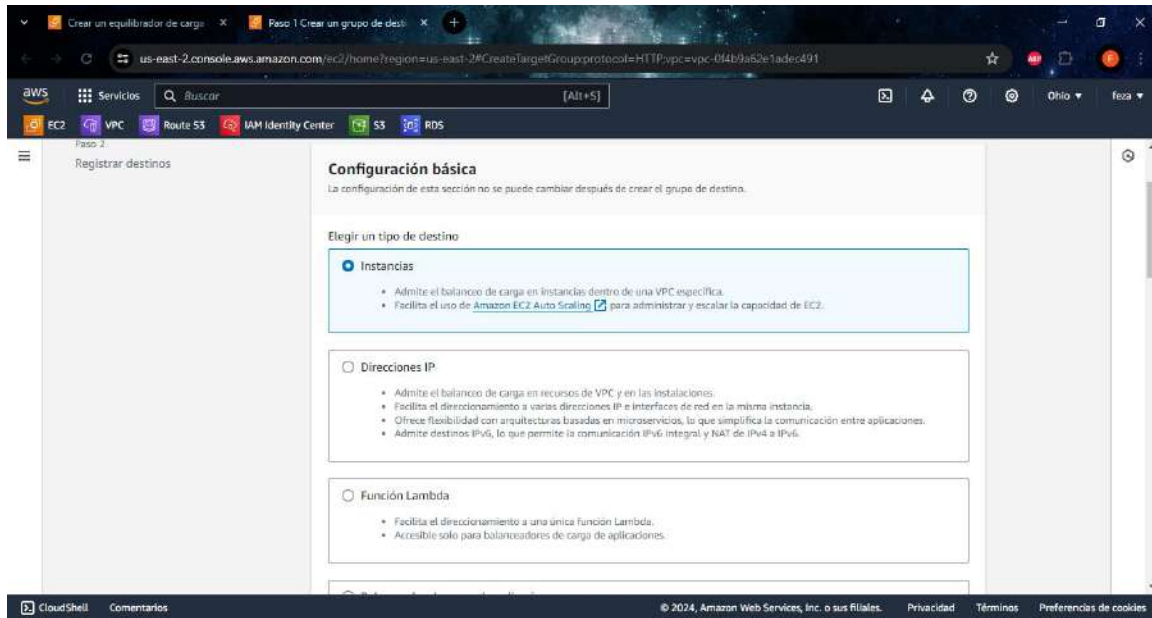


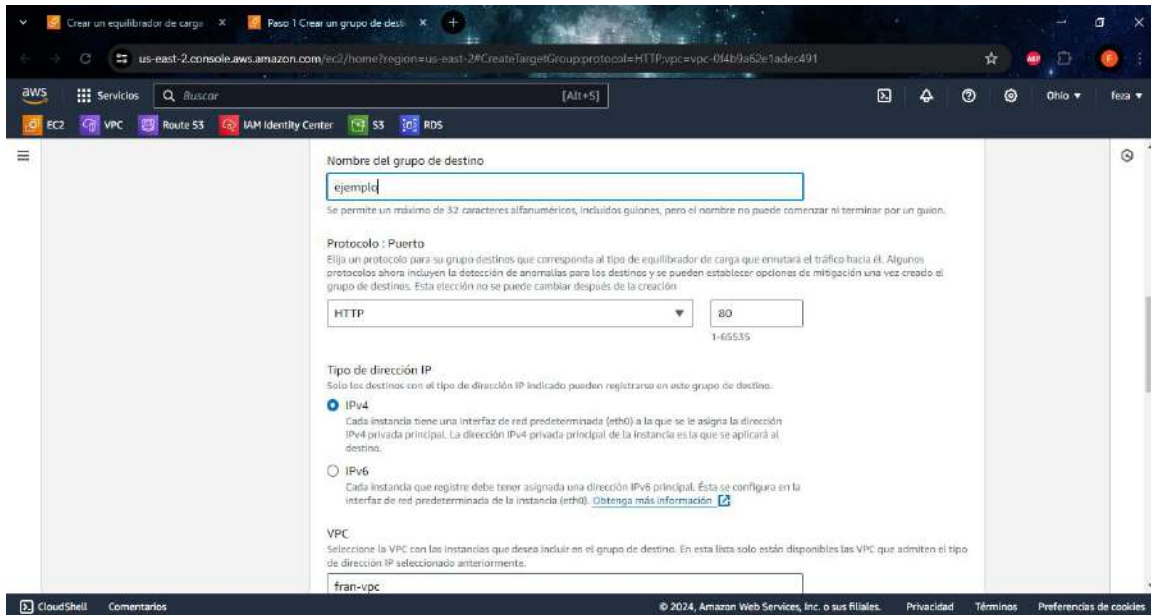
Ilustración 78. crear grupo de destino

Elegimos el tipo de destino en este caso es por instancias

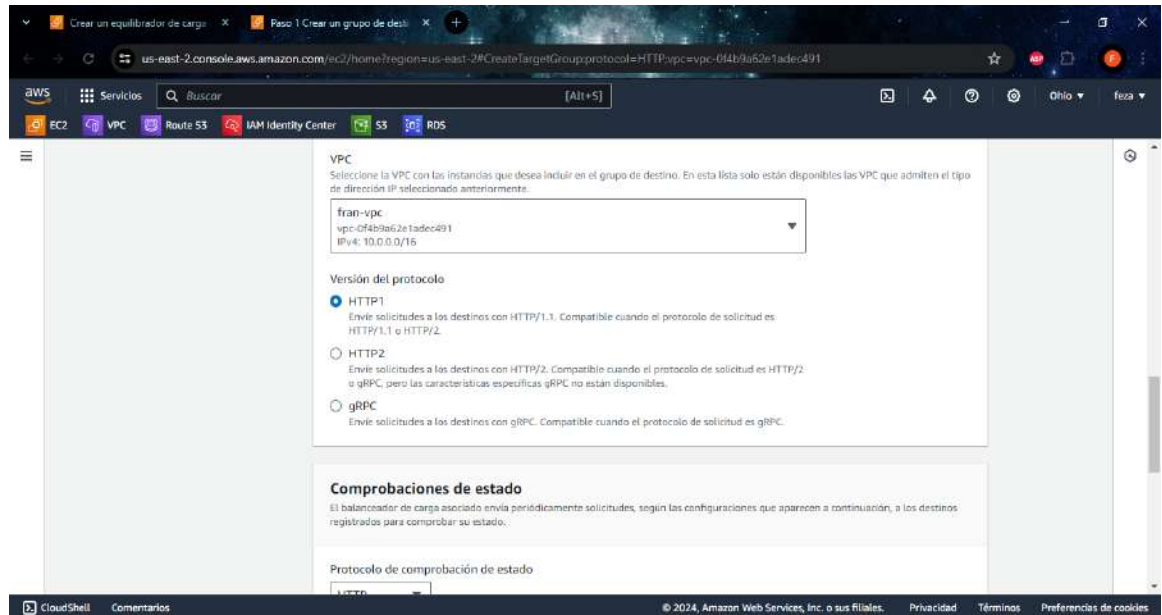


*Ilustración 79. configuración básica*

Asignamos un nombre con el cual vamos a identificar y seleccionamos nuestra vpc y le damos siguiente

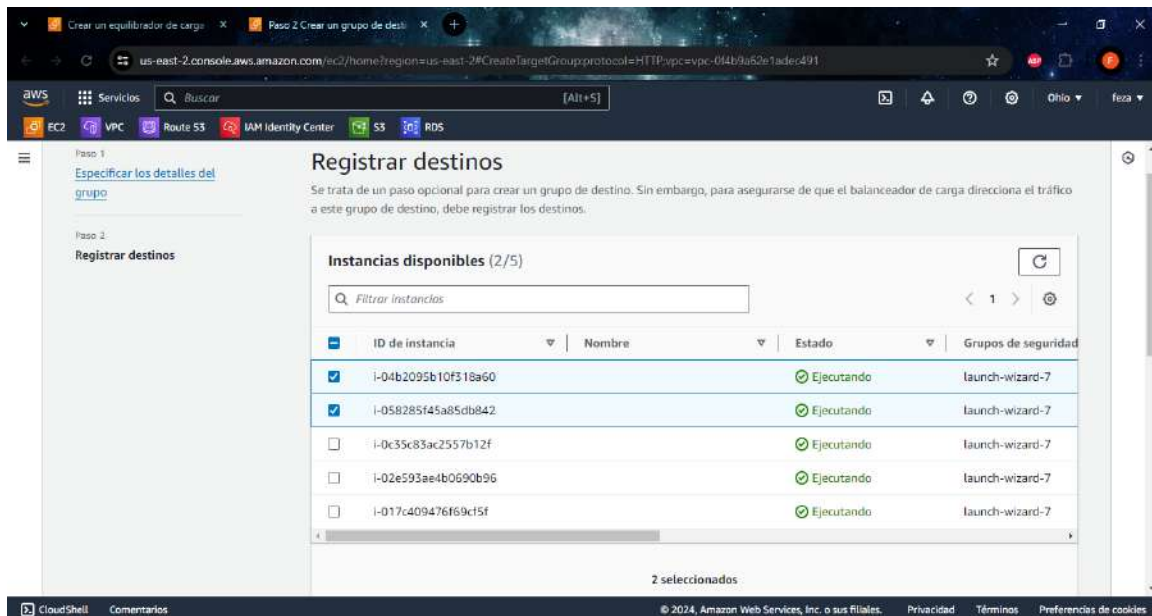


*Ilustración 80. nombre del grupo de destino*



*Ilustración 81. selección de la vpc para el grupo de trabajo*

Seleccionamos nuestras instancias para el balanceador de carga le damos en incluir y le damos en crear grupo de destino



*Ilustración 82. selección de las instancias*

Regresamos a donde estamos creando el balanceador y actualizamos el grupo de destino y hay ya nos aparece el destino creado lo seleccionamos y creamos nuestro balanceador de carga

