MINISTERIO DE TECNOLOGÍAS DE LA INFORMACIÓN Y LAS COMUNICACIONES

UNIVERSIDAD SERGIO ARBOLEDA

MisiónTIC2022

Despliegue máquinas virtuales

Oracle Cloud

Ciclo 3: Desarrollo de Software

AÑO 2022

Introducción

Para desplegar nuestras aplicaciones, crearemos una máquina virtual, que podemos comprender como un computador (así como el nuestro) pero que físicamente no existe debido a que es parte de algún "súper computador" en algún lugar del mundo.

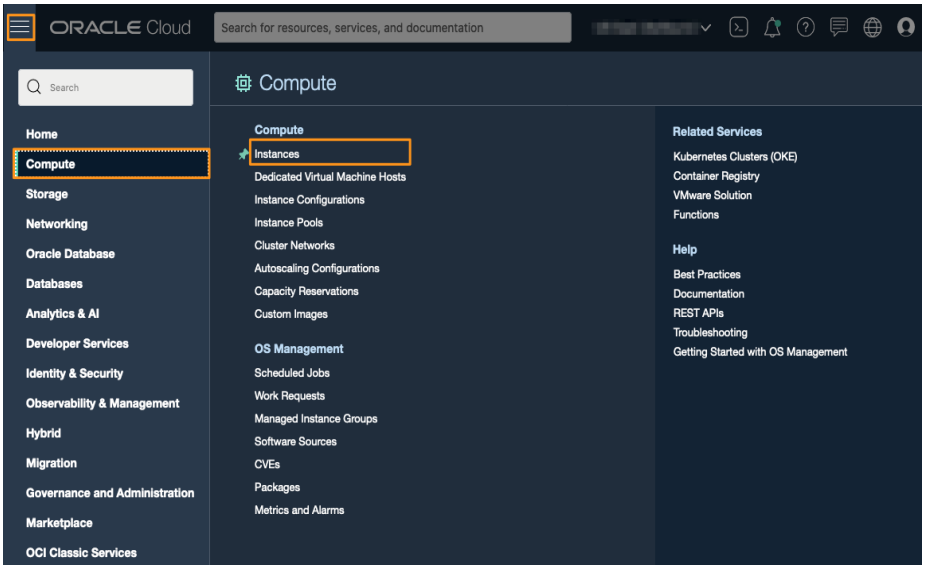
Esta máquina virtual, si bien físicamente no existe, sí funciona como cualquier otra máquina y podremos instalar en ella casi cualquier software mientras la configuración de la máquina se encuentre dentro de los límites requeridos por el software, incluyendo las especificaciones de hardware y software. Dentro de las especificaciones de software tenemos el sistema operativo, así mismo como otro software que se encuentre instalado en la máquina.

La máquina virtual suele estar conectada a una nube. Esta nube actuará como una red virtual, que son un conjunto de dispositivos virtuales (computadores y dispositivos de red) conectados entre si. Para que nuestra máquina virtual sea un servidor y podamos acceder a él, es necesario que nuestra máquina tenga el software necesario (servidor) y su configuración tanto de seguridad como de red sea la correcta.

