**Laboratorio**

**Curso:** ST0263 – Tópicos Especiales en Telemática.
**Título**: Arquitecturas Confiables, Disponibles y Escalables para el Despliegue de Aplicaciones Web Monolíticas.

**Objetivo**: Implementar y configurar una arquitectura para desplegar una aplicación web escalable y flexible considerando una infraestructura de TI robusta y confiable basada en un proveedor de nube pública como AWS.

**Duración:**

**Contenido del Laboratorio**

## 1 Introducción.

Cuando se diseña considerando alta confiabilidad y disponibilidad, así como aspectos de escalabilidad, uno de los aspectos fundamentales es la definir cuantas zonas de disponibilidad vamos a considerar para desplegar nuestra infraestructura y así soportar el despliegue de la aplicación.

Al respecto, al garantizar al menos dos zonas de alta disponibilidad se va a asegurar que los recursos de aplicación estén desplegados en centros de datos separados geográficamente por varios kilómetros.

En este laboratorio, desplegaremos una infraestructura robusta y escalable para soportar el despliegue de una aplicación web desarrollada a través de un sistema de gestión de contenidos (CMS) como wordpress.

## 2 Recursos

Para el desarrollo del siguiente laboratorio se dispondrán de los siguientes recursos web a través de la cuenta de AWS educate:

- Servicio de VPC.
- Servicio de EC2.
- Servicio de Load Balancer.
- Servicio de Autoscaling.

En el servidor se desplegará una aplicación web. Para estos efectos se utilizará un CMS como wordpress y el motor de bases de datos será MySQL.

## 3 Desarrollo

En este laboratorio vamos a crear una VPC. Esta VPC va a soportar el despliegue de una aplicación web como se mencionó anteriormente, específicamente wordpress. Para lograr un despliegue con alta disponibilidad, se van a implementar dos (2) zonas de disponibilidad (Availability Zone (AZ)) al igual que servicios de balanceo de cargas (LB), así como de auto escalamiento.

Una zona de disponibilidad (AZ) es uno o más centros de datos localizados en una región de AWS. Las AZs permiten que se desplieguen aplicaciones en un entorno de producción que satisface criterios o aspectos como alta disponibilidad, tolerancia a fallas y escalabilidad.

A nivel de arquitectura de la aplicación desde la perspectiva de despliegue, se ha considerado dos subredes: una subred pública y una privada por cada AZ tal como se observa en la Figura 1.

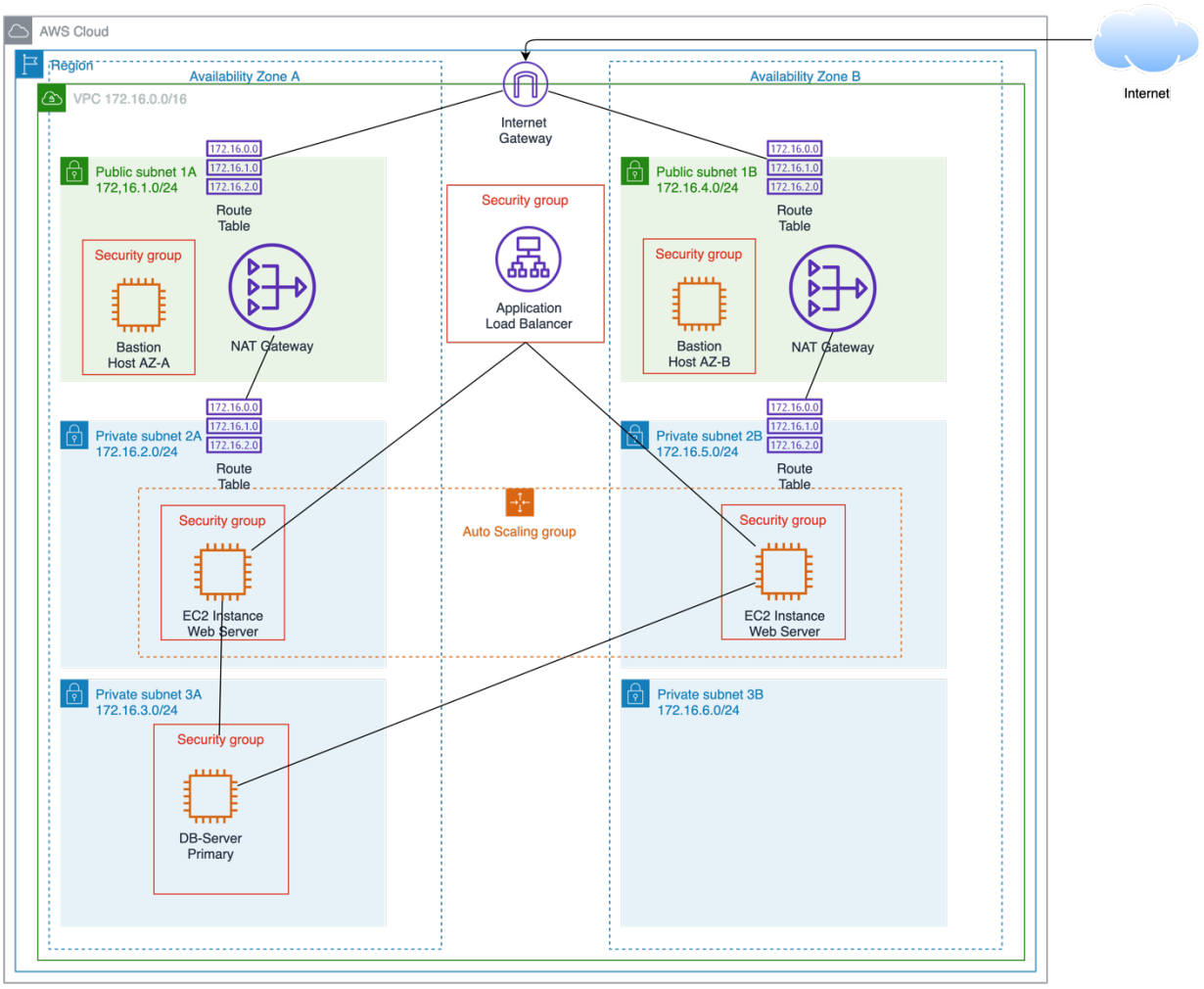



Figura 1. Vista de alto nivel para el despliegue de aplicación Web Multi Availability Zone.

