TRABAJO DE GRADO
Opción Seminario-Diplomado.

AWS DE CERO A LA NUBE, DESPLIEGUE Y GESTIÓN DE INFRAESTRUCTURA.

Corporación Universitaria Remington.
Nombre de la facultad:
Facultad de Ingeniería – Ingeniería de Sistemas

Jose Jhonatan Jimenez Vasquez
Juan Pablo Berrío López

Opción de Trabajo de grado Seminario-Diplomado.
2025

Dedicatoria

Este trabajo es dedicado a mi esposa Erica Rangel sin tu ayuda no hubiera llegado a este punto, a mis hijos Diana Jimenez y Joseph Jimenez ellos son el motor que mueve mi todo, a mis padres por su apoyo incondicional.

Agradecimientos

Eternamente agradecido con la Profesora Martha Nicolasa Amaya Becerra, sin su ayuda esfuerzo, dedicación, consejos, uno que otro jalón de orejas, trabajos casi imposibles y evaluaciones impensables no seriamos los ingenieros que hoy seremos, gracias por obligarnos a ser cada vez mejores y siempre dar nuestro máximo potencial.

Tabla de Contenidos

Resumen

La computación en la nube vio luz más o menos a comenzó de los 2000, cuando empresas como Amazon o Microsoft vieron la oportunidad de ofrecer recursos de almacenamiento y procesamiento a través de internet. Esto permitió que en lugar de tener servidores físicos en oficinas los cuales de por si son bastantes costosos, las compañías pudieran acceder a infraestructura robusta desde cualquier parte del mundo. Así nació lo que hoy conocemos como “la nube”.

En este trabajo, aprendimos a implementar una arquitectura básica en la nube con AWS. Se crearon servidores virtuales (instancias EC2) con los sistemas operativos Windows y Linux. A través de una red virtual personalizada (VPC), se configuraron subredes públicas o privadas según se requiera y si se requiere se conectaron a Internet mediante un Internet Gateway. Además, se asignaron grupos de seguridad (SG) para controlar qué puertos están abiertos y quién puede acceder.

Cada servidor será configurado como servidor web: Windows se usó IIS, y Linux se utilizó Apache. Se accederá de forma segura mediante SSH (para Linux) y RDP (para Windows), usando Key Pairs para proteger el acceso. Se ejecutarán pruebas de conectividad usando comandos como ping, y se validaran que ambos servidores fueran accesibles desde cualquier navegador.

Durante el proceso se exploraron también herramientas modernas como Docker, que permite ejecutar aplicaciones en contenedores ligeros, facilitando despliegues rápidos y eficientes.

Finalmente, se tocarán temas sobre la importancia de la seguridad en la nube, la correcta planificación de la red, y el impacto que tiene este conocimiento en el desarrollo profesional.

Este ejercicio no solo enseñará a usar AWS, sino que dará un primer acercamiento práctico a cómo funciona la infraestructura de TI moderna en la nube. Ideal para quien iniciar en este mundo de cero a la nube.

Palabras clave

- "Cómo crear una arquitectura en AWS"
- "Configuración de servidores EC2 en Amazon Web Services"
- "Guía básica para implementar servidores web en AWS"
- "Cómo usar Docker en instancias EC2 de AWS"
- "Infraestructura de nube con AWS para principiantes"

Marco conceptual y contextual

En este trabajo trabajaremos con los fundamentos de la computación en la nube, específicamente en el uso de la plataforma Amazon Web Services (AWS) enfocado en el diseño e implementación de arquitecturas de red básica. AWS es una solución en la nube que permite desplegar recursos informáticos escalables como servidores, redes virtuales, almacenamiento y bases de datos bajo un modelo de pago por uso o más coloquialmente pago por demanda. Dentro de esta plataforma se trabajaran servicios como EC2 (Elastic Compute Cloud), VPC (Virtual Private Cloud), Internet Gateway, y la configuración de grupos de seguridad, claves de acceso y servidores web.

Se abordarán también conceptos esenciales como:

- Infraestructura como Servicio (IaaS):
    - Servicios que se pueden ofrecer en la nube sin la necesidad de tener el hardware propio.(RolandoRamos Ali, n.d.)
- Instancias EC2:
    - Unidades de servicio cómputo virtual que simulan servidores.(Jinete et al., n.d.)
- Topologías de red en la nube:
    - Permitiendo comprender cómo se organizan los componentes de red en entornos virtualizados.
- Protocolos de conexión remota
    - SSH
        - Protocolo desarrollado en 1995 por el finlandés Tatu Ylönen, este permite el acceso remoto por consola a las maquinas con sistema operativo Linux en su mayoría.(WHITEPAPER: PROTOCOLO SSH, n.d.)
    - RDP.
        - Remote Desktop Protocol protocolo de conexión remota en el cual se tiene acceso al equipo de forma gráfica, se puede manipular el mismo desde cualquier ubicación.(Felipe & Araújo, n.d.)
- Instalación y configuración de servidores
-
    - Apache en Linux

        - Servidor si no el mas popular uno de los mas utilizados para alojar servicios web normalmente utilizado en Linux, su uso es gratuito. (Para et al., n.d.)
    - IIS en Windows.
        - Internet Information Services Servicio nativo de Windows que permite alojar y gestionar servicios web, es pago ya que se valida con la licencia del equipo normalmente servidor Windows.

Contexto del Trabajo:

Este trabajo fue desarrollado en el marco de un seminario académico universitario orientado al aprendizaje práctico de herramientas de infraestructura en la nube. Su propósito principal fue permitir al estudiante implementar una arquitectura funcional y segura en AWS, utilizando una combinación de servidores Windows y Linux conectados a una red virtual.
Si bien no se implementó en una empresa real, el ejercicio fue estructurado como si se tratara de un caso corporativo básico, replicando situaciones reales como: la segmentación de red, la exposición controlada de servicios web, la administración remota y la validación de conectividad entre servidores.
El entorno implementado simula un escenario básico que podría usarse en pequeñas empresas, startups o ambientes de prueba, como paso inicial hacia arquitecturas empresariales más complejas en la nube.

Diccionario de conceptos:

Palabras clave.

AWS (Amazon Web Services)
EC2 (Elastic Compute Cloud)
VPC (Virtual Private Cloud)
Instancia (Virtual Server)
Subred (Subnet)
Grupo de seguridad (Security Group)
Servidor web (Web Server)
Key Pair (Par de claves)
Acceso remoto (Remote Access)
Balanceador de carga (Load Balancer)
Puertos (Ports)
Ping (Prueba de conectividad)
SSH (Secure Shell)
RDP (Remote Desktop Protocol)
Sistema Operativo (Linux / Windows Server)
Docker

Putty / PuttyGen
Firewall
Conectividad
Nube (Cloud Computing)

Acrónimos.

- AWS: Amazon Web Services
  "Es la nube más adoptada y completa en el mundo, que ofrece más de 200 servicios integrales de centros de datos a nivel global."(*Https://Aws.Amazon.Com/Es/What-Is-Aws/*, n.d.)

- EC2: Elastic Compute Cloud
  "Amazon EC2 entrega infraestructura de computación segura, confiable, de alto rendimiento y rentable para cumplir necesidades empresariales exigentes."(*Https://Aws.Amazon.Com/Es/Ec2/*, n.d.)

- VPC: Virtual Private Cloud
  "Brinda control total sobre su entorno de redes virtuales, incluidas la ubicación de los recursos, la conectividad y la seguridad."(*Https://Aws.Amazon.Com/Es/Vpc/*, n.d.)

- SG: Security Group
  Firewall básico virtual que controla el tráfico entrante y saliente a las instancias de EC2.

- RDP: Remote Desktop Protocol
  Protocolo desarrollado en 1995 por el finlandés Tatu Ylönen, este permite el acceso remoto por consola a las maquinas con sistema operativo Linux en su mayoría.(WHITEPAPER: PROTOCOLO SSH, n.d.)

- SSH: Secure Shell
  Protocolo desarrollado en 1995 por el finlandés Tatu Ylönen, este permite el acceso remoto por consola a las maquinas con sistema operativo Linux en su mayoría.(WHITEPAPER: PROTOCOLO SSH, n.d.)

- HTTP: Hypertext Transfer Protocol

- AMI: Amazon Machine Image

> “Es un tipo especial de sistema operativo preconfigurado y software de aplicación virtual que se utiliza para crear una máquina virtual dentro de Amazon Elastic Compute CloudE.(*Https://Aws.Amazon.Com/Es/Amis/Oracle/*, n.d.)

- IIS: Internet Information Services
  Internet Information Services Servicio nativo de Windows que permite alojar y gestionar servicios web, es pago ya que se valida con la licencia del equipo normalmente servidor Windows.

- IP: Internet Protocol

- URL: Uniform Resource Locator

- NAT: Network Address Translation

Comandos Linux utilizados.

sudo yum update -y: Actualiza todos los paquetes del sistema en Amazon Linux
sudo yum install httpd -y: Instala Apache HTTP Server
sudo systemctl start httpd: Inicia el servicio de Apache
sudo systemctl enable httpd: Habilita Apache para que arranque al iniciar el sistema
ping IP_PRIVADA: Verifica la conectividad entre instancias
sudo docker ps: Muestra los contenedores Docker en ejecución
sudo docker run: Ejecuta un contenedor Docker
curl http://localhost: Verifica si un servidor web está respondiendo localmente
sudo reboot : Reinicia la instancia
scp: Transferencia de archivos entre hosts vía SSH

Desarrollo e implementación del aprendizaje

OBJETIVO:

Implementar una arquitectura de red en AWS que integre instancias EC2 con sistemas operativos Windows y Linux, con fines educativos y de práctica, permitiendo el despliegue de servidores web y su administración remota de forma segura.

