Seminario de Actualización Dev Ops

Práctica Formativa Obligatoria Nº 2

Comisión: D

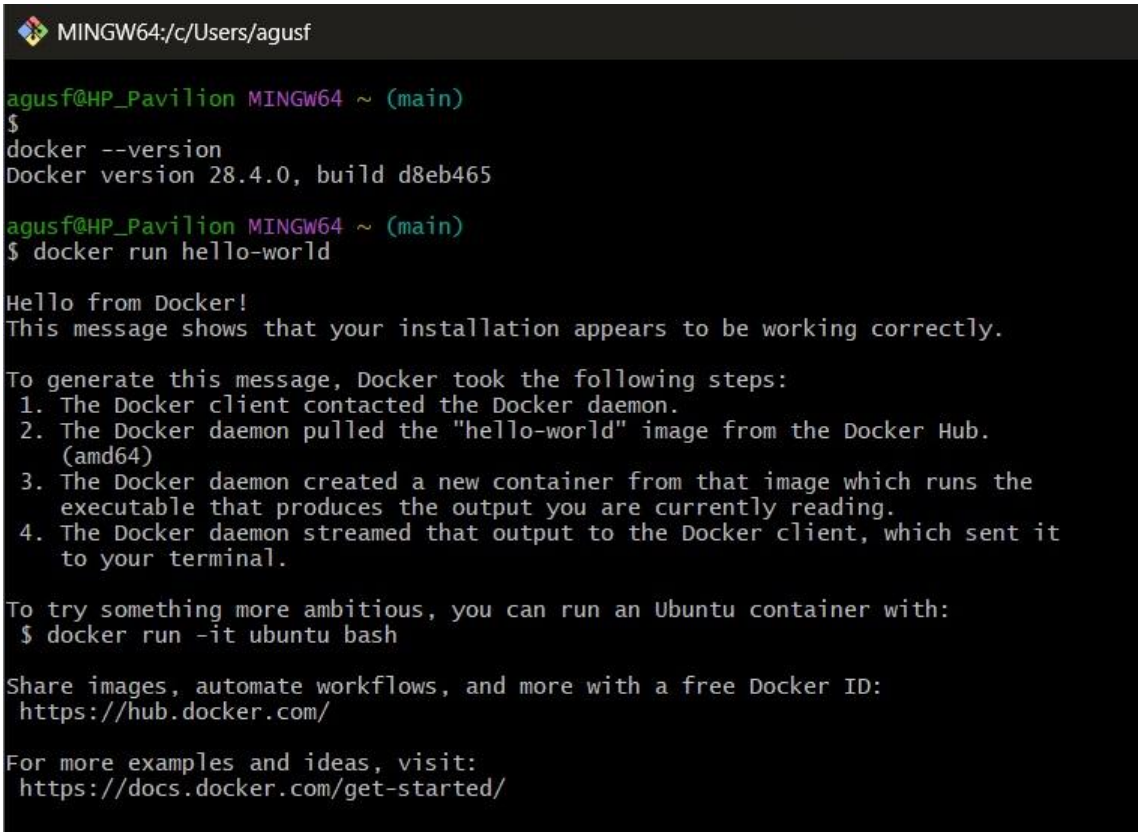
Grupo: 15

Integrantes:

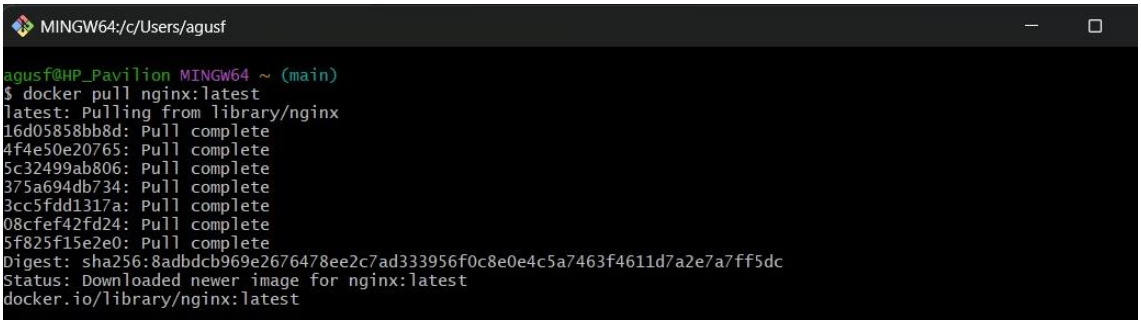
Arias Diego
Godoy Sergio
Ferrari Agustín

Parte I

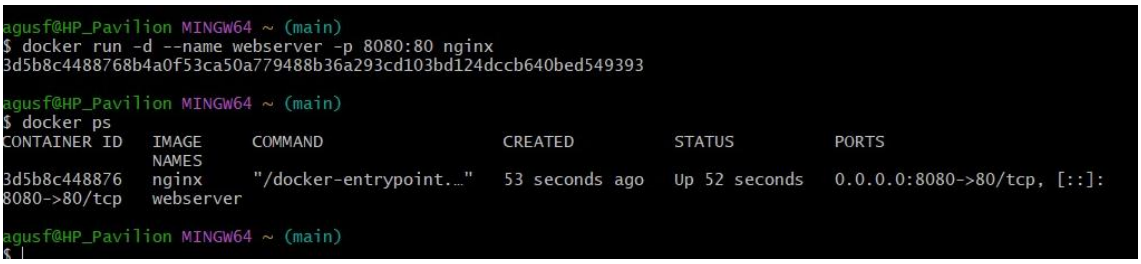
1. Una vez instalado Docker, se creó una cuenta en Docker Hub y se instalo Docker Desktop.
   Se ejecutaron los siguientes comandos para verificar la instalación de Docker:
   docker –version
   docker run hello-world