Como se puede observar, para realizar el despliegue se requiere definir una Virtual Prívate Cloud (VPC). Una VPC es equivalente a una red en su centro de datos, la cual le va a permitir definir, así como desplegar un conjunto de servicios en un entorno totalmente aislado de cualquier otro usuario. Para efectos de este laboratorio y como se puede observar en la figura 1, la dirección de red asignada para la VPC es la 172.16.0.0 /16. De igual forma, esta VPC se define en dos zonas de disponibilidad, así como se compone de cuatro (4) subredes: dos (2) públicas y (2) privadas. De esta forma, se deben definir y crear las subredes en cada una de las zonas de disponibilidad asociadas a la VPC.

En las subredes públicas se debe localizar todos los recursos como Bastion Host (BH) y NAT Gateway. Tenga en cuenta que el BH se define como una entidad que permite el acceso seguro a instancias EC2 Linux que se encuentran localizada en la subred privada de la VPC. El acceso seguro se da a través del establecimiento de una sesión ssh entre una máquina cliente y el BH. A partir de esta, entonces usted se conecta a las otras instancias. Por otro lado, el servicio de NAT Gateway, es un dispositivo/servicio que

permite la operación de Network Address Translation (NAT). De esta forma, este servicio o funcionalidad es implementada con el fin de que las estaciones (EC2 instances) ubicadas en las subredes privadas, puedan conectarse a servicios o recursos por fuera de la VPC (p.ej., conectarse a Internet). Sin embargo, servicios por fuera de la VPC no pueden establecer conexión con las máquinas en la subred privada a través del NAT Gateway.

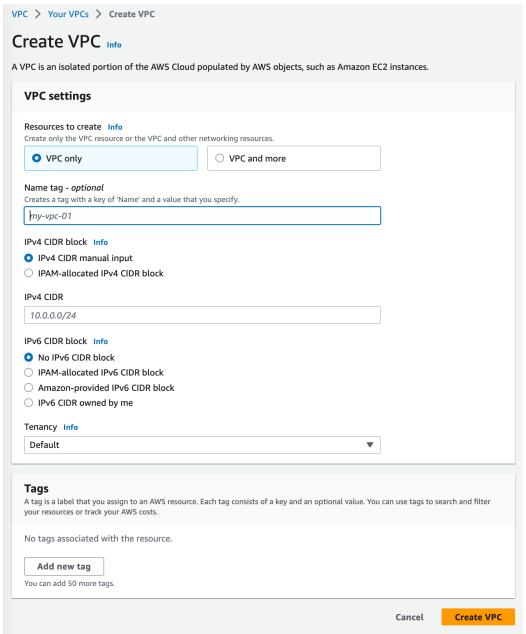
En la subred privada, se deben localizar, tanto la instancia que soporta la aplicación CMS así como la capa de persistencia de bases de datos.

Para lograr desplegar la arquitectura diseñada y propuesta para este laboratorio, se debe proceder y realizar los siguientes pasos:

- Crear la VPC.
    - Crear las subredes públicas y privadas en dos zonas de disponibilidad
    - Crear un Internet Gateway.
    - Crear y configurar un NAT Gateway.
- Crear los security groups que se requieren.
- Crear las diferentes instancias: bastion host, web server, bases de datos.
- Crear la AMI a partir del web server instalado.
- Crear y configurar el balanceador de carga.
- Crear y configurar el grupo de autoscaling

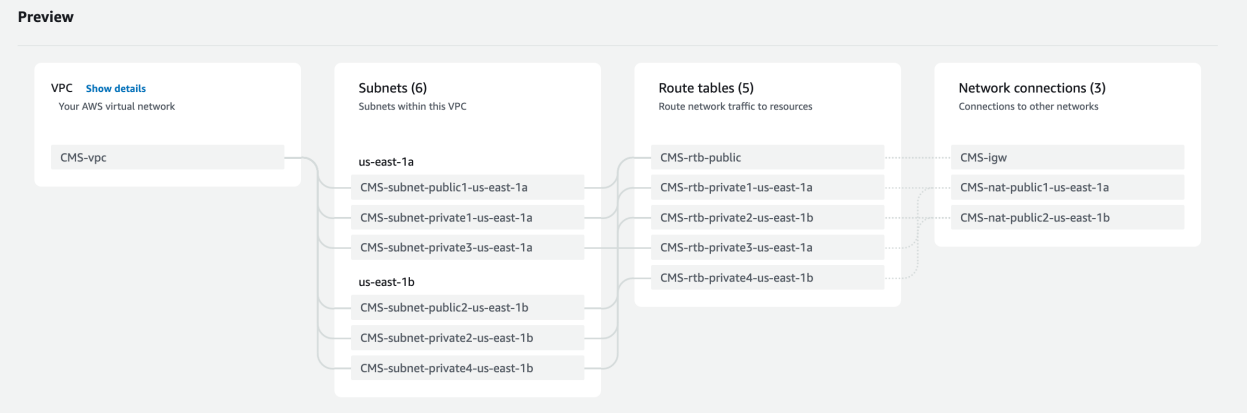
## 4 Crear y configurar la VPC.

- En la consola de AWS seleccione el servicio de VPC.
- En el panel izquierdo seleccione la opción de “Your VPCs”
- Click en Create VPC. Y debe observar una imagen como la siguiente:



- Seleccione la opción que indica “VPC and more”, y configure los siguientes parámetros:
    - **Autogenerate**: CMS
    - IPv4 CIDR **block:** 172.16.0.0/16
    - Availability Zones: 2. Seleccione las AZs us-east-1a y us-east-1b.
        - Customize AZs:
            - Number of public subnets: 2
            - Number of private subnets: 4
        - Customize subnet CIDR blocks:
            - Public subnet CIDR block in us-east-1a: 172.16.1.0 /24
            - Public subnet CIDR block in us-east-1b: 172.16.4.0 /24
            - Private subnet CIDR block in us-east-1a: 172.16.2.0 /24
            - Private subnet CIDR block in us-east-1b: 172.16.5.0 /24
            - Private subnet CIDR block in us-east-1a: 172.16.3.0 /24
            - Private subnet CIDR block in us-east-1b: 172.16.6.0 /24
            -
    - NAT Gateway: 1 per AZ (Seleccione uno por Zona de Disponibilidad).
    - VPC endpoints: None.

Si observa el preview, podrá observar una imagen parecida a la que se muestra a continuación.




- Click en Create VPC.

