

## ***Tarea 15: Modificación y Clusterización de un servicio REST en WildFly***



docker

## **Índice**

1. Introducción
2. Actualización de repositorios
3. Imagen de DockerFile
4. Creación del fichero dockerfile y los 3 nodos.

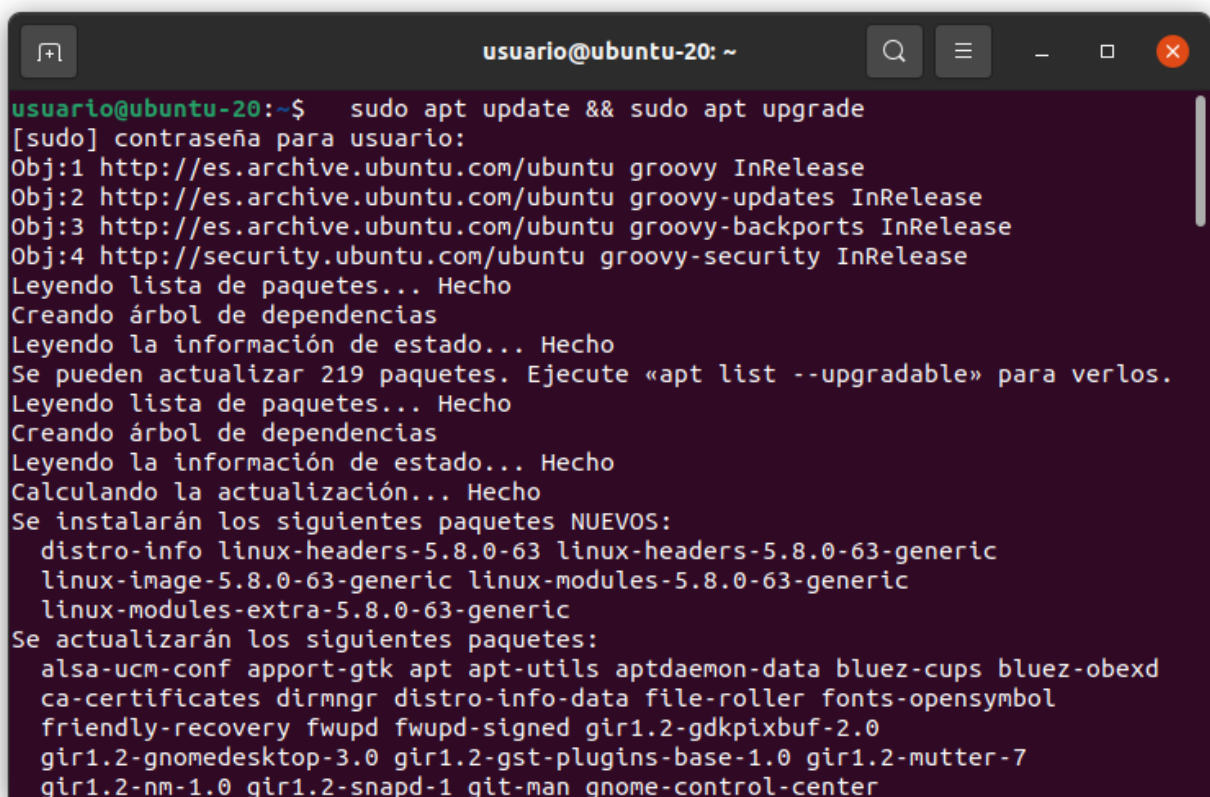
# 1. Introducción

Una vez realizas tareas anteriores, y comenzado a construir entornos de desarrollo ágiles para nuestras app's, en APACHE+PHP+BBDD, vamos a construir y desplegar una app, que requiere wildfly, bajo una api rest, y en el futuro una BBDD.

Además vamos a prevenir el correcto funcionamiento de nuestra app, en tiempos de respuesta y robustez, y vamos a desplegar nuestra solución en 3 nodos de wildfly.