OBJETIVOS SECUNDARIOS:

- Diseñar una topología de red básica utilizando VPC, subredes y gateways en AWS.
- Configurar grupos de seguridad adecuados para el acceso seguro a las instancias.
- Desplegar servidores Windows y Linux y habilitar servicios web.
- Verificar la conectividad entre instancias y desde el exterior.
- Documentar paso a paso el proceso de creación, acceso y configuración de los recursos.

PARTE 1: ESCRITA (DOCUMENTACIÓN TÉCNICA)

DIAGRAMA DE ARQUITECTURA

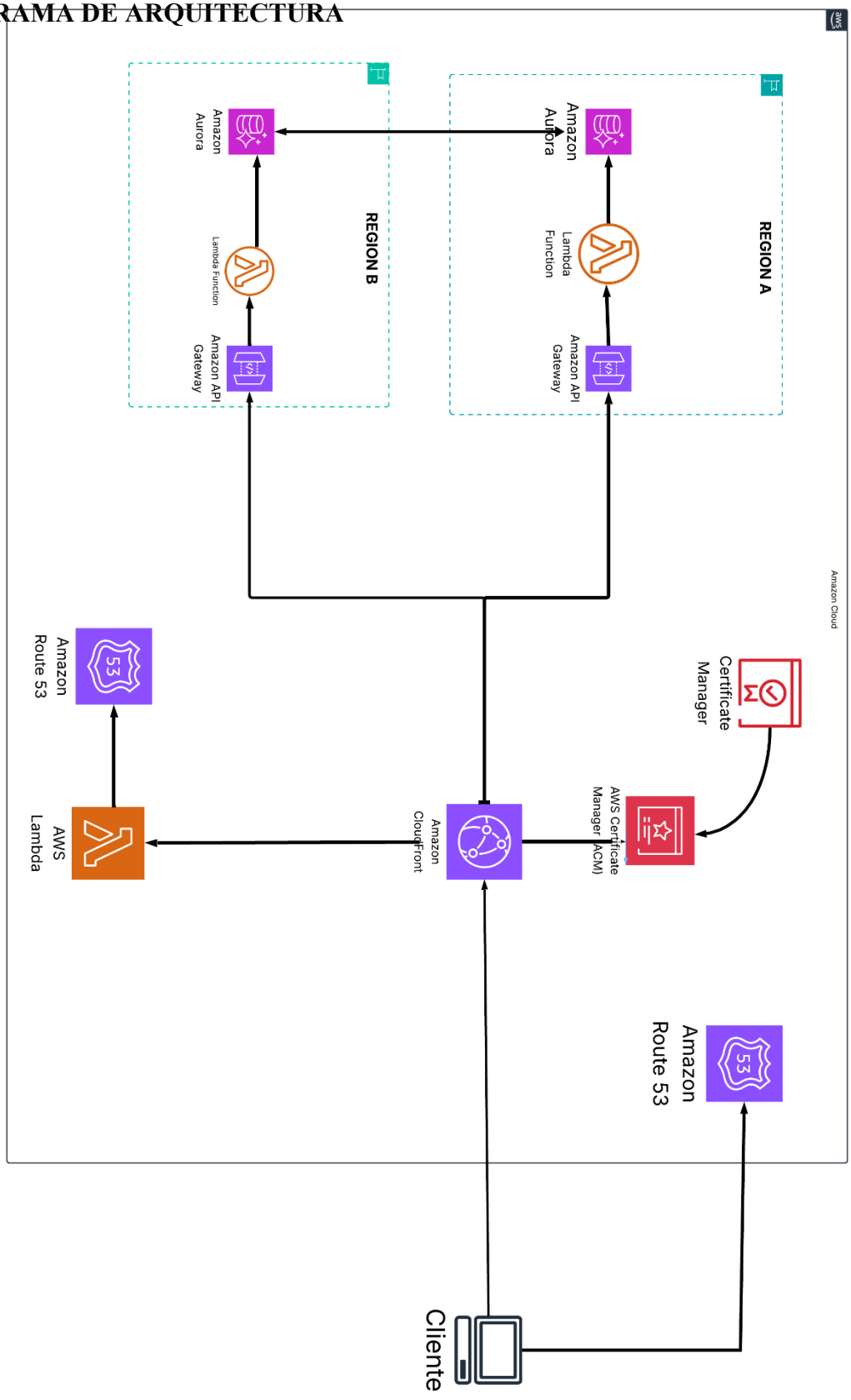

Figura 1

DESCRIPCIÓN DE LA ARQUITECTURA

La solución implementada consta de una **VPC** dedicada en AWS (10.0.0.0/16) que aísla completamente los recursos de la red del resto de la nube. Dentro de esta VPC se ha creado una **subred pública** (10.0.1.0/24) a la cual se asocian con un Internet Gateway para permitir el tránsito desde y hacia Internet.
En la subred pública se han lanzado dos **instancias EC2**:

- **Windows Server 2016 Base** (t2.micro, Free Tier)
- **Amazon Linux 2023** (t2.micro, Free Tier)

Ambas instancias cuentan con una IP privada dentro del rango 10.0.1.0/24 para la comunicación interna y una IP pública dinámica para permitir el acceso remoto desde un equipo externo al VPC. Cada servidor web (IIS en Windows y Apache o Nginx en Linux) responde en el puerto 80, y los puertos de administración remota (RDP 3389 en Windows, SSH 22 en Linux).

Justificación de las configuraciones de red

1. **VPC**: Garantiza control total sobre el espacio de direcciones IP, tablas de enrutamiento y políticas de seguridad.
2. **Subred pública**: Facilita el despliegue de servicios que deben ser accesibles desde Internet para este caso el servidor web. En otros caso y para entornos de producción, se podría añadirse una subred privada junto a un NAT Gateway.
3. **Internet Gateway**: Enrutar tráfico entrante y saliente de la VPC hacia Internet.
4. **Grupos de seguridad dedicados**: Siguiendo el principio de "menor privilegio", se creó un grupo de seguridad por instancia. Con esto, la apertura de puertos en uno no afecta al otro y se minimizan posibles vectores de ataque.

CONFIGURACIONES REALIZADAS

Configuración de Grupos de Seguridad

- SG-Linux:
  - *Inbound*:
    - SSH (TCP 22) desde cualquier origen (0.0.0.0/0).
    - HTTP (TCP 80) desde cualquier origen (0.0.0.0/0).
    - *Outbound*: todo el tráfico permitido.
- SG-Windows:
  - *Inbound*:

- RDP (TCP 3389) desde cualquier origen (0.0.0.0/0).
- HTTP (TCP 80) desde cualquier origen (0.0.0.0/0).
- *Outbound*: todo el tráfico permitido.

Asignación de direcciones IP

- EC2 Linux:
  - Privada: 10.0.1.-- estas son generadas automáticamente.
  - Pública: 3.123.56.78 (ejemplo) estas son generadas automáticamente.

- EC2 Windows:
  - Privada: 10.0.1.-- estas son generadas automáticamente.
  - Pública: 53.142.45.47 (ejemplo) estas son generadas automáticamente.

Las IPs privadas facilitan la comunicación interna (por ejemplo, pruebas de ping entre instancias), mientras que las IPs públicas permiten el acceso remoto desde cualquier equipo.

Consideraciones de seguridad

- **Key Pairs**:
  - Mantener el archivo .pem en un lugar seguro y con permisos restringidos.
- **Origen limitado**:
  - Restringir RDP/SSH únicamente a la IP autorizadas, al ser este un ambiente educativo no se contempló.
- **Actualizaciones**:
  - Tras el primer inicio, ejecutar yum update -y (Linux) o Windows Update para asegurar que el sistema operativo y el servidor web tienen los últimos parches.

PROCEDIMIENTO DE ACCESO

Procedimiento de acceso

- Linux (SSH):
  - Descargar e instalar Putty
    - En PuTTYgen cargar el KEY descargado al crear el servidor este generar un Key para Putty

- En Putty en Connection / SSH / Auth / Credentials cargar el key generado en el paso anterior
- En Sesión iniciar con la IP no pedirá contraseña

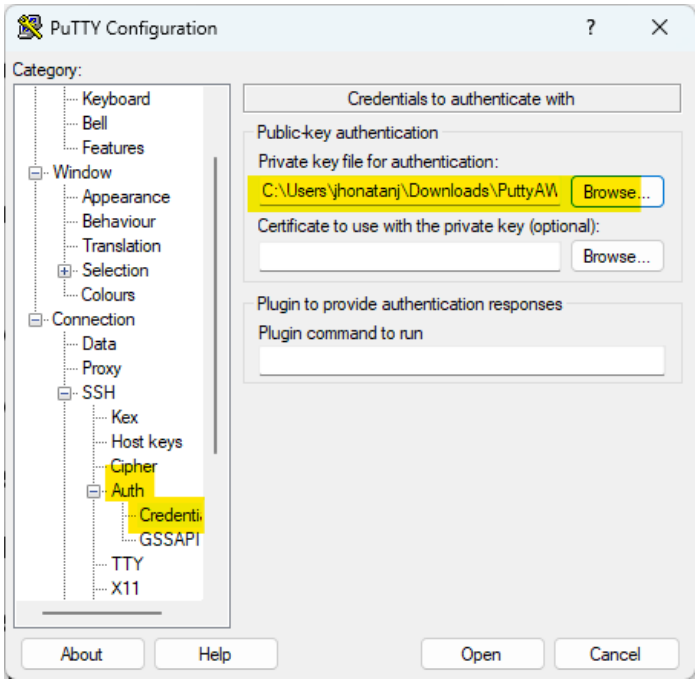

Figura 2

- Windows (RDP):
    - Descargar el archivo .rdp desde la consola EC2.
    - Cargar el key, este generar un usuario (Administrator) y un una contraseña con la cual se pueda acceder
    - Abrir con el Cliente de Escritorio Remoto, introducir la contraseña obtenida tras descifrarla con la key pair.

