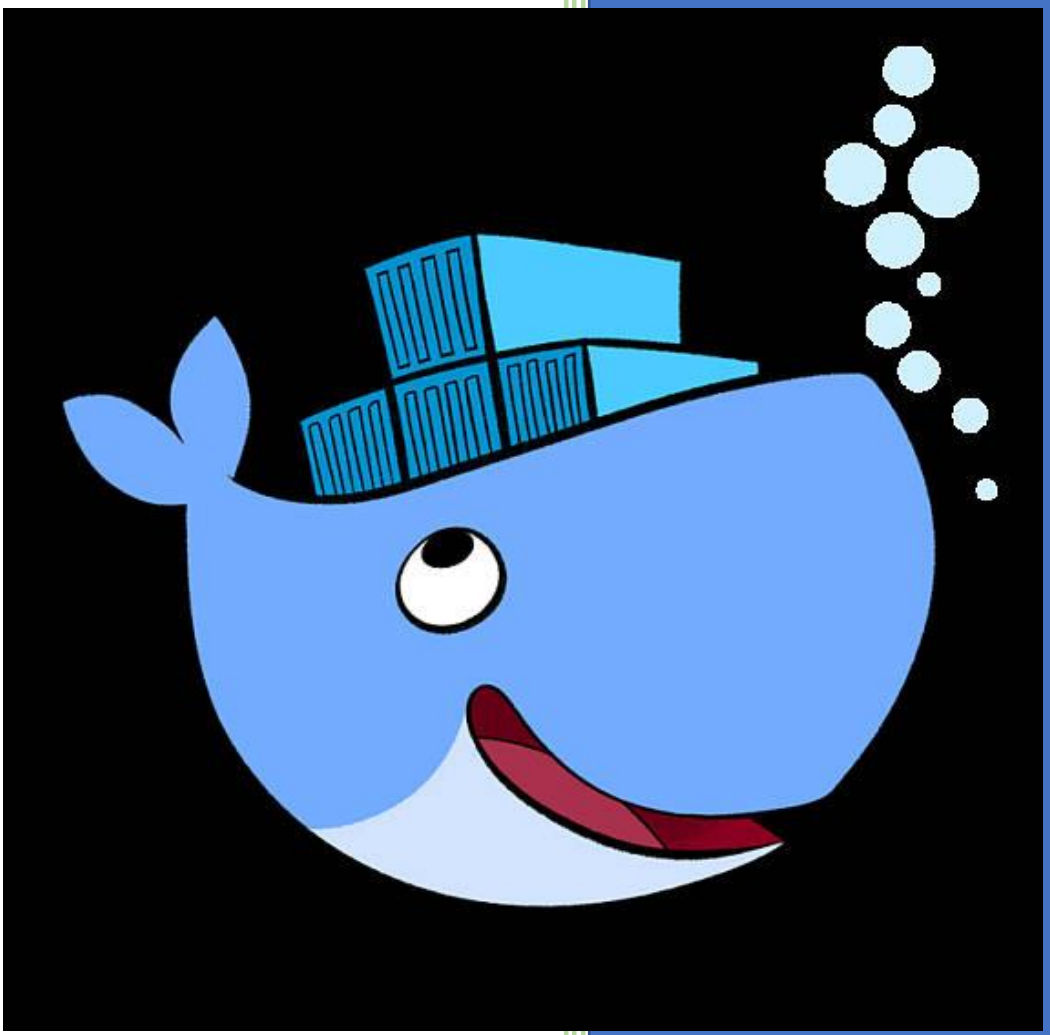
Docker



Victor Barrios Gonzalez

Máster Ingeniería Informática

ÍNDICE

DOCKER

INTRODUCCIÓN

En esta práctica se va a dockerizar la base de datos en la cual se guardan los usuarios y sus roles, a la cual el cliente acudirá para autenticarse y posteriormente la parte backend de nuestra tienda online, que se unirá a la base de datos para poder consultar y guardar los productos. Esto nos permitirá arrancar el servidor sin tener que utilizar Eclipse para ejecutarlo.

Posteriormente se subirán las imágenes de la base de datos y del backend a Docker Hub, así con tan solo descargar las imágenes en cualquier equipo, se podrán montar en un contenedor y ejecutar la aplicación, para la parte cliente se tendrá que utilizar Eclipse para arrancarla y pedir las consultas a la imagen del servidor y de la base de datos.