## 5 Bastion Host.

### 5.1 Crear security group.

En este apartado usted creará un grupo de seguridad el cual oficiará como un firewall virtual. De esta forma, cuando usted lance una instancia, usted asociará uno o varios grupos de seguridad.

En el panel de navegación ubicado en la izquierda, en la sección de **“Security”.** Seleccione la opción de “**Security Groups”.** Allí puede ver los diferentes **“security groups”** configurados. En el servicio de VPC, escoja la opción “create security group” y configure los siguientes parámetros:

**Basic Details**

- **Security group name:** SG-BastionHost
- **Description:** *Enable SSH Access*
- **VPC:** CMS-vpc

En la pestaña de **"Inbound rules"**:

- Click "Add rule". Configure los siguientes parámetros:
  - **Type:** *SSH*
  - **Source:** *Anywhere-IPv4*
  - **Description:** Allow ssh traffic

Click en "create security group".

5.2 Crear Instancia Bastion Host.

En esta sección crearemos una instancia EC2 la cual actuará como Host Bastion. Recuerde que esta VM estará asociada a la subred pública de la VPC por cada zona de disponibilidad.

Diríjase al **"home"** de la consola de administración de AWS. Escoja el servicio de EC2. En el panel izquierdo seleccione la opción de "**Instances**" seleccione la opción "**launch instances**" y ejecute lo siguientes pasos: En esta sección crearemos una instancia EC2 la cual actuará como Host Bastion. Recuerde que esta VM estará asociada a la subred pública de la VPC.

Diríjase al **"home"** de la consola de administración de AWS. Escoja el servicio de EC2. En el panel izquierdo seleccione la opción de "**Instances**" seleccione la opción "**launch instances**" y ejecute lo siguientes pasos:

- **Name and Tags**
  - Name: i-BastionHost.
  - Application and OS Images (Amazon machine image)
    - Escoja la imagen de Ubuntu Server 22.04 LTS (HVM), SSD Volutme Type. Free tier elegible.
  - Instance type: Seleccione el tipo de instancia **t2.micro (**columna **type)**
  - Key pair (login): Seleccione una llave existente o en su defecto cree una nueva.
  - Network Settings: Ahora configure, los siguientes parámetros (click en edit):
    - **VPC:** *CMS-vpc*
    - **Subnet:** *CMS-Subnet-public1-us-east-1a*
    - **Auto-assign Public IP:** *Enable*
    - Firewall (security groups):
      - Seleccione la opción de un security group existente.
      - Common Security Groups
        - Seleccione "SG-BastionHost"
    - Configure storage:
      - 1 x 8 Gib gp2 root volume

Al final puede ver lo siguiente en la vista de summary.

# 6 Instalar y Configurar la capa de Bases de Datos

## 6.1 Security Group para la Bases de Datos.

En el servicio de VPC, escoja la opción “Create security group” y configure los siguientes parámetros:

- **Basic details:**
  - **Security group name:** *SG-DB-CMS*
  - **Description:** *Allow SQL Access*
  - **VPC:** CMS-vpc
- **Inbound rules**: Click “Add rule”. Configure los siguientes parámetros:
  - **Type:** *MySQL/Aurora*
  - **Source:** *Custom. 172.16.2.0/24*
  - **Description:** Allow connections to the DB
- Click “Add rule”. Configure los siguientes parámetros:
  - **Type:** *MySQL/Aurora*
  - **Source:** *Custom. 172.16.5.0/24*
  - **Description:** Allow connections to the DB
- Click “Add rule”. Configure los siguientes parámetros:
  - **Type:** *SSH*
  - **Source:** custom. 172.16.1.0 /24
  - **Description:** Allow ssh traffic

De esta forma se configura el security group para la instancia de la base de datos. Esta configuración permite aceptar las peticiones entrantes sobre el puerto 3306 desde cualquier instancia EC2 (WebApp) ubicadas en las subredes privadas de los segmentos de red 172.16.2.0/24 y 172.16.5.0/24.

## 6.2 Crear la instancia del servidor de Bases de Datos.

Diríjase al **“home”** de la consola de administración de AWS. Escoja el servicio de EC2. En el panel izquierdo seleccione la opción de “**Instances**” seleccione la opción “**launch instances**” y ejecute lo siguientes pasos:

- Name and Tags
  - Name: i-DB
  - Application and OS Images (Amazon machine image)
    - Escoja la imagen de Amazon Machine Image (AMI) la cual contiene la imagen del sistema operativo. Seleccione Ubuntu server 22.04 LTS (HVM), SSD Volume Type. Free tier elegible.
  - Instance type: Seleccione el tipo de instancia **t2.micro (**columna **type)**
  - Key pair (login): Seleccione una llave existente o en su defecto cree una nueva.
  - Network Settings: Ahora configure, los siguientes parámetros (click en edit):
    - **VPC:** *CMS-vpc*
    - **Subnet:** *CMS-subnet-private2-us-east-1a (AZ:us-east-1a, CIDR:172.16.3.0/24)*
    - **Auto-assign Public IP:** *Disable*
    - Firewall (security groups):
      - Seleccione la opción de un security group existente.
      - Common Security Groups
        - Seleccione *“SG-DB-CMS”*

        - Configure storage:
            - 1 x 8 Gib gp2 root volume
- Click en “Launch instance”.

## 6.3 Conectarse al servidor de bases de datos (i-DB) via SSH.

En términos generales para efectos de poner a instalar y configurar los diferentes servidores (p.ej., web y bases de datos) se debe emplear una conexión ssh. Para esto debe tener en cuenta que cuando está creando las instancias debe utilizar la llave que género (p.ej., archivo con extensión *.pem).

Para el caso del poder configurar y administrar el servidor de bases de datos de forma segura, debe pasar por el bastion host y, una vez ubicado en el Bastion Host, debe levantar una sesión ssh hasta los servidores ubicados en la subred privada, en este caso, el servidor de bases de datos (p.ej, DB). Para esto vamos a emplear SSH agent forwarding y así de esta forma, vamos a conectarnos de manera segura a las instancias localizadas en la subred privada (p.ej., I-DB-CMS)

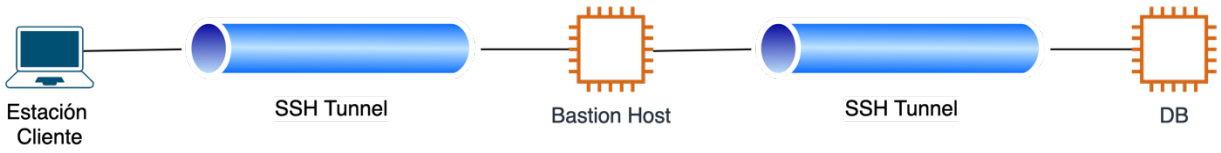



*Figura 2. Conexión SSH*

Es así como para lograr esto, debe aplicar el siguiente comando ubicado en la ruta donde se encuentra almacenada su llave. Tenga en cuenta que esto es su estación de trabajo. Recuerde que para poder conectarse a través de una conexión ssh, requiere un cliente ssh. Para estaciones Windows, bien puede utilizar Putty o bien puede utilizar “Power Shell” y se hace necesario configurar de igual forma la opción de “agent forwarding”.