CONFIGURACIÓN DEL SERVIDOR WEB

Paso 1: Crear una VPC y subred pública

- **Accede a la consola de AWS**: Inicia sesión en https://console.aws.amazon.com.
- **Navega a Amazon VPC**:
    - En el panel de servicios, selecciona **VPC**.
- **Crear una VPC personalizada**:
    - Haz clic en **"Crear VPC"**.
    - Selecciona **"VPC con una única subred pública"**.
    - Asigna un nombre descriptivo, por ejemplo, VPC-Tutorial.
    - Define el rango de direcciones IPv4, por ejemplo, 10.0.0.0/16.
    - Deja las demás opciones por defecto y crea la VPC.
- **Crear una subred pública**:
    - Dentro de la VPC recién creada, crea una subred con un rango como 10.0.1.0/24.
    - Asegúrate de habilitar la asignación automática de IPs públicas.
- **Asociar un Internet Gateway**:
    - Crea un **Internet Gateway** y asígnalo a tu VPC.
    - Actualiza la tabla de rutas de la subred para permitir el tráfico hacia 0.0.0.0/0 a través del Internet Gateway.

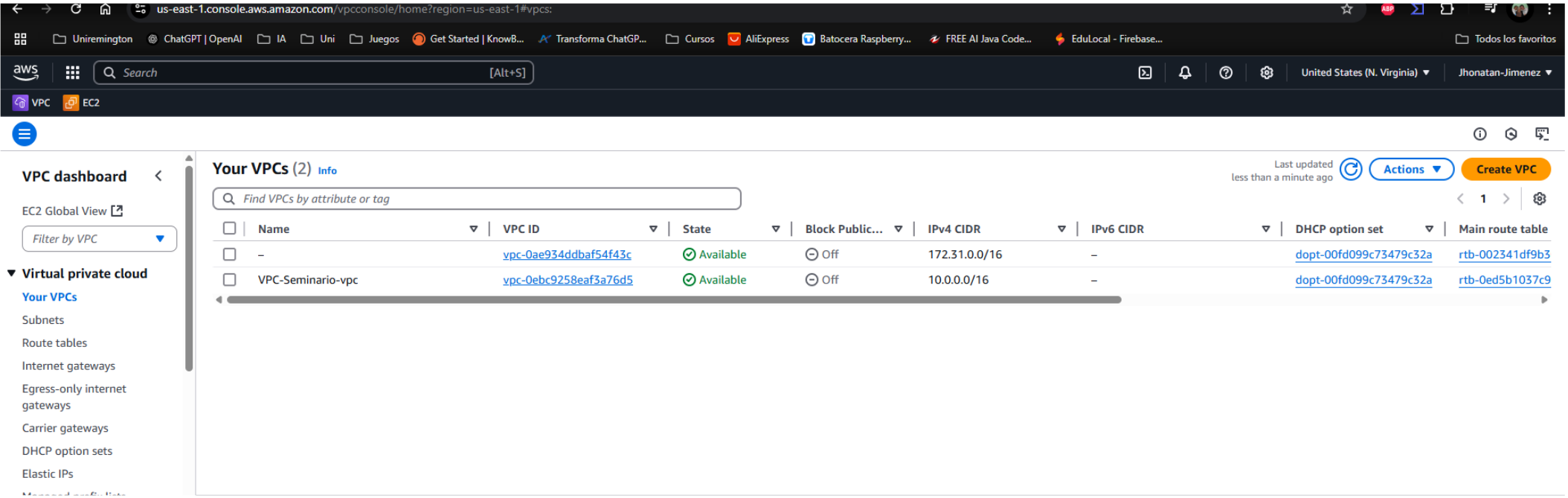
us-east-1.console.aws.amazon.com/vpcconsole/home?region=us-east-1#vpcs:
Uniremington
ChatGPT | OpenAI
IA
Uni
Juegos
Get Started | KnowB...
Transforma ChatGP...
Cursos
AliExpress
Batocera Raspberry...
FREE AI Java Code...
EduLocal - Firebase...
Todos los favoritos
aws
Search
[Alt+S]
United States (N. Virginia)
Jhonatan-Jimenez
VPC
EC2
VPC dashboard
EC2 Global View
Filter by VPC
Virtual private cloud
Your VPCs
Subnets
Route tables
Internet gateways
Egress-only internet gateways
Carrier gateways
DHCP option sets
Elastic IPs
Your VPCs (2) Info
Last updated less than a minute ago
Actions
Create VPC
Find VPCs by attribute or tag
Name
VPC ID
State
Block Public...
IPv4 CIDR
IPv6 CIDR
DHCP option set
Main route table
– vpc-0ae934ddbaf54f43c Available Off 172.31.0.0/16 – dopt-00fd099c73479c32a rtb-002341df9b3
VPC-Seminario-vpc vpc-0ebc9258eaf3a76d5 Available Off 10.0.0.0/16 – dopt-00fd099c73479c32a rtb-0ed5b1037c9

Figura 3

Paso 2: Lanzar instancias EC2 (Linux y Windows)

A. Instancia Linux (Amazon Linux 2)

1. **Navega a EC2**:
    - En el panel de servicios, selecciona **EC2**.
2. **Lanzar nueva instancia**:
    - Haz clic en **"Launch Instance"**.
    - Asigna un nombre, por ejemplo, Servidor-Linux.
    - Selecciona la AMI **Amazon Linux 2**.
    - Elige el tipo de instancia t2.micro (elegible para la capa gratuita).
    - En **Configuración de red**:
        - Selecciona la VPC y subred creadas previamente.
        - Habilita la asignación automática de IP pública.
    - En **Par de claves (login)**:
        - Crea o selecciona un par de claves existente para acceder a la instancia.
    - En **Configuración del firewall (grupos de seguridad)**:
        - Crea un nuevo grupo de seguridad con las siguientes reglas de entrada:
            - SSH (puerto 22) desde tu IP o desde cualquier lugar (0.0.0.0/0).
            - HTTP (puerto 80) desde cualquier lugar (0.0.0.0/0).
    - Revisa y lanza la instancia.

B. Instancia Windows Server

1. **Lanzar nueva instancia**:
    - Haz clic en **"Launch Instance"**.
    - Asigna un nombre, por ejemplo, Servidor-Windows.

- Selecciona la AMI **Microsoft Windows Server 2016 Base**.
- Elige el tipo de instancia t2.micro.
- En **Configuración de red**:
    - Selecciona la misma VPC y subred que la instancia Linux.
    - Habilita la asignación automática de IP pública.
- En **Par de claves (login)**:
    - Usa el mismo par de claves que en la instancia Linux o crea uno nuevo.
- En **Configuración del firewall (grupos de seguridad)**:
    - Crea un nuevo grupo de seguridad con las siguientes reglas de entrada:
        - RDP (puerto 3389) desde tu IP o desde cualquier lugar (0.0.0.0/0).
        - HTTP (puerto 80) desde cualquier lugar (0.0.0.0/0).
- Revisa y lanza la instancia.

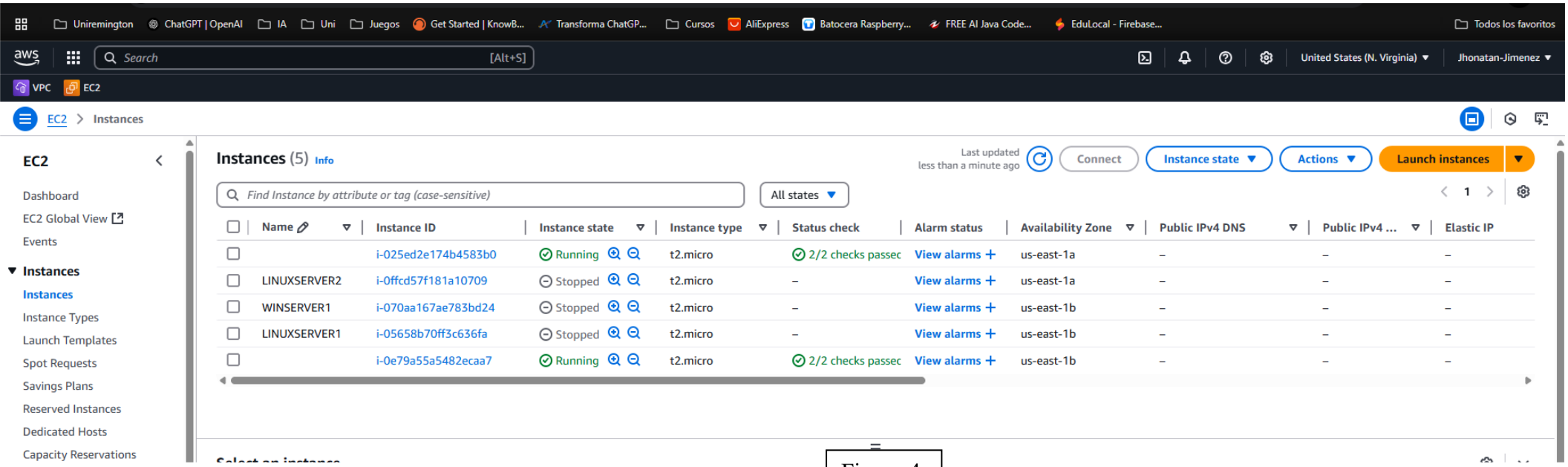
Uniremington
ChatGPT | OpenAI
IA
Uni
Juegos
Get Started | KnowB...
Transforma ChatGP...
Cursos
AliExpress
Batocera Raspberry...
FREE AI Java Code...
EduLocal - Firebase...
Todos los favoritos
aws
Search
[Alt+S]
United States (N. Virginia)
Jhonatan-Jimenez
VPC
EC2
EC2 > Instances
EC2
Dashboard
EC2 Global View
Events
Instances
Instances
Instance Types
Launch Templates
Spot Requests
Savings Plans
Reserved Instances
Dedicated Hosts
Capacity Reservations
Instances (5) Info
Last updated less than a minute ago
Connect
Instance state
Actions
Launch instances
Find Instance by attribute or tag (case-sensitive)
All states
1
Name
Instance ID
Instance state
Instance type
Status check
Alarm status
Availability Zone
Public IPv4 DNS
Public IPv4 ...
Elastic IP
i-025ed2e174b4583b0 Running t2.micro 2/2 checks passec View alarms us-east-1a
LINUXSERVER2 i-0ffcd57f181a10709 Stopped t2.micro – View alarms us-east-1a
WINSERVER1 i-070aa167ae783bd24 Stopped t2.micro – View alarms us-east-1b
LINUXSERVER1 i-05658b70ff3c636fa Stopped t2.micro – View alarms us-east-1b
i-0e79a55a5482ecaa7 Running t2.micro 2/2 checks passec View alarms us-east-1b