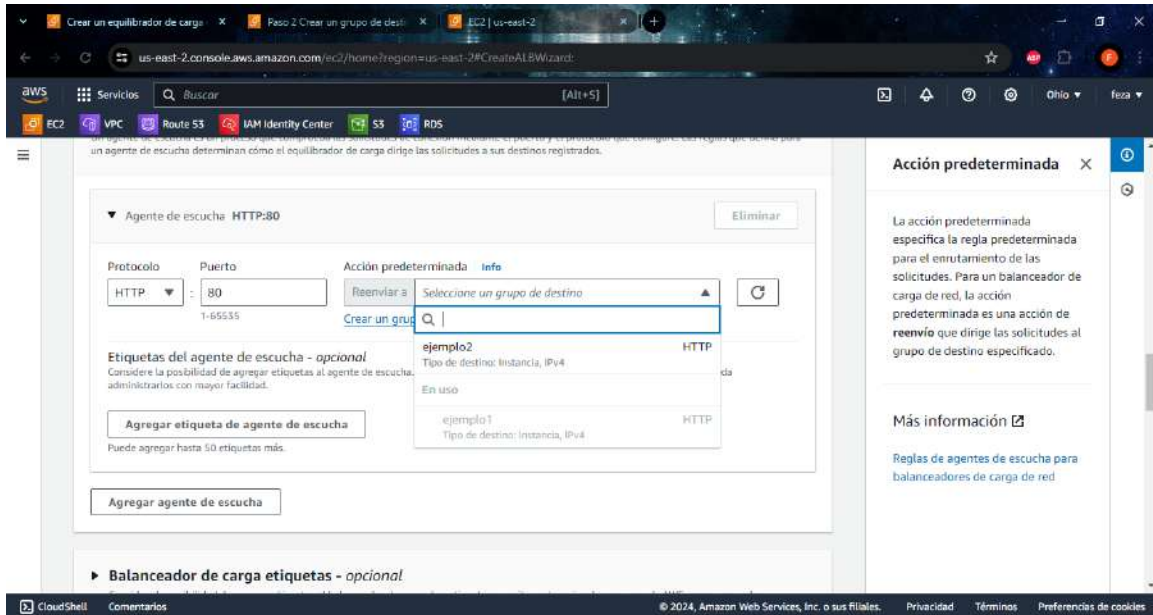
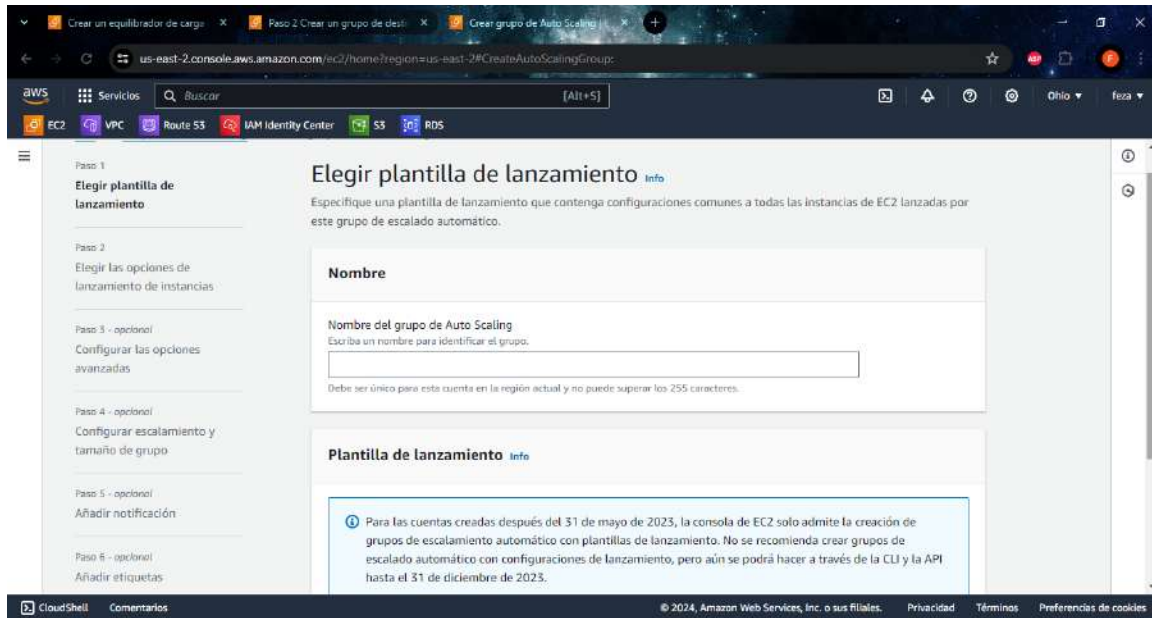


Ilustración 83. creamos balanceador de carga

Ahora crearemos un grupo de auto escalado asignamos nombre



*Ilustración 84. grupo de auto escalado*

## Y creamos una configuración de lanzamiento

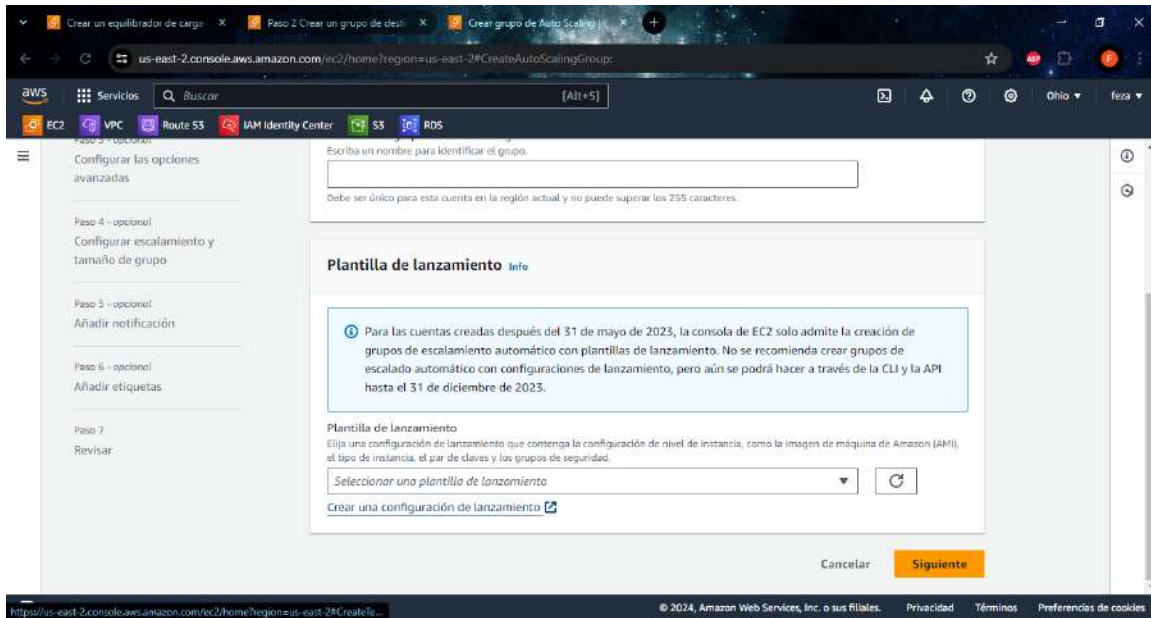


Ilustración 85. configuración de lanzamiento

## Le asignamos un nombre

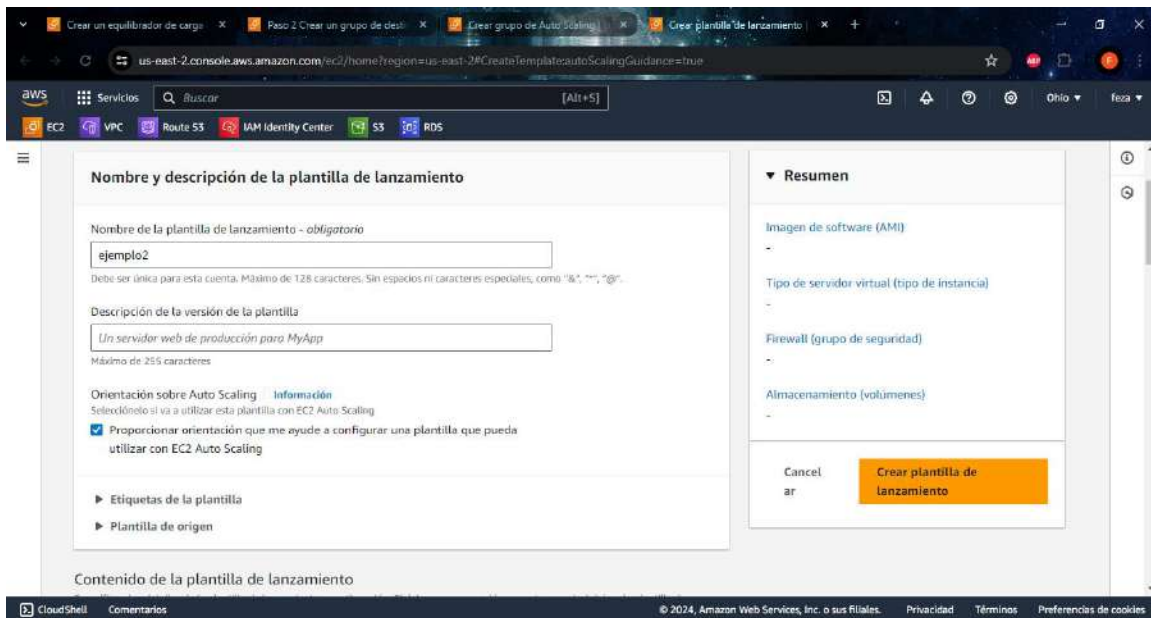


Ilustración 86. nombre del lanzamiento

Seleccionamos nuestra ami ya anterior mente creada

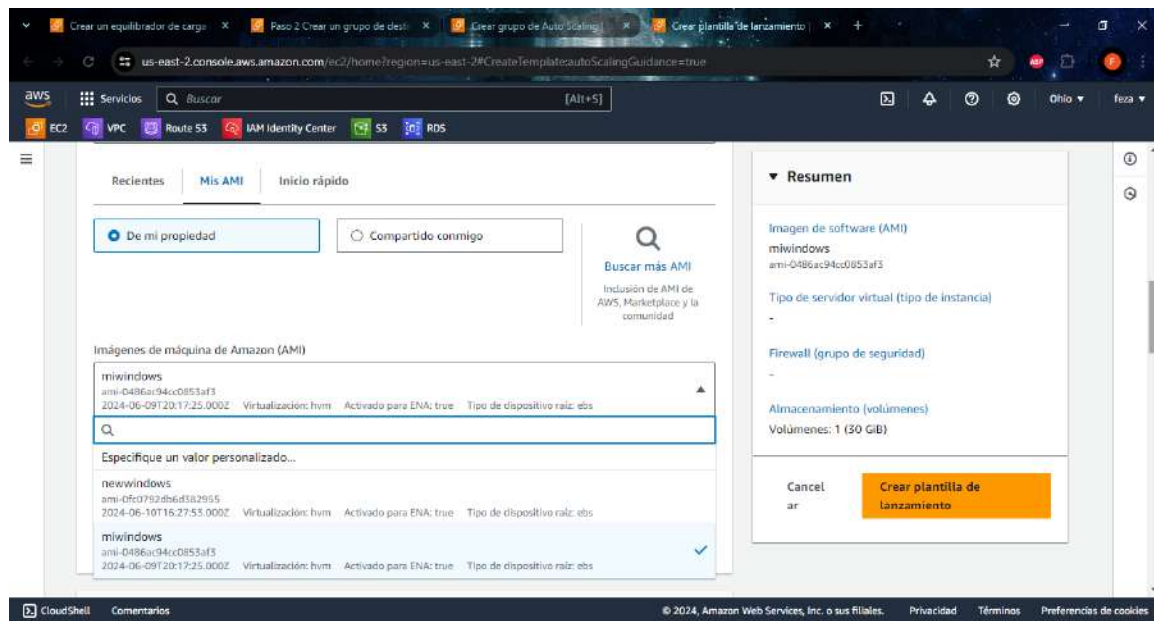


Ilustración 87. seleccionamos nuestra ami

Asignamos nuestra arquitectura para las instancias que se crearan y el tipo de seguridad para las claves de inicio de sección