CREACIÓN DE LA IMAGEN DE LA BASE DE DATOS

Para crear el contenedor y la imagen de la base de datos que nos servirá para la parte cliente debemos seguir los siguientes pasos.

Se abre un terminal en el cual escribiremos las siguientes sentencias:

- Creamos la imagen de la base de datos SQL, entramos en la carpeta donde tengamos el **dockerfile** de MySQL y escribimos:

```
#docker build -t victorsql --label victorsql 5.7/
```

- Ahora creamos la base de datos **practicadocker** a partir de la imagen creada anteriormente de **victorsql:**

```
#docker run --name victorsql -e MYSQL_ROOT_PASSWORD=123456 -e
MYSQL_DATABASE=practicadocker -e MYSQL_USER=victor -e MYSQL_PASSWORD=123456 -p
3306:3306 -d victorsql:latest
```

- Ahora abrimos la consola de mysql, así podemos comprobar si se crean las listas, donde deben aparecer los usuarios y administradores y los productos.

```
#docker exec -it victorsql mysql -uroot -p
```

Antes de arranca el cliente debemos modificar el archivo **application.yml** para que se creen las tablas dentro de la base de datos.

application.yml --> créate

Una vez creadas las tablas, debemos parar el cliente y volver a modificar el **application.yml**:

application.yml --> validate

Dentro del **UserDatabase** se debe comentar la creación de los usuarios.

Para subir la imagen a Docker Hub, en la consola escribimos:

```
#docker login
NOMBRE: evaurjc2017
CONTRASEÑA: qwertyuiop
```

Preparamos la imagen para subirla y hacemos un push al repositorio:

```
#docker tag victorsql evaurjc2017/Victor:basedatos

#docker push evaurjc2017/victor
```

CONFIGURACION DE LA PARTE BACKEND

A continuación, se detallan los pasos necesarios para configurar el servidor para poder dockerizarlo.

I. Configuración del archivo **pom.xml**.

artifacId → SpringServerRestTestDockerAvanzado
packaging → jar

II. Configuración del archivo **application.yml**.

- Cambiar los parámetros para que al iniciar la primera vez nos cree las tablas dentro de la base de datos.

```
ddl-auto: create
```

- En la url, añadir el nombre del contenedor donde esta la base de datos, así creamos el enlace con est.

```
url: jdbc:mysql://victorsql/practicadocker
```

- Creación del archivo dockerfile.
  Dentro del archivo introduciremos lo siguiente:

```
FROM java:8
MAINTAINER Victor Barrios Gonzalez
EXPOSE 8080
VOLUME /tmp
ADD /target/SpringServerRestTestDockerAvanzado-0.0.1-SNAPSHOT.jar
practicavictor.jar
ENTRYPOINT ["java", "-jar", "practicavictor.jar"]
```

III. Una vez modificados los archivos y creado el dockerfile, en Eclipse, buscamos en el menú **Run As**, **Maven Build,** dentro de la pestaña Main escribimos lo siguiente en el apartado de **Goals** para crear el archivo .jar en la carpeta target:

clean install verify package

Comprobamos que dentro del proyecto se ha creado la carpeta target, dentro de la cual estará el archivo .jar.

Ya tenemos configurada la parte backend para proceder a la creación de la imagen de Docker.

CREACIÓN DE LA IMAGEN

Para crear la imagen de nuestro servidor abrimos un terminal e iniciamos Docker con la siguiente sentencia:

 # service Docker start

Una vez iniciado Docker procedemos a la creación del contenedor con la imagen del servidor ya dentro, para ello vamos hasta la carpeta del proyecto e introducimos en la consola:

 # docker build -t practicavictor .

```
root@sephit:~/Programas# cd WSEclipse
root@sephit:~/Programas/WSEclipse# ls
 Jose                    SpringClientRestTestDockerAvanzado
 Lorena                  SpringServerRestTestDockerAvanzado
 PracticaServerOMDBMysql 'Templates VideoClub'
'Práctica IWCN'          VideoClubClient
 RemoteSystemsTempFiles  'Videoclub Eva'
 RestTest                pMysql
root@sephit:~/Programas/WSEclipse# cd SpringServerRestTestDockerAvanzado/
root@sephit:~/Programas/WSEclipse/SpringServerRestTestDockerAvanzado# ls
Dockerfile  pom.xml  src  target
root@sephit:~/Programas/WSEclipse/SpringServerRestTestDockerAvanzado# docker build -t practicavictor .
Sending build context to Docker daemon   29.6MB
Step 1/6 : FROM java:8
 ---> d23bdf5b1b1b
Step 2/6 : MAINTAINER Victor Barrios Gonzalez
 ---> Using cache
 ---> ba572a43b83b
Step 3/6 : EXPOSE 8080
 ---> Using cache
 ---> 78e997407253
Step 4/6 : VOLUME /tmp
 ---> Using cache
 ---> 38c48586ace8
Step 5/6 : ADD /target/SpringServerRestTestDockerAvanzado-0.0.1-SNAPSHOT.jar practicavictor.jar
 ---> 6ca4a89a3621
Removing intermediate container 8aca307497a6
Step 6/6 : ENTRYPOINT java -jar practicavictor.jar
 ---> Running in a1e2d1ceb124
 ---> bd3318093495
Removing intermediate container a1e2d1ceb124
Successfully built bd3318093495
root@sephit:~/Programas/WSEclipse/SpringServerRestTestDockerAvanzado#
```

Ilustración 1

Una vez creado unimos los contenedores de la base de datos con el servidor:

 # docker run --name practicavictor -p 8080:8080 --link victorsql -d practicavictor

```
root@sephit:~/Programas/WSEclipse/SpringServerRestTestDockerAvanzado# docker run --name practicavictor -p 8080:8080 --link victorsql -d practicavicto
r
5c487473cdb5e506c5f84a4060f84944ba5a41b344a48354882ec9d2d540fe40
root@sephit:~/Programas/WSEclipse/SpringServerRestTestDockerAvanzado#
```

Ilustración 2

Para poder unirlos, deberemos arrancar antes los dos contenedores como se ve en la siguiente imagen.

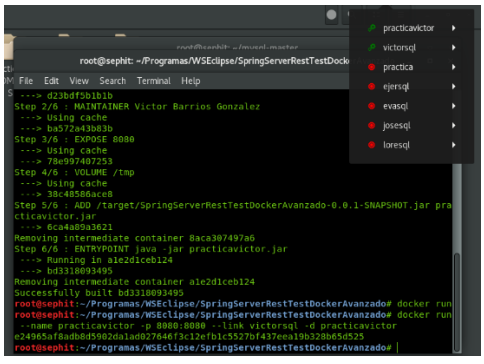


Ilustración 3

Ya tenemos unidos y arrancados los dos contenedores con las imágenes del backend en el puerto 8080 y la base de datos en el puerto 3306.

```
root@sephit:~/Programas/WSEclipse/SpringServerRestTestDockerAvanzado# docker ps
CONTAINER ID        IMAGE               COMMAND                  CREATED             STATUS              PORTS                    NAMES
5c487473cdb5        practicavictor      "java -jar practic..."   7 minutes ago       Up 7 minutes        0.0.0.0:8080->8080/tcp   practicavictor
35312e36839a        victorsql:latest    "docker-entrypoint..."   34 minutes ago      Up 34 minutes       0.0.0.0:3306->3306/tcp   victorsql
root@sephit:~/Programas/WSEclipse/SpringServerRestTestDockerAvanzado#
```

Ilustración 4

PRUEBA DEL SERVIDOR

Con los contenedores de la base de datos y el servidor arrancados, ejecutamos el cliente en Eclipse y lo abrimos en el navegador.

