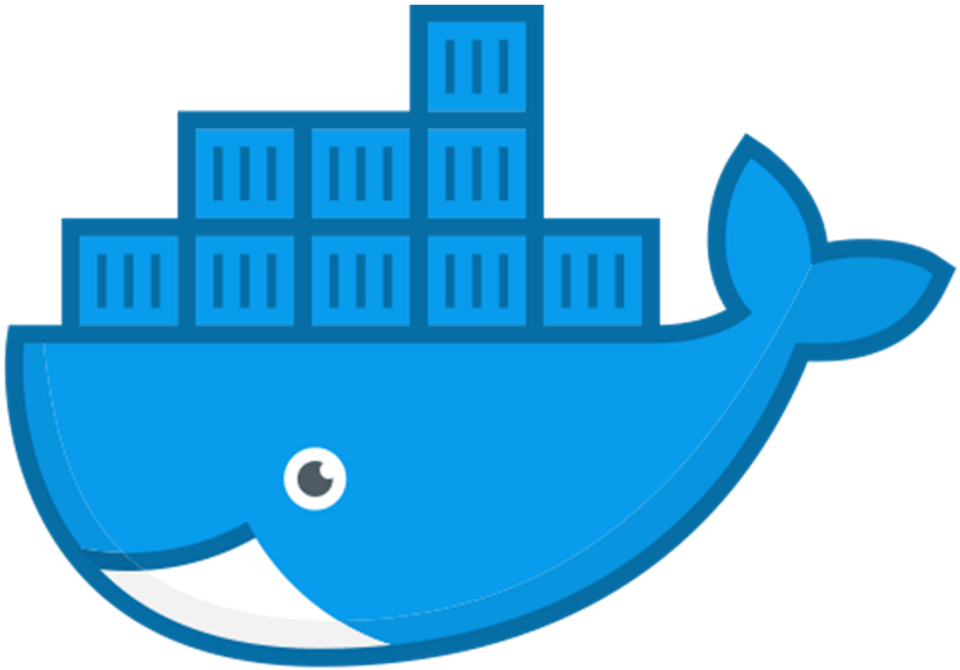
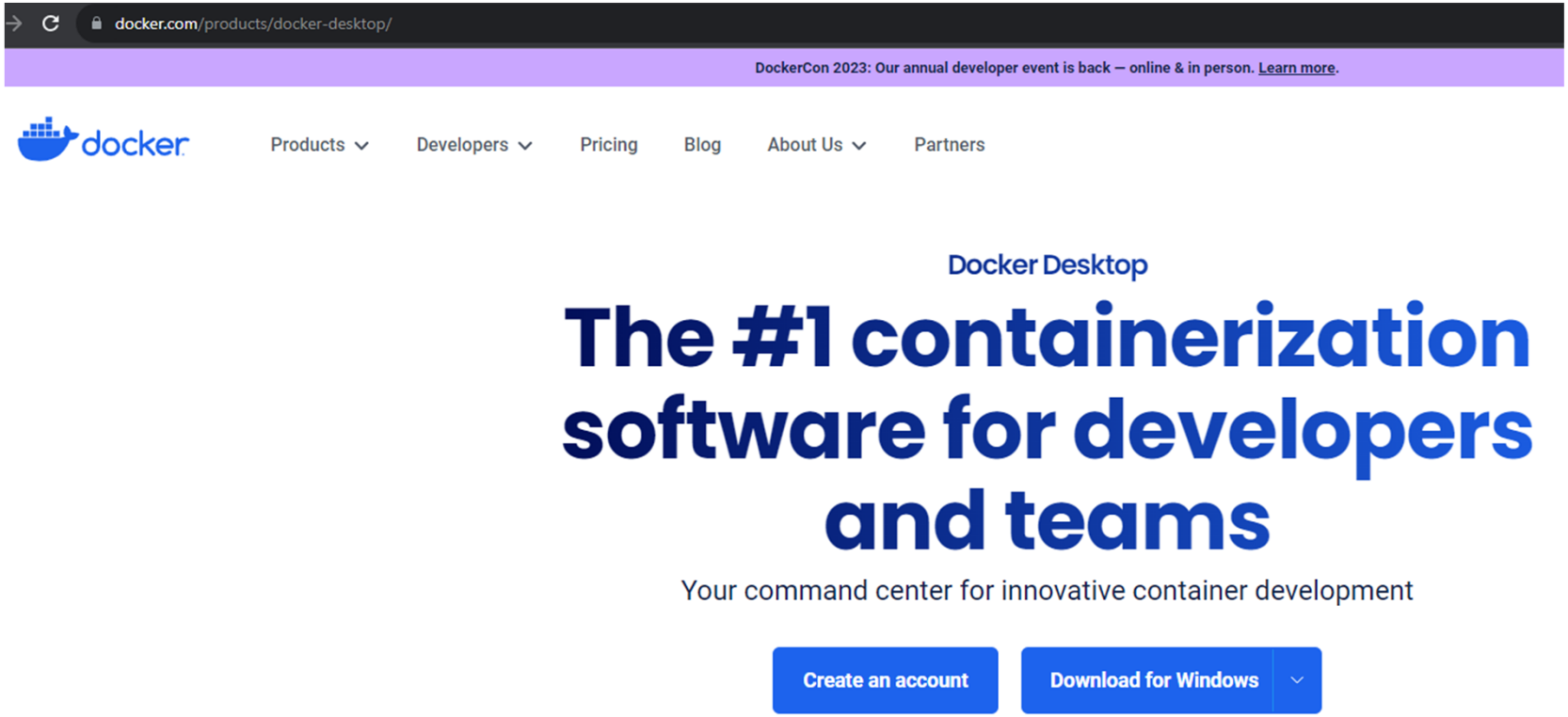
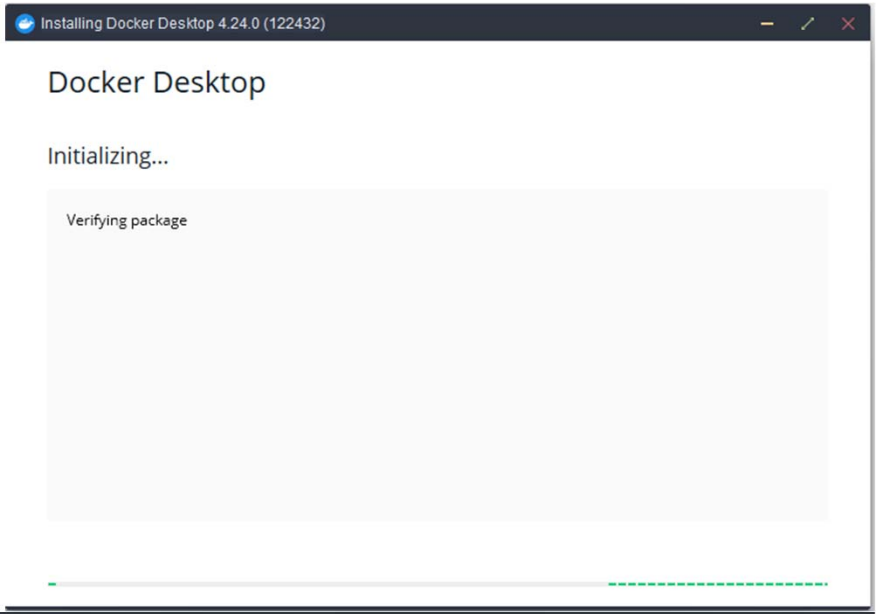


