3.1 Portainer. Gestión de contenedores

Imagen obtenida de https://www.portainer.io/

Esta herramienta sirve para...

Gestionar los distintos contenedores, imágenes, stacks,... que tengamos en nuestra Raspberry Pi a través de un entorno gráfico en lugar de hacerlo a través del terminal del sistema operativo. Cuenta con una versión BE (Business Edition) y otra CE (Community Edition), usaremos la 2ª.

Web de proyecto y otros enlaces de interés

Página web oficial: https://www.portainer.io/

Repositorio de la versión CE en github: https://github.com/portainer/portainer

Documentación del proyecto: https://docs.portainer.io/

Despliegue

> Si crees que instalar portainer del modo que a continuación se explica es complicado puedes instalarlo a través del método que explicamos en el capítulo 3.3 Linux Media Delivery System (LMDS). No te librará de utilizar la terminal pero quizás te resulte mas amigable.

En la propia documentación podemos encontrar como desplegar Portainer (https://docs.portainer.io/start/install-ce/server/docker/linux). Vamos a recopilar aquí qué hay que hacer y explicar los comandos

```
docker volume create portainer_data
docker run -d -p 8000:8000 -p 9443:9443 --name portainer --restart=always -v
/var/run/docker.sock:/var/run/docker.sock -v portainer_data:/data portainer/portainer-ce:latest
```

En la primera línea creamos un volumen llamado portainer_data

En la segunda línea lanzamos, desplegamos un contenedor:

- -d (--detach): Ejecuta un contenedor en segundo plano.
- -p (--expose): Nos permite indicar qué puerto del contenedor se corresponde con qué puerto de la máquina anfitriona.
- --name: Nos permite establecer el nombre del contenedor.
- --restart: Nos permite establecer qué queremos que ocurra en caso de que el contenedor falle. En este caso establecemos que se reunicie siempre.
- -v (--volume): Nos permite mapear rutas del contenedor con rutas de la máquina anfitriona.
- El último parámetro que aparece en la ruta `portainer/portainer-ce:latest` es la imagen que se va a ejecutar.

> Visto y explicado cómo realizar la instalación según indica la documentación oficial, nosotros/as vamos a hacerlo de otro modo.

La forma que hemos visto con anterioridad funciona. Podéis usarla sin ningún problema. Ahora bien, dado que en este curso desconozco el nivel de partida de cada compañero/a que lo cursa voy a optar por utilizar un modo de despliegue semejante para cada servicio y, por ello, voy a hacer uso de docker-compose. Vamos allá:

Para ello accedemos al terminal y escribimos lo siguiente:

```
cd $HOME
mkdir portainer
cd portainer
nano docker-compose.yml
```

Dentro del fichero escribimos el siguiente contenido:

```
version: '2'
services:

  portainer:
    container_name: portainer
    image: portainer/portainer-ce
    restart: unless-stopped
    ports:
      - 9000:9000
    volumes:
      - /var/run/docker.sock:/var/run/docker.sock
      - ./volumes/data:/data
```

Para salir del fichero pulsaremos `control + x` y guardaremos los cambios. Posteriormente ponemos en marcha los contenedores con `docker compose up -d` Aparecerá en pantalla algo similar a

```
pablo@raspberrypicatedu:~/portainer $ docker compose up -d
[+] Running 12/12
 ✔ portainer 11 layers [⣿⣿⣿⣿⣿⣿⣿⣿⣿⣿⣿⣿⣿⣿⣿⣿]      0B/0B      Pulled                    16.1s
   ✔ 772227786281 Pull complete                                                              0.6s
   ✔ 96fd13befc87 Pull complete                                                              0.8s
   ✔ 23f2184d3136 Pull complete                                                              3.4s
   ✔ 96063559028e Pull complete                                                              8.9s
   ✔ 4c3b3bc85ff4 Pull complete                                                             10.1s
   ✔ 9d8366b4fa62 Pull complete                                                             11.3s
   ✔ 569864db7e82 Pull complete                                                             11.5s
   ✔ 3f3e23a6219f Pull complete                                                             12.7s
   ✔ 36cd77991df0 Pull complete                                                             13.6s
   ✔ db99531051cb Pull complete                                                             13.7s
   ✔ 4f4fb700ef54 Pull complete                                                             13.8s
[+] Running 2/2
 ✔ Network portainer_default  Created                                                        0.2s
 ✔ Container portainer        Started                                                       14.8s
pablo@raspberrypicatedu:~/portainer $
```