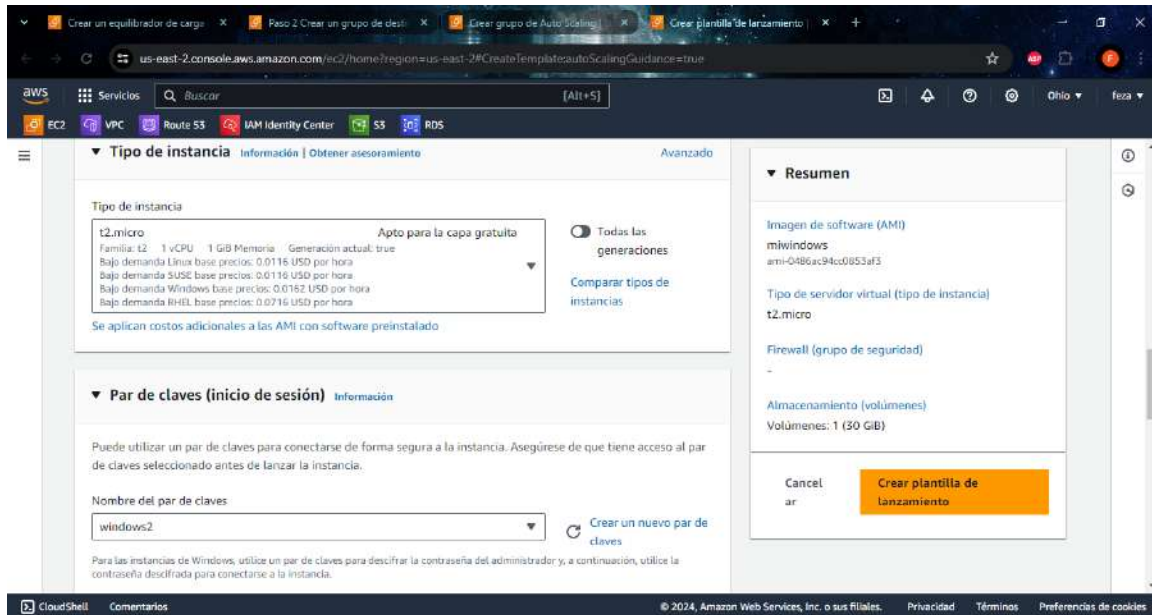
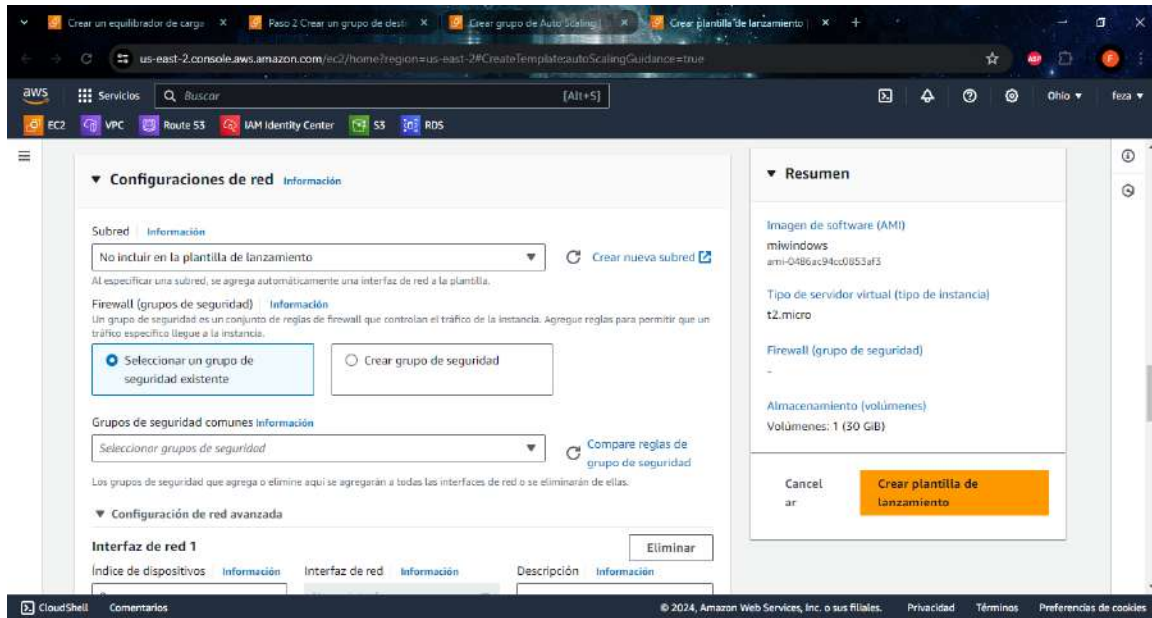


Ilustración 88. claves de inicio de sección

Seleccionamos esta opción para nuestra subred



*Ilustración 89. subred*

Asignamos ip publica para poder conectarnos a nuestras instancias creadas es muy importante habilitarla por si no se habilita no se podrá conectar seleccionamos