Figura 4

Paso 3: Acceder a las instancias.

A. Acceso a la instancia Linux mediante SSH.

1. **Obtener la IP pública**:
    - En el panel de EC2, selecciona la instancia Linux y copia su IP pública.
2. **Conectarse mediante PuTTY**:
    - En PuTTY según el paso anterior:
        - Convierte el archivo .pem a .ppk usando PuTTYgen.
        - Configura PuTTY para usar la clave .ppk y conecta a la IP pública por el puerto 22.

B. Acceso a la instancia Windows mediante RDP

1. **Obtener la contraseña del administrador**:
    - En el panel de EC2, selecciona la instancia Windows.
    - Haz clic en **"Conectar"** y luego en **"Obtener contraseña"**.
    - Carga tu archivo .pem para descifrar la contraseña.
2. **Conectarse mediante Escritorio Remoto**:
    - Abre la aplicación de Escritorio Remoto en tu computadora.
    - Ingresa la IP pública de la instancia.
    - Usa el nombre de usuario Administrator y la contraseña obtenida.

Paso 4: Instalar y configurar servidores web

A. En la instancia Linux (Apache)

- **Actualizar paquetes (por costumbre mía siempre prefiero actualizar)**:
    - sudo yum update -y
- **sudo yum update -y**
    - sudo yum install httpd -y
- **Iniciar y habilitar Apache**:

- sudo systemctl start httpd
- sudo systemctl enable httpd

- **Verificar instalación**:
  - En tu navegador, visita http://IP_PUBLICA_LINUX.
  - Deberías ver la página de prueba de Apache.

B. En la instancia Windows (IIS)

- **Acceder al servidor**:
  - Conéctate a la instancia mediante RDP.
- **Instalar IIS**:
  - Abre el **Administrador del servidor**.
  - Ve a **Agregar roles y características**.
  - Selecciona **Servidor web (IIS)** y completa la instalación.
- **Verificar instalación**:
  - En tu navegador, visita http://IP_PUBLICA_WINDOWS.
  - Deberías ver la página de inicio de IIS.

Paso 5: Verificar conectividad y pruebas finales.

- **Desde tu navegador local**, accede a ambas instancias mediante sus IPs públicas:
  - http://IP_PUBLICA_LINUX
  - http://IP_PUBLICA_WINDOWS
- **Desde la instancia Linux**, prueba la conectividad a la instancia Windows:
  - ping IP_PRIVADA_WINDOWS
- **Desde la instancia Windows**, prueba la conectividad a la instancia Linux:
  - Abre la terminal de comandos y ejecuta:
    - ping IP_PRIVADA_LINUX

URL Balanceador:
http://alb-seminario-1975934159.us-east-1.elb.amazonaws.com/

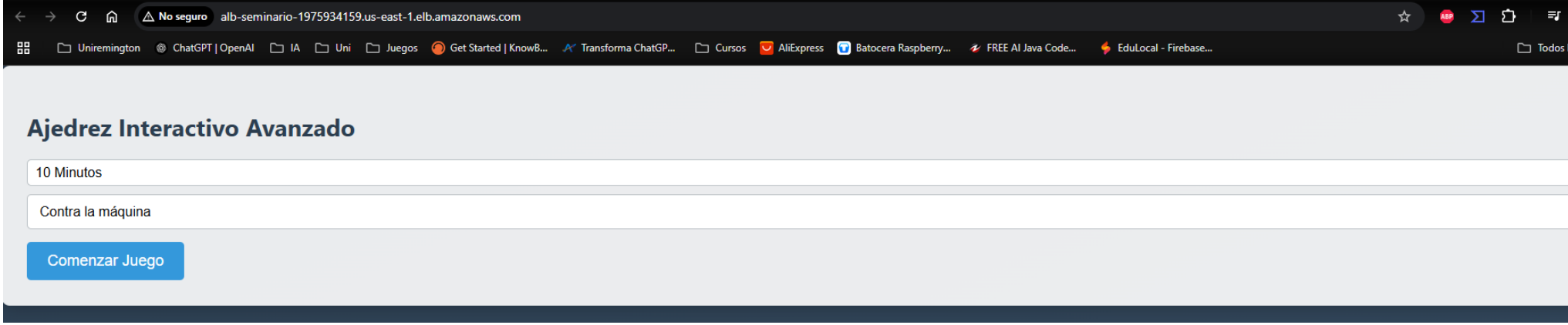

Figura 5

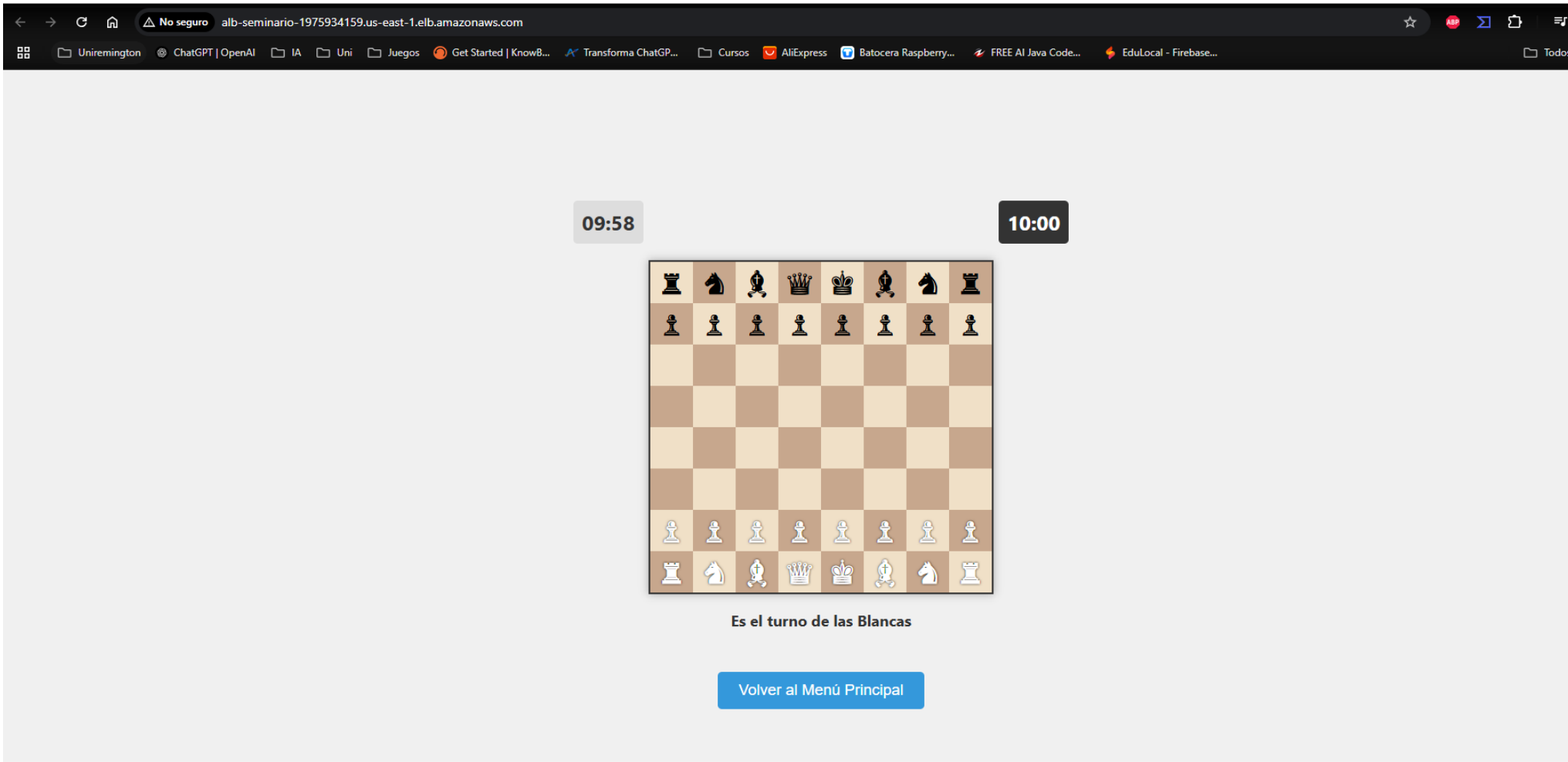

Figura 6

PARTE 2: PRÁCTICA (IMPLEMENTACIÓN Y PRUEBAS)