Elaboración propia

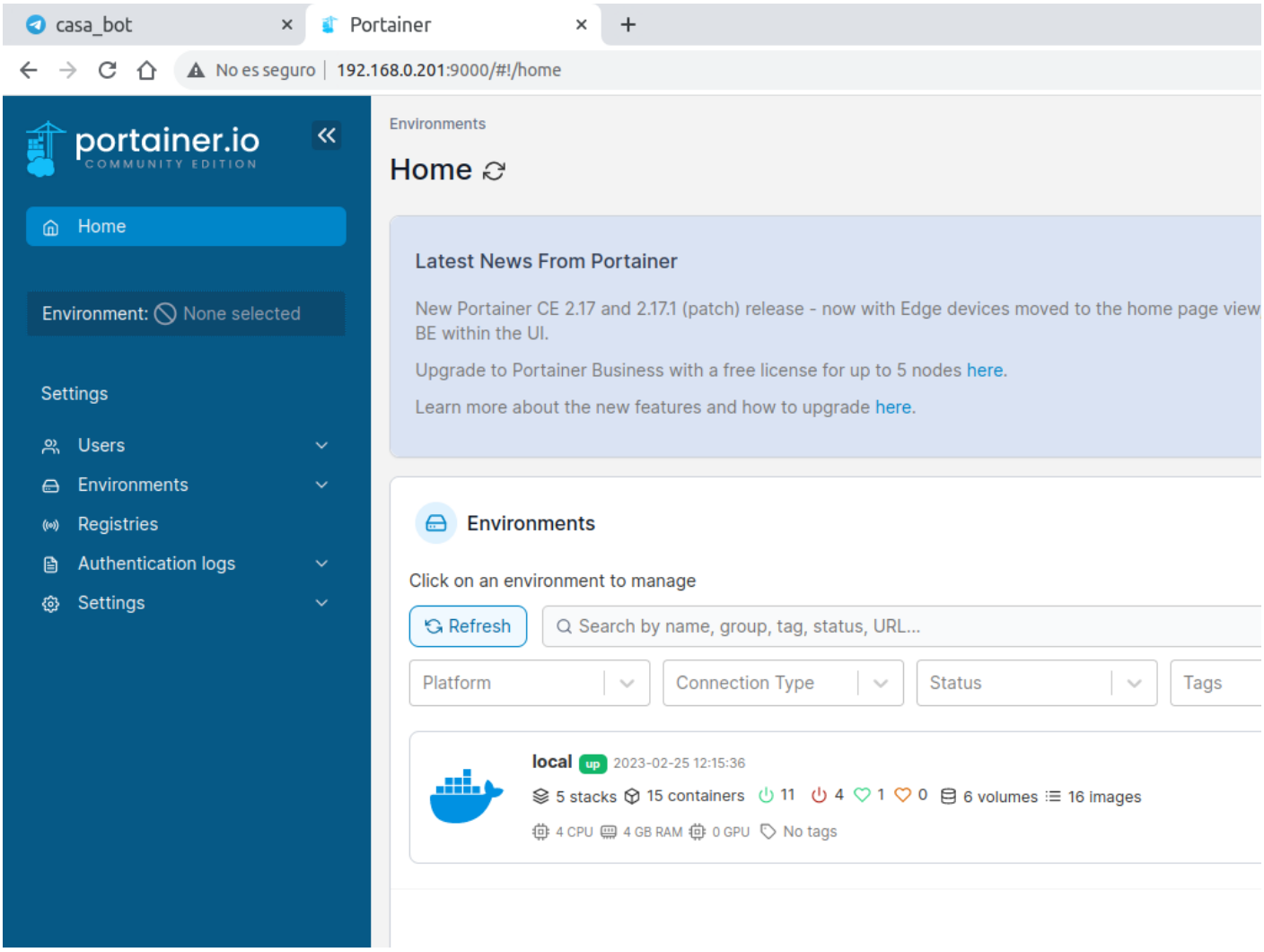
Si queremos comprobar que el contenedor está en marcha podemos ejecutar `docker ps --all` lo que nos mostrará todos los contenedor que hay en la máquina. Si queremos ver si, concretamente, está disponible el que acabamos de crear podemos ejecutar docker `ps --all | grep portainer`.