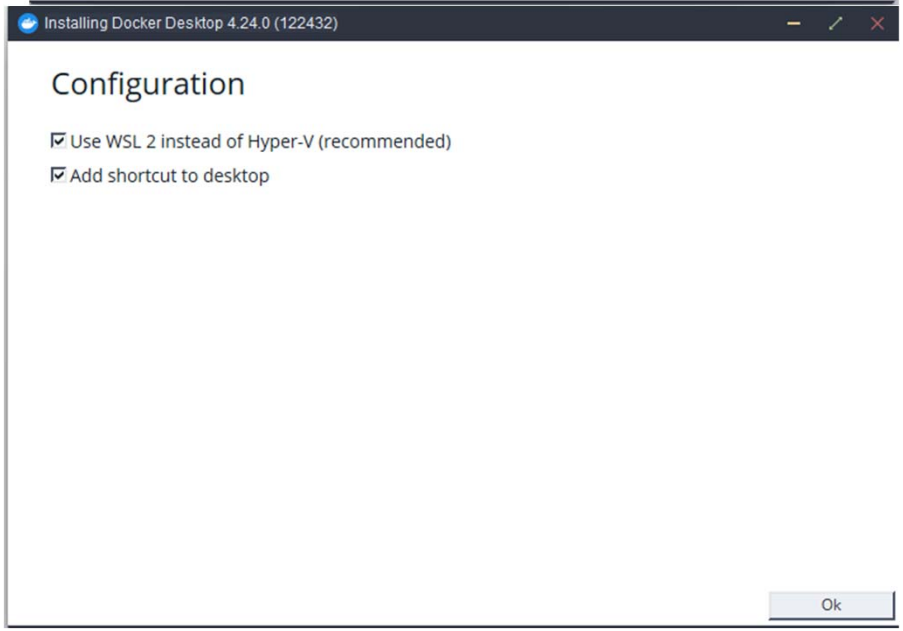
docker®

Docker Desktop

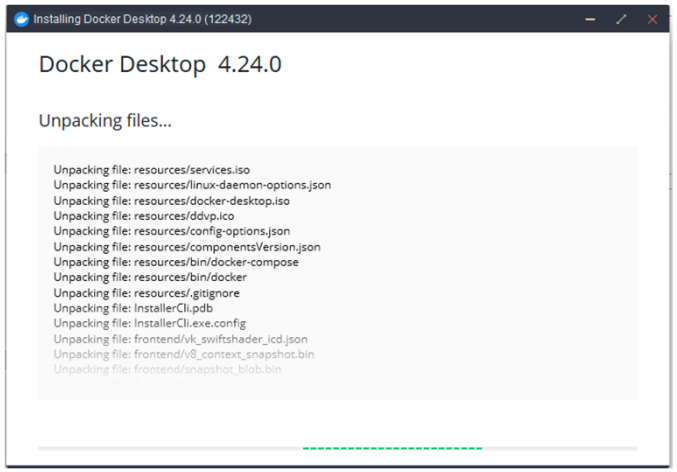


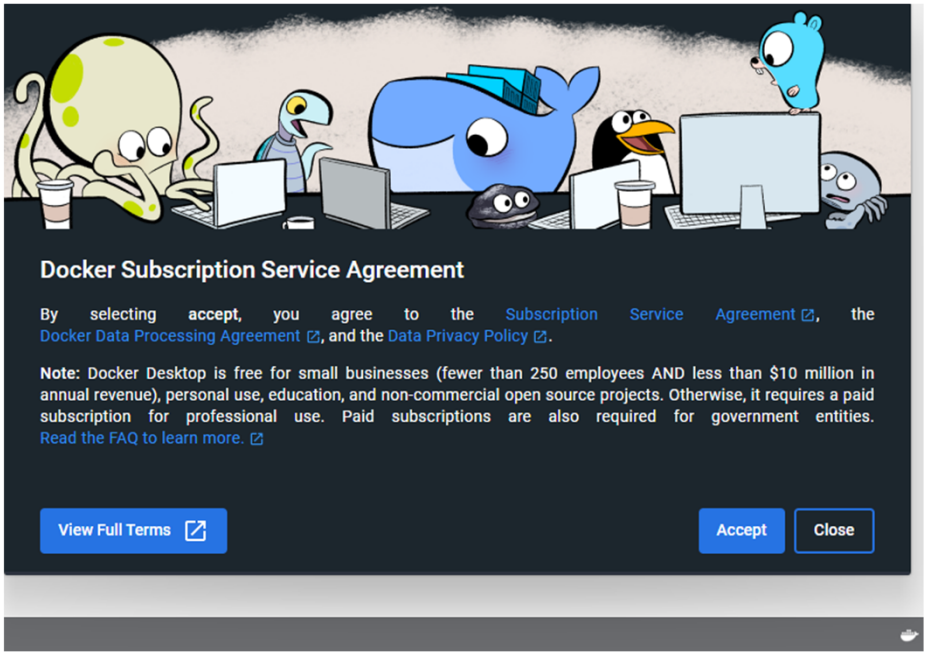
El instalador



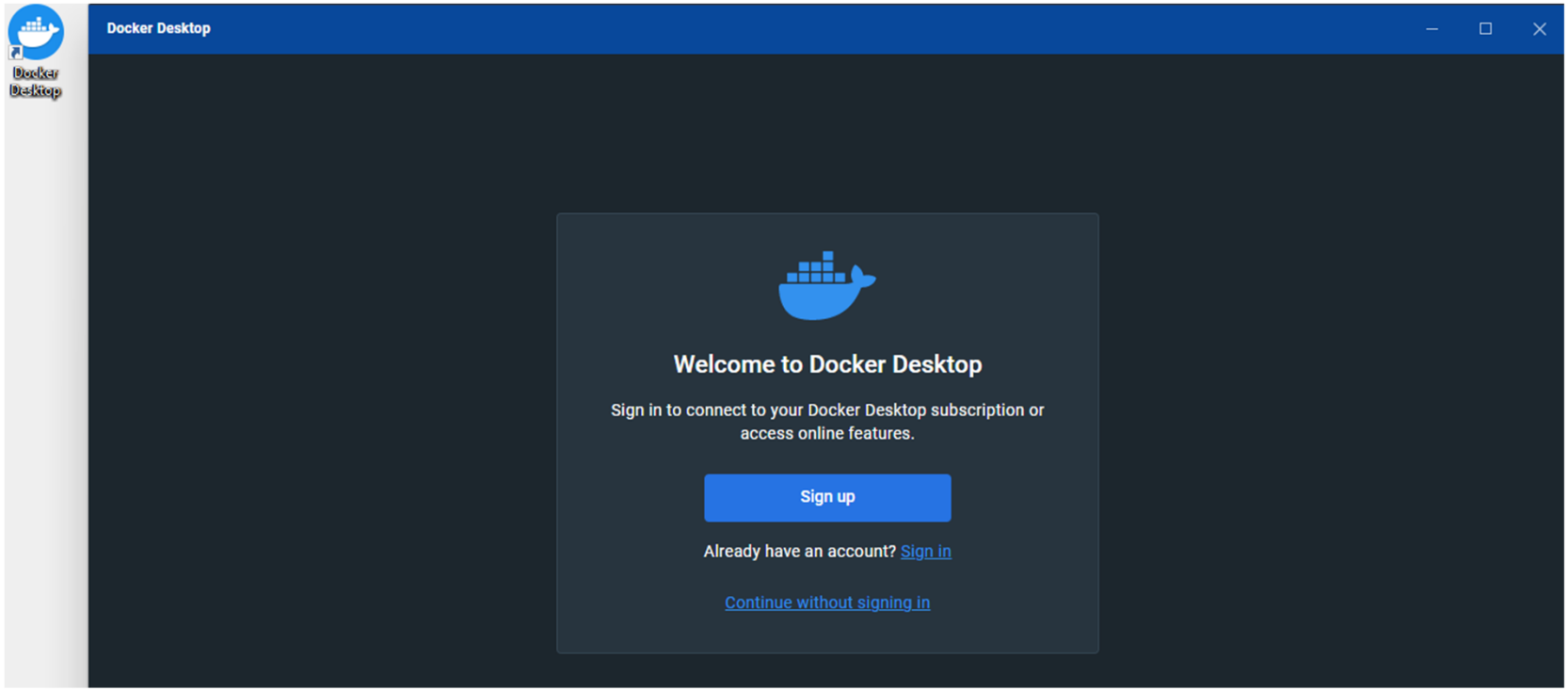


Instalando ...

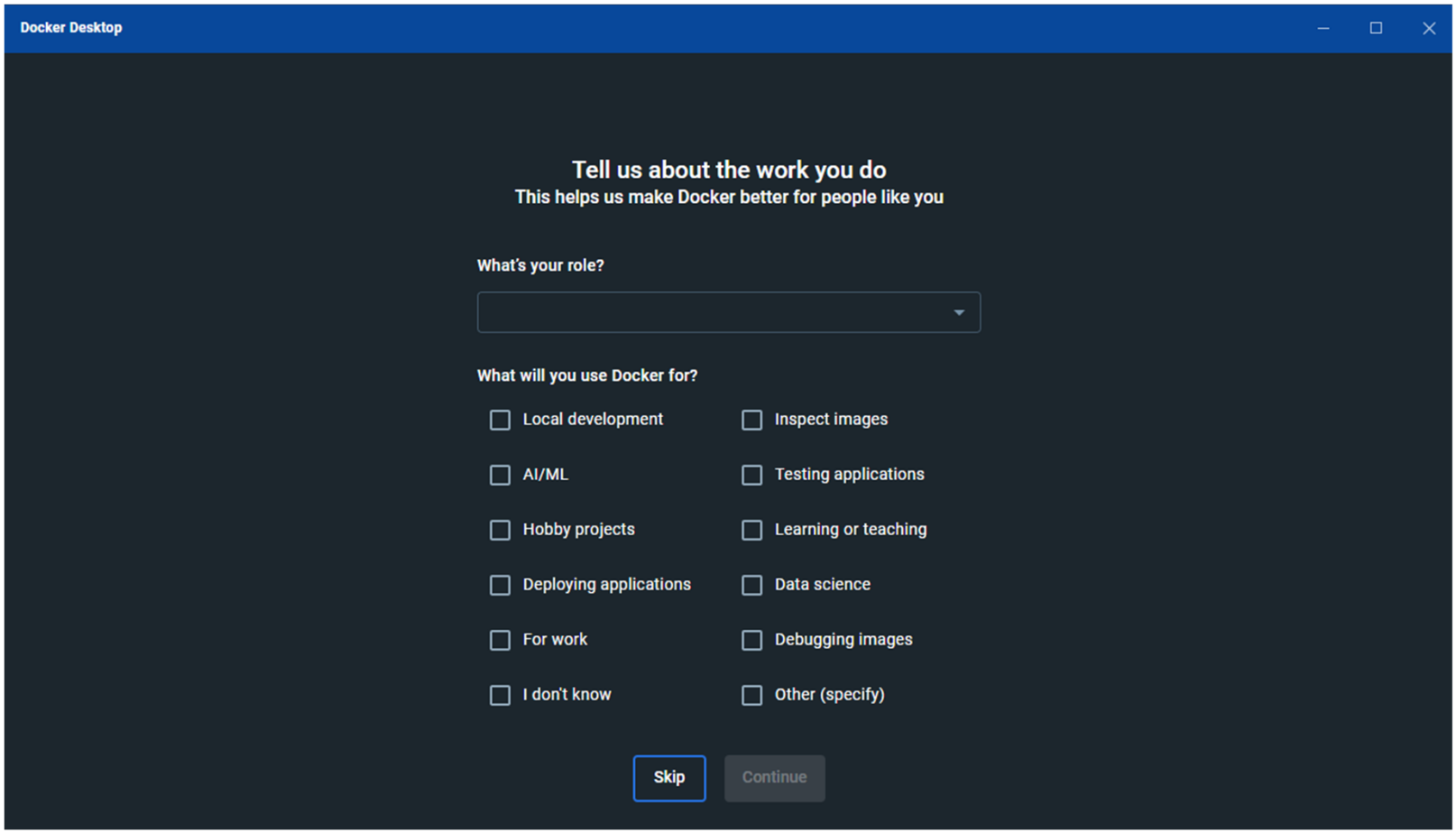

Installing Docker Desktop 4.24.0 (122432)
Docker Desktop 4.24.0
Unpacking files...
Unpacking file: resources/services.iso
Unpacking file: resources/linux-daemon-options.json
Unpacking file: resources/docker-desktop.iso
Unpacking file: resources/ddvp.ico
Unpacking file: resources/config-options.json
Unpacking file: resources/componentsVersion.json
Unpacking file: resources/bin/docker-compose
Unpacking file: resources/bin/docker
Unpacking file: resources/.gitignore
Unpacking file: InstallerCli.pdb
Unpacking file: InstallerCli.exe.config
Unpacking file: frontend/vk_swiftshader_icd.json
Unpacking file: frontend/v8_context_snapshot.bin
Unpacking file: frontend/snapshot_blob.bin