2. Se buscó una imagen Nginx en Docker Hub y se descargó usando los siguientes comandos:
   docker search nginx
   docker pull nginx:latest

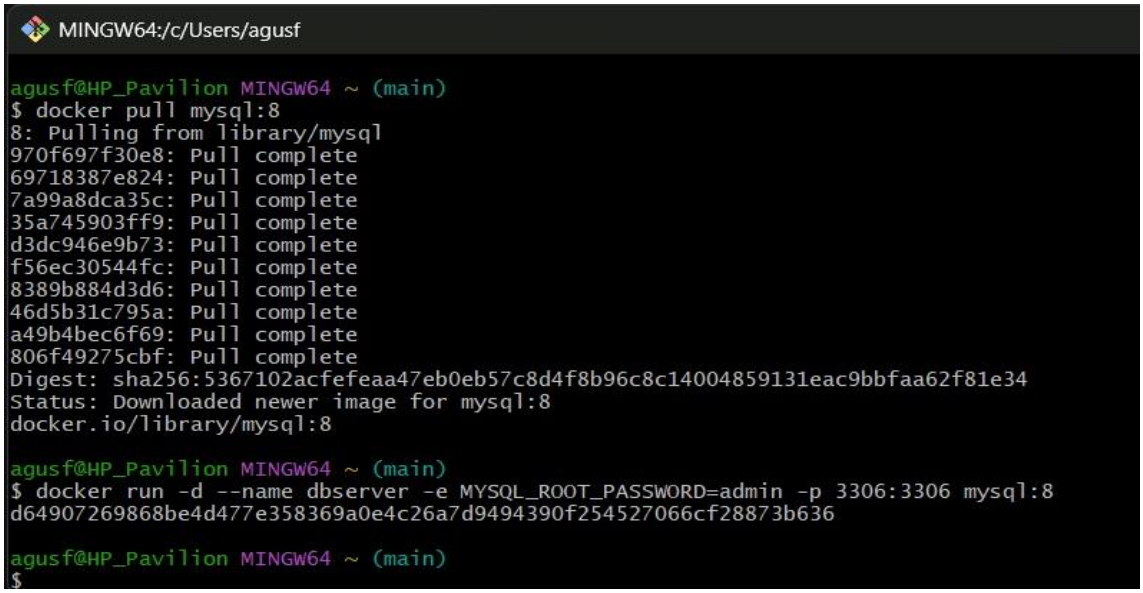


3. Luego se crea un contenedor con esa imagen Nginx y se comprueba con los siguientes comandos:
   docker run -d --name webserver -p 8080:80 nginx
   docker ps



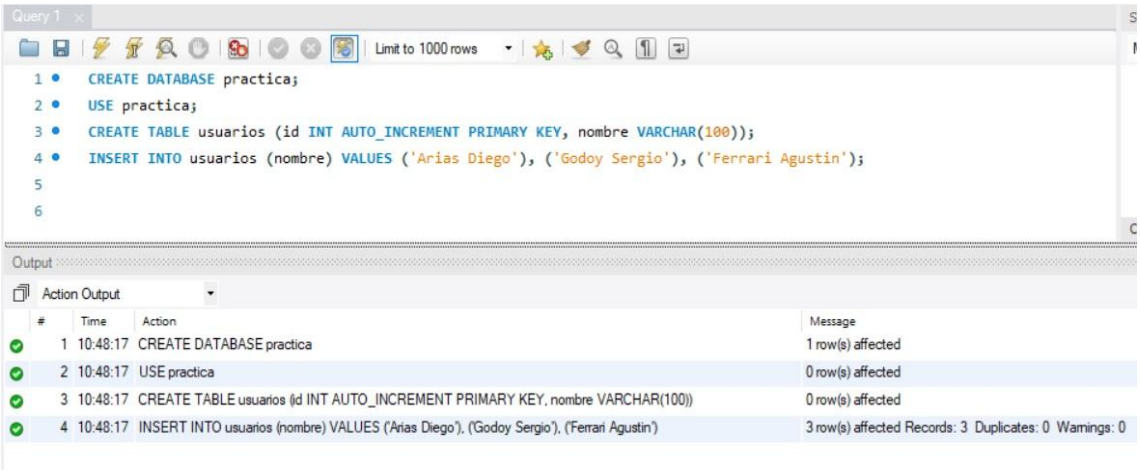
   El contenedor se va a comunicar a través del puerto 8080 con la computadora, lo que se verifica con el navegador en la dirección: *http://localhost:8080*

4. Se crea un contenedor MySql con una imagen descargada de Docker Hub, utilizando los siguientes comandos:
   docker pull mysql:8
   docker run -d --name dbserver -e MYSQL_ROOT_PASSWORD=admin -p 3306:3306 mysql:8



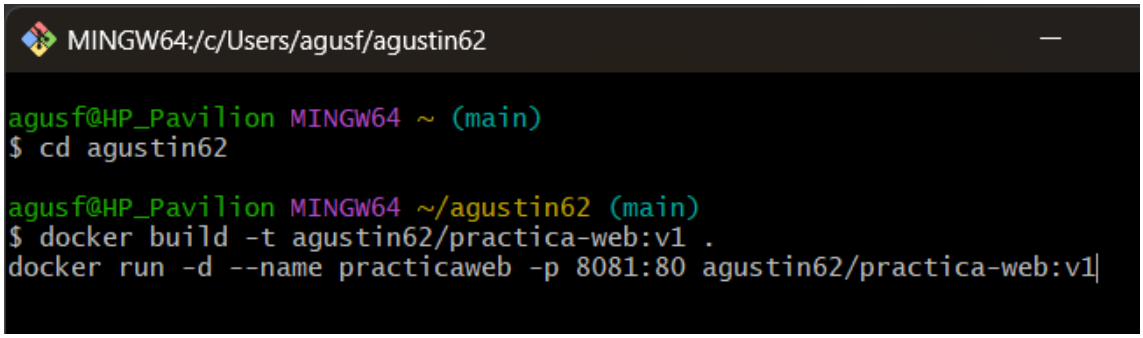
5. Linkeamos los contenedores, usando una red brige, con los siguientes comando:
   docker network create mynet
   docker network connect mynet webserver
   docker network connect mynet dbserver
   y comprobamos con:
   docker network inspect mynet