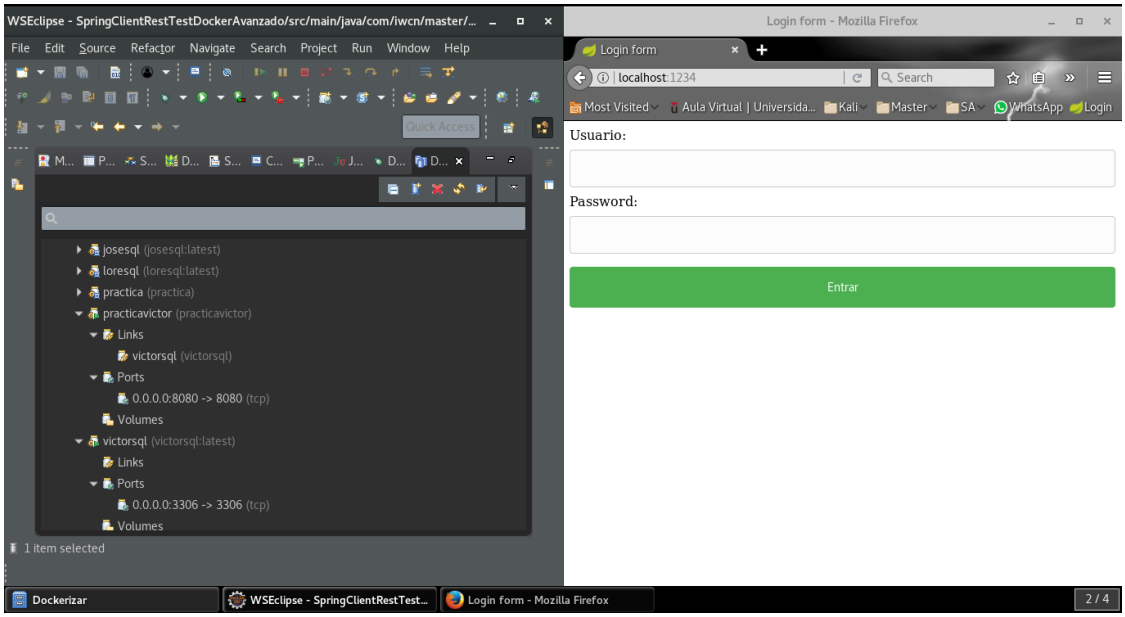


Ilustración 5

Una vez abierto nos autenticamos y guardamos un producto en la base de datos y lo comprobamos en la consola pidiendo las listas en la consola:

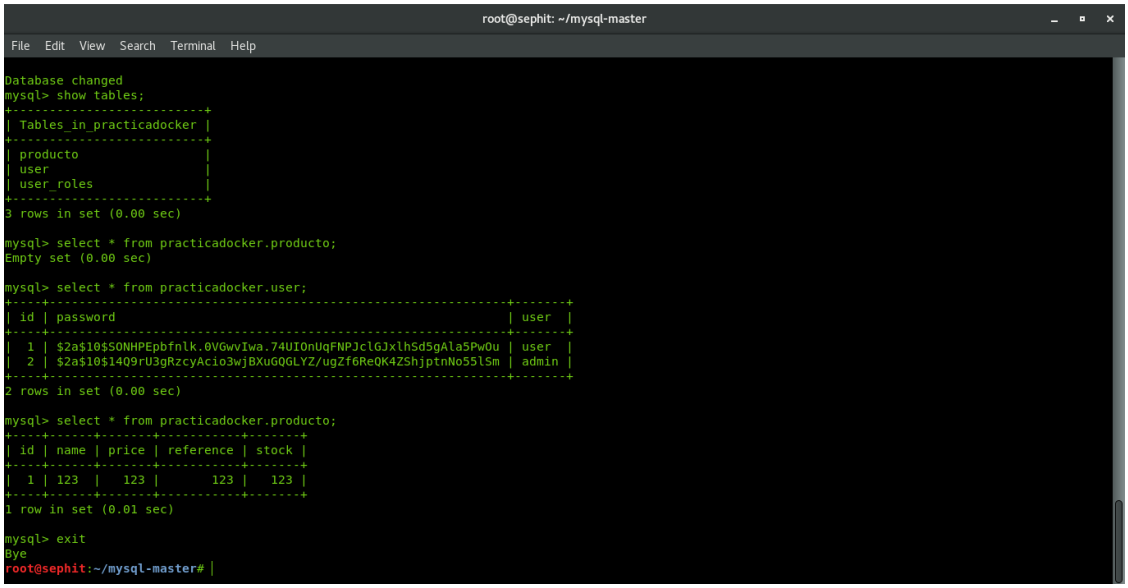


Ilustración 6

Una vez guardado abrimos el servidor en el navegador en el puerto 8080 y le pedimos la lista de productos, para ello escribimos en la barra de direcciones:

localhost:8080/ListaDeProductos → Permite ver la lista de productos desde el servidor