también nuestro grupo de trabajo y creamos plantilla

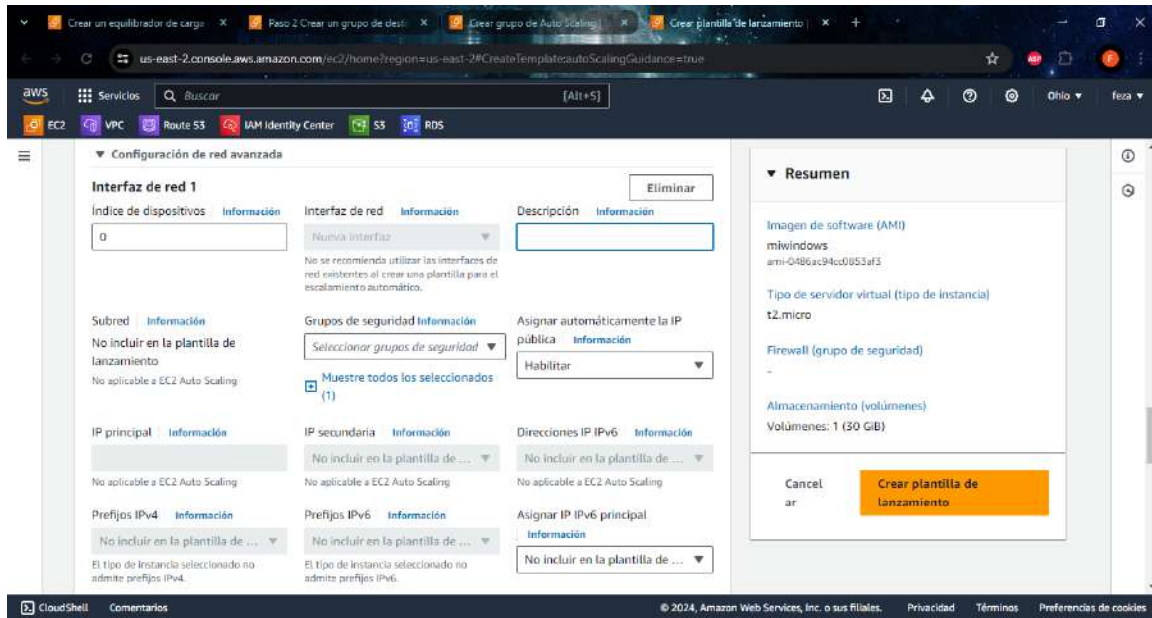


Ilustración 90. configuraciones de red y seguridad

Ya nos aparece nuestro lanzamiento lo seleccionamos y le damos en siguiente

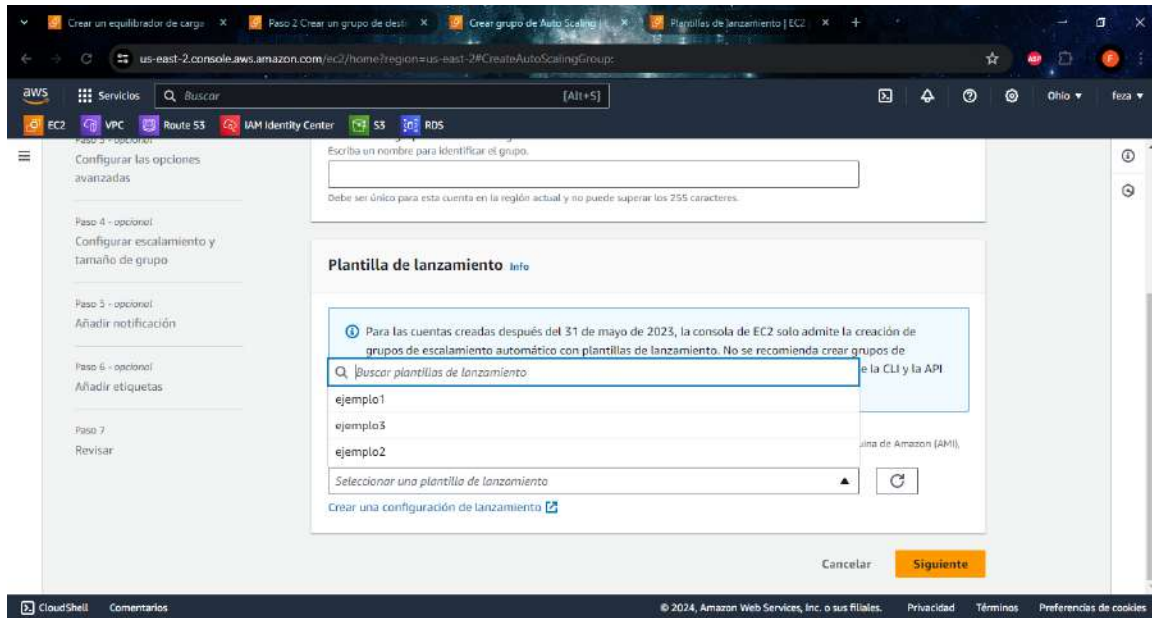


Ilustración 91. vista de lanzamiento

Seleccionamos nuestra vpc y nuestras ip públicas y le damos siguiente

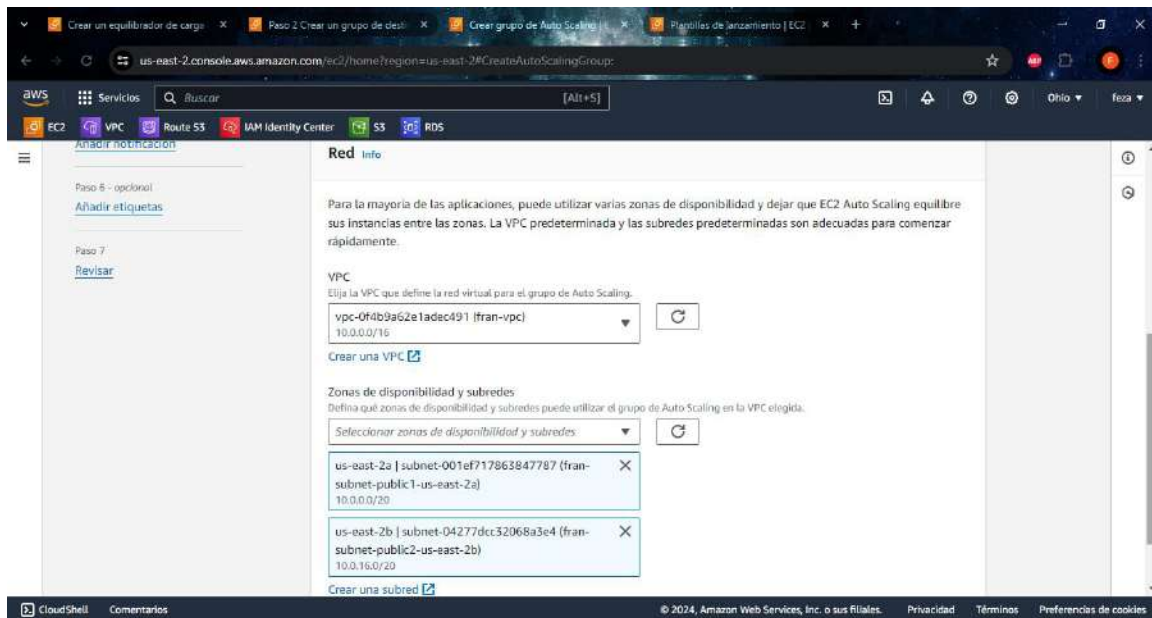
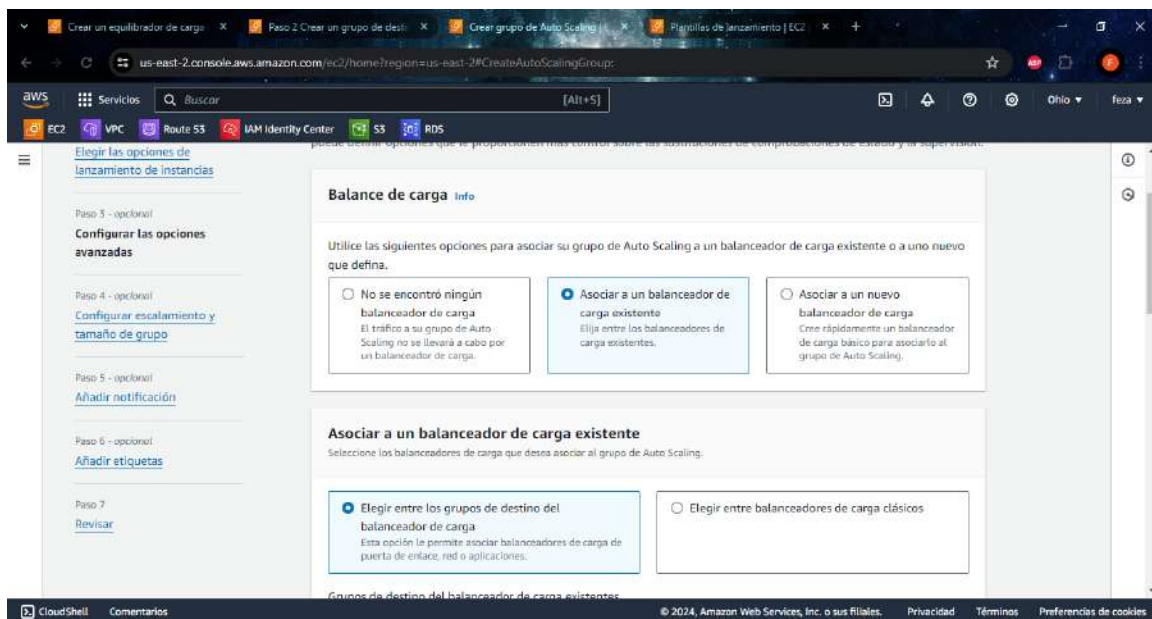
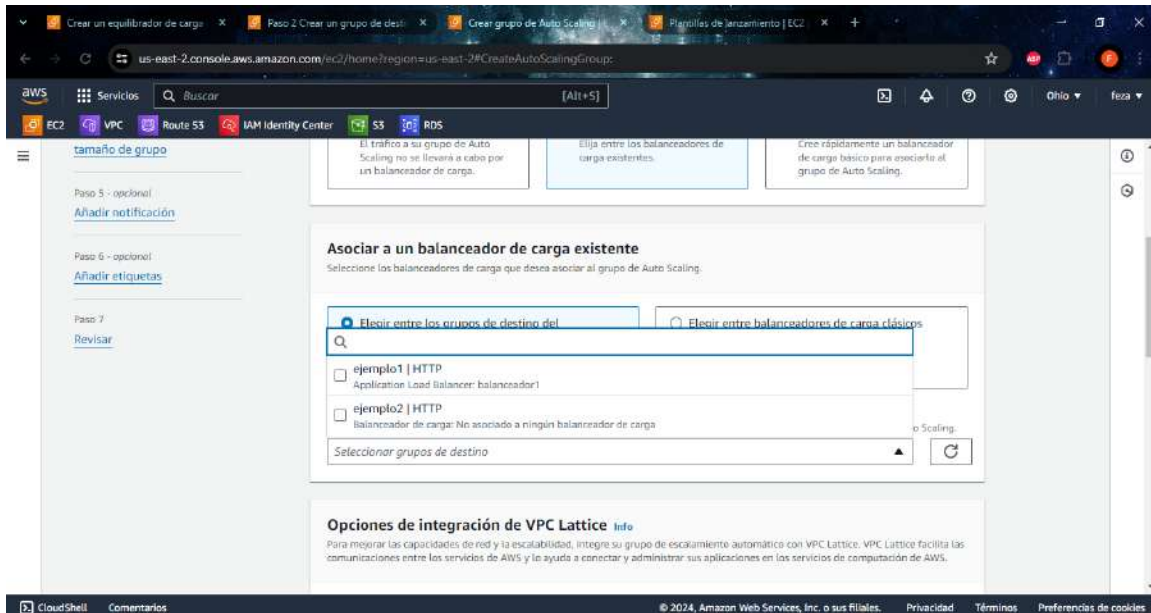


Ilustración 92. selección de vpc e ip públicas

Asociamos un balanceador de carga ya existente y seleccionamos el que se creó anteriormente y le damos en siguiente