Obteniendo unos resultados similares a los siguientes:

```
pi@mediacenter:~/portainer $ docker ps --all
CONTAINER ID   IMAGE                                COMMAND        CREATED             STATUS                     PORTS                                                                                                NAMES
950d6cb766ca   portainer/portainer-ce               "/portainer"   About a minute ago  Up 59 seconds              8000/tcp, 9443/tcp, 0.0.0.0:9000->9000/tcp                                                           portainer
9dd6a4624109   crazymax/diun:latest                 "diun serve"   7 days ago          Up 7 days                                                                                                                        diun
5dccdf56873b   filebrowser/filebrowser:s6           "/init"        8 days ago          Exited (111) 8 days ago                                                                                                          quizzical_raman
58d473db4066   filebrowser/filebrowser:s6           "/init"        8 days ago          Exited (111) 8 days ago                                                                                                          nifty_dubinsky
9d868df0e951   filebrowser/filebrowser:s6           "/init"        8 days ago          Exited (111) 8 days ago                                                                                                          relaxed_murdock
32baad3eabc2   lscr.io/linuxserver/wireguard:latest  "/init"        3 weeks ago         Up 9 days                  0.0.0.0:51820->51820/udp                                                                              wireguard
aca7f70a7079   pihole/pihole:latest                 "/s6-init"     2 months ago        Up 9 days (healthy)        0.0.0.0:53->53/udp, 0.0.0.0:53->53/tcp, 0.0.0.0:80->80/tcp, 0.0.0.0:67->67/udp                       pihole
ed3cb8e33da9   ghcr.io/linuxserver/duckdns          "/init"        2 months ago        Exited (0) 2 days ago                                                                                                            duckdns
pi@mediacenter:~/portainer $
pi@mediacenter:~/portainer $
pi@mediacenter:~/portainer $ docker ps --all | grep portainer
950d6cb766ca   portainer/portainer-ce               "/portainer"   About a minute ago  Up About a minute          8000/tcp, 9443/tcp, 0.0.0.0:9000->9000/tcp                                                           portainer
```

Elaboración propia

También podemos tratar de acceder a la interface gráfica a través de un navegador web. Para ello accedemos a través del navegador la Raspberry Pi y al servicio Portainer del siguiente modo `http://<IP>:puerto` en mi caso tengo configurada la raspberry Pi con la IP `192.168.0.201` y portainer con el puerto `9000` por lo que escribo `http://192.168.0.201:9000` y así accedo a la interface web de portainer.