6. Luego utilizando MySql Workbench nos conectamos a dbserver y creamos una base de datos.



7. Creamos localmente la aplicación de consulta de registros de la base de datos con el nombre index.php

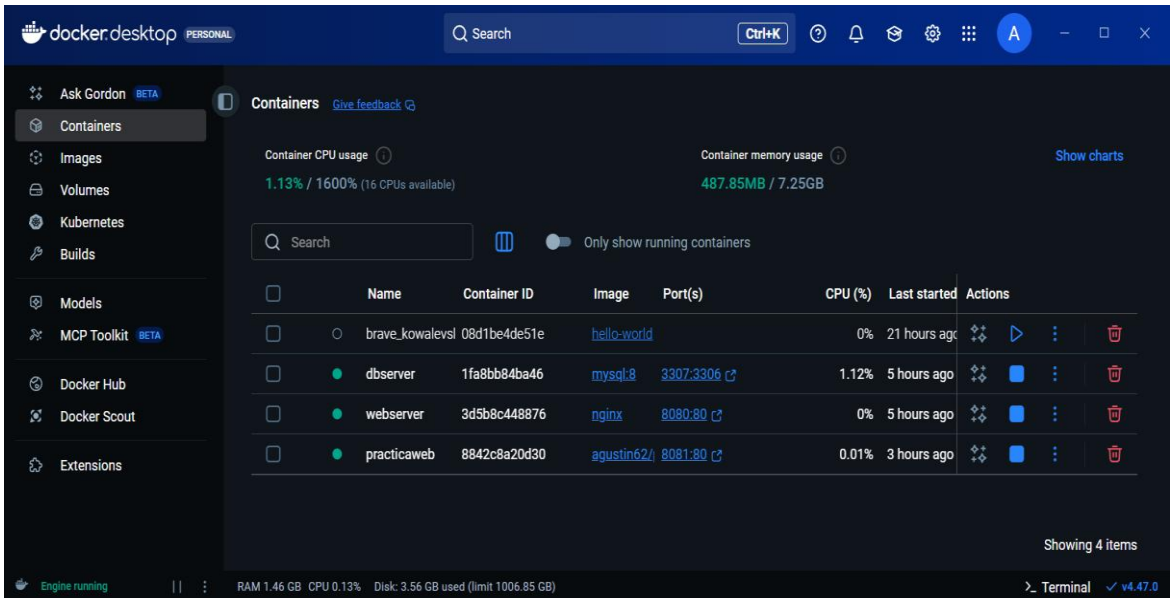
8. Utilizamos un dockerfile, con el siguiente contenido para construir la imagen con la aplicación incorporada:
   FROM php:8.2-apache
   COPY index.php /var/www/html/
   RUN docker-php-ext-install mysqli
   Luego construimos la imagen con los siguientes comandos
   docker build -t agustin62/practica-web:v1 .
   docker run -d --name practicaweb -p 8081:80 agustin62/practica-web:v1



