R03: AWS

3.1 Crear una instància EC2

2. Accedir a la consola de gestió

Un cop hagis iniciat sessió, accedeix a la AWS Academy Learner Lab. Un cop allà, Vas a "Contenidos" > "Lanzamiento del Laboratorio para el alumnado de AWS Academy". Un cop ho iniciem clickant "Start Lab", ens apareixerà AWS amb una icone en verd, hi clickem i ens redirigeix a la "Página de inicio de la consola".

Li donem a l'opció de "AWS Details" i a l'apartat de SSH Key premem a Download PPK (en cas de Windows) o Download PEM (en cas de Linux o Mac) per connectar-ho per SSH.

Creació de l'instància EC2

Un cop hem clickat a AWS quan està verd, ens redirigirà a la "Página de inicio de la consola", i llavors premem l'opció de EC2 i fem scroll fins l'apartat on posa "Lanzar la instancia" i llavors comencem amb configuració de la màquina.

Hi posem el nom, a "Imagenes de màquina de Amazon" escollim la que vulguem, en el nostre cas, escollim "Ubuntu Server 20.04 LTS (HVM), SSD Volume Type", i li donem a "Seleccionar".

Després, a "Seleccionar tipus d'instància" escollim la que vulguem, en el nostre cas, escollim "t2.micro" i a "Par de claves (inicio de sesión)", escollim l'opció per defecte de vockey. A continuació, a "Configuraciones de red", premem els checkboks de "Permitir el tráfico de HTTPS desde Internet" i de "Permitir el tráfico de HTTP desde Internet".

Per últim, configurem l'emmagatzemmatge amb 30 GiB i un cop fet tot això, li donem a "Lanzar la instancia".

3.2 Connexió SSH a la instància EC2 (Linux/ubuntu)

Podem connectar-nos a la instància EC2 mitjançant SSH. Per fer-ho, podem utilitzar el programari PuTTY (en cas de Windows) o la terminal mateixa amb OpenSSH.

3.2.1 Descarreguem PuTTY en local i ens connectem a la instància EC2 (Windows)

Descarreguem el software PuTTY a Windows, per tal de poder connectar-nos a la nostre instància EC2. Amb la clau privada (.ppk) que hem descarregat anteriorment, obrim PuTTYgen, "File" > "Load Private Key", escollim la .ppk, i li donem a l'opció de "Save private key".

Ara obrim PuTTY i configurem la connexió SSH amb la nostra instància EC2. Introduïm l'adreça IP de la instància (en el nostre cas, 44.210.150.195) a la casella "Host Name (or IP address)" i ens

assegurem de que el port sigui el 22. A la secció "Connection" > "SSH" > "Auth" > "Credencials", on posa "Private key file for authentication" seleccionem el fitxer .ppk que hem creat amb PuTTYgen. Finalment, fes clic a "Open" per connectar-te a la instància.

Per agilitza-ho, podem guardar la configuració de la connexió al subapartat "Saved Sessions" per tal de no haver de configurar-la cada vegada que ens connectem a la instància.

Finalment, després de donar-li a "Open", ens hauria de sortir una finestra de connexió a la instància EC2. Ens demanarà un usuari, en el nostre cas, l'usuari és "ubuntu".

Un cop ens haguem connectat a la instància, ja podrem començar a treballar amb ella.

3.2.2 Amb OpenSSH(Windows)

Simplement hem d'anar a EC2 > Instancias > ID de la instancia > Conectarse a la instancia i llavors a la finestra de "Cliente SSH".

Allà on posa "Ejemplo:", copiem la comanda, com per exemple la següent: ssh -i "vockey.pem" ubuntu@ec2-54-83-183-202.compute-1.amazonaws.com i obrim la terminal amb drets d'administrador. Ens movem a la carpeta on tenim el fitxer .PEM, i només hem de canviar el nom del fitxer per defecte de "vockey.pem" per el nostre i executar la comanda.

3.3 Instal·lació del web server a AWS EC2

- 1. Actualitzem el sistema amb la comanda:

```
# sudo apt update -y
```

```
# sudo apt upgrade -y
```

- 2. Instal·lem el servidor web Apache amb la comanda:

```
# sudo apt install apache2 -y
```

- 2.1 Per tal de que el servei apache pugui permetre reescriure les URL, cal activar el mòdul de reescriptura amb la comanda:

```
# sudo a2enmod rewrite
```

- 2.2 Per tal de que el servei apache s'habiliti automàticament en iniciar el sistema, cal activar el servei amb la comanda:

```
# sudo systemctl restart apache2
```

- 2.3 Creem l'arxiu de configuració de torrenova:

```
# cd /etc/apache2/sites-available/
# sudo nano torrenova.conf
```