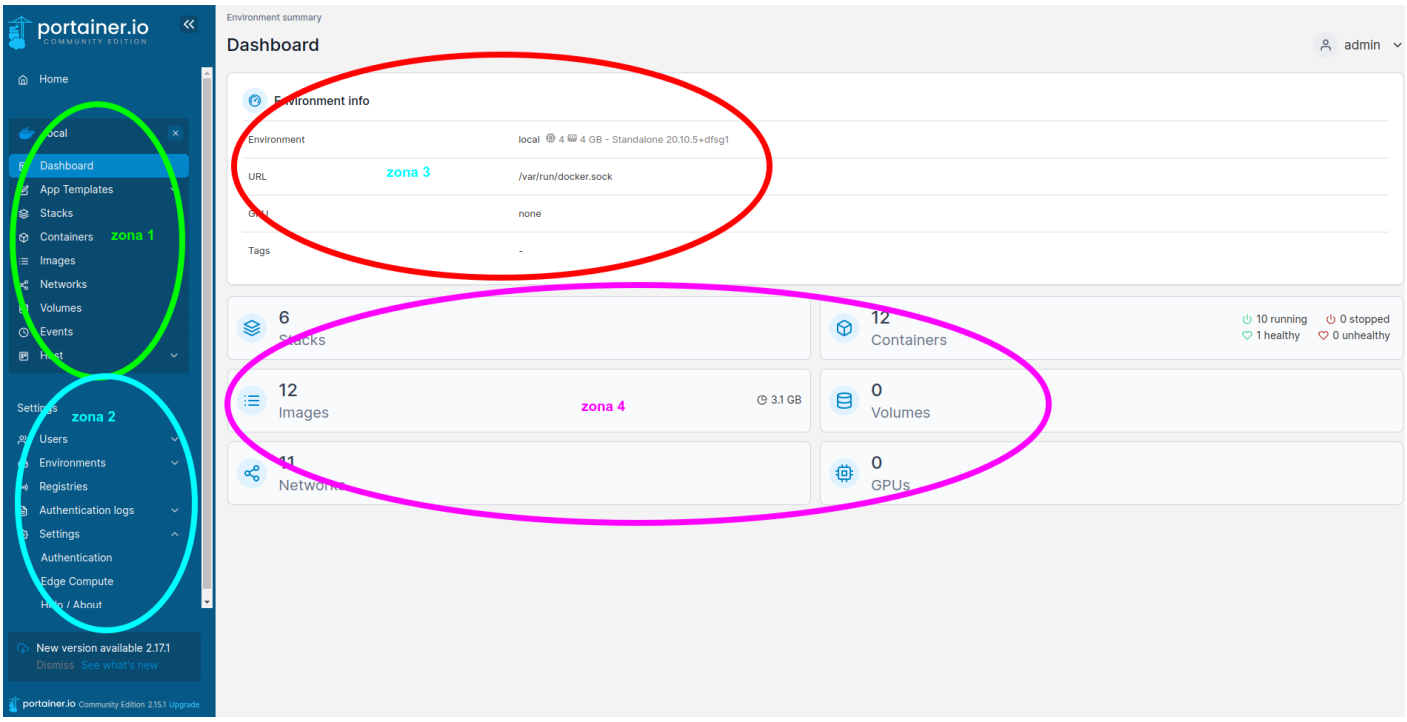
Elaboración propia

Funcionamiento

> En el resto del curso el despliegue de los distintos servicios lo haré siempre a través de comandos pero debes saber que con esta herramienta puedes hacer lo mismo en un entorno gráfico.

Como ya indicamos en la introducción esta herramienta es una interface web para docker lo cual facilitará enormemente la vida a aquellas personas que no estén acostumbradas a trabajar desde una interface de texto (terminal).

Nada mas acceder veremos una imagen como la que hemos visto un par de párrafos más arriba en ella podemos gestionar los usuarios, entornos, logs, parámetros de configuración... de este primer menú lateral no voy a comentaros nada. Si pinchamos en local veremos una imagen como la siguiente:



Elaboración propia

En la misma os he remarcado 4 zonas:

- zona 1 (verde): Tenemos acceso a los diferentes contenedores, imágenes, redes y volúmenes de nuestro entorno local.
- zona 2 (azul claro): Acceso a las mismas secciones que teníamos antes de entrar en el entorno local.
- zona 3 (rojo): Información del entorno.
- zona 4 (morado): Similar a la zona 1 con algo de información adicional.

Si accedemos a, por ejemplo, `Containers` tendremos un listado de todos los contenedores descargados:

Containers

Container list

admin

Containers

Search... | Start | Stop | Kill | Restart | Pause | Resume | Remove | + Add container

Name	State	Quick Actions	Stack	Image	Created	IP Address	GPUs	Published Ports
bazarr	running		lmds	linuxserver/bazarr	2023-03-05 16:39:02	172.24.0.3	none	6767:6767
deluge	running		lmds	linuxserver/deluge	2023-03-05 16:39:02	172.24.0.5	none	58847:58847 58946:58946 58947:58947 58948:58948 8112:8112 58949
diun	running		diun	crazymax/diun:latest	2023-03-04 23:06:20	172.19.0.2	none	-
duckdns	running		duckdns	lscr.io/linuxserver/duckdns:latest	2023-03-05 10:49:38	172.20.0.2	none	-
file-browser_filebrowser_1	running		file-browser	hurlenko/filebrowser	2023-03-05 19:01:56	172.21.0.2	none	8080:8080
jackett	running		lmds	linuxserver/jackett	2023-03-05 16:39:02	172.24.0.7	none	9117:9117
lidarr	exited		lmds	linuxserver/lidarr	2023-03-05 16:39:02	172.24.0.8	none	8686:8686
pihole	healthy		pi-hole	pihole/pihole:latest	2023-03-19 07:50:02	172.25.0.2	none	67:67 8088:80 53:53
portainer	running		lmds	portainer/portainer-ce	2023-03-05 16:39:02	172.24.0.2	none	9000:9000
radarr	running		lmds	linuxserver/radarr	2023-03-05 16:39:02	172.24.0.6	none	7878:7878
sonarr	running		lmds	linuxserver/sonarr	2023-03-05 16:39:02	172.24.0.4	none	8989:8989
wireguard	running		wireguard	lscr.io/linuxserver/wireguard:latest	2023-02-02 10:10:56	172.18.0.2	none	51820:51820