Docker Subscription Service Agreement
By selecting accept, you agree to the Subscription Service Agreement, the Docker Data Processing Agreement, and the Data Privacy Policy.
Note: Docker Desktop is free for small businesses (fewer than 250 employees AND less than $10 million in annual revenue), personal use, education, and non-commercial open source projects. Otherwise, it requires a paid subscription for professional use. Paid subscriptions are also required for government entities. Read the FAQ to learn more.
View Full Terms
Accept
Close

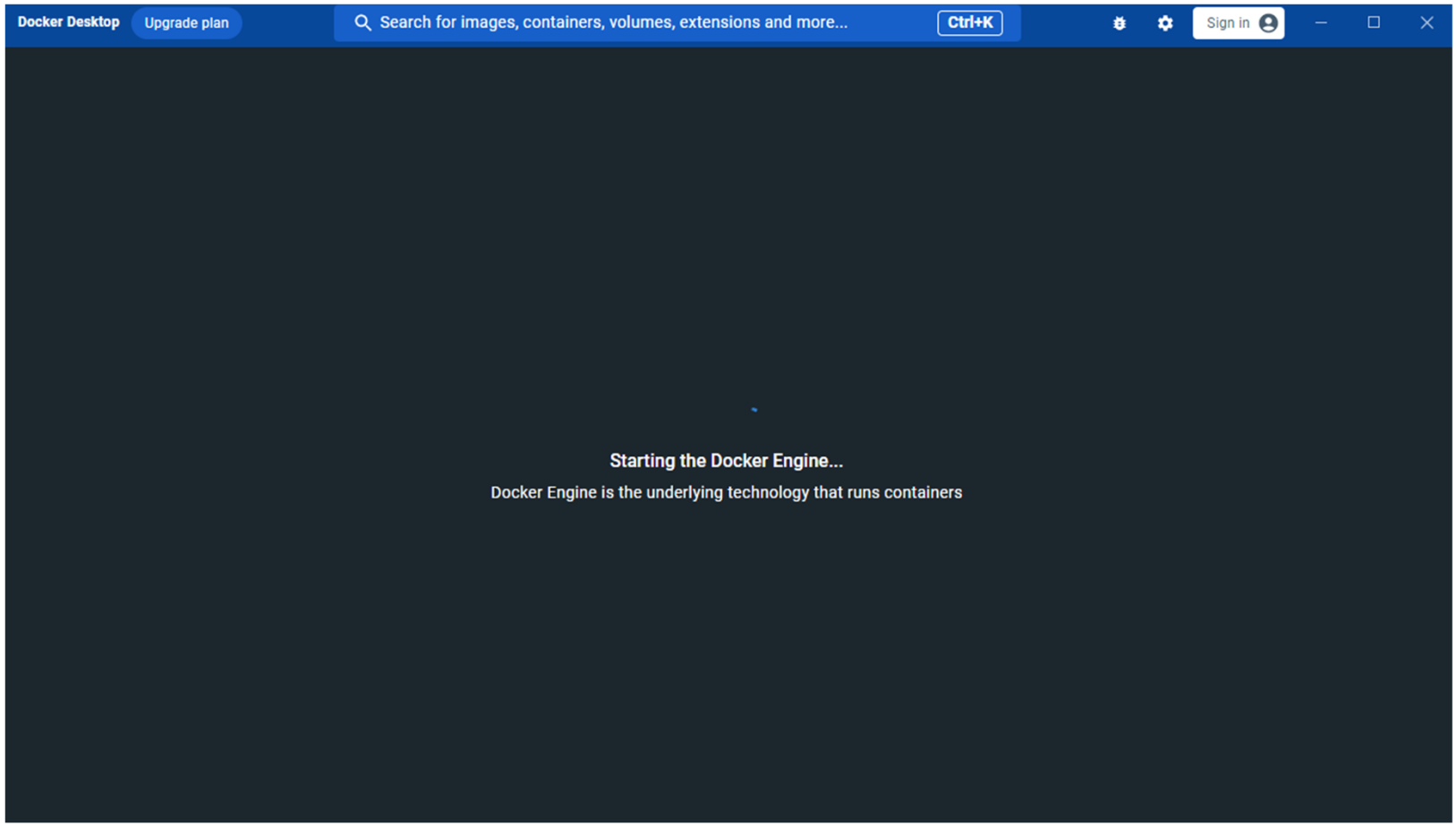
Instalado ¿?


Docker Desktop
Docker Desktop
Welcome to Docker Desktop
Sign in to connect to your Docker Desktop subscription or access online features.
Sign up
Already have an account? Sign in
Continue without signing in