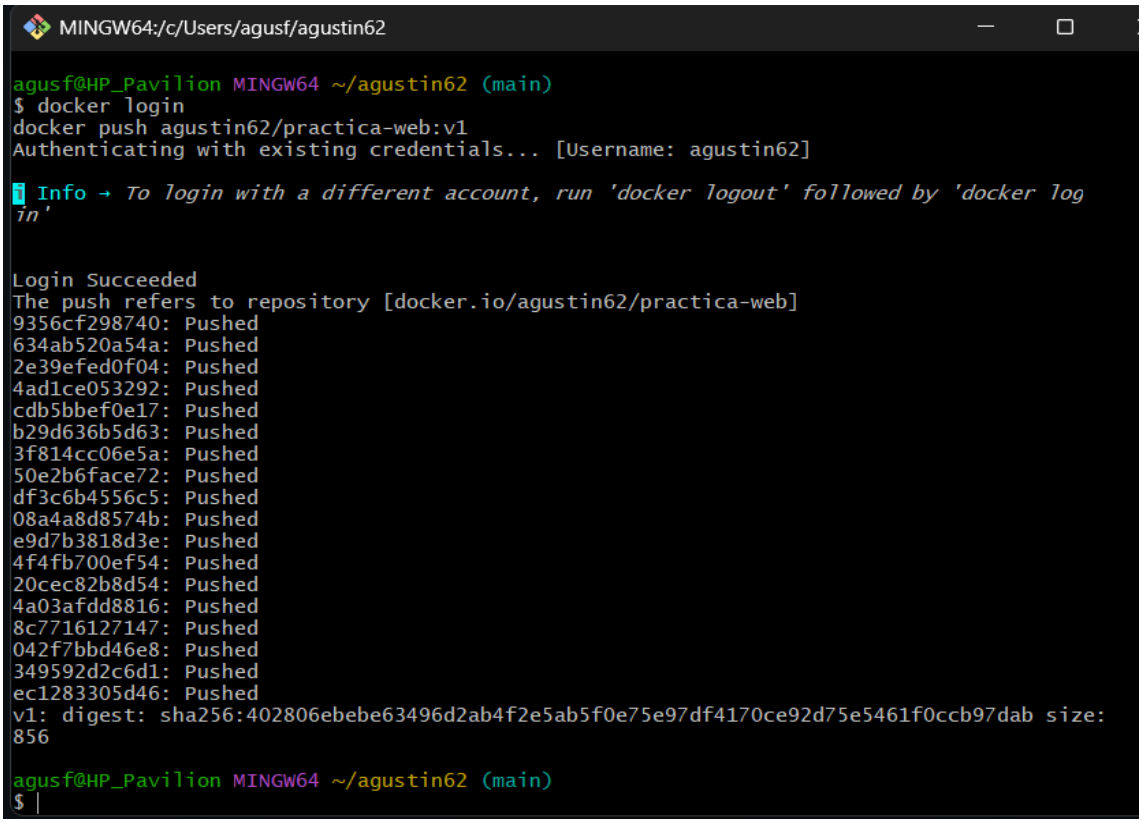
```
MINGW64:/c/Users/agusf/agustin62
[+] Building 41.4s (9/9) FINISHED                                docker:desktop-linux
 => [internal] load build definition from Dockerfile                              0.0s
 => => transferring dockerfile: 124B                                              0.0s
 => [internal] load metadata for docker.io/library/php:8.2-apache                 3.0s
 => [auth] library/php:pull token for registry-1.docker.io                        0.0s
 => [internal] load .dockerignore                                                 0.0s
 => => transferring context: 2B                                                   0.0s
 => [internal] load build context                                                 0.1s
 => => transferring context: 251B                                                 0.1s
 => [1/3] FROM docker.io/library/php:8.2-apache@sha256:b3876890595b471c1eee      22.0s
 => => resolve docker.io/library/php:8.2-apache@sha256:b3876890595b471c1eeeb      0.0s
 => => sha256:4f4fb700ef54461cfa02571ae0db9a0dc1e0cdb5577484a6d75e 32B / 32B     0.4s
 => => sha256:50e2b6face7299e43069931af7f0db8c72d1edb20763ffbcb2 891B / 891B     0.7s
 => => sha256:cdb5bbef0e17cc369fe94c20c6641182328878775ab6b87579 247B / 247B     0.9s
 => => sha256:b29d636b5d63db81da11d665e918098a1474c86258807e167d 252B / 252B     0.9s
 => => sha256:4ad1ce053292f2375b8f6a0517dcb2ebcb95355c64319b 2.46kB / 2.46kB     0.5s
 => => sha256:ec1283305d46a106c5964550157c480c0ea7f4d2e726 11.46MB / 11.46MB     2.1s
 => => sha256:2e39efed0f048e4687c422b747beb5b09d50fd4b1910bafa3a 487B / 487B     0.3s
 => => sha256:634ab520a54aba8c5a6d62d3ddbe679ad8ece38b85bc 12.33MB / 12.33MB     4.3s
 => => sha256:08a4a8d8574bdab5eba763e3bf4a4db8fd3cf95640dcf336dd 485B / 485B     0.8s
 => => sha256:4a03afdd881641c99da1c6e6b7ebd9f598387b9a52df2c8525 427B / 427B     0.3s
 => => sha256:042f7bbd46e8cf11ba540d911d86b5906080f716737ab7 4.22MB / 4.22MB     1.6s
 => => sha256:e9d7b3818d3e96587ed711eca7cf7b0c26da753cb25862d3aa 225B / 225B     0.5s
 => => sha256:349592d2c6d1cb2f129cf094d77965d5666bb0637 117.84MB / 117.84MB      13.8s
 => => sha256:3f814cc06e5ab25037f69a4038c0dceeae6a11dfe9f8327ad4 225B / 225B     0.5s
 => => sha256:8c7716127147648c1751940b9709b6325f2256290d32 29.78MB / 29.78MB     5.7s
 => => extracting sha256:8c7716127147648c1751940b9709b6325f2256290d3201662ec     1.7s
 => => extracting sha256:3f814cc06e5ab25037f69a4038c0dceeae6a11dfe9f8327ad41     0.0s
 => => extracting sha256:349592d2c6d1cb2f129cf094d77965d5666bb06373dd45d9dcd     4.3s
 => => extracting sha256:e9d7b3818d3e96587ed711eca7cf7b0c26da753cb25862d3aa0     0.0s
 => => extracting sha256:042f7bbd46e8cf11ba540d911d86b5906080f716737ab7a94b0     0.3s
 => => extracting sha256:4a03afdd881641c99da1c6e6b7ebd9f598387b9a52df2c85250     0.0s
 => => extracting sha256:08a4a8d8574bdab5eba763e3bf4a4db8fd3cf95640dcf336ddc     0.0s
 => => extracting sha256:634ab520a54aba8c5a6d62d3ddbe679ad8ece38b85bcb152f82     0.1s
 => => extracting sha256:2e39efed0f048e4687c422b747beb5b09d50fd4b1910bafa3a9     0.0s
 => => extracting sha256:ec1283305d46a106c5964550157c480c0ea7f4d2e7263cb746f     0.5s
 => => extracting sha256:4ad1ce053292f2375b8f6a0517dcb2ebcb95355c64319ba2e9a     0.0s
 => => extracting sha256:b29d636b5d63db81da11d665e918098a1474c86258807e167d2     0.0s
 => => extracting sha256:cdb5bbef0e17cc369fe94c20c6641182328878775ab6b875799     0.0s
 => => extracting sha256:50e2b6face7299e43069931af7f0db8c72d1edb20763ffbcb21     0.0s
 => => extracting sha256:4f4fb700ef54461cfa02571ae0db9a0dc1e0cdb5577484a6d75     0.0s
 => [2/3] COPY index.php /var/www/html/                                           0.8s
 => [3/3] RUN docker-php-ext-install mysqli                                      15.0s
 => exporting to image                                                            0.4s
 => => exporting layers                                                           0.2s
 => => exporting manifest sha256:85613a93efa8b7da74c3a1e7d31bc0d2788d411d9b1     0.0s
 => => exporting config sha256:ed6659213f21ce1c819becf62dbc0a355b89d5550ab69     0.0s
 => => exporting attestation manifest sha256:44266722531ca9fdccbc5e51a9b92ae     0.0s
 => => exporting manifest list sha256:402806ebebe63496d2ab4f2e5ab5f0e75e97df     0.0s
 => => naming to docker.io/agustin62/practica-web:v1                              0.0s
 => => unpacking to docker.io/agustin62/practica-web:v1                           0.1s
8842c8a20d30e2620c89a0d8bb391992a48b18e21dcd902ca084fff5dc39fc1f

agusf@HP_Pavilion MINGW64 ~/agustin62 (main)
```