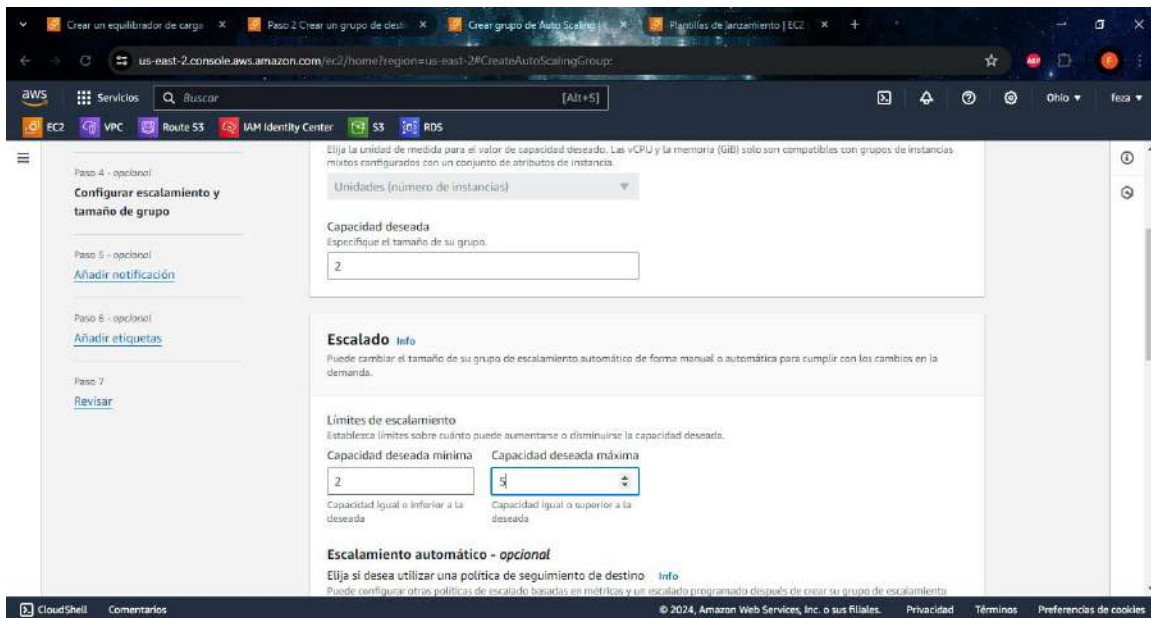


*Ilustración 93. asignar balacedor de cargar*



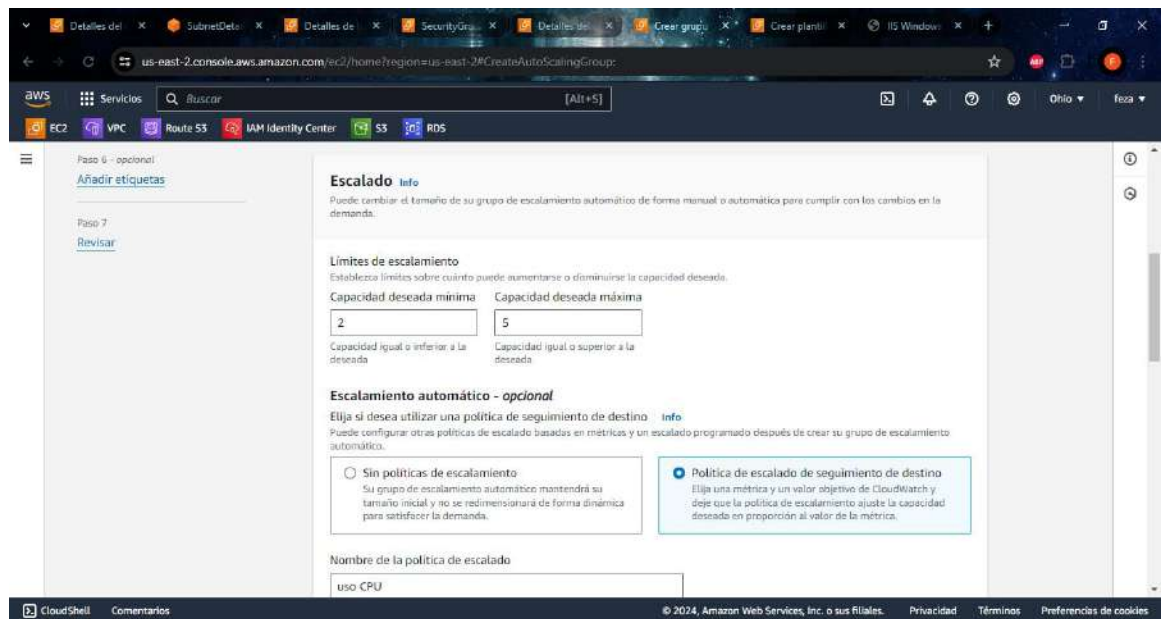
*Ilustración 94. selección del balanceador de carga*

Asignamos nuestros valores a crear de las instancias



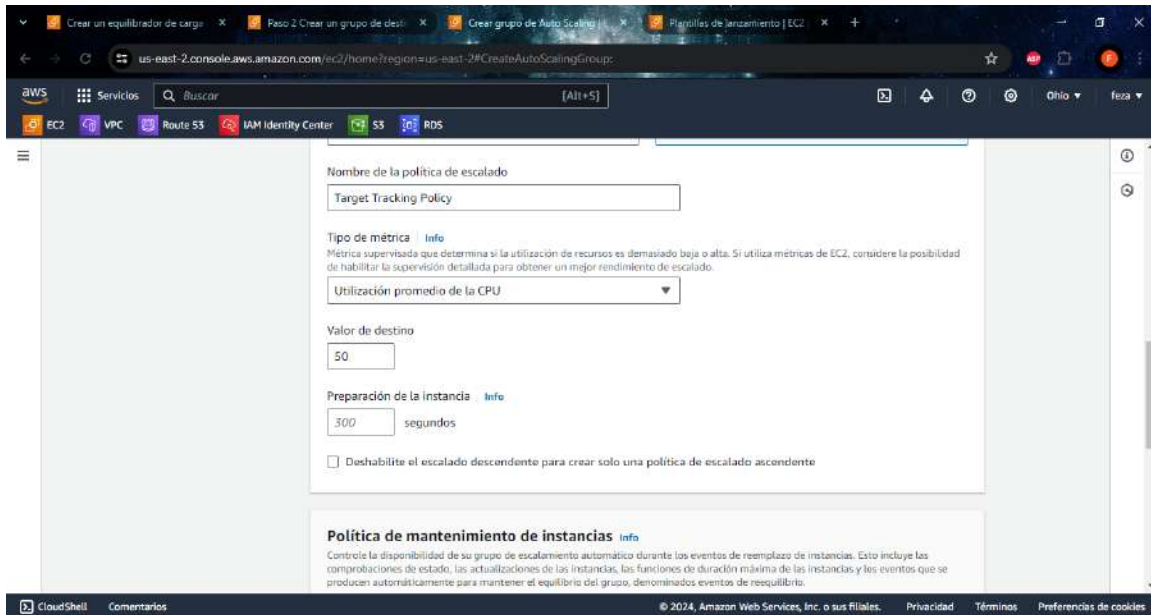
*Ilustración 95. límites de implementación de las instancias*

En la opción de escalado de nuestro grupo de escalamiento seleccionamos la opción de política de escalado de seguimiento de destino para poder asignarle el escalamiento por uso de CPU y le asignamos los valores que deseamos asignarles para poder automáticamente que el escaldo cree nuestras instancias



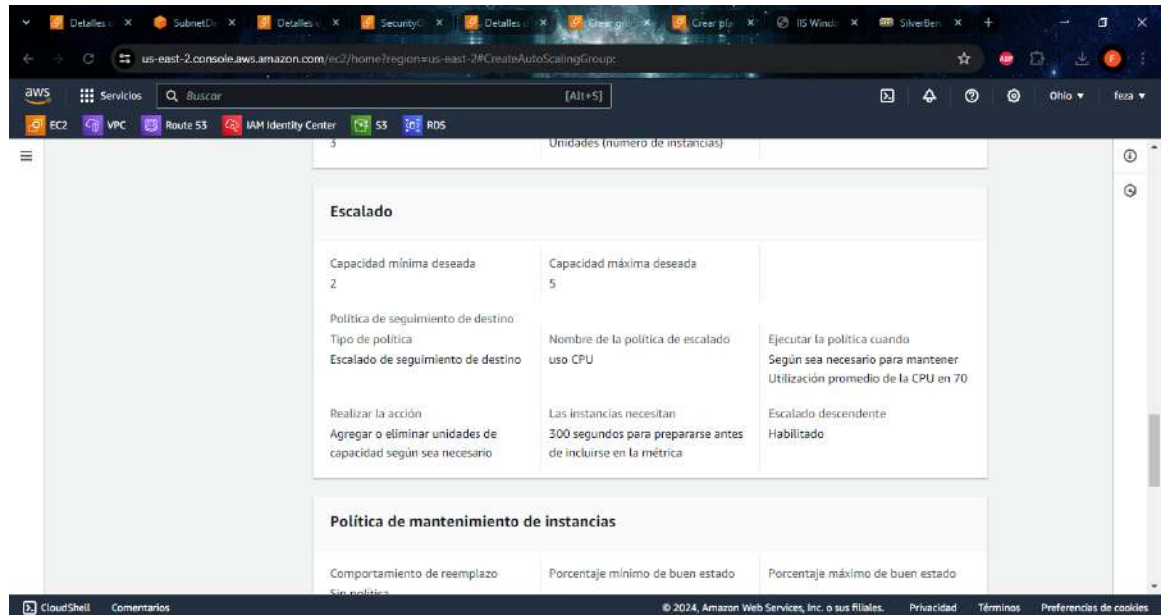
*Ilustración 96. política de escaldo*

Al activar la política nos aparecerá estos campos para crear unas matrices de uso de cpu y le damos en siguiente



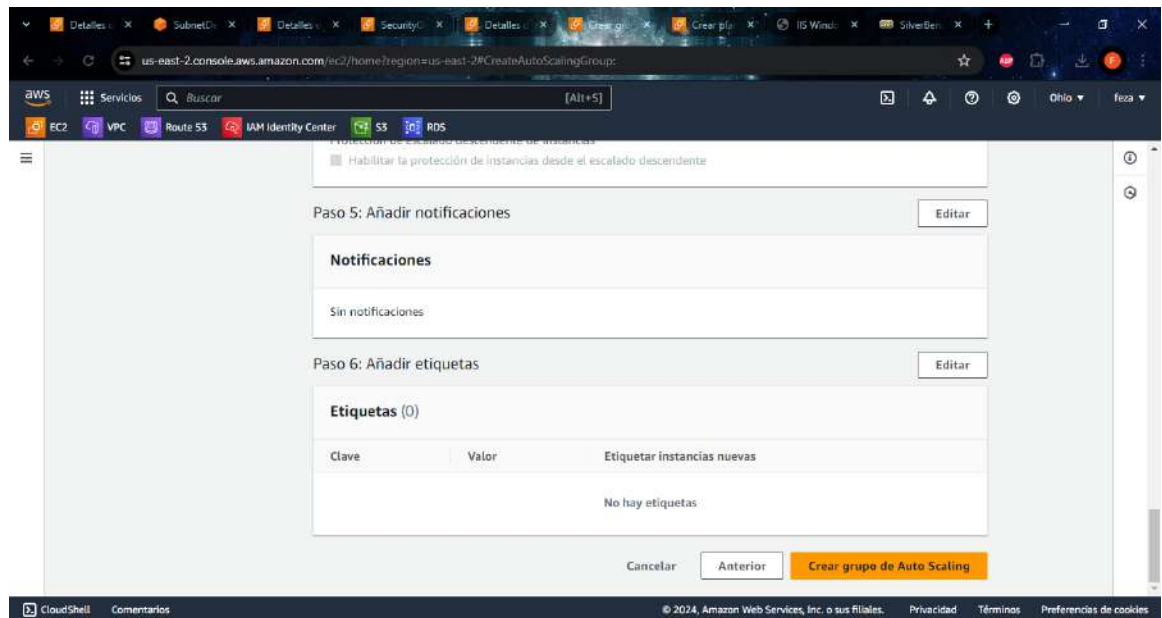
*Ilustración 97. matrices de escaldo de uso de CPU*

Ya al darle siguiente nuestro grupo de escalamiento va a quedar con estas matrices en la opción del escalado