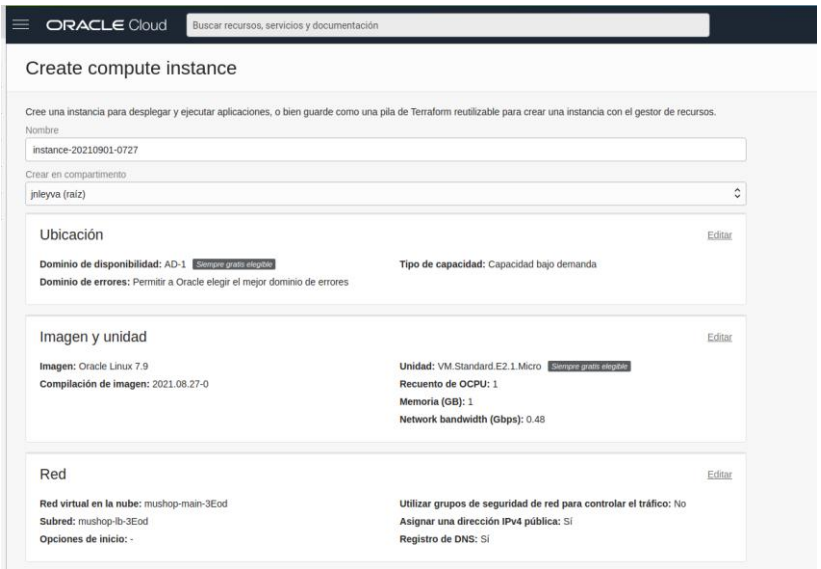
En este documento abordaremos un paso a paso de cómo desplegar una máquina como Servidor para una aplicación web construida en java, también haremos el despliegue de otro servidor para nuestra base de datos y finalmente explicaremos la manera en que podremos gestionar nuestra aplicación.

El primer servidor

La nube que hemos elegido y que utilizaremos como PaaS (Plataforma como un servicio) es Oracle Cloud. Para ello, hemos de iniciar sesión y a continuación en el menú superior elegiremos "Recursos Informáticos" y posteriormente instancias.



Una vez allí, haremos clic en el botón de nueva Instancia. Una instancia será entonces una máquina con características de hardware modificables para ajustarla a nuestras necesidades. Para el desarrollo del curso elegiremos las máquinas cuya configuración siempre se encuentren bajo el rótulo de "siempre gratis" lo que nos garantiza que podemos utilizarla durante un largo tiempo sin tener que hacer ningún pago por utilizarla.



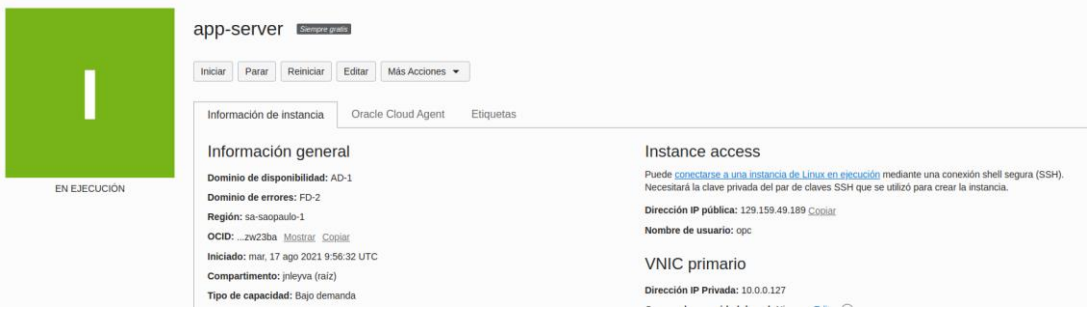
En la sección de nombre, escribiremos un nombre podemos elegir un nombre con el que identificaremos la máquina (en adelante, instancia). La sección de Imagen y Unidad nos muestra que tenemos un sistema operativo Oracle Linux y la configuración que se encuentra disponible tiene el rótulo de "Siempre gratis elegible".

En la sección de red, nos aseguraremos de asignar una dirección IPv4 pública.

En la sección de Agregar claves SSH debemos tener mucho cuidado, pues en este paso generaremos un archivo que será siempre necesario para conectarnos con nuestra instancia y acceder a ella. Este archivo es generado una sola vez y no es posible volver a descargarlo. El archivo que mayor interés nos genera es el archivo que se guarda cuando hacemos clic en Guardar clave privada. También descargaremos la clave pública tendremos la misma precaución de guardarla y evitar su pérdida.

Finalmente haremos clic en el botón inferior de Crear. Luego de un breve tiempo nuestra instancia estará creada.