Para el caso de estaciones de trabajo con sistema operativo MAC OS o Linux, el cliente ssh viene nativamente. Para efectos de este laboratorio, asumimos que estamos utilizando una máquina con sistema operativo Linux o MAC OS y la llave (p.ej. MyKey.pem) se encuentra en el directorio “keys”. Para esto ubíquese en dicho directorio:

```
$ cd /home/keys
$ chmod 400 MyKey.pem
$ ssh-add -K MyKey.pem
$ ssh-add -L MyKey.pem
```

Ahora se procede a conectarse al bastion host aplicando el siguiente comando:

```
$ ssh -A <user@nombre de la máquina Bastion Host>
```

Una vez este en la instancia del bastion host, se inicia una sesión ssh contra su servidor de bases de datos, tal como se observa en la figura 2.

```
$ ssh <user@nombre de la máquina i-DB-Host>
```

### 6.3.1 Instalar el servidor de bases de datos

A continuación, se presenta el conjunto de comandos para instalar el servidor de bases de datos mariadb/mysql.

Primero se actualizarán el índice de paquetes en su instancia, para esto ejecute la secuencia de comandos:

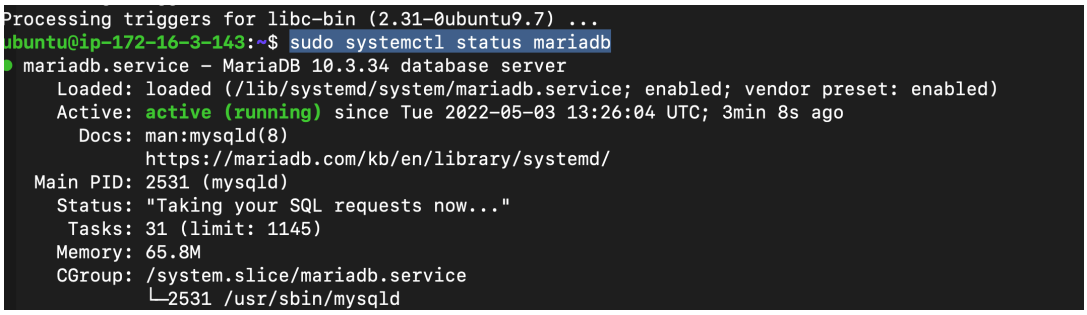
$ sudo apt-get update -y

Ahora, vamos a proceder a instalar el motor de bases de datos mariaDB. Por favor ejecute el siguiente comando:

$ sudo apt-get install mariadb-server

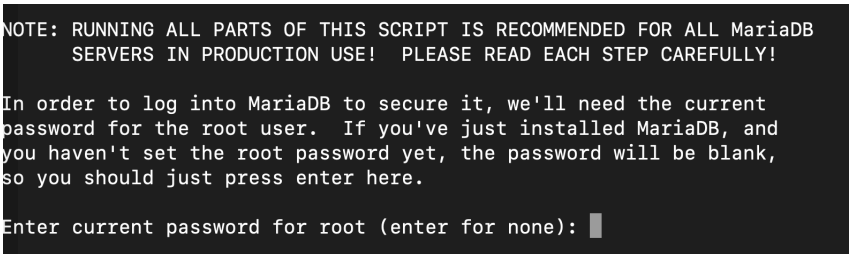
Por favor verifique que el servicio este corriendo, para esto digite el comando:

$ sudo systemctl status mariadb




Ahora, se procederá a configurar la base de datos para que pueda ser utilizada por el CMS, en este caso, wordpress.

$ sudo mysql_secure_installation




Presione enter, para indicar que no tenemos password de root configurado aún. A continuación, se puede observar un conjunto de preguntas a las cuales puede contestar que si tal como se muestra en la figura a continuación.

```
Switch to unix_socket authentication [Y/n] n
 ... skipping.

You already have your root account protected, so you can safely answer 'n'.

Change the root password? [Y/n] n
 ... skipping.

By default, a MariaDB installation has an anonymous user, allowing anyone
to log into MariaDB without having to have a user account created for
them.  This is intended only for testing, and to make the installation
go a bit smoother.  You should remove them before moving into a
production environment.

Remove anonymous users? [Y/n] Y
 ... Success!

Normally, root should only be allowed to connect from 'localhost'.  This
ensures that someone cannot guess at the root password from the network.

Disallow root login remotely? [Y/n] Y
 ... Success!

By default, MariaDB comes with a database named 'test' that anyone can
access.  This is also intended only for testing, and should be removed
before moving into a production environment.

Remove test database and access to it? [Y/n] Y
 - Dropping test database...
 ... Success!
 - Removing privileges on test database...
 ... Success!

Reloading the privilege tables will ensure that all changes made so far
will take effect immediately.

Reload privilege tables now? [Y/n] Y
 ... Success!

Cleaning up...

All done!  If you've completed all of the above steps, your MariaDB
installation should now be secure.

Thanks for using MariaDB!
```

A continuación, por favor, se sugiere que no haga copiar y pegar de estos comandos. Digítelos usted manualmente.