LINK YOUTUBE

En el siguiente Link de detalla el paso a paso de los procesos ejecutados, como se realiza y una serie de pruebas y validaciones de este ejercicio:

https://youtu.be/nrILSRFcmuM

PARTE 3: IMPLEMENTACIÓN DE DOCKER EN EC2 Y OBSERVACIÓN DE RENDIMIENTO

Configuraciones realizadas

- Instancia base (EC2):
    - Sistema Operativo: Amazon Linux 2.
    - Tipo: t2.micro (Free Tier).
    - Acceso: SSH con par de llaves (.pem).
    - Grupo de seguridad: puertos 22 (SSH), 80 (HTTP) abiertos.
- Instalación de Docker:
    - Comandos Linux utilizados
        - sudo yum update -y
            - Actualiza todos los paquetes del sistema a sus últimas versiones disponibles. El parámetro -y acepta automáticamente todas las confirmaciones.
        - sudo yum install docker -y.
            - Instala el motor de Docker desde los repositorios de Amazon Linux.
        - sudo systemctl start Docker
            - Inicia el servicio de Docker en la máquina
        - Habilitar Docker al arranque:

- sudo systemctl enable Docker
    - Hace que Docker se inicie automáticamente cada vez que se reinicia la instancia.

▪ sudo systemctl enable Docker

- Configura Docker para que se inicie automáticamente al encender la instancia.

▪ sudo usermod -aG docker ec2-user

- Agrega al usuario ec2-user al grupo docker, permitiéndole ejecutar comandos Docker sin usar sudo.

o Con estos comandos ya se tiene instalado el servicio de Docker.

- **Verificación de instalación:**

o Comandos Linux utilizados

▪ docker versión

- Muestra la versión del cliente y del servidor de Docker instalados.

▪ docker info

- Proporciona información general sobre el entorno de Docker: contenedores en ejecución, imágenes, versión, almacenamiento, entre otra.

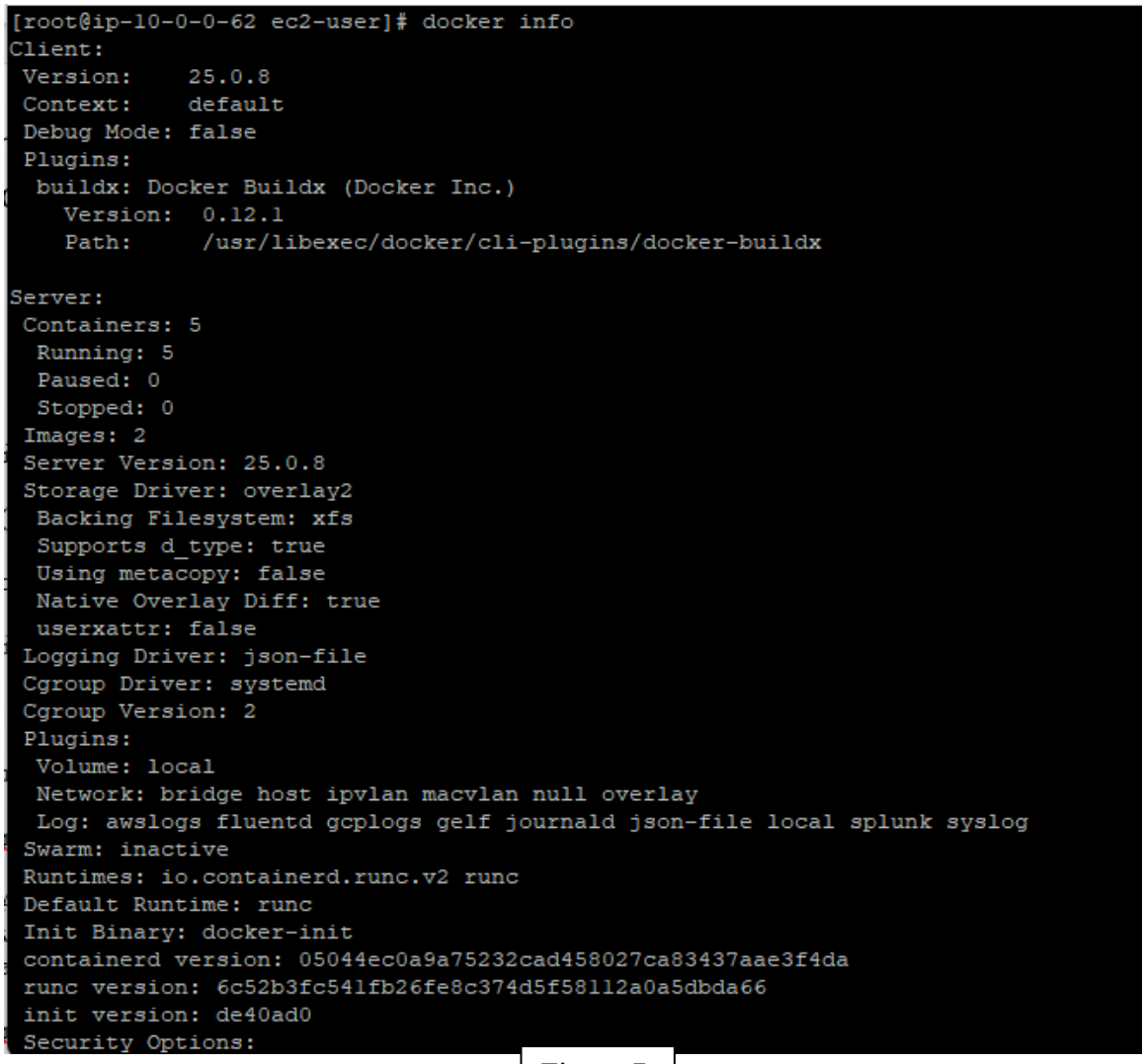

Figura 7

Instalar Apache

Para este primer caso como servidor web instalaremos Apache

- Comandos Linux utilizados
    - sudo yum install httpd -y
        - Instala Apache HTTP Server.
    - sudo systemctl start httpd
        - Inicia el servidor web.
    - sudo systemctl enable httpd

- Configuración para que se inicie automáticamente.

Cargar un contenido a el servidor.

- /home/ec2-user/app1 en esta ubicación cargaremos el contenido que deseamos para nuestra pagina en este caso el juego de ajedrez en HTML en este caso index.html.

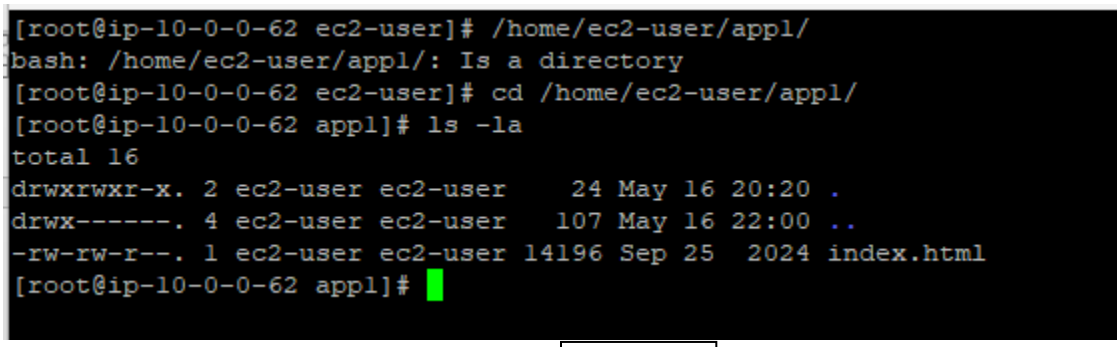

Figura 8

Ejecutar contenedores en el Apache instalado

- Comandos Linux utilizados
    - docker run -dit --name app2 --restart unless-stopped -p 8082:80 -v /home/ec2-user/app1/:/usr/local/apache2/htdocs httpd
        - Explicación del comando:
            - docker run:
                - Inicia un nuevo contenedor.
            - -dit:
                - Lo ejecuta en segundo plano (detached), interactivo y con TTY.
            - --name app2:
                - Asigna nombre al contenedor.
            - --restart unless-stopped:
                - Se reinicia automáticamente si se apaga o reinicia el host.

- -p 8082:80:
    - Mapea el puerto 8082 del host al 80 del contenedor.
- -v /home/ec2-user/app1/:/usr/local/apache2/htdocs:
    - Usa el contenido estático creado en app1 será nuestra base y el contenido que se mostrara.
- httpd:
    - Usa la imagen oficial del servidor Apache en contenedor.