Elaboración propia

Dónde podemos ver si hay contenedores detenidos, fallando, sanos,... también podemos acceder al detalle de cualquiera de ellos y dentro del detalle detenerlo, reiniciarlo, pausarlo, borralo,... ver los logs, acceder al terminar del contenedor,... todo ello sin necesidad de conocer los comandos docker que hay por detrás:

Containers > deluge

Container details

Actions

Start | Stop | Kill | Restart | Pause | Resume | Remove | Recreate | Duplicate/Edit

Container status

ID	b47f1be5aaedb40c84e241577a7b00a4758c7964c8aa25b0ef8339573bf5b458
Name	deluge
Status	Running for 7 hours
Created	2023-03-05 16:39:02
Start time	2023-03-25 09:26:08

Container webhook — Business Edition Feature

Logs | Inspect | Stats | Console | Attach

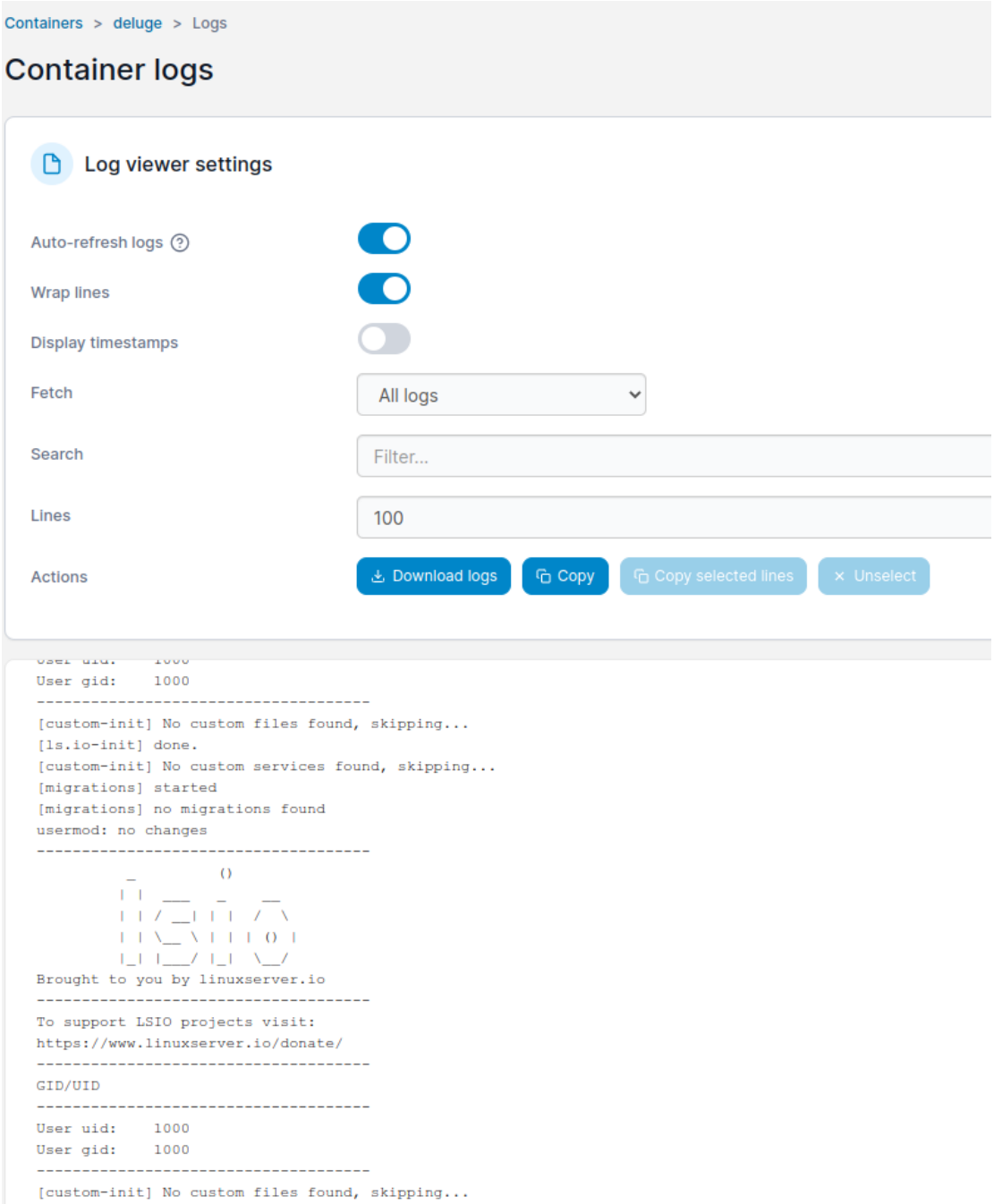
Access control

Ownership	administrators

Change ownership

Create image

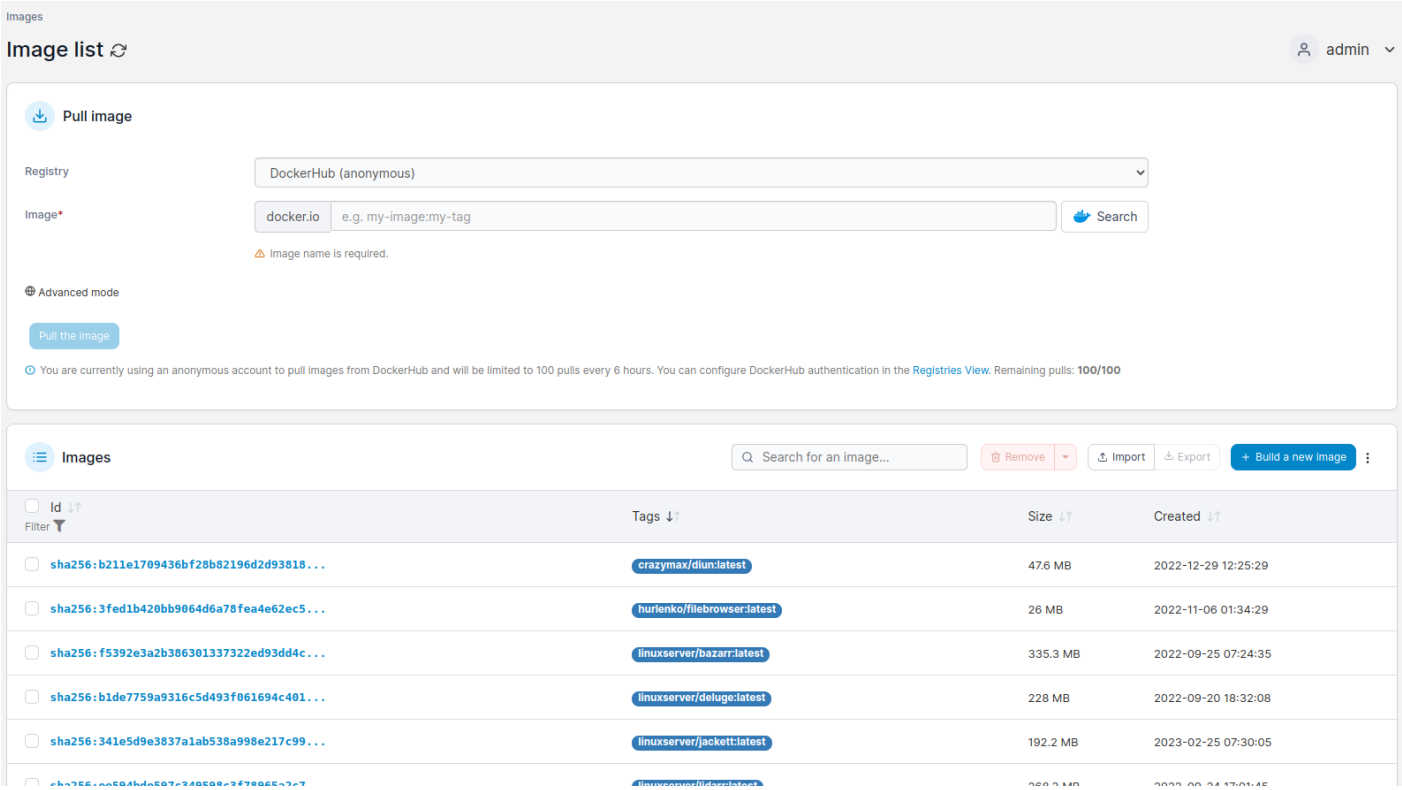
Elaboración propia



Elaboración propia

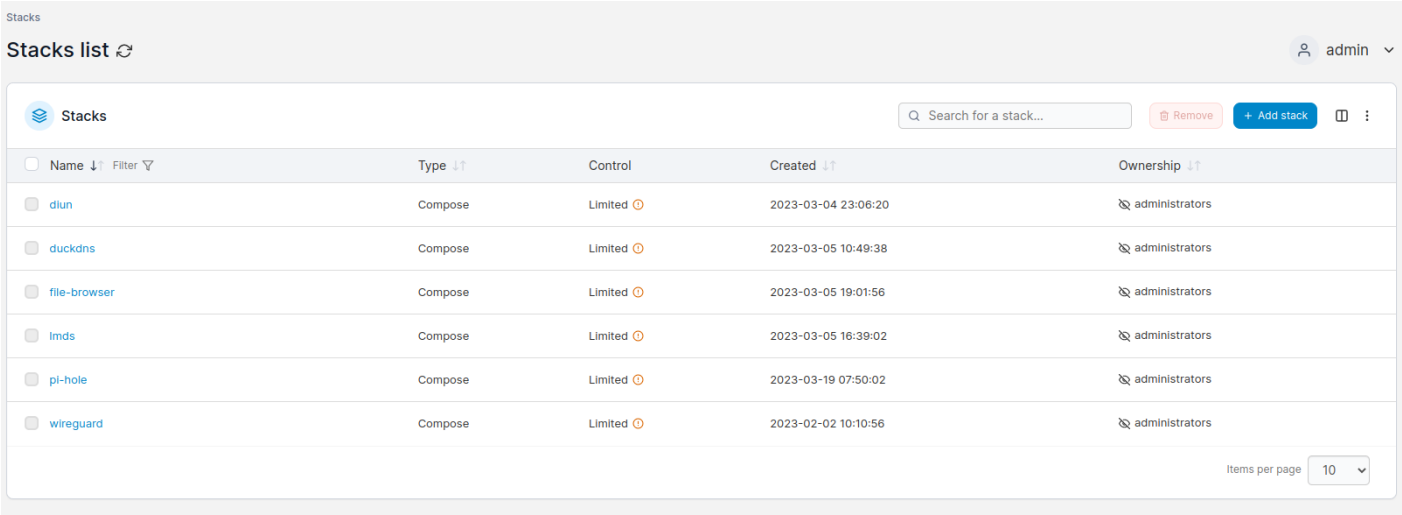
Es bastante probable que nunca tengáis que recurrir a estas opciones pero si en alguna ocasión las necesitáis no tendréis que buscar que con `docker logs nombre_del_contenedor --follow` podéis hacer lo mismo que haciendo 4 clicks.

Acceder a las secciones de `Images` o `Volumes` nos puede resultar muy útil para de un vistazo ver, respectivamente, que imágenes o volúmenes no están en uso y así borrarlas para que dejen de ocupar espacio en nuestro disco duro.



Elaboración propia

Además, por si fuera poco, nos agrupa los contenedores por stacks de modo que nos facilita ver si los contenedores asociados a un determinado servicio (hay servicios que pueden requerir el funcionamiento de mas de 1 contenedor) están operativos o no.



Elaboración propia

Stacks > lmds

Stack details

Stack

Information

This stack was created outside of Portainer. Control over this stack is limited.