$ sudo mariadb

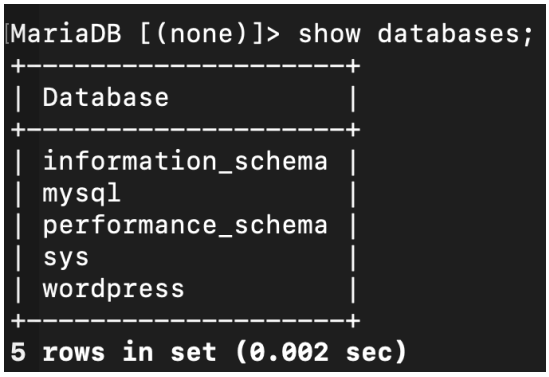
MariaDB [(none)]> GRANT ALL ON *.* TO 'admin'@'localhost' IDENTIFIED BY 'pass123' WITH GRANT OPTION;

MariaDB [(none)]> FLUSH PRIVILEGES;

MariaDB [(none)]>  CREATE DATABASE wordpress DEFAULT CHARACTER SET utf8 COLLATE utf8_unicode_ci;

Verifiquemos que se haya creado exitosamente la base de datos para wordpress, para eso digite el comando

MariaDB [(none)]> show databases;




MariaDB [(none)]> CREATE USER 'wpuser'@'172.16.2.%' IDENTIFIED BY 'wppassword';
MariaDB [(none)]> CREATE USER 'wpuser'@'172.16.5.%' IDENTIFIED BY 'wppassword';
MariaDB [(none)]> GRANT ALL PRIVILEGES ON *.* TO 'wpuser'@'172.16.2.%';
MariaDB [(none)]> GRANT ALL PRIVILEGES ON *.* TO 'wpuser'@'172.16.5.%';
MariaDB [(none)]> FLUSH PRIVILEGES;
MariaDB [(none)]> exit

Con los comandos anteriores se crea el usuario 'wpuser' con el password 'wppassword' y se le da privilegio sobre las diferentes tablas que se tiene configurado sobre la bases de datos.

Finalmente, le vamos a indicar a mariadb que permita conexiones remotas. Para esto vamos a modificar la opción 'bind-address' en este archivo. De esta forma, busque la línea y modifique el valor.

sudo nano /etc/mysql/mariadb.conf.d/50-server.cnf
bind-address = 0.0.0.0

Salga y guarde los cambios con ctrlX + Y+ enter.

Reinicie el servicio de la base de datos:

$ sudo systemctl restart mariadb

## 7 Configurando la Capa Web

7.1 Crear el security para el tráfico Web

- **Basic Details.** Configure los siguientes parámetros:
  - **Security group name:** SG-WebCMS
  - **Description:** *Enable HTTP Access*
  - **VPC:** CMS-vpc

En la pestaña de **"Inbound rules"**, click en "add rule". Configure los siguientes parámetros:

- **Type:** *HTTP*
- **Source:** *Custom. 172.16.1.0 /24*
- **Description:** *Permit Web Requests*

Click en “add rule”:

- **Type:** *HTTP*
- **Source:** *Custom. 172.16.4.0 /24*
- **Description:** *Permit Web Requests*

Click en “add rule”:

- **Type:** *SSH*
- **Source:** *Custom. 172.16.1.0/24*
- **Description:** *Permit ssh connections*

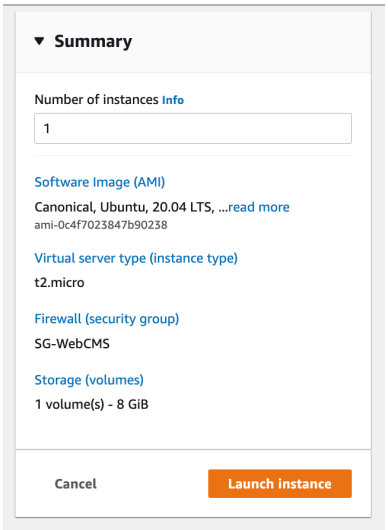
Click **“Create security group”.**

7.2 Crear y configurar la Instancia del Wervidor Web.

En esta sección crearemos una instancia EC2 la cual actuará como Web Server. Recuerde que esta VM estará asociada a la subred privada 1 de la VPC. Diríjase al **“home”** de la consola de administración de AWS. Escoja el servicio de EC2. En el panel izquierdo seleccione la opción de “**Instances**” seleccione la opción “**launch instances**” y ejecute lo siguientes pasos:

- Name and Tags
    - Name: i-WebServer.
    - Application and OS Images (Amazon machine image)
        - Escoja la imagen de Amazon Machine Image (AMI) la cual contiene la imagen del sistema operativo. Seleccione Ubuntu server 22.04 LTS (HVM), SSD Volume Type. Free tier elegible.
    - Instance type: Seleccione el tipo de instancia **t2.micro (**columna **type)**
    - Key pair (login): Seleccione una llave existente o en su defecto cree una nueva.
    - Network Settings: Ahora configure, los siguientes parámetros (click en edit):
        - **VPC -required:** *CMS-vpc*
        - **Subnet:** *CMS-subnet-private1-us-east-1a (CIDR 172.16.2.0/24)*
        - **Auto-assign Public IP:** *Disable*
        - Firewall (security groups):
            - Seleccione la opción de un security group existente.
            - Common Security Groups
                - Seleccione “SG-WebCMS”
        - Configure storage:
            - 1 x 8 Gib gp2 root volume

Al final puede ver lo siguiente en la vista de summary.




7.3 Instalar y configurar el Servidor Web/PHP.

Una vez conectado a la máquina del servidor web via ssh, se procede a instalar apache y PHP. Para esto se requiere que ejecute los siguientes comandos.

$ sudo apt-get update -y

Ahora procederemos a instalar Apache, para esto por favor digite el siguiente comando:

$ sudo apt-get install apache2

Ahora, procedemos a instalar PHP, el cual es necesario para poder desplegar el wordpress.

$ sudo apt-get install php libapache2-mod-php php-mysql

En el caso particular de wordpress, se requieren algunos complementos a nivel de php, los cuales se instalan con el siguiente comando:

$ sudo apt-get install php-curl php-gd php-mbstring php-xml php-xmlrpc php-soap php-intl php-zip

En este punto se hace necesario reiniciar el servicio de apache, por favor digite el siguiente comando:

$ sudo systemctl restart apache2

Se procede a crear un directorio en la ruta /var/www/

$ sudo mkdir /var/www/wordpress

Vamos a habilitar el uso de archivos .htaccess Para esto vamos a modificar el archivo 000-default.conf que está localizado en la ruta /etc/apache2/sites-available. Para esto aplique el siguiente comando:

$sudo nano /etc/apache2/sites-available/000-default.conf

Modifique la línea de DocumentRoot de la siguiente forma:

DocumentRoot /var/www/wordpress

Ahora, agregue la siguiente directiva:

```
<Directory /var/www/wordpress/>
        AllowOverride All
</Directory>
```

Ahora, procedemos a habilitar el modo de sobreescritura.