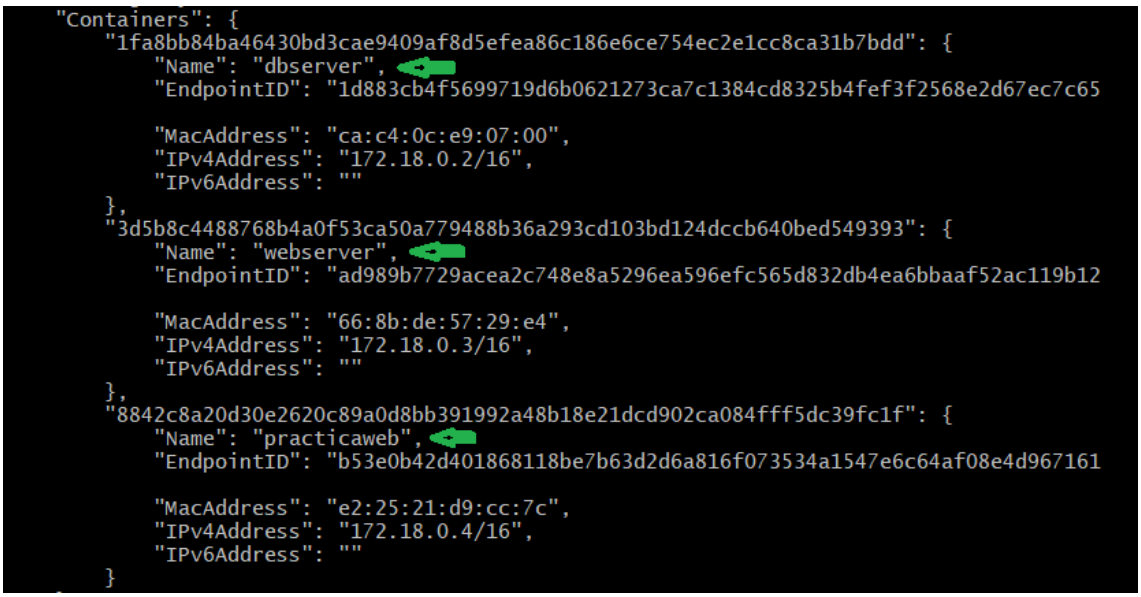
Podemos visualizar los contenedores agregados en Docker Desktop



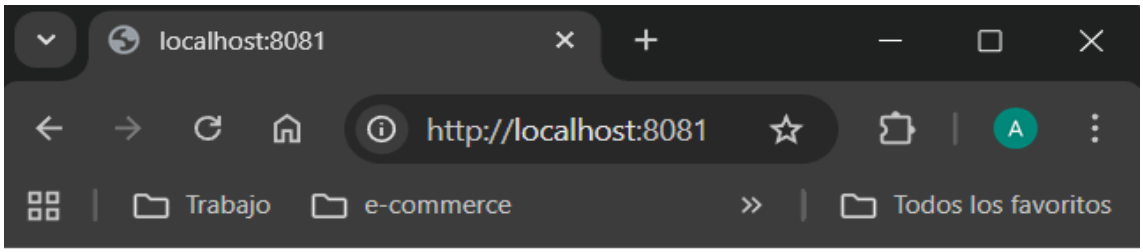
Luego hacemos el push a Docker Hub, utilizando los siguientes comandos:
docker login
docker push agustin62/practica-web:v1



Conectamos el contenedor practicaweb a la red mynet, donde están conectados dbserver y webserver, mediante el comando:
docker network connect mynet practicaweb
Y luego inspeccionamos la red para confirmar la incorporación del contenedor:
docker network inspect mynet



Ahora, utilizando el navegador podemos ver en la dirección:
http://localhost:8081
Se comprueba la ejecución de la aplicación web index.php leyendo los datos de la tabla usuarios.