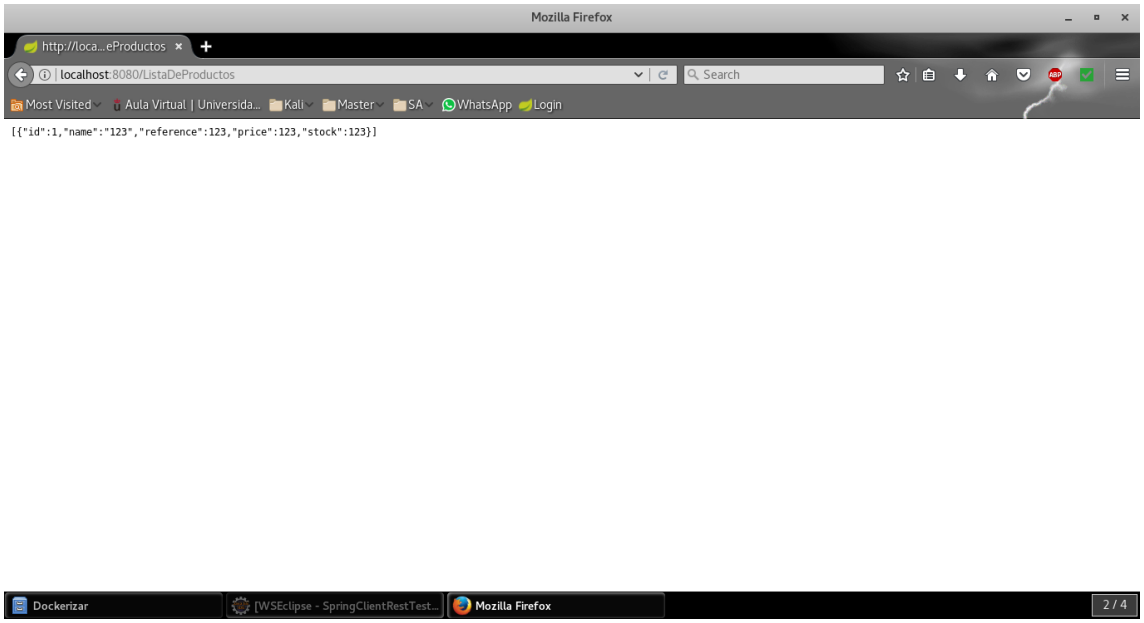


Ilustración 7

SUBIR IMÁGENES A DOCKERHUB

Al igual que la imagen de la base de datos vamos a subir al repositorio de Docker Hub la imagen del servidor, las sentencias son similares a las de la base de datos.

Nos autenticamos en Docker Hub:

```
#docker login
NOMBRE: evaurjc2017
CONTRASEÑA: qwertyuiop
```

Preparamos la imagen para subirla al repositorio y hacemos un push:

```
#docker tag practicavictor evaurjc2017/victor:servidor
#docker push evaurjc2017/Victor
```

Para comprobar que se ha subido correctamente:

```
#docker search evaurjc2017
```

En el caso de querer descargar las imágenes que hay en el repositorio escribiríamos la siguiente sentencia:

#docker pull evaurjc2017/Victor

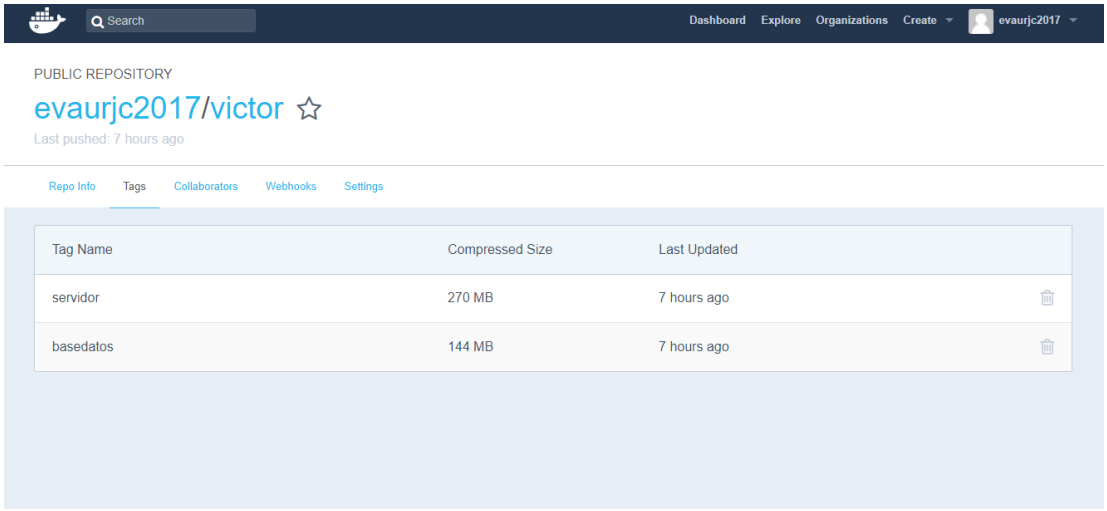


Ilustración 8

Descargando el repositorio desde otra maquina vamos a probar que podemos arrancar el servidor y la base de datos directamente desde la consola.

Introducimos en la cosola:

#docker run –name servidor -p 8080:8080 –link victorsql -i evaurjc2017/Victor:servidor

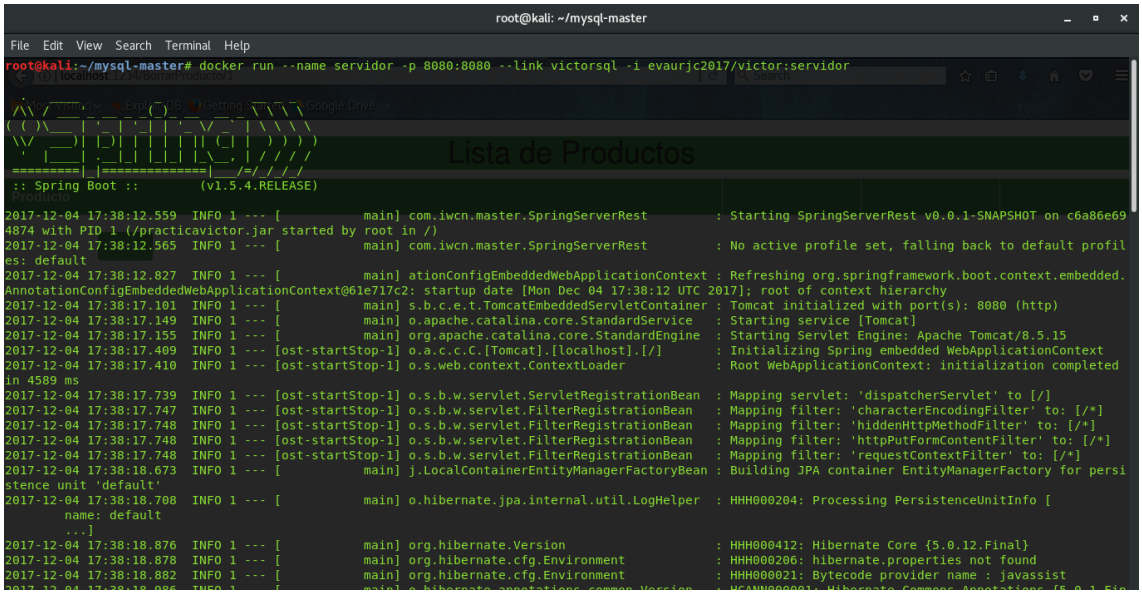


Ilustración 9

Arrancamos el cliente con Eclipse e introducimos en el navegador la dirección del cliente.