$ sudo a2enmod rewrite

En este punto se hace necesario reiniciar el servicio de apache, por favor digite el siguiente comando:

$ sudo systemctl restart apache2

Se procede a descargar el archivo fuente de wordpress. Para esto, vamos primero a crear un directorio y desde allí se ejecuta el comando para la descarga.

$ sudo mkdir descarga
$cd descarga
$ sudo wget https://wordpress.org/latest.tar.gz

Descomprimimos el archivo:

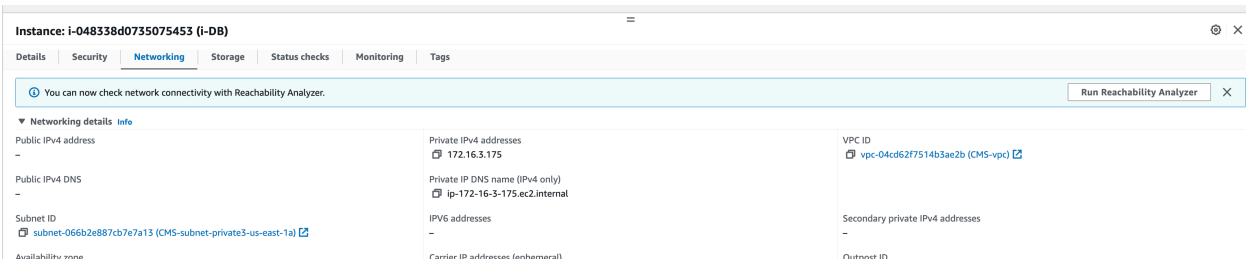
$ sudo tar zxvf latest.tar.gz
$ cd wordpress
$ sudo cp wp-config-sample.php wp-config.php

#Aqui se configuran los parámetros definidos durante la instalación y configuración de mariadb/mysql en el archivo creado…

$ sudo nano wp-config.php

Por favor modifique los siguientes parámetros tal como se indica a continuación:

```
define( 'DB_NAME', 'wordpress' );
define( 'DB_USER', 'wpuser' );
define( 'DB_PASSWORD', 'wppassword' );
define( 'DB_HOST', 'DirIPPrivadaDBServer'); //Debe buscar la dirección IP privada del servidor empleado para la bases de datos
```




Ahora vamos a proceder a crear un archivo health.html, el cual será empleado para validar que la instancia está operativa (se empleará en la configuración del target group). Para esto digite el siguiente comando:

```
$ sudo touch health.html
```

A continuación, se procede a copiar el contenido de la carpeta wordpres a esta nueva ruta:

```
$ sudo cp -a /home/ubuntu/descarga/wordpress/. /var/www/wordpress
```

Se procede a dar los permisos al usuario y grupo www-data quien se encarga de ejecutar el servicio de apache. Dicho usuario debe tener los permisos necesario para poder leer y escribir en los archivos de wordpress.

```
$ sudo chown -R www-data:www-data /var/www/wordpress
```

A continuación, se requiere establecer permisos correctos de los directorios y archivos de la carpeta wordpress:

```
$ sudo find /var/www/wordpress/ -type d -exec chmod 750 {} \;
$ sudo find /var/www/wordpress/ -type f -exec chmod 640 {} \;
```

## 8 Configurando el Servicio de Balanceador de Cargas.

Hasta este punto, se ha configurado la capa de persistencia de datos. En esta se ha configurado el servidor de bases de datos. Igualmente, se configuró la capa de aplicación, en la cual se creó la instancia del servidor web. Sin embargo, dado que la instancia del servidor web se encuentra en una red privada, aún no podemos alcanzarlo desde Internet. Para poder solventar esto, se hace necesario configurar un balanceador de carga o proxy inverso. Este dispositivo o servicio lo que nos permitirá es recibir las peticiones y redireccionarlas al servidor web.

Esto lo puede lograr de dos formas:

- La primera es creando una instancia en la subred pública y desplegando un proxy inverso (p.ej., NGINX, apache, HaProxy, etc). Para esto debe ubicar una instancia EC2 en cada zona de disponibilidad (AZ) e instalar y configurar la solución que usted seleccione. Es importante notar que en este escenario hay que configurar el servicio de DNS para que distribuya las peticiones a cada uno de los proxys inversos configurados en las subredes públicas (172.16.1.0/24 y 172.16.4.0/24).

- La segunda alternativa es empleando un servicio gestionado de un proveedor de nube. Para efectos de este laboratorio, se procederá a emplear esta segunda opción. Es así como se configurará el servicio de ELastic Load Balancer (ELB) de AWS. De esta forma, las peticiones entrantes serán recibidas por el ELB y las redireccionará a la capa de aplicación.

En esta sección, se procederá a configurar el servicio ELB de AWS. Este será el encargado de recibir las peticiones entrantes http al puerto 80 y las redirecciona al servidor CMS.

## 8.1 Creación de AMI.

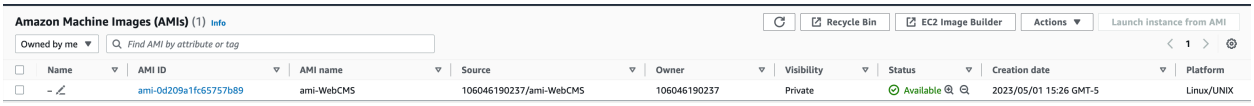
A continuación, vamos a crear una AMI del servidor web que contiene el wordpress. De esta forma se guardarán el contenido del boot disk y las nuevas instancias desplegadas a partir de esta, se van a instanciar con un contenido idéntico. Es así como una Amazon Machine Image se convierte en una plantilla que contiene una configuración básica la cual sirve para instanciar posteriormente máquinas.

Para crear una AMI, en el home de la consola de administración, seleccione el servicio de EC2.

- En el panel izquierdo, click en Instances. Confirme que la instancia de la cual vamos a realizar la AMI este corriendo (Running, 2/2 checks passed). Click en refresh para actualizar si es necesario.
- Seleccione i-WebServer.
- En el menú de Actions, click en Image and Templates> Create Image y configure los siguientes parámetros:
    - Image name: ami-WebCMS
    - Image description: Lab AMI for Web Server
    - Click en Create Image. Observará un mensaje de confirmación que muestra el AMI ID para la nueva AMI.
    - Click en Close.

Ahora podrá usar esta AMI en el servicio de Auto Scaling.