Stack details

lmds

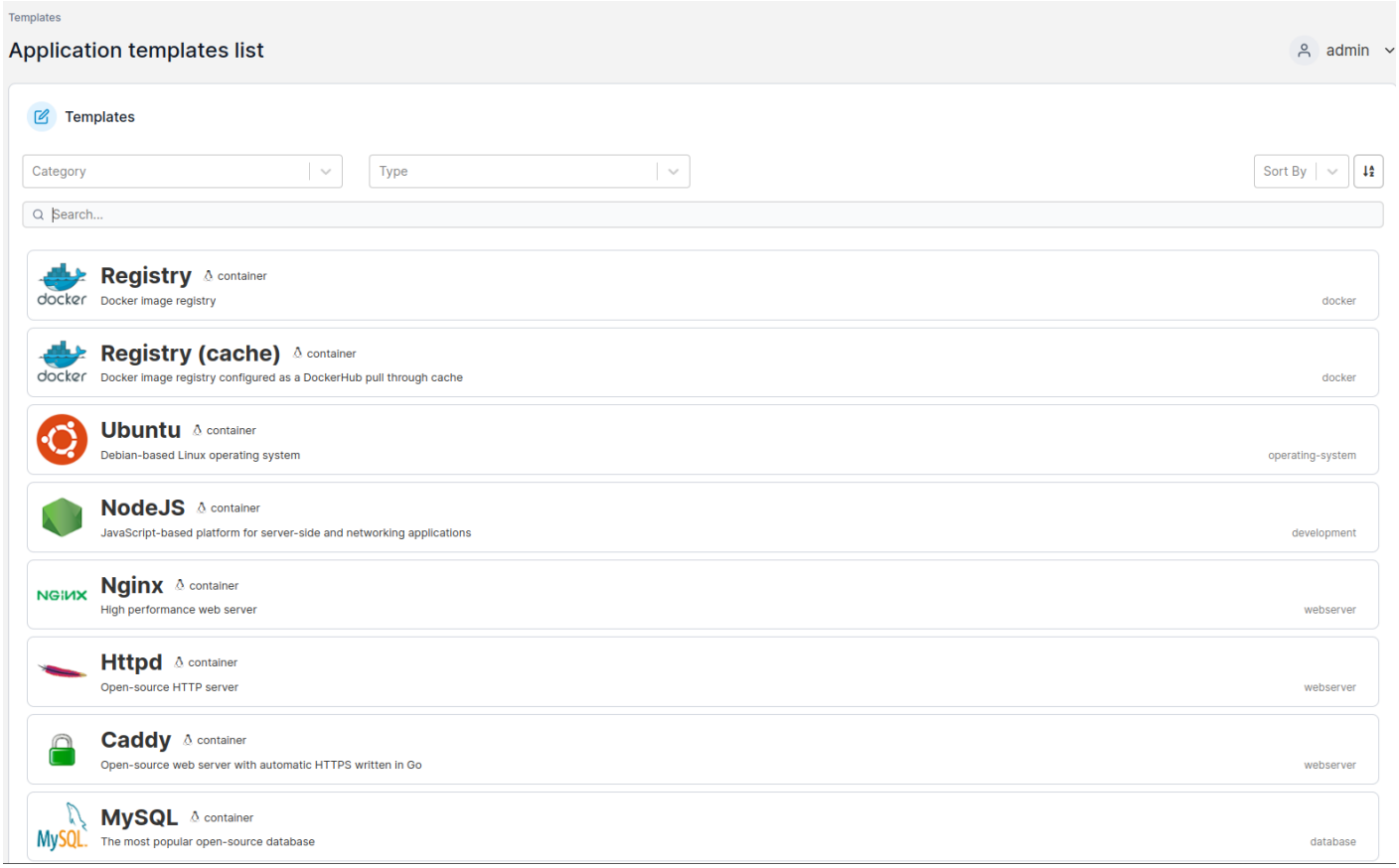
Containers

Search... Start Stop Kill Restar

Name	State	Quick Actions	Stack	Image	Created	IP Address	GPUs	Published Ports
bazarr	running		lmds	linuxserver/bazarr	2023-03-05 16:39:02	172.24.0.3	none	6767:6767
deluge	running		lmds	linuxserver/deluge	2023-03-05 16:39:02	172.24.0.5	none	58949:58949 58950:58950 58846:58846 58848:58
jackett	running		lmds	linuxserver/jackett	2023-03-05 16:39:02	172.24.0.7	none	9117:9117
lidarr	exited		lmds	linuxserver/lidarr	2023-03-05 16:39:02	172.24.0.8	none	8686:8686
portainer	running		lmds	portainer/portainer-ce	2023-03-05 16:39:02	172.24.0.2	none	9000:9000
radarr	running		lmds	linuxserver/radarr	2023-03-05 16:39:02	172.24.0.6	none	7878:7878
sonarr	running		lmds	linuxserver/sonarr	2023-03-05 16:39:02	172.24.0.4	none	8989:8989

Elaboración propia

Y, ya para terminar pero no por ello menos importante, nos ofrece también la posibilidad de crear contenedores a partir de plantillas de aplicaciones preexistentes. Esta funcionalidad puede permitirnos desplegar servicios en segundos sin conocer ni un solo comando de docker (si bien no es lo deseable):



Elaboración propia

Como nos ocurrirá en servicios posteriores, esta herramienta podría dar por si misma para un curso dedicado. Os presento aquellas cuestiones que me parecen mas relevantes a fin de que seáis capaces de continuar vosotros/as desde este punto de partida.

Revision #17
Created 2 February 2023 16:01:51 by Pablo Ruiz
Updated 20 July 2023 17:08:56 by Pablo Ruiz