- 2.4 Afegim el següent contingut a l'arxiu de configuració amb el seu directori arrel i els certificats d'SSL:

```
# <VirtualHost *:443>
ServerAdmin admin@example.conf
ServerName laravel.example.com
DocumentRoot /var/www/html/hotel-grup1/src/public
SSLEngine on
SSLCertificateFile /etc/ssl/certs/ssl-cert-snakeoil.pem
SSLCertificateKeyFile /etc/ssl/private/ssl-cert-snakeoil.key
<Directory "/var/www/html/hotel-grup1/src/public">
  AllowOverride All
  Require all granted
  Options Indexes FollowSymLinks
</Directory>
ErrorLog ${APACHE_LOG_DIR}/error.log
CustomLog ${APACHE_LOG_DIR}/access.log combined
</VirtualHost>
<VirtualHost *:80>
  RewriteEngine On
# RewriteCond %{HTTP_HOST} ^192\.168\.60\.170$
  RewriteRule ^(.*)$ https://%{HTTP_HOST}%{REQUEST_URI} [L,R=301]
</VirtualHost>
```

- 2.5 Desabilitem l'arxiu de configuració de 000-default.conf amb la comanda:

```
# sudo a2dissite 000-default.conf
```

- 2.6 Habilitem l'arxiu de configuració de torrenova.conf amb la comanda:

```
# sudo a2ensite torrenova.conf
```

- 2.7 Activem el mòdul de SSL amb la comanda:

```
# sudo a2enmod ssl
```

- 2.8 Reiniciem el servei apache amb la comanda:

```
# sudo systemctl restart apache2
```

3.4 Instal·lació de PHP a AWS EC2

- 1. Instal·lem PHP i les seves dependències amb la comanda:

```
# sudo apt install php-cli php-xml libapache2-mod-php8.3 php-cgi php-mysql
```

- 1.2 Instal·lem PHP i les seves dependències amb la comanda:

```
# sudo nano /etc/php/8.3/cli/php.ini

Un cop a dins, descomentem les següents extensions:
extension=curl
extension=mbstring
extension=openssl
extension=pdo_mysql
```

3.5 Instal·lació de Git a AWS EC2

- 1. Instal·lem Git amb la comanda:

```
# sudo apt install git -y
```

- 1.2 Clonem el repositori del nostre projecte amb les següents comandes:

```
# cd /var/www/html/
#sudo git clone https://gitlab.com/hotel-grup1/hotel-grup1.git
```

- 1.3 Un cop hem creat el nostre projecte, canviem l'arxiu .env.example a .env per així configurar l'entorn d'aplicació:

```
# cd /var/www/html/hotel-grup1/src/
# sudo cp .env.example .env
```

3.6 Instal·lació del Composer a AWS EC2

- 1. Instal·lació i actualització del composer i les dependències pertinents amb les següents comandes:

```
# sudo apt install composer -y
# sudo composer update
# sudo composer install
```

3.7 Instal·lació de l'npm a AWS EC2

- 1. Instal·lem NPM amb la comanda:

```
# sudo apt install npm -y
```

- 1.2 Executem la següent comanda per tal d'instal·lar les dependències del nostre projecte:

```
# sudo npm install
```

3.8 Configuració de l'aplicació AWS EC2

- 1.1 Compilem les dades de Vite a la aplicació amb la comanda:

```
# sudo npm run build
```

- 1.2 Connectem l'storage amb en public storage amb la comanda:

```
# sudo php artisan storage:link
```

- 1.3 Generem una clau d'aplicació utilitzant l'artisan amb la comanda:

```
# sudo php artisan key:generate
```

- 1.4 Afegim el grup i l'usuari www-data:www-data a storage amb la comanda:

```
# sudo chown -R www-data:www-data /var/www/html/hotel-grup1/src/storage
```

- 1.4.2 Donem permisos a la carpeta storage amb la comanda:

```
# sudo chmod -R 775 /var/www/html/hotel-grup1/src/storage/
```

3.9 Instal·lació del Docker AWS EC2

- 1. Primer, amb la següent comanda desinstalem totes les versions antigues de Docker:

```
# for pkg in docker.io docker-doc docker-compose docker-compose-v2 podman-docker
containerd runc; do sudo apt-get remove $pkg; done
```

Ara si, instal·lem docker

- 1.2 Actualitzem el sistema amb la comanda:

```
# sudo apt update -y
```

- 1.3 Afegim la clau GPG oficial de Docker:

```
# sudo apt install -y ca-certificates curl
# sudo install -m 0755 -d /etc/apt/keyrings
# sudo curl -fsSL https://download.docker.com/linux/ubuntu/gpg -o
/etc/apt/keyrings/docker.asc
# sudo chmod a+r /etc/apt/keyrings/docker.asc
```

- 1.3.2 Afegim el repositori a les fonts d'APT:

```
# echo \
  "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.asc]
https://download.docker.com/linux/ubuntu \
  $(. /etc/os-release && echo "$VERSION_CODENAME") stable" | \
  sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
```

- 1.3.3 Actualitzem el la clau i el repositori

```
# sudo apt update
```

- 1.3.4 Instalem el docker

```
# sudo apt install -y docker-ce docker-ce-cli containerd.io \
docker-buildx-plugin docker-compose-plugin
```

- 1.3.5 Puguem el docker amb la BBDD amb la comanda:

```
# cd /var/www/html/hotel-grup1/server/
# sudo docker compose up -d
```

- 1.3.6 Creem les dades de la BBDD amb la comanda:

```
# cd /var/www/html/hotel-grup1/src/
# sudo php artisan migrate:fresh --seed
```

Manual a seguir després de cada git pull:

```
# cd /var/www/html/hotel-grup1/src/
# sudo run build
# php artisan migrate:fresh --seed
```