- Se pueden lanzar múltiples contenedores variando el nombre app3, app4, …, junto al puerto 8083, 8084, … partiendo que no podemos repetir el nombre ni el puerto en especial este ultimo

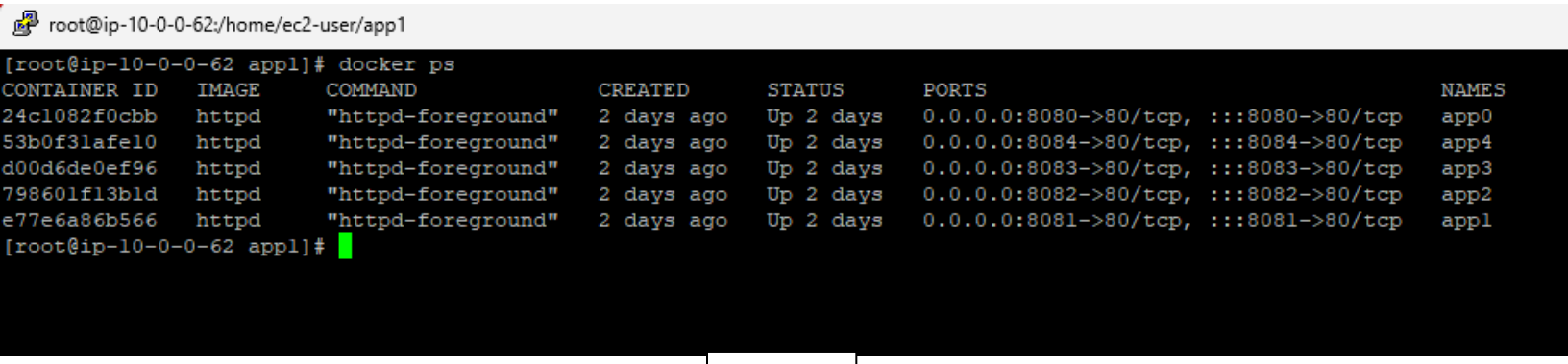

Figura 9

Nginx, Ejecución de Contenedores con Balanceador o proxy Reverso

Para este caso procedemos a instalar el servidor web Nginx para validar una opción diferente a apache.

- Comandos Linux utilizados
    - sudo yum install nginx -y
        - Instala Nginx
    - nano /etc/nginx/nginx.conf
        - Acceder por consola a un editor de texto que nos permitirá editar la configuración de Nginx.

- Nueva configuración

```
events {}

http {
   upstream seminario {


      server localhost:8082;
      server localhost:8083;
   }

   server {
      listen 80;
      server_name localhost;

      location / {
         proxy_pass http://seminario;
      }
   }
}
```

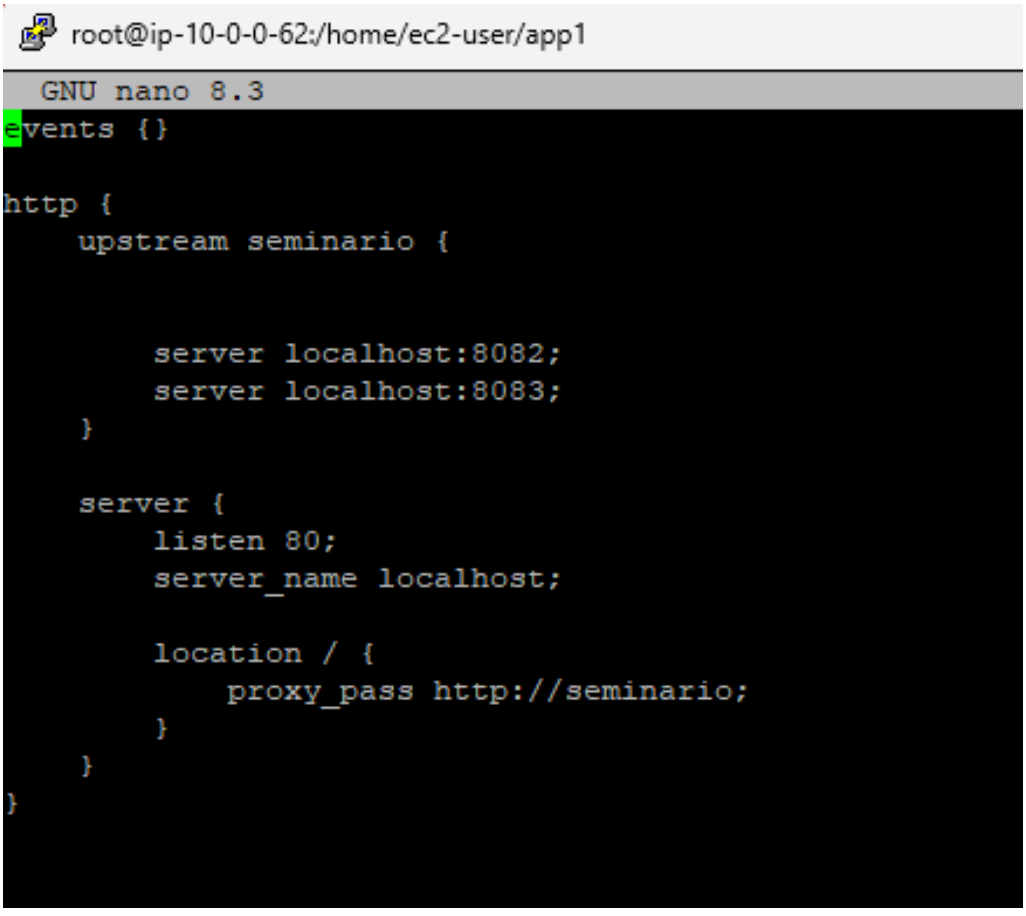

Figura 10

- sudo systemctl start nginx
  - Inicia el servicio de Nginx
- sudo systemctl enable nginx -y
  - Ejecuta el servicio automáticamente.

Acceder al contenido de los contenedores desde el navegador

- Identifica la IP pública de tu instancia EC2
  - Ve a la consola de AWS.
  - Ingresa al servicio EC2.
  - Selecciona tu instancia y copia la IP pública (por ejemplo: 54.123.45.67).
- **Accede desde el navegador:**
  - http://54.123.45.67:8081
  - http://54.123.45.67:8082