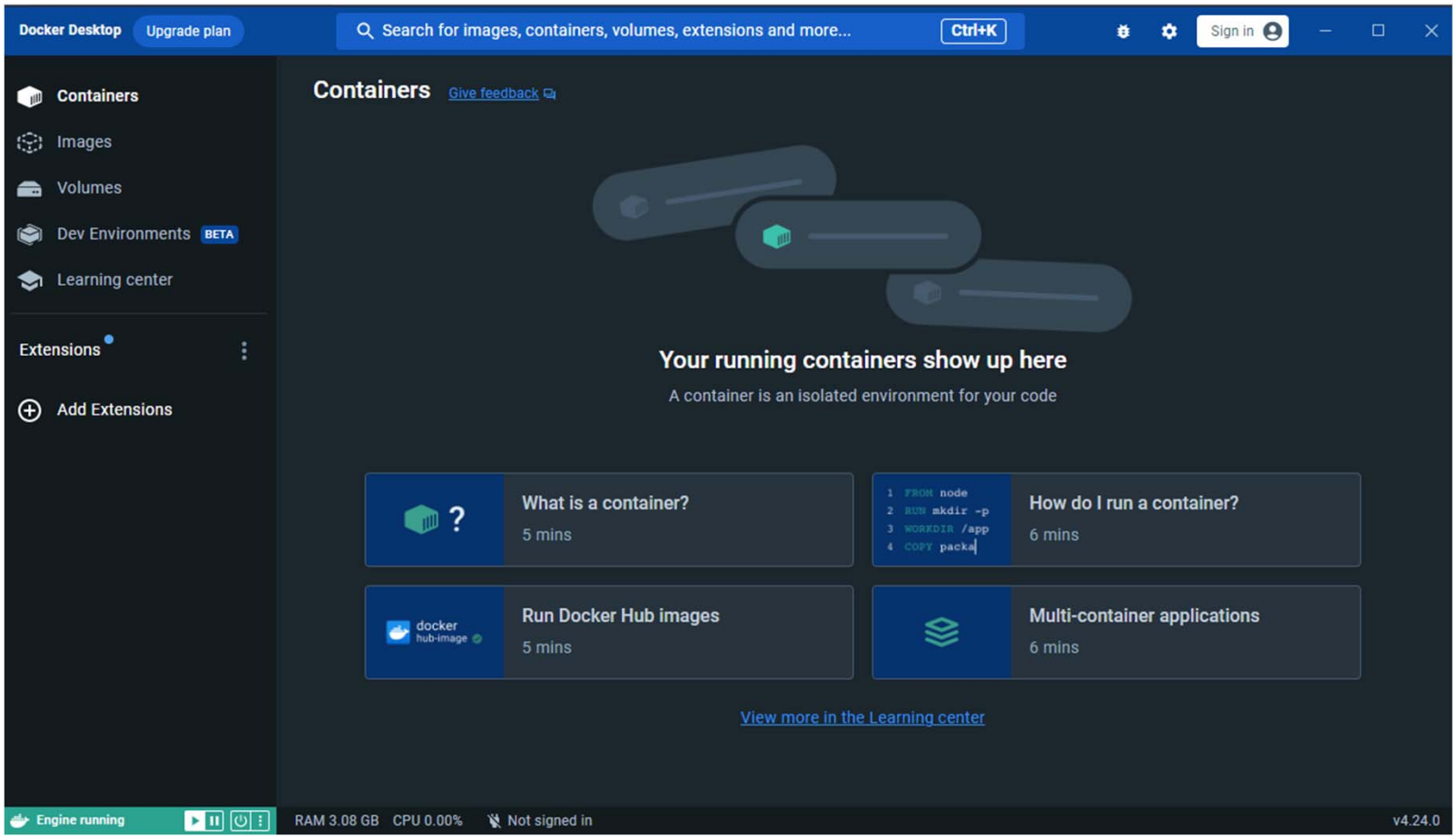
Más preguntas



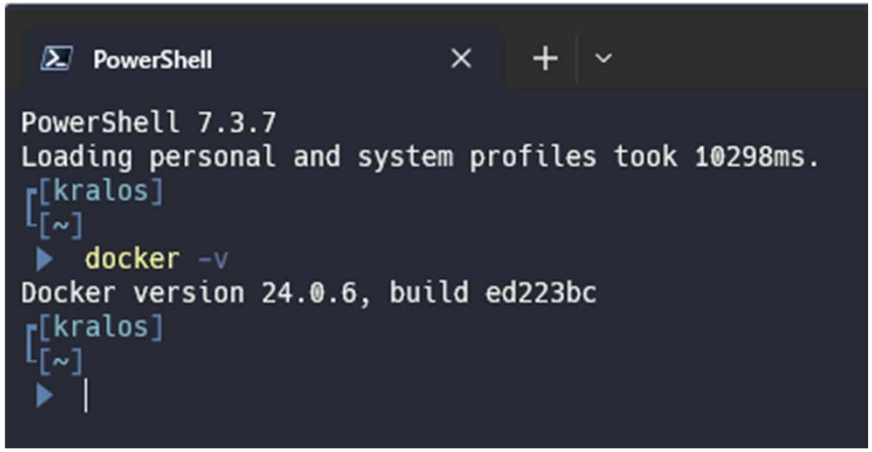
Ahora parece que ya inicio



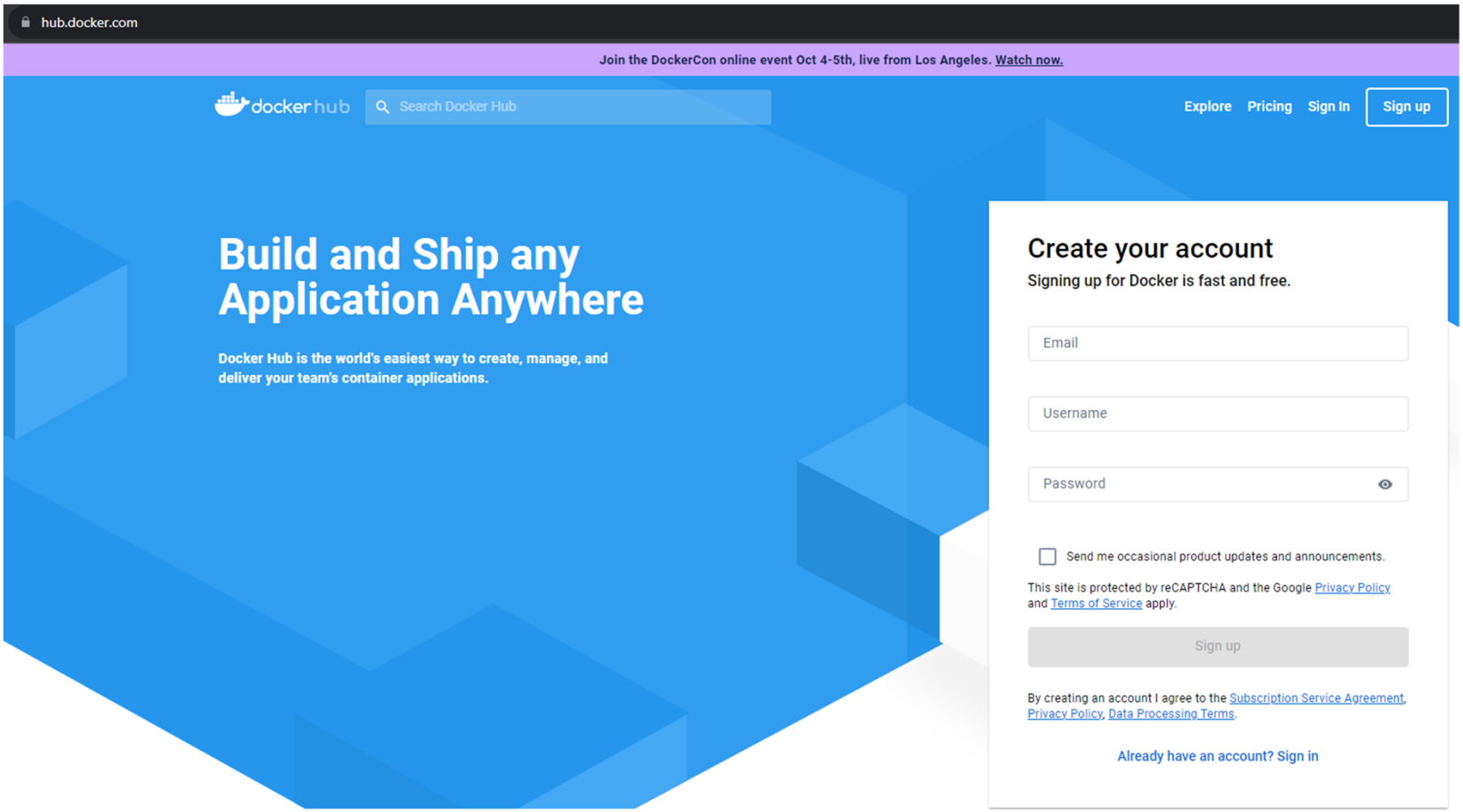
Según ya esta ejecutándose …