## 2. Actualizacion de repositorios

Actualizamos los repositorios



```
usuario@ubuntu-20: ~  
usuario@ubuntu-20:~$ sudo apt update && sudo apt upgrade  
[sudo] contraseña para usuario:  
Obj:1 http://es.archive.ubuntu.com/ubuntu groovy InRelease  
Obj:2 http://es.archive.ubuntu.com/ubuntu groovy-updates InRelease  
Obj:3 http://es.archive.ubuntu.com/ubuntu groovy-backports InRelease  
Obj:4 http://security.ubuntu.com/ubuntu groovy-security InRelease  
Leyendo lista de paquetes... Hecho  
Creando árbol de dependencias  
Leyendo la información de estado... Hecho  
Se pueden actualizar 219 paquetes. Ejecute «apt list --upgradable» para verlos.  
Leyendo lista de paquetes... Hecho  
Creando árbol de dependencias  
Leyendo la información de estado... Hecho  
Calculando la actualización... Hecho  
Se instalarán los siguientes paquetes NUEVOS:  
  distro-info linux-headers-5.8.0-63 linux-headers-5.8.0-63-generic  
  linux-image-5.8.0-63-generic linux-modules-5.8.0-63-generic  
  linux-modules-extra-5.8.0-63-generic  
Se actualizarán los siguientes paquetes:  
  alsa-ucm-conf apport-gtk apt apt-utils aptdaemon-data bluez-cups bluez-obexd  
  ca-certificates dirmngr distro-info-data file-roller fonts-opensymbol  
  friendly-recovery fwupd fwupd-signed gir1.2-gdkpixbuf-2.0  
  gir1.2-gnomedesktop-3.0 gir1.2-gst-plugins-base-1.0 gir1.2-mutter-7  
  gir1.2-nm-1.0 gir1.2-snapd-1 git-man gnome-control-center
```

## 3. Imagen del docker WildFly

Vemos las imágenes descargadas del docker:

```
usuario@ubuntu-20:~$ sudo docker images
[sudo] contraseña para usuario:
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
wildfly_2            latest             141332b19546       46 hours ago       736MB
wildfly_1            latest             630887e98047       46 hours ago       736MB
jboss/wildfly        25.0.0.Final-config 2a88339acee3       7 days ago         736MB
jboss/wildfly        25.0.0.Final       856694040847       5 weeks ago        736MB
jboss/wildfly        latest             856694040847       5 weeks ago        736MB
hello-world          latest             feb5d9fea6a5       6 weeks ago        13.3kB
```

Comprobamos que las imagenes previamente cargadas del servidor y comprobamos que al acceder a la url veremos la instalacion de wildfly.



Comprobamos que la salida sea la correcta:



Ahora construimos el fichero Dockerfile

```

🐳 Dockerfile
1  FROM jboss/wildfly
2
3  ARG WAR_FILE=target/*.war
4  ##COPY ${JAR_FILE} app.jar
5
6  ADD ${ARG} /opt/jboss/wildfly/standalone/deployments/
7
8  ARG WILDFLY_NAME
9  ARG CLUSTER_PW
10
11  ENV WILDFLY_NAME=${WILDFLY_NAME}
12  ENV CLUSTER_PW=${CLUSTER_PW}
13
14  ENTRYPOINT /opt/jboss/wildfly/bin/standalone.sh -b=0.0.0.0 -bmanageme
15  |

```

Y después construimos el fichero .yml

```

version: '3.5'
services:

  wildfly1:
    build:
      context: .
      args:
        WILDFLY_NAME: wildfly_1
        CLUSTER_PW: secret_password
    image: wildfly_1
    ports:
      - 8080:8080
    networks:
      wildfly_network:

  wildfly2:
    build:
      context: .
      args:
        WILDFLY_NAME: wildfly_2
        CLUSTER_PW: secret_password
    image: wildfly_2
    ports:
      - 8081:8080
    networks:
      wildfly_network:

  wildfly3:
    build:
      context: .
      args:
        WILDFLY_NAME: wildfly_3
        CLUSTER_PW: secret_password
    image: wildfly_3
    ports:
      - 8082:8080
    networks:
      wildfly_network:

```

Y por último realizamos el siguiente comandos para crear el cluster de tres nodos

```
usuario@ubuntu-20:~/Escritorio/prueba/app-web-demo$ sudo docker-compose up
[sudo] contraseña para usuario:
Starting app-web-demo_wildfly2_1 ...
Starting app-web-demo_wildfly1_1 ...
Starting app-web-demo_wildfly3_1 ...
```