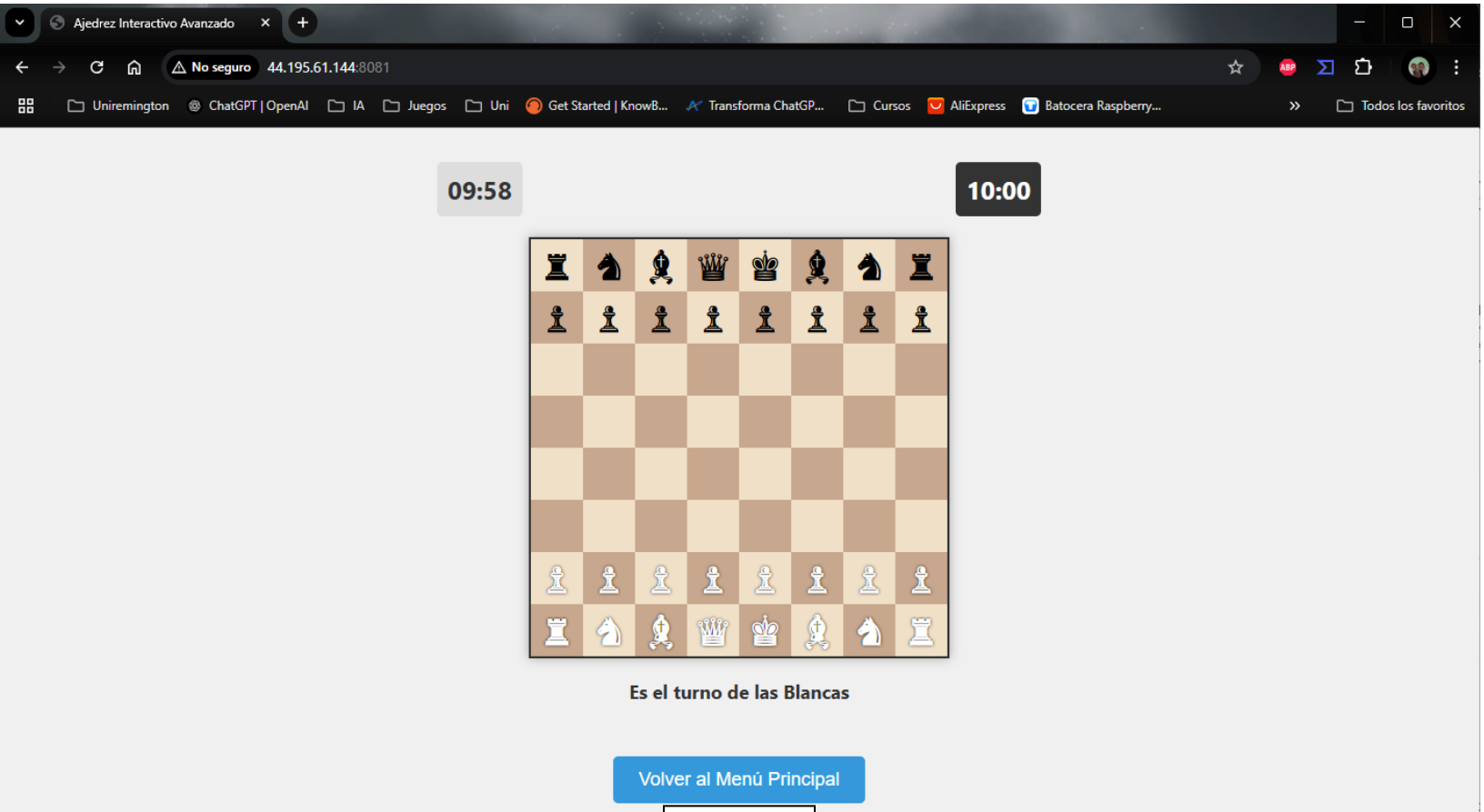

Figura 11

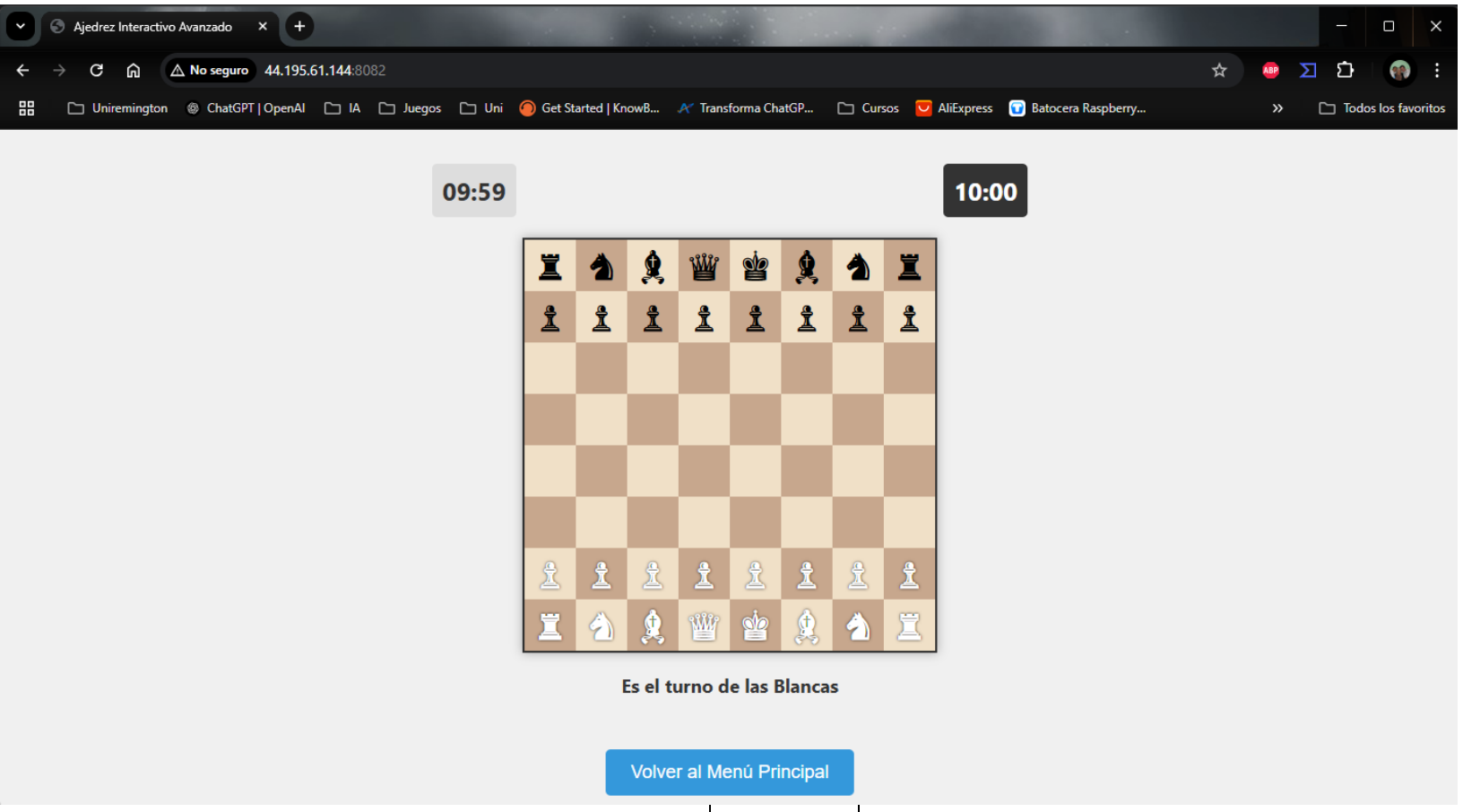

Figura 12

- **Verifica el balanceador (Nginx)**:
    - El balanceador de carga escucha en el puerto 80.
    - Para acceder al contenido distribuido entre los contenedores.
        - http://54.123.45.67 no se requiero el mencionar el puero 80 ya que este es el puerto por defauld

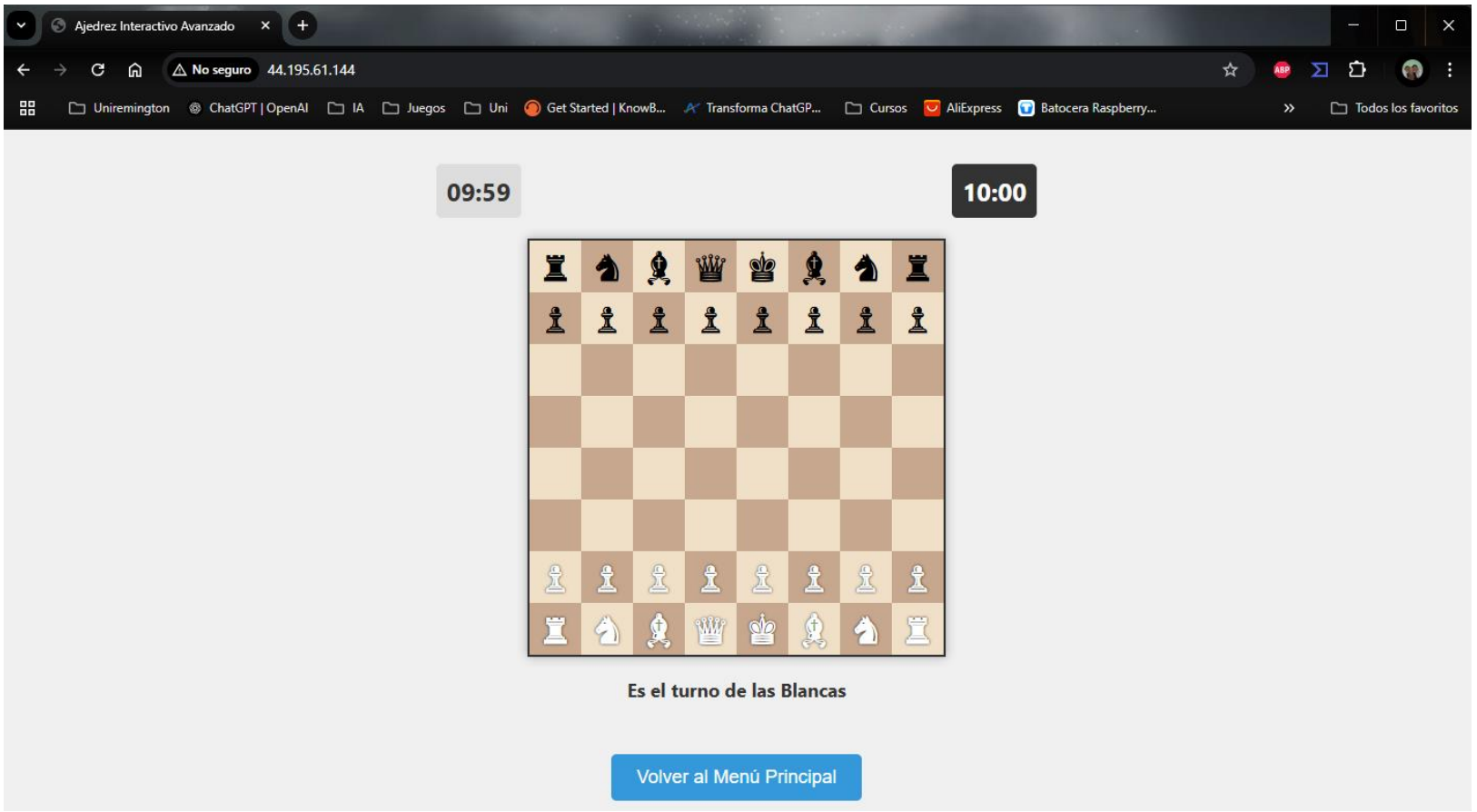

Figura 13

- **Verifica el balanceador de AWS**:
    - Vamos a la cuenta de AWS.
    - Accedemos EC2/ Load Balancers
        - Validamos la ruta que nos da el DNS en este caso LINUX-DOCKER-LB-243664107.us-east-1.elb.amazonaws.com