Una vez creada veremos una pantalla semejante a la siguiente:



Es importante tener en cuenta la dirección IP para crear la conexión con el servidor.

En este momento es necesario revisar el enlace que se presenta junto a la dirección ip pública que dice "conectarse a una instancia de Linux en ejecución"

En este tutorial proponemos utilizar el programa Git Bash como terminal para realizar la conexión. Sin embargo no es obligatorio, debido a que cualquier cliente terminal estará en condiciones de realizar la conexión.

En el directorio en donde se guardó la clave privada hacer clic derecho y elegir "open git bash here". De otro modo, abrir git bash e ir hasta la ubicación de la llave privada.

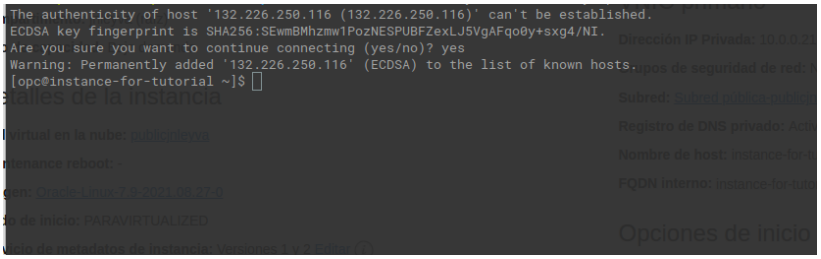
Una vez allí se recomienda escribir el comando chmod 400 llave-privada.key donde llave-privada.key corresponde al nombre del archivo de la llave privada que allí se encuentra. Este paso hacerlo solamente si está utilizando git bash o si se encuentra en linux. Para otras alternativas consultar la documentación planteada en el link "conectarse a una instancia de Linux en ejecución" que se nombró previamente.

Posteriormente para conectarnos a la máquina recién creada, escribiremos lo siguiente

ssh -i llave-privada.key opc@ip-instancia

en el que llavep-privada.key es el archivo de la llave privada y se debe reemplazar ip-instancia por la dirección ip pública de la instancia.

A la pregunta que haga respecto a querer continuar con la conexión, escribiremos la palabra yes y daremos enter. Deberíamos ver algo así:



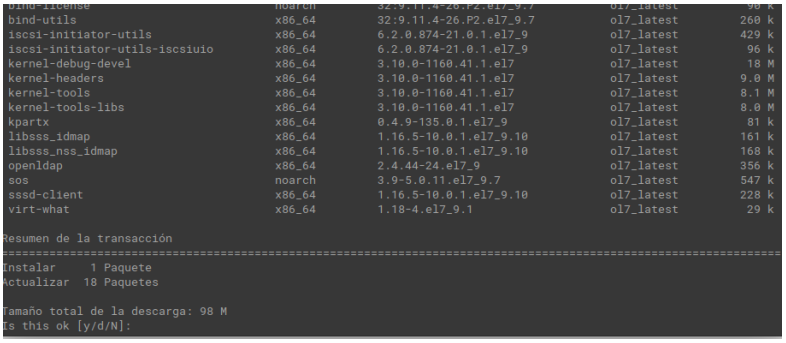
En este momento estamos autenticados en nuestra máquina a través de un tunel seguro. Empezaremos a configurar nuestra máquina y nuestra infraestructura para que este computador sea un servidor.

Instalando el software necesario

La primera instrucción que ejecutaremos será:

sudo yum update

Presionaremos enter y esperaremos un proceso largo en el que la máquina actualizará las definiciones y versiones del software que tiene por instalar. Después de unos instantes nos preguntará si queremos hacer la descarga del software actualizado, le diremos que sí presionando la tecla y, presionamos enter y esperamos a que el proceso de descarga inicie.



Una vez termine el proceso de actualización de software y de definiciones, procederemos a instalar 3 aplicaciones que necesitamos para que nuestro programa pueda ser ejecutado sin inconvenientes:

git, maven y java.