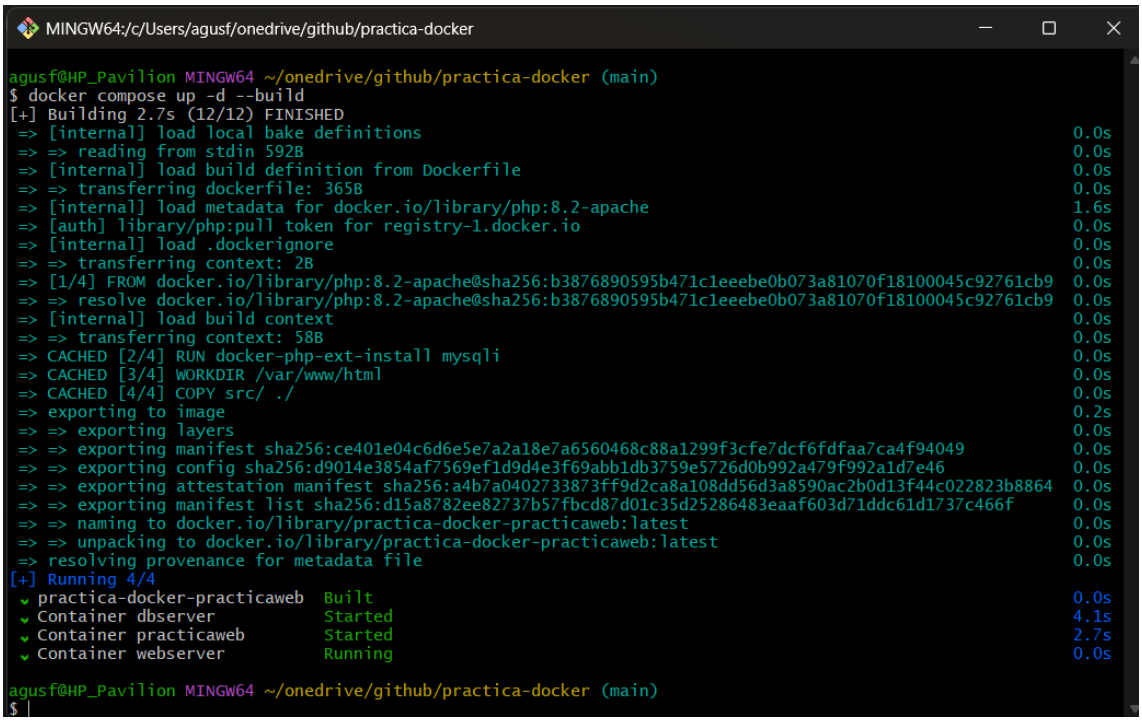


1 - Arias Diego
2 - Godoy Sergio
3 - Ferrari Agustin

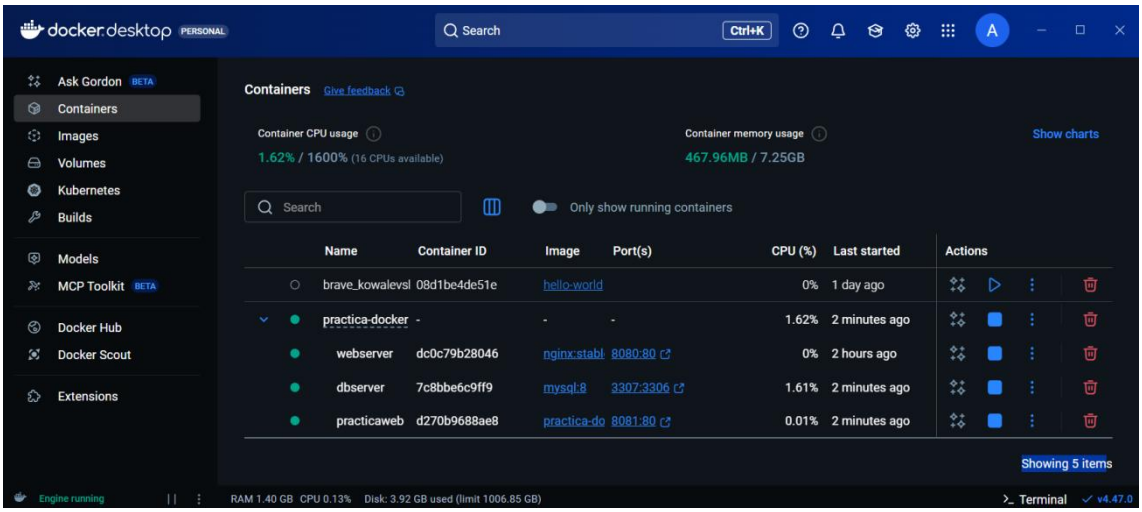
Parte II

Se creó un repositorio en Github, y se clono posteriormente en la PC, utilizando los siguientes comandos:
git init
git remote add origin https://github.com/AgustinCFerrari/practica-docker.git

Una vez hechos el Dockerfile y el docker-compose.yml , se levantan los contenedores con la siguiente instrucción:
Docker compose up –d --build



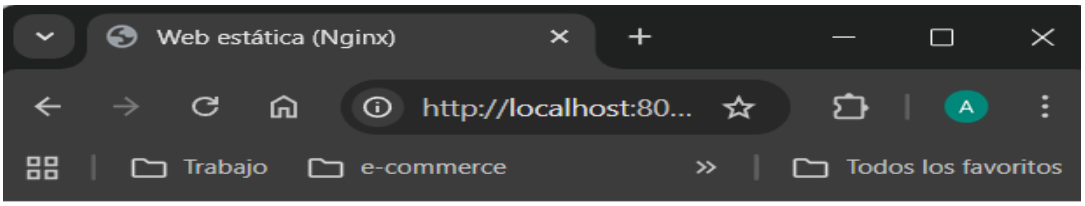
Se verifica que los contenedores están corriendo en el Docker desktop



También se verifica el acceso a los servicios a través del navegador:

http://localhost:8080 ==> Nginx (contenedor webserber)

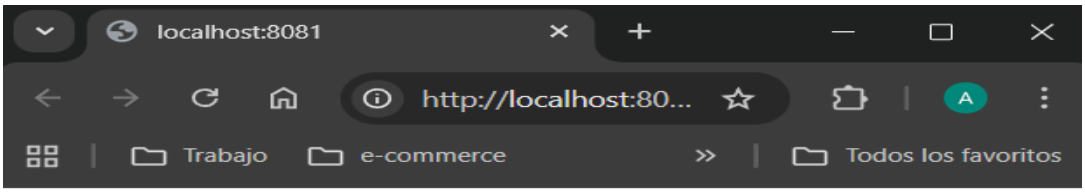
http://localhost:8081 ==> Apache+PHP (contenedor practicaweb), la app PHP consulta la tabla Usuarios de la BD en el contenedor dbserver.