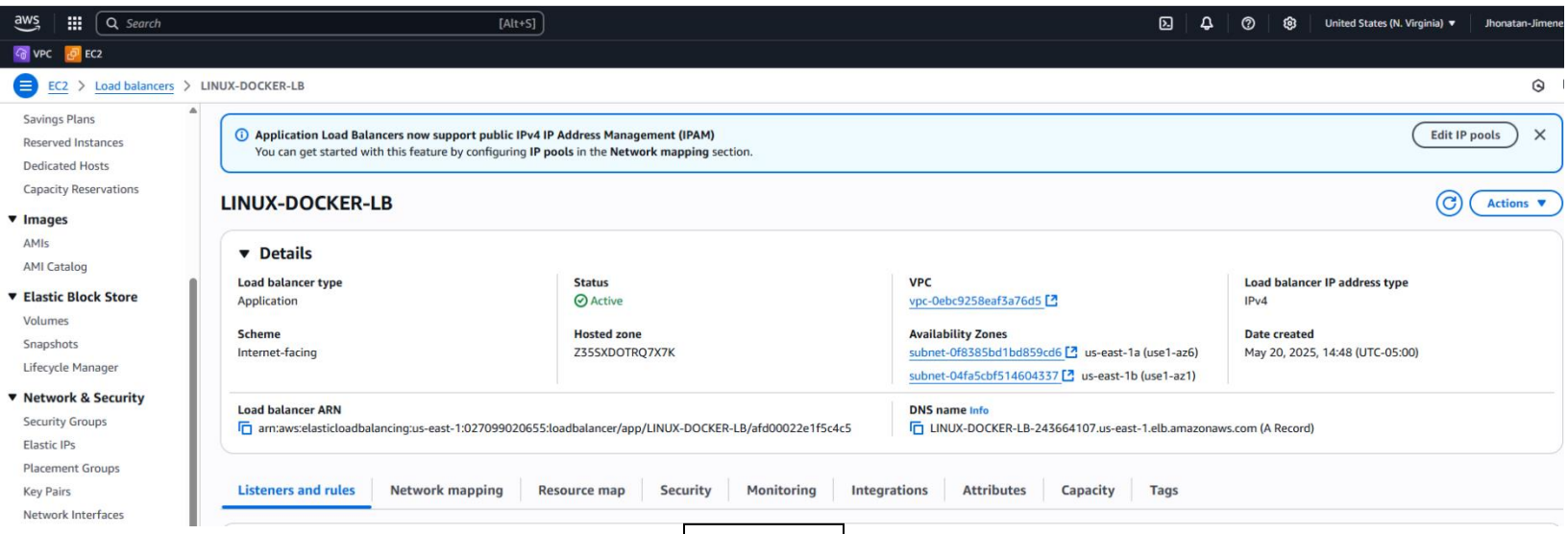

Figura 14

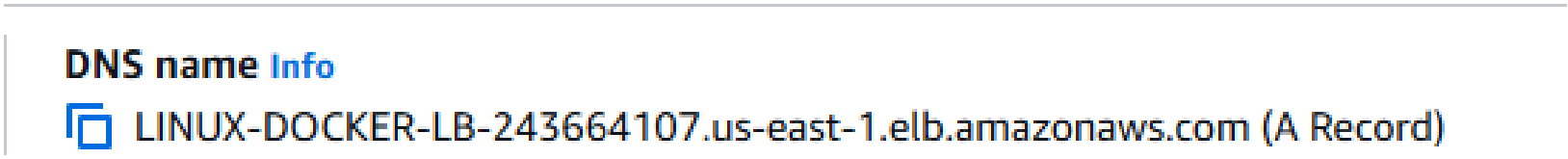

Figura 15

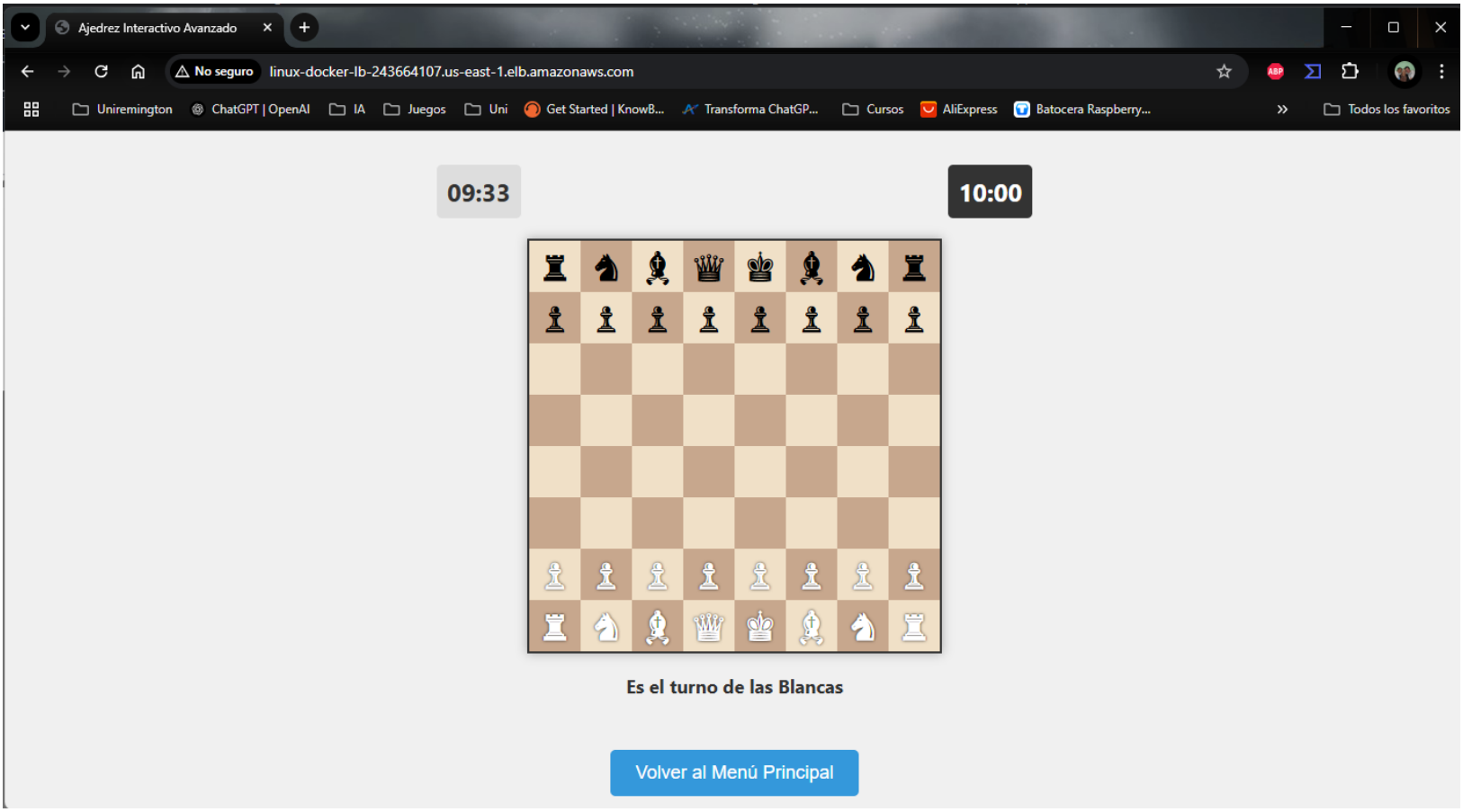

Figura 16

PARTE 4: PRÁCTICA (IMPLEMENTACIÓN Y PRUEBAS)

LINK YOUTUBE

En el siguiente Link de detalla el paso a paso de los procesos ejecutados, en el proceso de creación de los Docker y sus configuraciones junto a los servicios montados en AWS:

https://youtu.be/xvTewbUaeHw

Figuras y tablas

Conclusiones

La implementación de una arquitectura de red en AWS con servidores Windows y Linux permitió aprender conceptos clave como direccionamiento IP, configuración de reglas de firewall, y administración de instancias virtuales. Esta práctica evidenció la importancia de planificar adecuadamente la topología de red y aplicar buenas prácticas de seguridad, como el uso de grupos de seguridad restrictivos y el manejo adecuado de credenciales de acceso.

Adicionalmente, este trabajo se fortaleció con la incorporación de contenedores Docker dentro de instancias EC2, simulando un entorno de alto uso de CPU para observar el rendimiento, la eficiencia en el despliegue de múltiples aplicaciones ligeras, y el comportamiento de los servicios balanceados con Nginx. Esta fase permitió contrastar la virtualización tradicional con los beneficios de los contenedores Docker, entre ellos: rapidez de despliegue, menor consumo de recursos, escalabilidad sencilla y portabilidad.

Este ejercicio no solo es relevante como experiencia técnica, sino que también representa un punto de partida para entender cómo funcionan los servicios en la nube como EC2, VPC, Security Groups, Elastic Load Balancer y Docker. Nos brinda una visión práctica y cercana de cómo las soluciones modernas pueden aplicarse en proyectos empresariales reales.

En lo profesional, esta experiencia se traduce en habilidades concretas que son altamente demandadas en el campo de la ingeniería de sistemas. Comprender y manejar servicios de AWS, junto con herramientas de automatización y contenedores, se ha vuelto esencial en la era digital. Saber cómo diseñar, implementar y escalar una solución cloud permite ofrecer respuestas técnicas ágiles, seguras y adaptables a las necesidades de cualquier organización.

Finalmente, este trabajo deja en evidencia cómo los servicios en la nube han transformado la manera en que se construyen e implementan soluciones tecnológicas. Lejos de ser un concepto distante, hoy la nube es el ecosistema donde convergen innovación, eficiencia y oportunidad, y nosotros como ingenieros debemos estar preparados para liderar ese cambio.

Referencias

Felipe, A., & Araújo, A. (n.d.). *"CREACION DE UN CLIENTE REMOTE DESKTOP PROTOCOL MULTIPLATAFORMA" JAVARDP Por*.

https://aws.amazon.com/es/amis/oracle/. (n.d.).

https://aws.amazon.com/es/ec2/. (n.d.).

https://aws.amazon.com/es/vpc/. (n.d.).

https://aws.amazon.com/es/what-is-aws/. (n.d.).

Jinete, D., Aviles, J., & Rojsa-Cordero, A. (n.d.). *SUBORDINACIÓN DE SERVIDORES PARA CLÚSTER EN CLOUD USANDO AWS-EC2*.

Para, Q., El, O., & De, T. (n.d.). *UNIVERSIDAD AUTÓNOMA DEL ESTADO DE HIDALGO INSTITUTO DE CIENCIAS BÁSICAS E INGENIERÍA M O N O G R A F Í A*.

RolandoRamos Ali, J. (n.d.). *Infraestructura como Servicio (IaaS)*. http://translate.google.com.bo/translate?hl=e

WHITEPAPER: PROTOCOLO SSH. (n.d.).