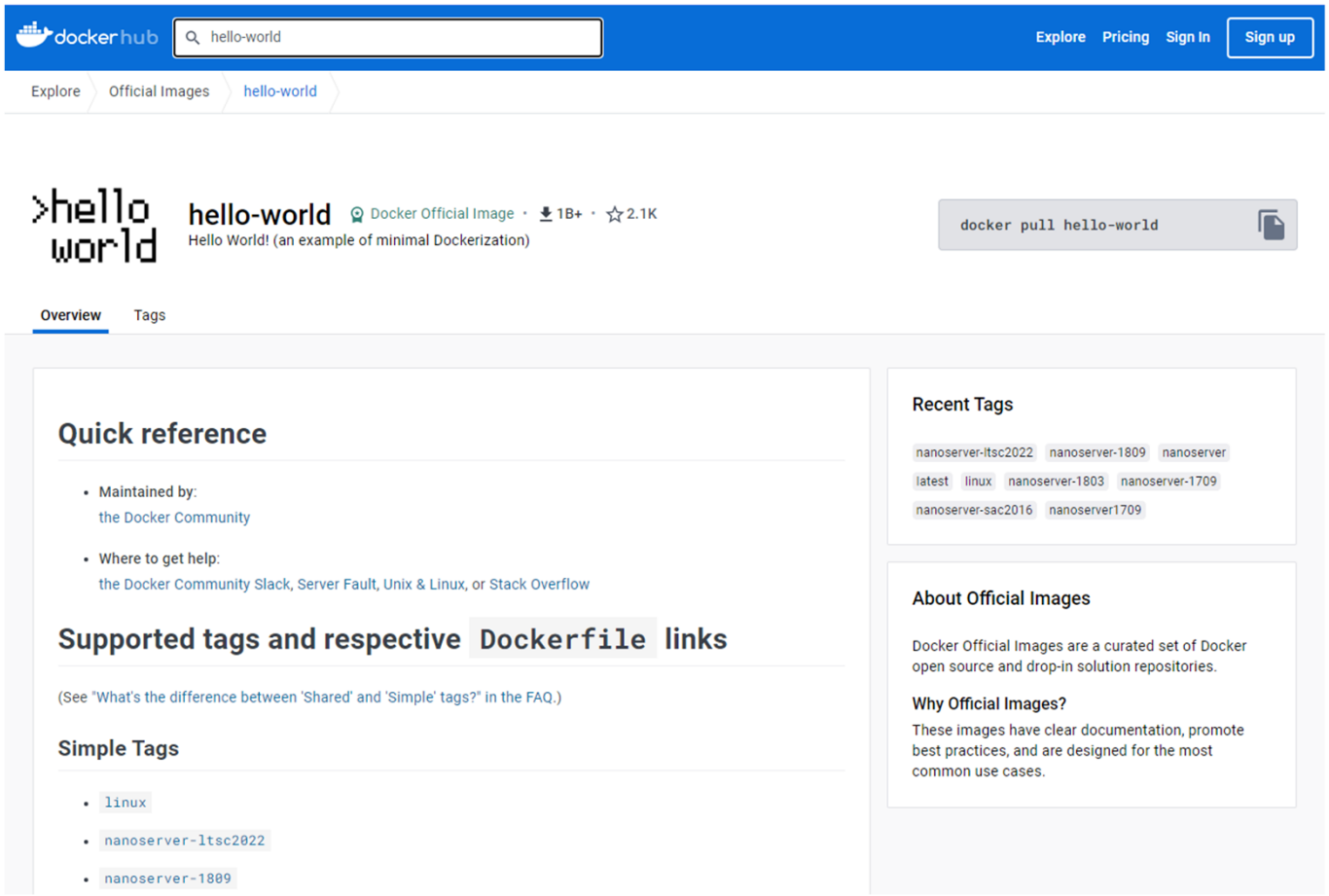
Verificación de que todo esta bien



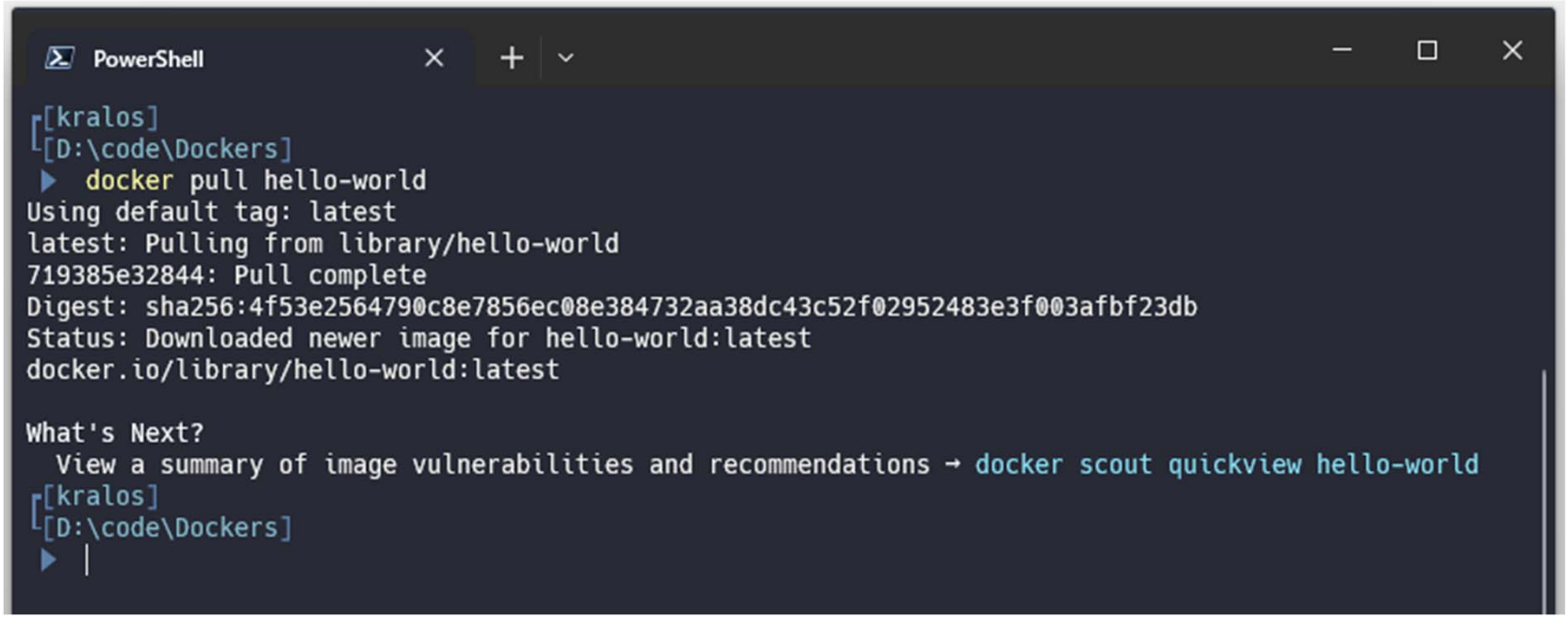
hub.Docker.com



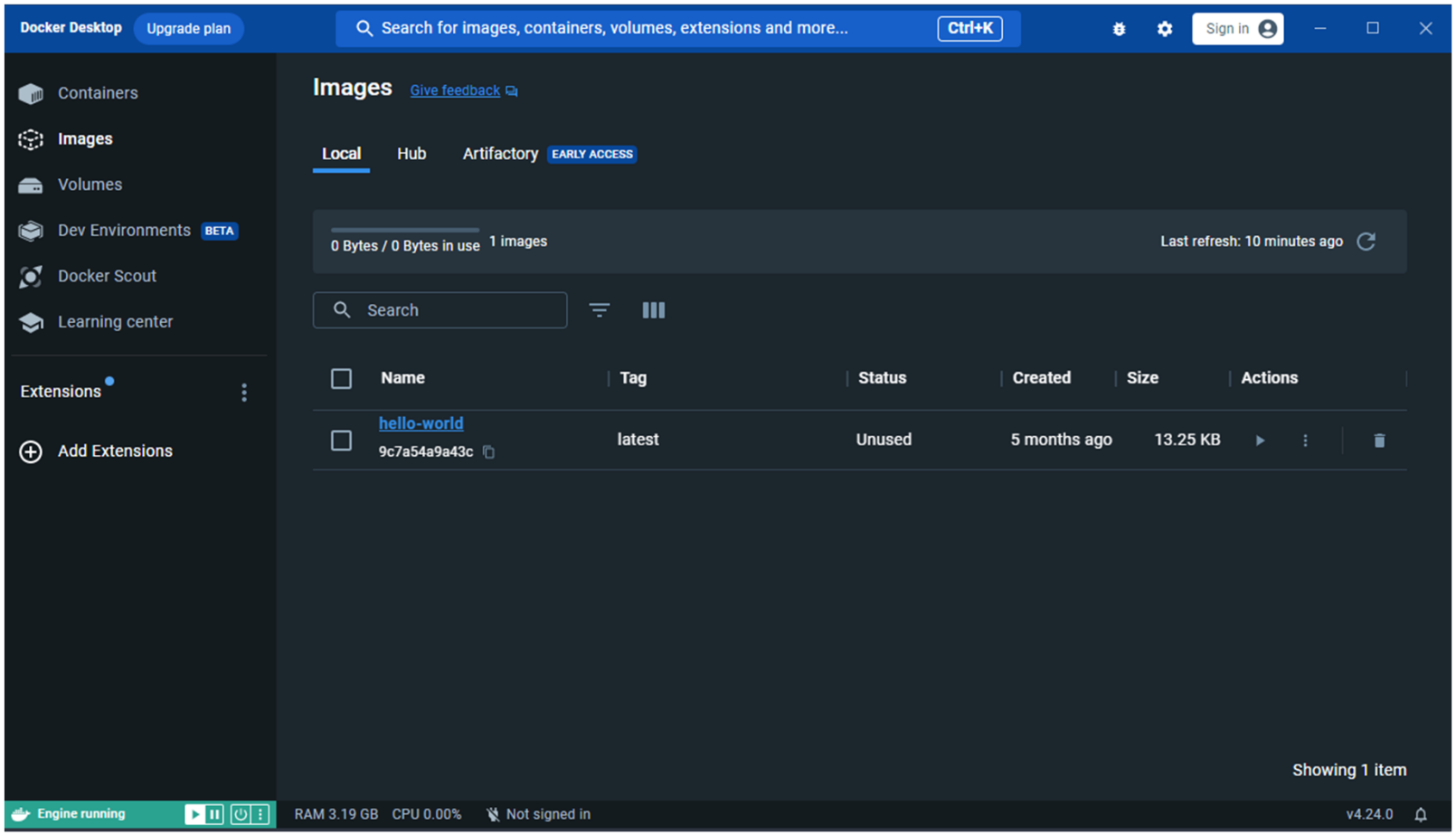
El hola mundo de docker