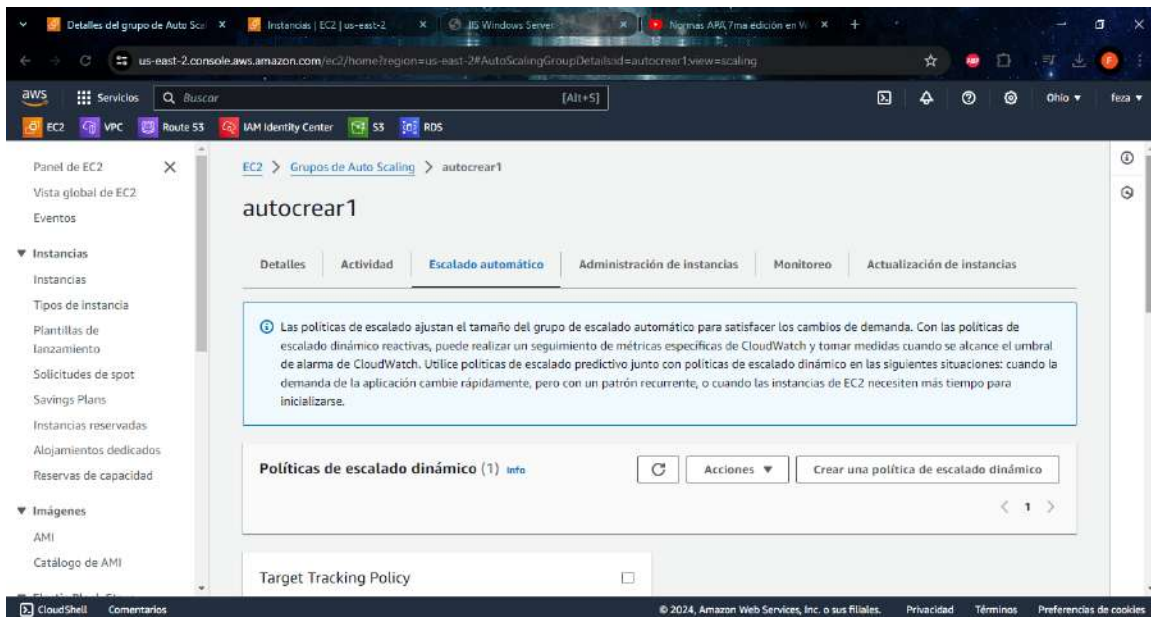
*Ilustración 98. escaldo final*

Le damos en crear el grupo de escalamiento para terminar



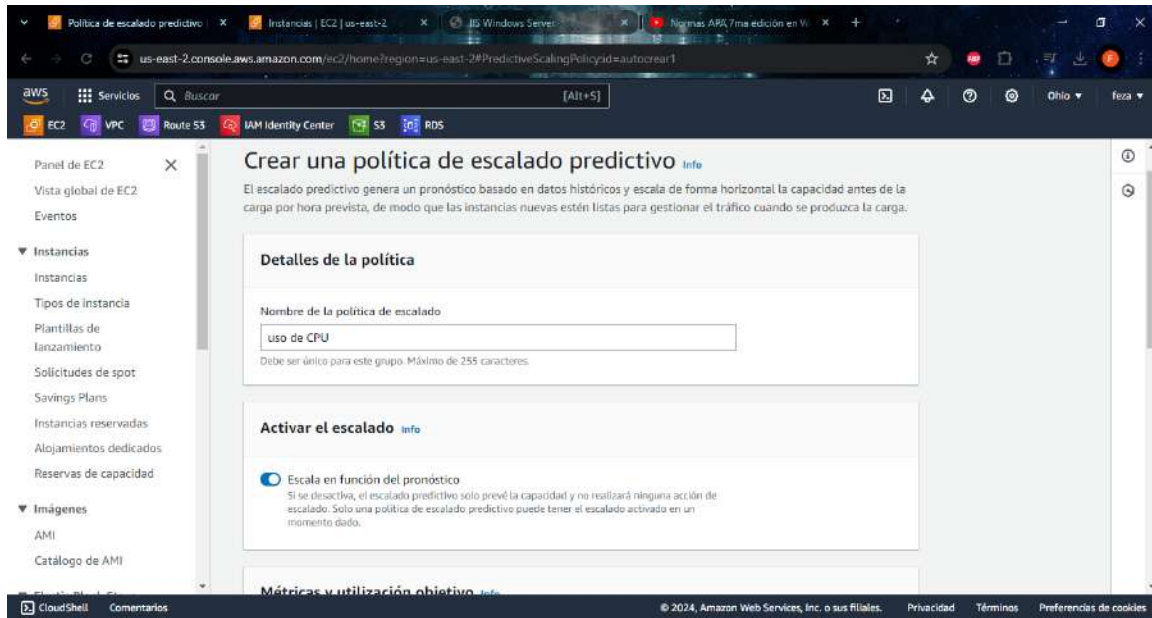
*Ilustración 99. crear grupo de escalamiento*

Una vez creado nuestro grupo de escalamiento nos vamos a la pestaña de escalado automático donde asignaremos los mismos valores cuando activamos la política de escalado de seguimiento de destino



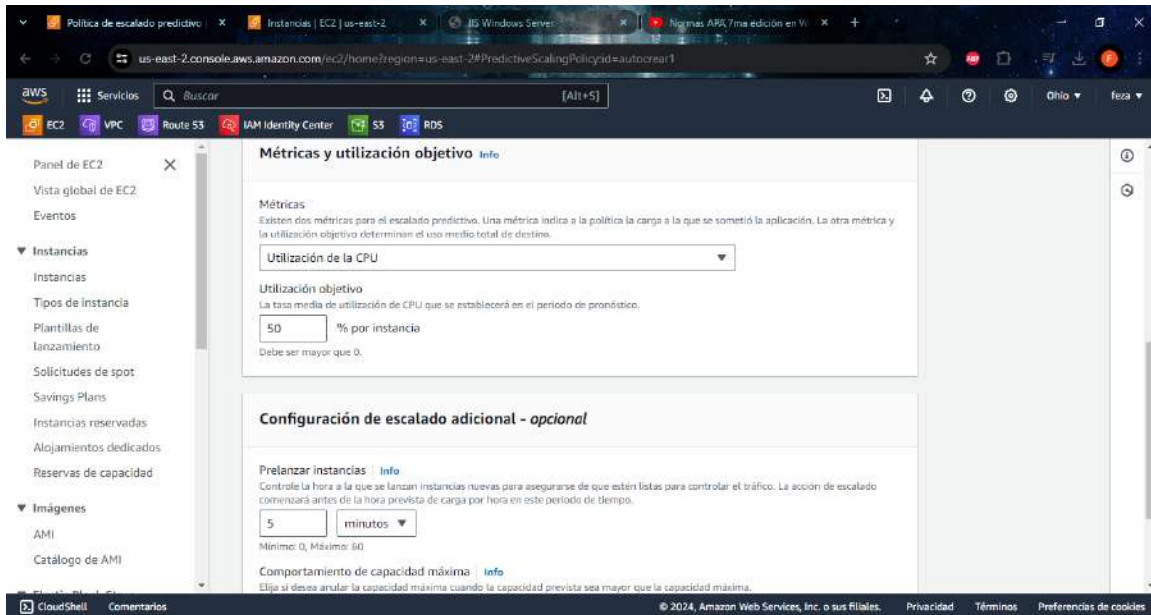
*Ilustración 100. escalado automatico*

Creamos una política creamos un nombre y activamos él escalado



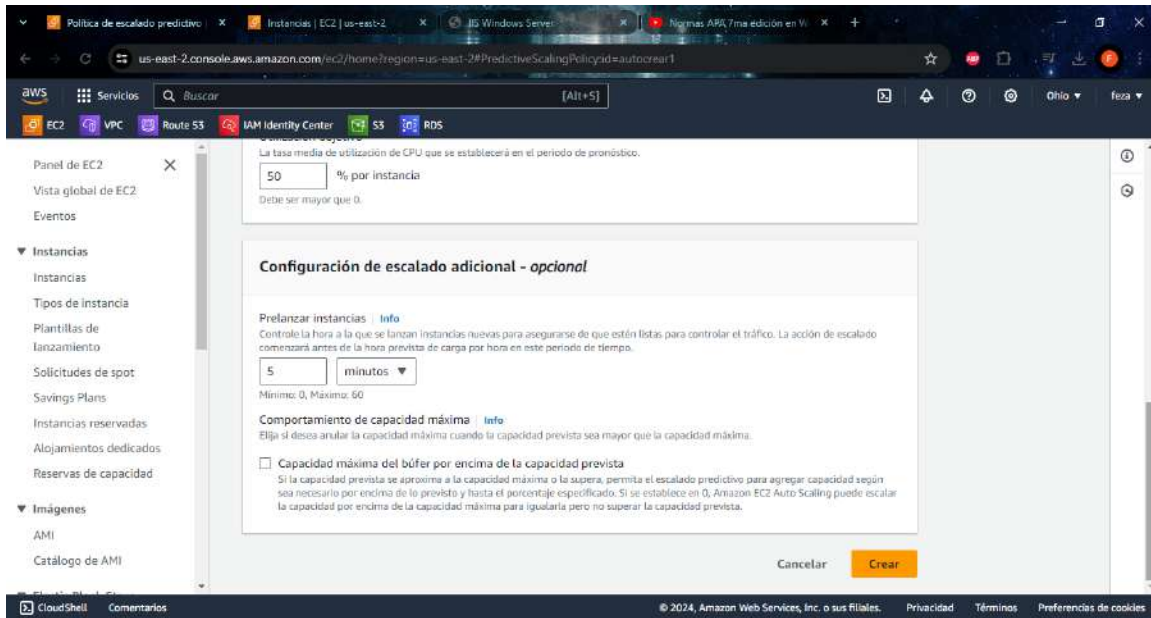
*Ilustración 101. politica uso de CPU*

Seleccionamos la métrica por uso de CPU, también el % de utilización y asignamos el tiempo en que será creada



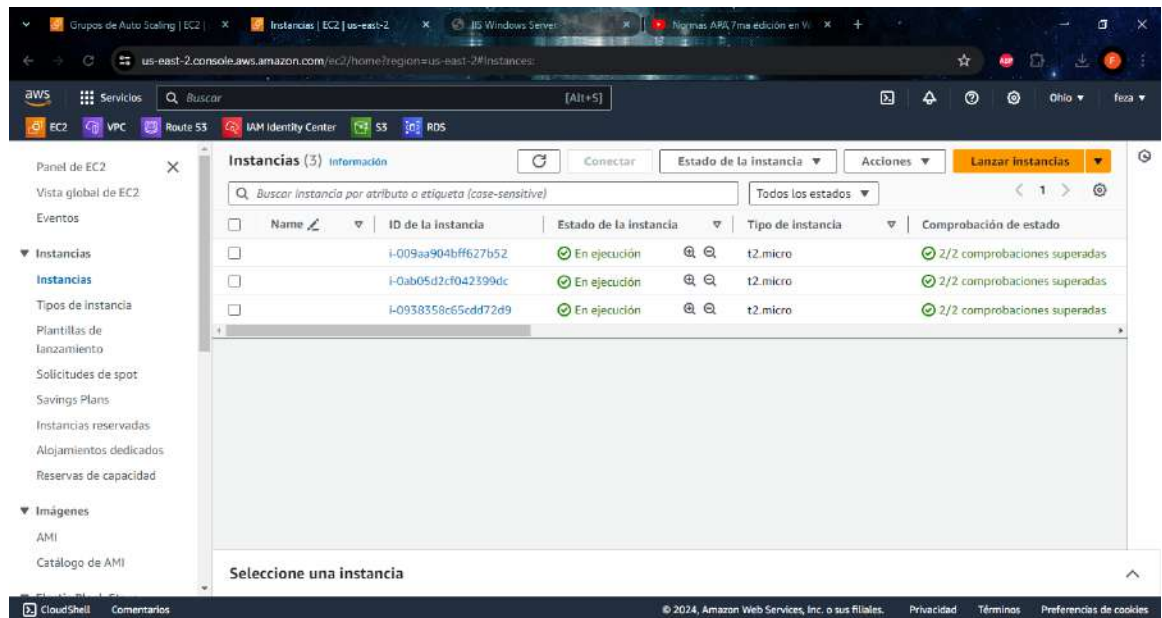
*Ilustración 102. detalles de la política de uso de CPU*