Ahora, se debe verificar que la imagen ya está disponible para su uso. En el menú izquierdo, seleccione Images, click en AMIs. Se debe esperar que la imagen se muestre como disponible, tal como se ilustra en la figura.




## 8.2 Creación de Target Group.

A continuación, se procede a la creación de un Target Group. Para esto, en el menú lateral izquierdo, en la sección de Load Balancing, se selecciona la opción y da click en Target Groups.

**Creación de Target Group:**

- Basic Configuration

- Choose a target type: Instances.
- Target group name: tg-CMS
- Protocol: HTTP: 80
- VPC: CMS-vpc.
- Protocol version: HTTP1

- Health Checks.
  - Health check path: /health.html

Expanda la sección de Health Checks y configure los siguientes parámetros:

- Healthy threshold: 2
- Inteval: 10
- Success codes: agregue una coma (,) y los valores 300-399. Debe quedar de la siguiente forma: 200,300-399

Estos parámetros lo que indican es que se va a estar chequeando cada 10 segundos las instancias y si éstas dan la respuesta de forma adecuada dos en una fila, se consideran que son instancias que están operando de manera correcta.

Click en Next. En este momento aparece la ventana de Register targets

Registers Target

- Available instances:
- Review Target:

**Nota**: Por favor no realice ninguna configuración en este paso.

Click en 'create target group'.

## 8.3 Creando Balanceador de Carga

Ahora se procede a configurar el balanceador de carga. En el menú de EC2, por favor localice la opción de 'Load Balancers'. Click en "Create load balancer". Seleccione la opción de 'Application Load Balancer (ALB)'

**Basic Configuration.**

- Load Balancer Name: lb-WebCMS
- Scheme: Internet Facing.

**Network Mappings**

- VPC: CMS-vpc
- Mappings: Aquí se debe seleccionar las dos zonas de disponibilidad que definimos en nuestra arquitectura.
  - Marque la casilla para la AZ1 (us-east-1a).
    - Seleccione la subred pública de esta AZ. (CMS-subnet-public1-us-east-1a)
  - Marque la casilla para la AZ2 (us-east-1b)

        - Seleccione la subred pública de esta AZ. (CMS-subnet-public2-us-east-1b)
- Security Groups: En caso tal aparezca un security group seleccionado, elimínelo.
    - Ahora se procede a crear un nuevo security group (SG-LB). Se debe crear con una regla que permita el ingreso de tráfico http al puerto 80 desde cualquier dirección IPv4 e IPv6 (Anywhere).
    - Una vez creado el security group, selecciónelo.

**Listeners and Routing:**

- Listener:
    - Protocol: HTTP
    - Port:80
    - Forward to: En esta sección, se debe crear o seleccionar un target group.
        - Seleccione el target group creado en el punto anterior 8.2. En caso tal no aparezca, de click en refresh. Si no aparece, seleccione la opción crear uno y ejecute los pasos del punto 8.2.
- Click en "create load balancer".

## 9 Configurando Servicio de Launch Template y AutoScaling Group.

En esta sección se procederá a configurar el servicio de autoscaling, de tal forma que se permita de forma automática y basado en algún criterio poder aumentar o disminuir el número de instancias que se tienen del CMS desplegado. De esta manera, las instancias se van desplegando y distribuyendo en las dos zonas de disponibilidad que se tienen definidas y lograr así la alta disponibilidad en el servicio que se ofrece.

Para esto, primero vamos a crear una imagen base de nuestro servidor CMS.

### 9.1 Crear y configurar Launch Template.

En el home de la consola de administración, seleccione el servicio de EC2.

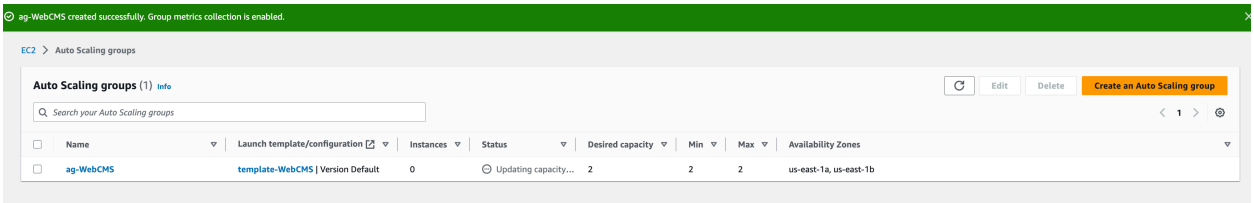
- En el panel izquierdo, seleccione Launch Templates.
- Click en Create launch template.
- Launch Template Name and Description:
    - Launch template name: template-WebCMS
    - Template version description: Template for web cms
    - Auto Scaling guidance: Active la casilla.
- Launch template contents
    - Application and OS Images (Amazon Machine Image): En MyAMIs, click en Owned by me, ami-WebCMS.
    - Instance type: t2.micro
- Network Settings:
    - Click en select existing security group. Por favor seleccionar el security group SG-WebCMS

Deje el resto de las opciones en los valores por defecto y click en 'create launch template'.

9.2 Crear y configurar Auto Scaling Groups.

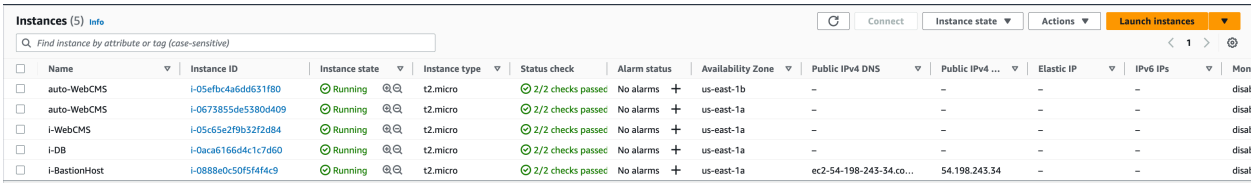
En el menú de instances, seleccione la opción de ‘Auto Scaling Groups’.