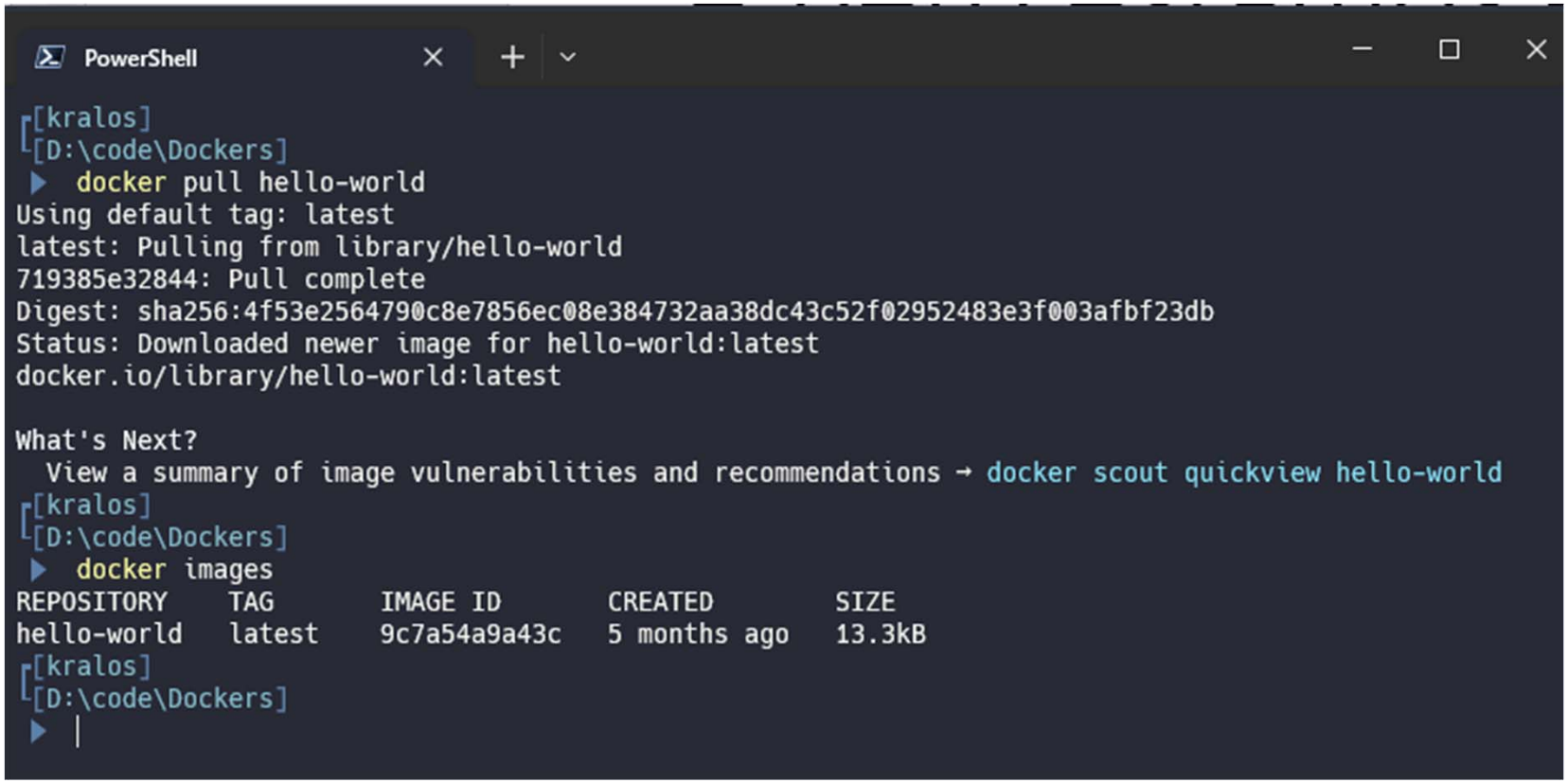
Ahora si, primeros pasos



Por si las flies …



A pero estamos en CLI …



Comandos para imágenes docker

Imagenes

`docker images`

Lista las imagenes almacenadas en el host