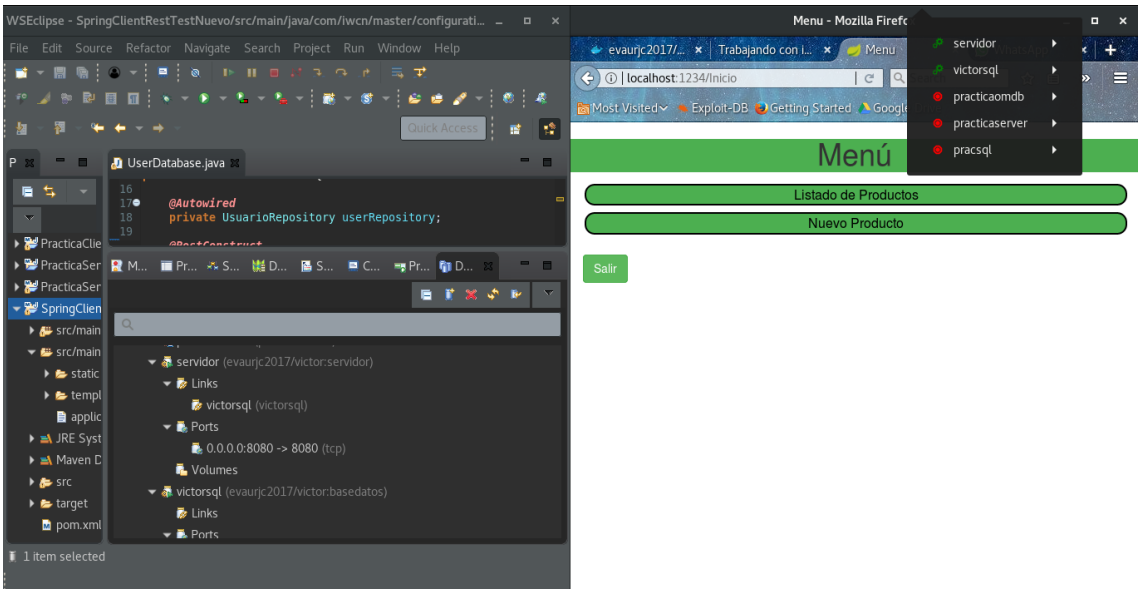


Ilustración 10

Creamos un producto y comprobamos que se ha guardado en la base de datos a través de la consola.

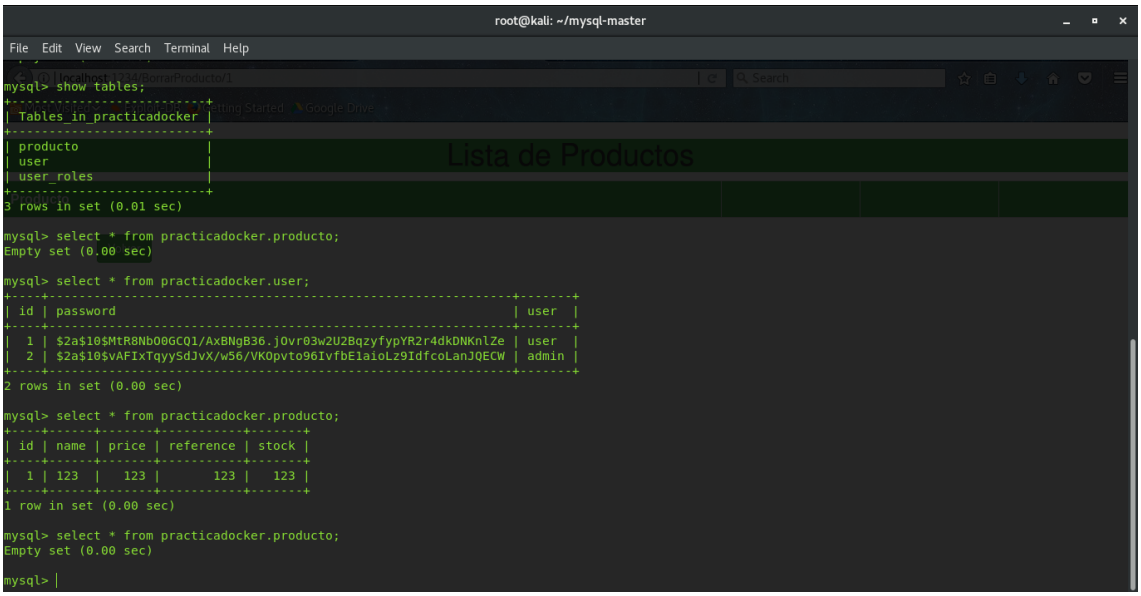


Ilustración 11