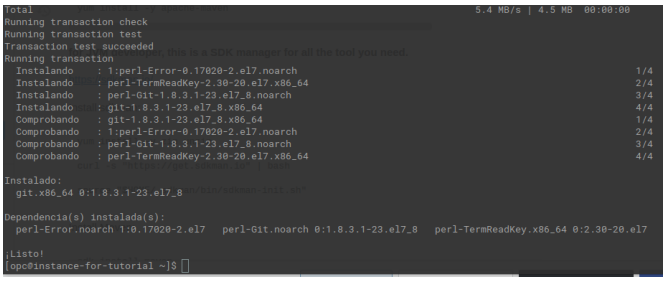
La instalación de maven la haremos según lo propuesto por la página oficial con el fin de obtener la última versión. Java y Git serán instalados desde sus repositorios.

Lo primero que instalaremos será Git, seguiremos con Java y por último instalaremos maven. Es importante instalar maven después de java.

Para instalar git escribiremos el comando

sudo yum install git

luego presionaremos la tecla y, presionamos enter y esperamos el proceso. Ahora, está instalado git en nuestra instancia.



Para comprobar que hemos hecho la instalación de forma correcta simplemente escribiremos

git --version

Nos arrojará en pantalla la versión instalada de git.

A continuación instalaremos Java en su versión 11. Para ello contaremos con OpenJDK.

La instalación la realizaremos de la siguiente manera:

sudo yum install java-11-openjdk-devel

Debemos presionar y, luego enter cuando nos pida confirmar el proceso. Esto instalará java y todo lo que necesitemos.

Para validar simplemente escribiremos

java -version

y nos mostrará en pantalla que tenemos instalado OpenJDK.

En este momento está todo listo para instalar maven. Para ello ejecutaremos las siguientes instrucciones:

sudo yum-config-manager --add-repo http://repos.fedorapeople.org/repos/dchen/apache-maven/epel-apache-maven.repo

sudo yum-config-manager --enable epel-apache-maven

sudo yum install -y apache-maven

Este proceso instalará maven, pero modificará la versión instalada de java, para ello hemos de corregir el proceso con

sudo alternatives --config java

sudo alternatives --config javac

Al ejecutar cada instrucción nos preguntará cuál es la versión que se quiere utilizar, es probable que la correspondiente a OpenJDK versión 11 sea la opción 1. Basta con escribir el número 1 y presionar enter para que quede seleccionado.

```
[opc@instance-for-tutorial ~]$ sudo alternatives --config java

Hay 3 programas que proporcionan 'java'.

  Selección    Comando
-----------------------------------------------
   1           java-11-openjdk.x86_64 (/usr/lib/jvm/java-11-openjdk-11.0.12.0.7-0.0.1.el7_9.x86_64/bin/java)
*+ 2           java-1.8.0-openjdk.x86_64 (/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.302.b08-0.el7_9.x86_64/jre/bin/java)
   3           /usr/java/jre1.8.0_301-amd64/bin/java

Presione Intro para mantener la selección actual[+], o escriba el número de la selección:  1
[opc@instance-for-tutorial ~]$ sudo alternatives --config javac

Hay 2 programas que proporcionan 'javac'.

  Selección    Comando
-----------------------------------------------
   1           java-11-openjdk.x86_64 (/usr/lib/jvm/java-11-openjdk-11.0.12.0.7-0.0.1.el7_9.x86_64/bin/javac)
*+ 2           java-1.8.0-openjdk.x86_64 (/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.302.b08-0.el7_9.x86_64/bin/javac)

Presione Intro para mantener la selección actual[+], o escriba el número de la selección:  1
[opc@instance-for-tutorial ~]$ java -version
openjdk version "11.0.12" 2021-07-20 LTS
OpenJDK Runtime Environment 18.9 (build 11.0.12+7-LTS)
OpenJDK 64-Bit Server VM 18.9 (build 11.0.12+7-LTS, mixed mode, sharing)
[opc@instance-for-tutorial ~]$
```

Creando nuestro servidor de Base de Datos

Para cumplir nuestro objetivo y almacenar la información, tendremos dos alternativas. Estas alternativas son:

Configurar un servidor de de base de datos mysql el cual será una instancia que vendrá preconfigurada para utilizarla o utilizar una base de datos basada en archivos en el mismo servidor de nuestra aplicación.