Hola desde Nginx (webserver)

Este sitio sirve archivos estáticos.

La app PHP está en http://localhost:8081



Usuarios

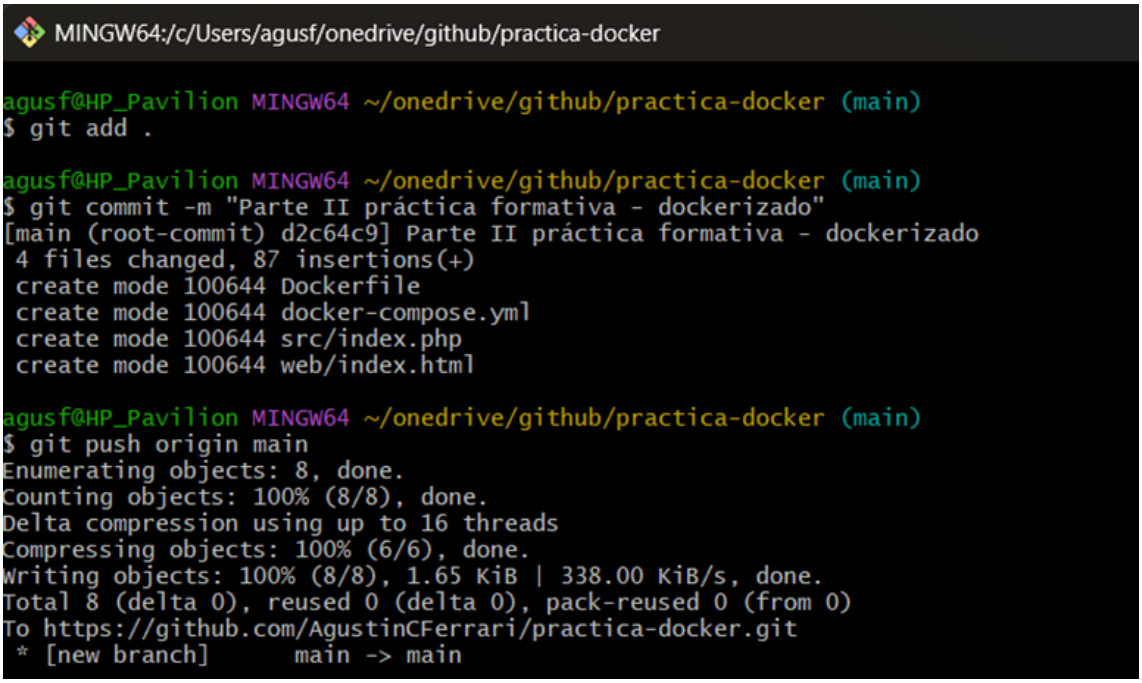
1 - Arias Diego
2 - Godoy Sergio
3 - Ferrari Agustin

Ahora subimos al repositorio de Github los archivos de la práctica formativa, mediante los siguientes comandos:

git add .

git commit -m "Parte II práctica formativa - dockerizado"

git push origin main



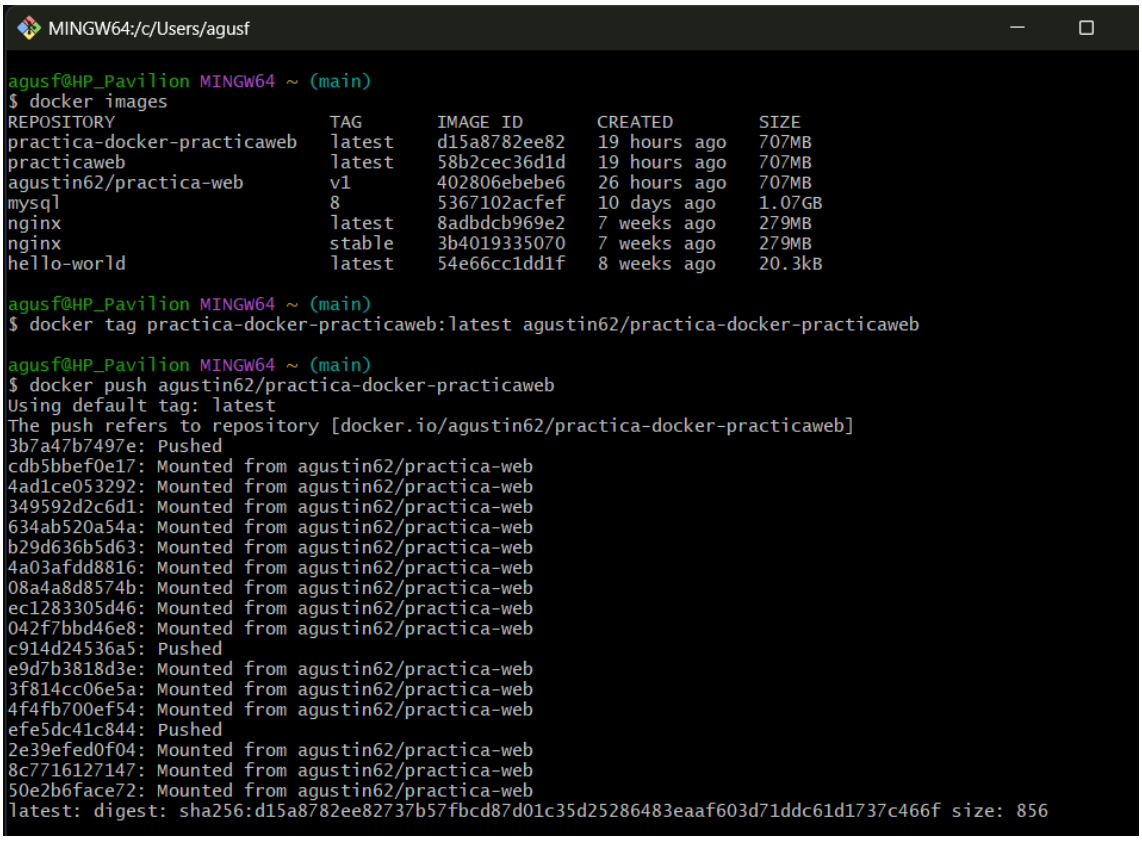
Por último subimos la imagen donde se encuentra la aplicación PHP, al repositorio Docker Hub, con los siguientes comandos