Por le damos en crear



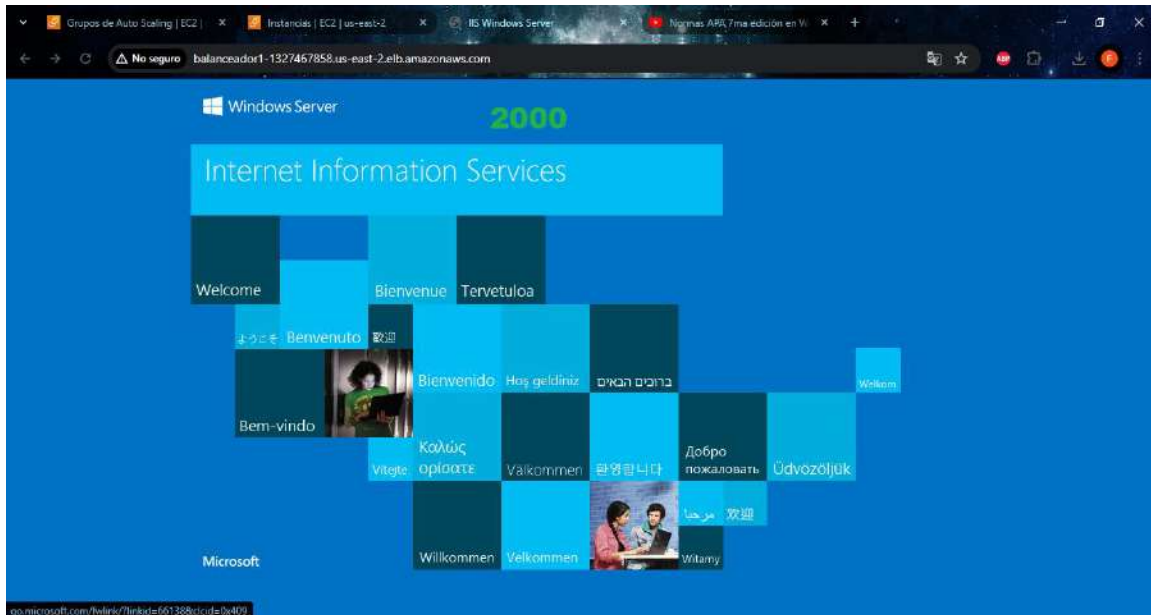
*Ilustración 103. crear politica*

Hay nos muestra las 3 nuevas creadas instancias



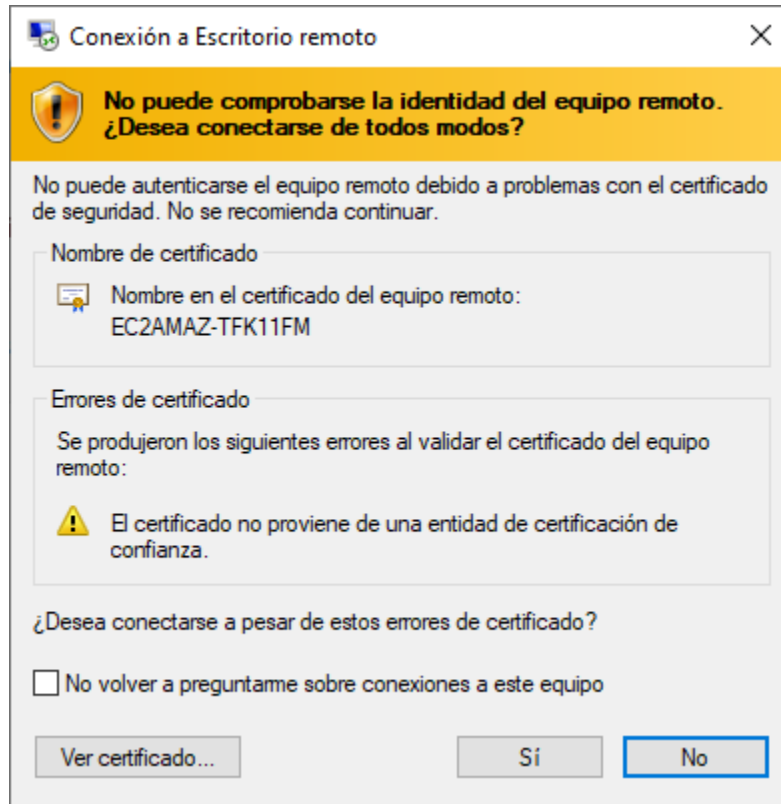
*Ilustración 104. primer resultado*

Probamos nuestro dns



*Ilustración 105. nuestro dns*

Ahora probaremos nuestro auto escalamiento ingresando a cualquiera de nuestras 3 instancias



*Ilustración 106. conexión a escritorio remoto*

Una vez hagamos ingresado abrimos el navegador de internet y buscamos esta aplicación web llamada silverbench para sobre cargar el procesador dándole clic en el

botón de benchmark

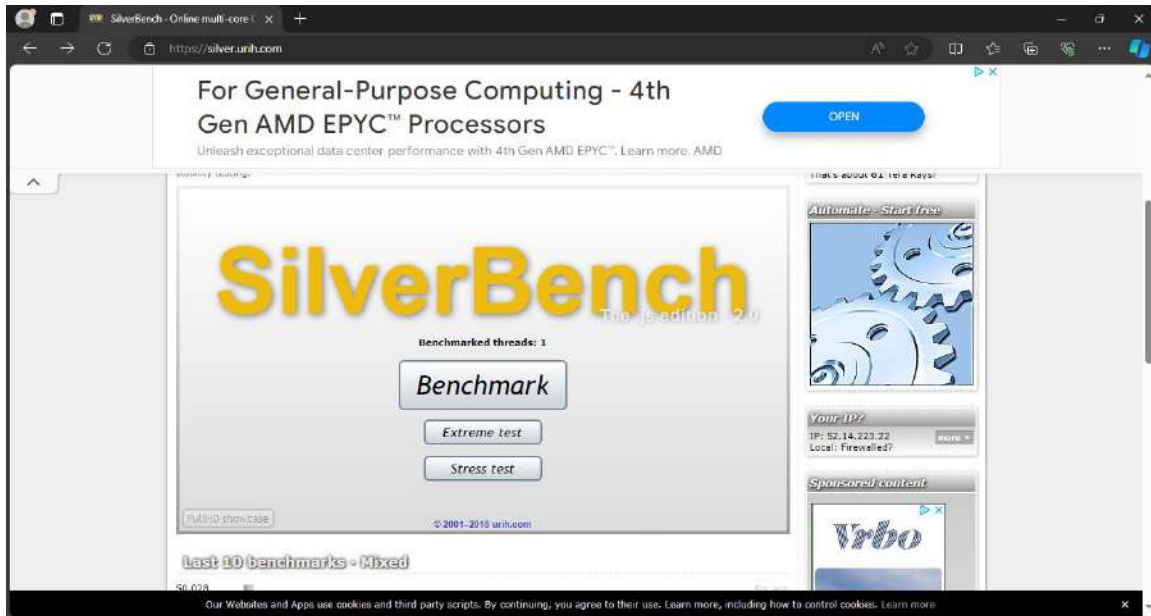
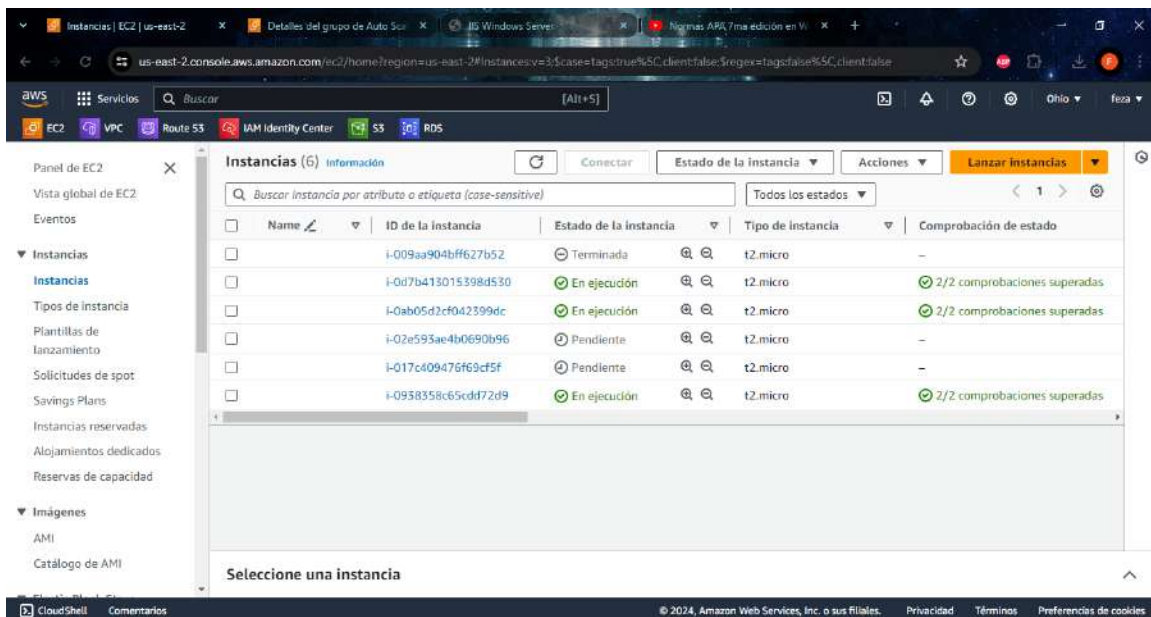