Debido a que hemos utilizado JPA, la direfencia estará en el String de conexión. Para poder lograr una evaluación automática sin problemas, hemos ya sugerido en los retos que la estrategia de generación de tablas sea create en lugar update, así cada vez que ejecutamos la aplicación, la base de datos será creada desde cero.

La elección que hagamos de base de datos no afecta el funcionamiento de nuestra aplicación. En el momento de crear nuestras cuentas de Oracle Cloud, hemos recibido un saldo inicial de $300 (dólares) los cuales serán utilizados en los servicios que no tengan etiqueta de "Siempre gratis". Las instancias de MySql que utilizaremos tendrán un costo de $1,5 dólares por día (valor aproximado).

A continuación contaremos los dos escenarios:

Escenario 1) Crear un servidor MySql:

Para crear un servidor de Bases de datos MySql simplemente nos dirigimos al menú de Oracle y seleccionamos bases de datos, ahí haremos clic en MySql

En la sección que aparece haremos clic en el botón Crear Sistema de base de datos MySQL.

En la siguiente pantalla hay varios elementos que tener en cuenta para la creación de la instancia de base de datos:

El nombre: un nombre con el que identificaremos la instancia de BD.

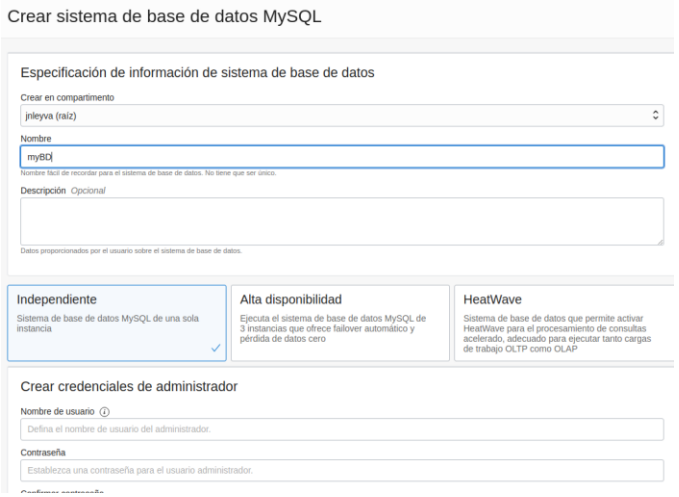
El tipo: Independiente. Para el propósito del curso es la más económica, sin ser gratuita.

Las credenciales de administrador:

El nombre de usuario se sugiere que sea admin. Este nombre estará posteriormente en el archivo de propiedades en el perfil producción.

La contraseña, la cual se debe confirmar es muy importante que no la olvidemos y que cumpla con los requerimientos.

En la configuración de red, hay que asegurarnos de establecer la misma red virtual y la misma subred que la instancia previamente creada, importante elegir la subred pública.



Posteriormente al presionar el botón crear avanzamos a la pantalla de la instancia. El proceso de creación de la instancia puede tardar unos minutos. Ahí encontraremos los datos necesarios para nuestra aplicación, el más importante: la dirección ip que escribiremos en la cadena de conexión de nuestro archivo properties.

Escenario 2) Base de datos local, basada en archivos en Disco duro o en memora.

Para el escenario, que es establecer un sistema de archivos en nuestro servidor de aplicaciones, bastará con ajustar el archivo de configuración .properties en el perfil producción y agregar en el archivo pom.xml la dependencia del conector JDBC para el motor de base de datos basado en archivos. Es muy recomendable que en SISTEMAS DE PRODUCCIÓN NO SE UTILICE este sistema de base de datos, es muy útil en entornos de desarrollo.