- Choose launch template or configuration
  - Name:
    - Auto Scaling group name: ag-WebCMS
  - Launch Template:
    - Launch Template : Seleccione el template que se creó template-WebCMS
    - Click en ‘Next’.
- Choose instance launch options:
  - Network:
    - VPC: CMS-vpc
    - Availability Zones and subnets: Seleccione las dos subredes privadas en las cuales se van a desplegar las instancias: 172.16.2.0/ 24 – 172.16.5.0/24
    - Click en ‘Next’.
- Configure Advanced Options:
  - Seleccionar la opción de Attach to an existing load balancer
  - Seleccionar la opción de ‘Choose from your load balancer target groups’
  - Existing load balancer target groups:
    - Seleccione el target group tg-CMS.
- Health Checks
  - Active la casilla Turn on Elastic Load Balancing health checks
  - Healch Checkgrace period: 90
- Additional Settings:
  - Marque la casilla de Enable group metrics collection within CloudWatch.
  - Click en Next.
- En la sección de Group size, configure los siguientes parámetros
  - Desired capacity: 2
  - Minimum capacity: 2
  - Maximum capacity: 2
- Resource Add Tags:
  - Click en Add tag.
  - Key: Name
  - Value: auto-WebCMS



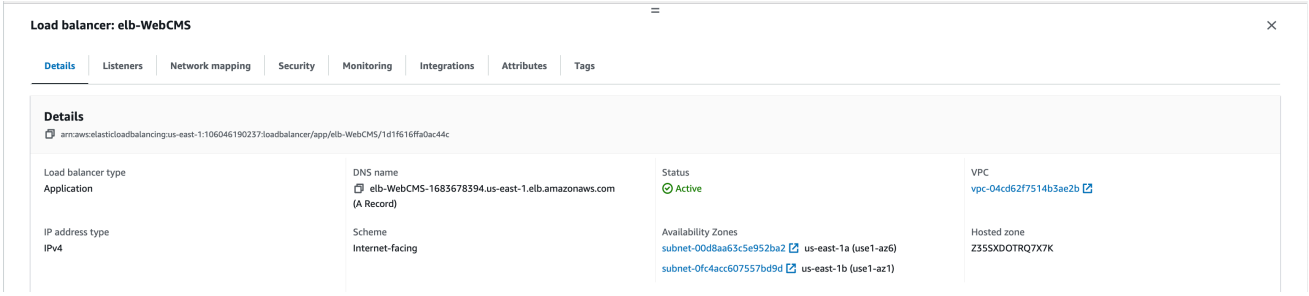

Para efectos de este laboratorio, se van a mantener siempre dos instancias con el fin de garantizar la alta disponibilidad. Pro esta razón, no se implementará ninguna política de escalamiento.

En este momento, en el menú Instances, puede observar que se inicializa las dos instancias que configuro en el Auto Scaling Group. Por favor espere a que terminen de estar lista. Verifique que el ‘status check’, este listo (2/2 checked passed).

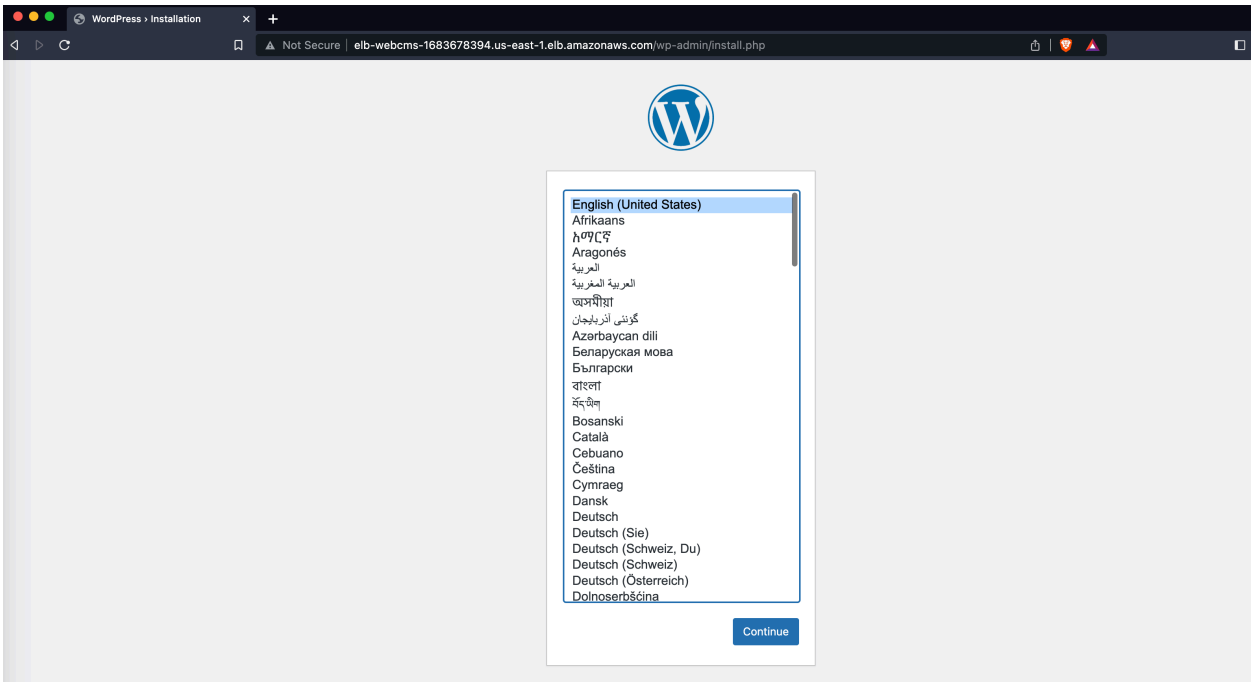



Igualmente, puede en el menú de EC2, Load Balancing, ‘Target Groups’. Seleccione el TG creado en esta guía, y en la pestaña ‘Targets’ debe visualizar algo parecido a lo que se observa en la siguiente figura:


En este punto puede probar el funcionamiento de lo configurando de la siguiente forma. En el menú de EC2, Load Balancing, Load Balancers, seleccione el balanceador de carga que se configuro previamente. En la pestaña localizada en la parte inferior, copie el DNS name.




Ahora, pegue ese nombre en un browser y debe observar la pantalla de configuración de wordpress.


WordPress › Installation
Not Secure | elb-webcms-1683678394.us-east-1.elb.amazonaws.com/wp-admin/install.php
English (United States)
Afrikaans
አማርኛ
Aragonés
العربية
العربية المغربية
অসমীয়া
گؤنئی آذربایجان
Azərbaycan dili
Беларуская мова
Български
বাংলা
བོད་ཡིག
Bosanski
Català
Cebuano
Čeština
Cymraeg
Dansk
Deutsch
Deutsch (Sie)
Deutsch (Schweiz, Du)
Deutsch (Schweiz)
Deutsch (Österreich)
Dolnoserbšćina
Continue