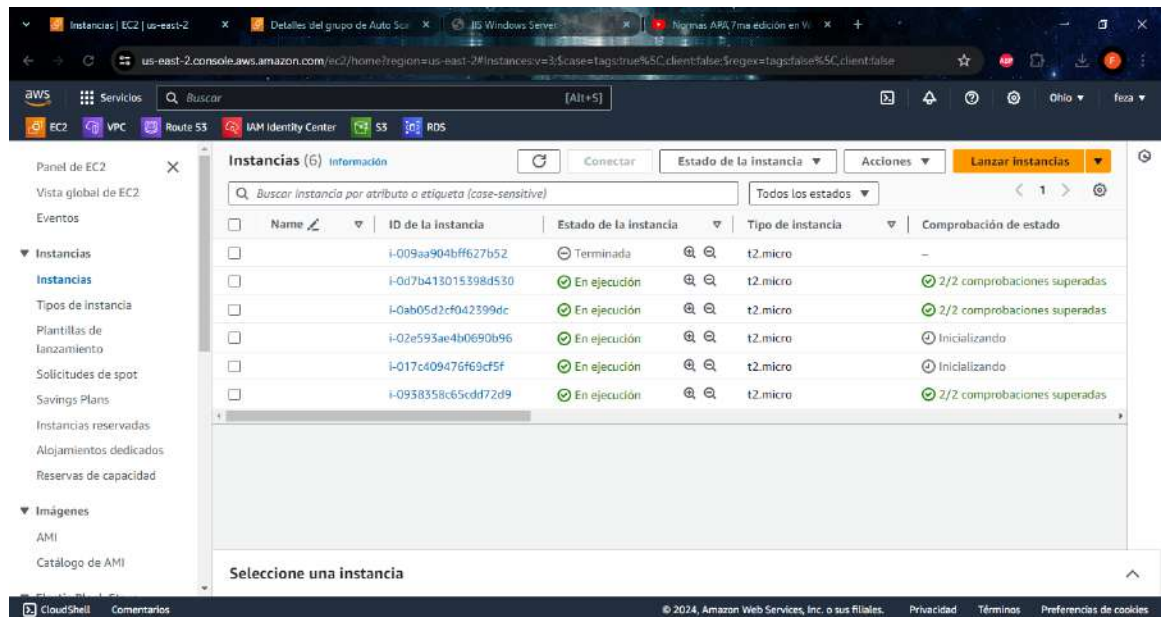
Ilustración 107. Siverbench

Después de esperar el tiempo estipulado para crear las instancias el automático nos crea las otras 2 instancias

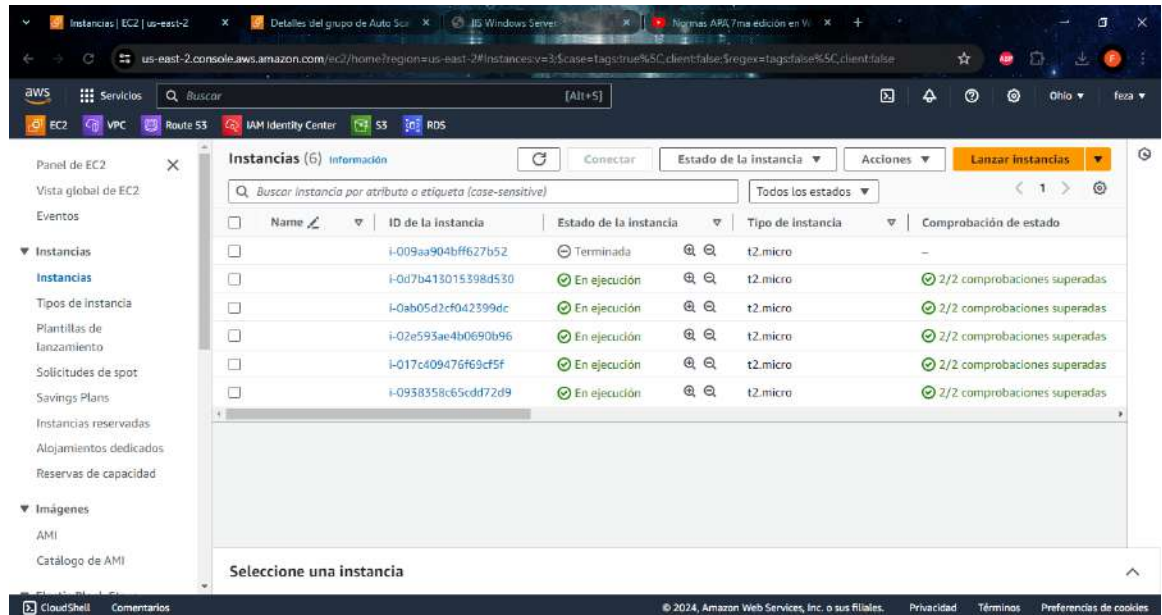


*Ilustración 108. segundo resultado*

Está probando la instalación inicializándolas

*Ilustración 109. inicializando las instancias creadas*

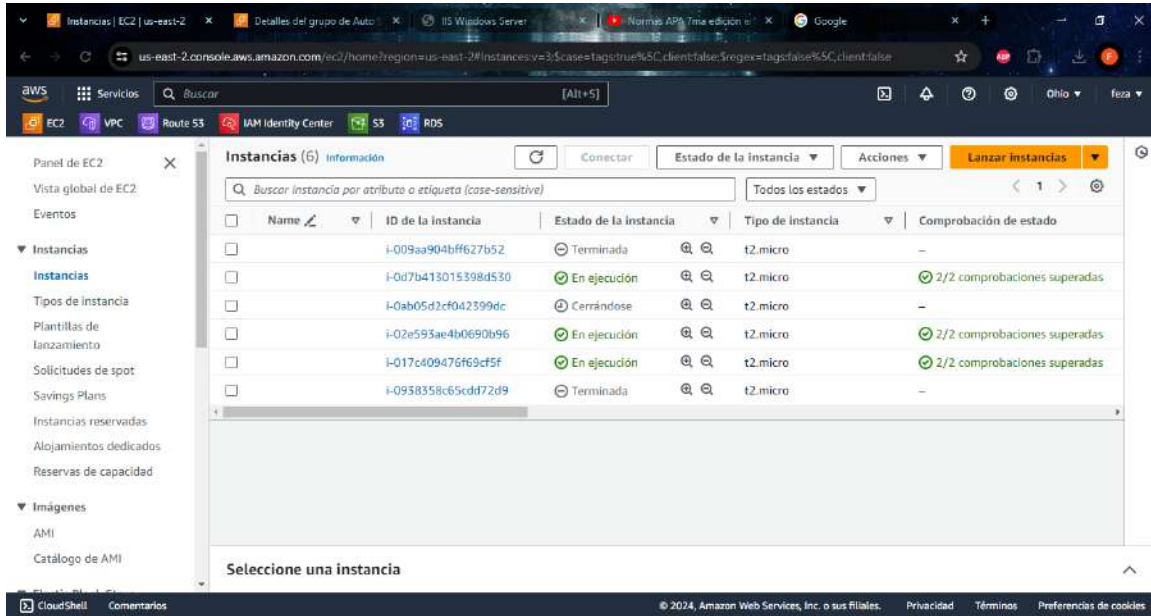
Y hay ya quedan listas y funcionales



*Ilustración 110. conectadas las instancias creadas*

Automáticamente cuando el procesador vuelve a su funcionamiento normal de uso las 2 instancias creadas de más desaparecen automáticamente una detrás de otra

podemos observar como la cierra y la termina y hay procede con la otra



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

Name	ID de la instancia	Estado de la instancia	Tipo de instancia	Comprobación de estado
	i-009aa904b627b52	Terminada	t2.micro	-
	i-0d7b413015398d530	En ejecución	t2.micro	2/2 comprobaciones superadas
	i-0ab05d2cf042399dc	Cerrándose	t2.micro	-
	i-02e595ae4b0690b96	En ejecución	t2.micro	2/2 comprobaciones superadas
	i-017c409476f69cf5f	En ejecución	t2.micro	2/2 comprobaciones superadas
	i-0958358c65cd472d9	Terminada	t2.micro	-

The interface includes a search bar at the top, a left-hand navigation menu with categories like 'Instancias', 'Tipos de instancia', and 'Imágenes', and a footer with 'CloudShell' and 'Comentarios' options.

*Ilustración 111. resultado final*

### **Conclusiones**

Bueno ya para concluir es muy fundamental señalar que el desarrollo de la implantación de la computación en la nube por medio de la aplicación Amazon Web Services (AWS) fue muy importante y de gran enseñanza a tal punto de enriquecer los conocimientos que tenía sobre Amazon Web Services. Donde más adelante este conocimiento adquirido me puede ayudar a fortalecer el trabajo en equipo, y a ayudar a dar más aportes significativos en cualquier lugar donde se haga la implementación de computación en la nube de Amazon Web Services.

Ya que gracias a las charlas sobre Amazon Web Services que se hizo anteriormente pude obtener unas pautas y o directrices claras de cómo funciona la computación en la nube utilizando Amazon Web Services para darle un inicio a proyectos y o implementaciones en el lugar donde viva, labore o donde tengan la necesidad de un servicio de computación en la nube utilizando Amazon Web Services.

## Referencias

*accenture*. (15 de 04 de 2022). Obtenido de <https://www.accenture.com/co-es/insights/cloud-computing-index#:~:text=Hoy%20en%20d%C3%ADa%20la%20tecnolog%C3%ADa,operaciones%20y%20reducir%20los%20costos>.

*amazon*. (16 de 05 de 2024). *aws*. Obtenido de <https://aws.amazon.com/es/what-is/virtualization/#:~:text=La%20virtualizaci%C3%B3n%20es%20una%20tecnolog%C3%ADa,en%20una%20C3%BAnica%20m%C3%A1quina%20f%C3%ADsica>.

*desarrollo*, D. d. (2 de 8 de 2023). *Medium*. Obtenido de <https://devdiaryacademy.medium.com/how-to-install-nginx-on-an-amazon-linux-2-instance-developer-diary-6007e9d7e141>

Fernández, E. C. (05 de 08 de 2021). *tokio school*. Obtenido de <https://www.tokioschool.com/noticias/historia-computacion-nube/>

*hpe*. (26 de 05 de 2024). Obtenido de <https://www.hpe.com/lamerica/es/what-is/virtualization.html#:~:text=El%20origen%20de%20la%20virtualizaci%C3%B3n,un%20proceso%20a%20la%20vez>.