Modificación en el archivo pom.xml:

-Agregar dependencia.

```
    <dependency>
            <groupId>com.h2database</groupId>
             <artifactId>h2</artifactId>
             <version>1.4.200</version>
            <scope>runtime</scope>
    </dependency>


</dependencies>
```

Modificación del archivo .properties del entorno de producción

spring.datasource.url=jdbc:h2:mem:testdb

spring.datasource.driverClassName=org.h2.Driver

spring.datasource.username=sa

spring.datasource.password=password

spring.jpa.database-platform=org.hibernate.dialect.H2Dialect

La línea url establece que la base de datos será guardada en memoria, por lo que al cerrar la aplicación, se borrará. Si se quiere guardar en un archivo:

spring.datasource.url=jdbc:h2:file:/data/demo

La ubicación del archivo debe ser accesible por java.

Es recomendable crear un archivo con un perfil solamente para esta configuración.

Desplegando nuestra aplicación

Para desplegar nuestra aplicación, el proceso se resumirá en:

-Descargar la última versión del código fuente a través de git.

-Compilar el código a través de maven.

-Ejecutar el archivo jar a través de java

Para descargar la úlitma versión utilizaremos el comando git clone.

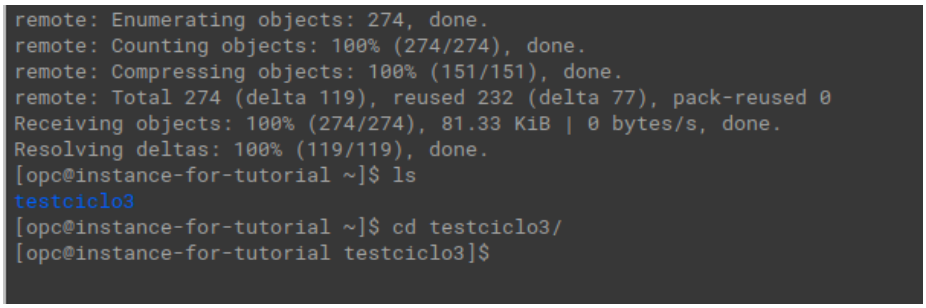
git clone url-repositorio

donde url-repositorio corresponde a la url proporcionada por github. Es posible que requieras una contraseña de aplicación para poder hacer la descarga dependiendo de la configuración de privacidad.

Una vez hecho esto, se inicia la descarga y posteriormente debemos entrar a la carpeta escribiendo

cd nombrecarpeta.

Para conocer el nombre de la carpeta, podemos escribir ls y veremos la lista de carpetas.



Una vez hecho esto procedemos a compilar el código. Para compilar es recomendable saltar las pruebas así no tendremos problemas al elegir el perfil. Por lo general la primera compilación implica la descarga de todas las dependencias.

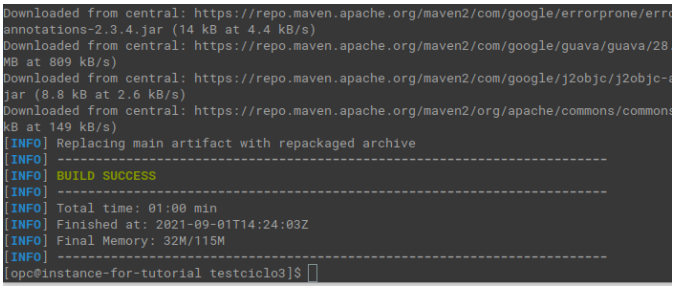
Para ello escribiremos

mvn clean package -DskipTests

El proceso de compilar deberá iniciar sin problemas mientras nuestro código esté funcionando. Cualquier inconveniente que tengamos acá lo debemos resolver desde el código que hemos hecho, las correcciones las haremos en nuestra máquina local y subiremos los cambios al repositorio.