Etiqueta imagen

docker tag practicaweb:latest agustin62/practica-docker-practicaweb

#Sube al repositorio

docker push agustin62/practica-docker-practicaweb

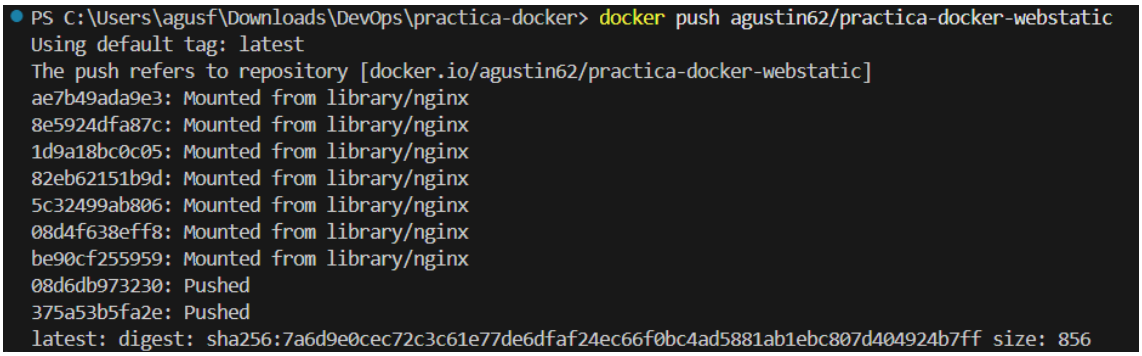


A partir del archivo Dockerfile.web creamos la imagen Nginx donde estará ubicado el archivo index.html, con el comando:

docker build -t agustin62/practica-docker-webstatic -f Dockerfile.web .

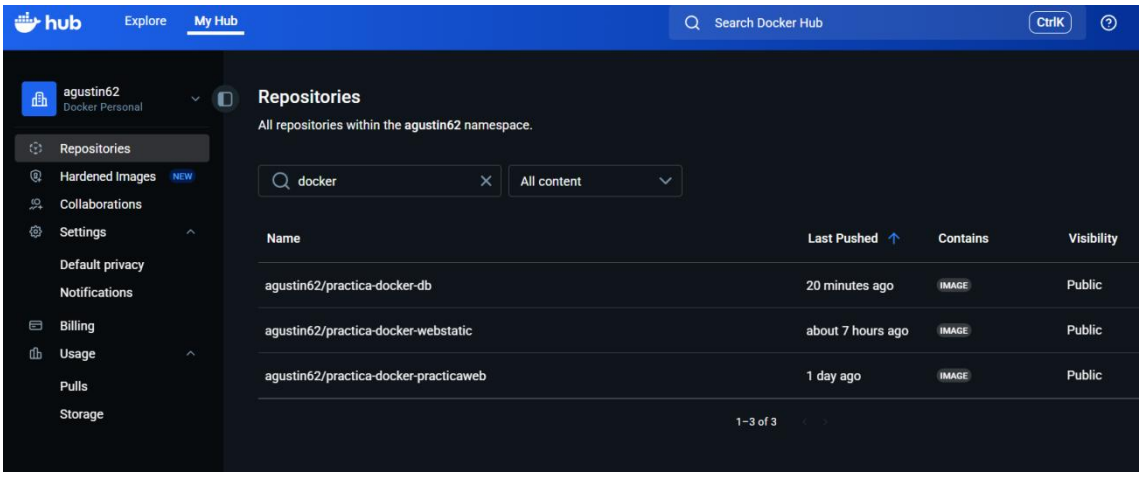
Luego subimos el contenedor al repositorio de Docker Hub, con la instrucción:

docker push agustin62/practica-docker-webstatic



Con el archivo Dockerfile.db creamos la imagen MySql donde estará ubicado el archivo init.sql que que crea la base de datos, mediante el comando:
docker build -t agustin62/practica-docker-db -f Dockerfile.db .
Luego subimos el contenedor al repositorio de Docker Hub, con la instrucción:
docker push agustin62/practica-docker-db

Accediendo a Docker Hub por el navegador vemos ambos repositorios subidos



Con Docker Desktop previamente instalado y además con el archivo docker-compose.yml ubicado en la misma carpeta donde se ejecuta desde la consola de cualquier otra máquina:
Docker compose down –v # borra contenedores + volumen (datos)
Docker compose up –d
Se levantaran aplicaciones multi-contenedor definidas en el archivo docker-compose.yml en segundo plano.