Cada vez que deseemos actualizar los cambios escribiremos git pull.

La compilación exitosa debe verse así:



El archivo ejecutable se encontrará en la carpeta target.

Para ello, cd target.

Para ejecutar nuestra aplicación, haremos lo siguiente:

sudo java -jar -Dspring.profiles.active=prod archivo.jar

En el comando anterior hemos utilizado sudo para tener permisos de super usuario con el fin de poder utilizar el puerto 80 ya que en el archivo de perfil de producción el puerto seleccionado es el puerto 80.

En el comando también hemos puesto la palabra prod, asumiendo que así se llama el perfil de producción. La palabra archivo.jar se debe cambiar por el nombre correcto del archivo .jar que se encuentre en la carpeta target.

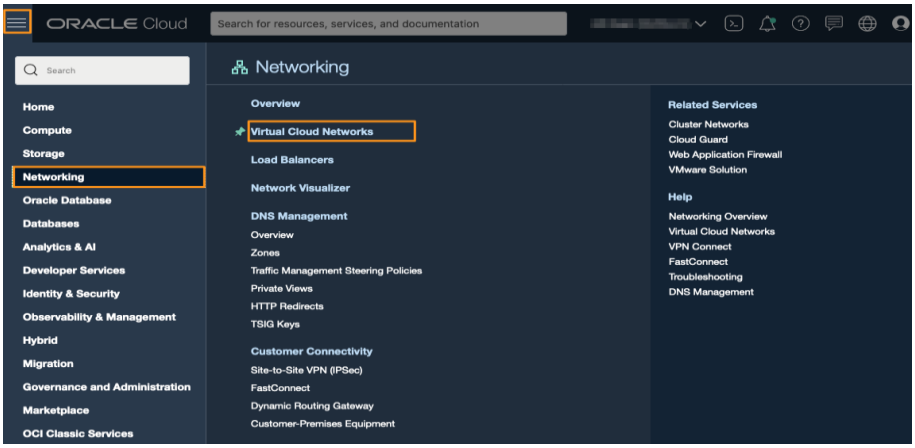
```
[opc@instance-for-tutorial target]$ sudo java -jar -Dspring.profiles.active=prod RentCar-0.0.1-SNAPSHOT.jar

  .   ____          _            __ _ _
 /\\ / ___'_ __ _ _(_)_ __  __ _ \ \ \ \
( ( )\___ | '_ | '_| | '_ \/ _` | \ \ \ \
 \\/  ___)| |_)| | | | | || (_| |  ) ) ) )
  '  |____| .__|_| |_|_| |_\__, | / / / /
 =========|_|==============|___/=/_/_/_/
 :: Spring Boot ::                (v2.5.3)

2021-09-01 14:36:02.690  INFO 14663 --- [           main] com.usa.rentcar.RentCarApplication       : Starting RentCarApplicat
on v0.0.1-SNAPSHOT using Java 11.0.12 on instance-for-tutorial with PID 14663 (/home/opc/testciclo3/target/RentCar-0.0.1-SNAP
HOT.jar started by root in /home/opc/testciclo3/target)
2021-09-01 14:36:02.693  INFO 14663 --- [           main] com.usa.rentcar.RentCarApplication       : The following profiles a
e active: prod
2021-09-01 14:36:09.702  INFO 14663 --- [           main] .s.d.r.c.RepositoryConfigurationDelegate : Bootstrapping Spring Dat
 JPA repositories in DEFAULT mode.
2021-09-01 14:36:10.291  INFO 14663 --- [           main] .s.d.r.c.RepositoryConfigurationDelegate : Finished Spring Data rep
sitory scanning in 488 ms. Found 7 JPA repository interfaces.
2021-09-01 14:36:16.401  INFO 14663 --- [           main] o.s.b.w.embedded.tomcat.TomcatWebServer  : Tomcat initialized with
ort(s): 80 (http)
2021-09-01 14:36:16.494  INFO 14663 --- [           main] o.apache.catalina.core.StandardService   : Starting service [Tomcat
2021-09-01 14:36:16.494  INFO 14663 --- [           main] org.apache.catalina.core.StandardEngine  : Starting Servlet engine:
[Apache Tomcat/9.0.50]
```

Infraestructura de Red

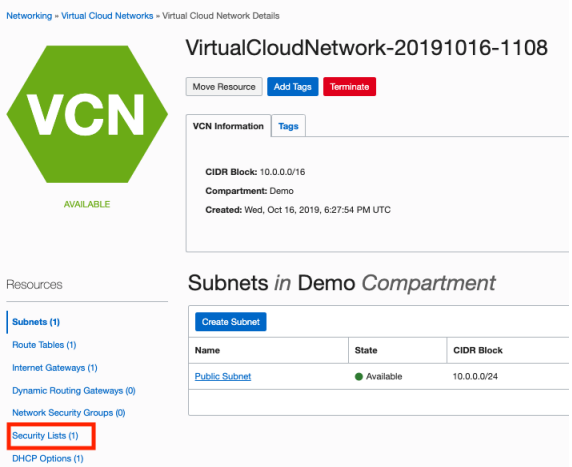
Por politicas de seguridad es muy importante restringir el tráfico entrante hacia los dispositivos de nuestra nube virtual. Por tal razón los puertos suelen estar cerrados para la entrada de tráfico. Lo que haremos a continuación es abrir el puerto 80 sobre el cual está corriendo nuestra aplicación.

Para ello:

Vamos a la sección de Redes en el menú general:



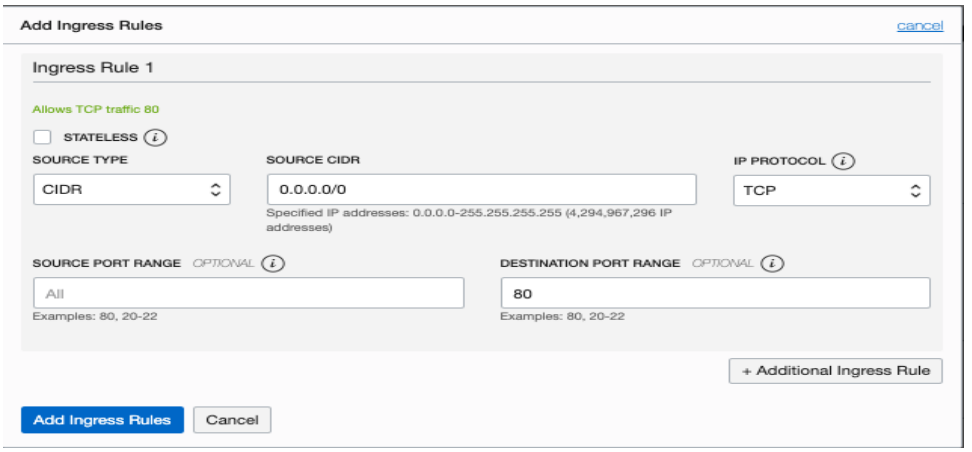
Allí, seleccionamos la Nube Privada Virtual a la que pertenecen nuestras instancias, por ejemplo:



Una vez allí, elegimos Listas de Seguridad. Las listas de seguridad tienen la configuración del tráfico entrante, los puertos y direcciones permitidas o explícitamente negados.

De esta manera, habilitaremos el ingreso de tráfico por un puerto a nuestra red.

Para eso agregar una nueva regla de ingreso y nos aseguramos que se vea con la información de la siguiente imagen:



Una vez ejecutado esto, hace falta ingresar las siguientes instrucciones en la instancia de nuestro servidor de la aplicación, con el fin de permitir que el firewall reciba el tráfico:

sudo firewall-cmd --permanent --zone=public --add-service=http

sudo firewall-cmd --reload

Con esto, hemos terminado, nuestro servidor está desplegado en la nube y es accesible a través de la dirección IP.