`docker search imagen`

Busca una imagen en el Docker hub

`docker inspect id/nombre`

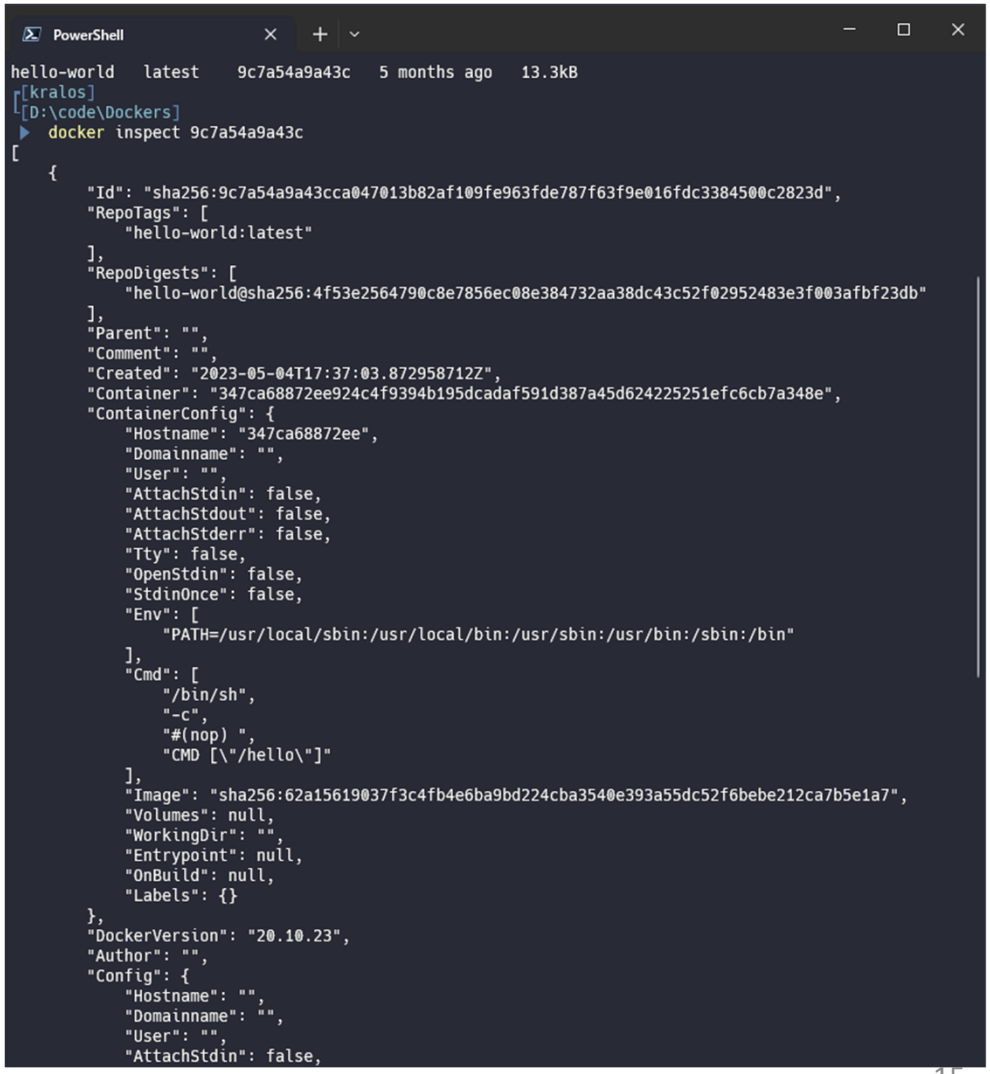
Muestra los detalles de una imagen a partir de un id o un nombre

`docker pull nombre`

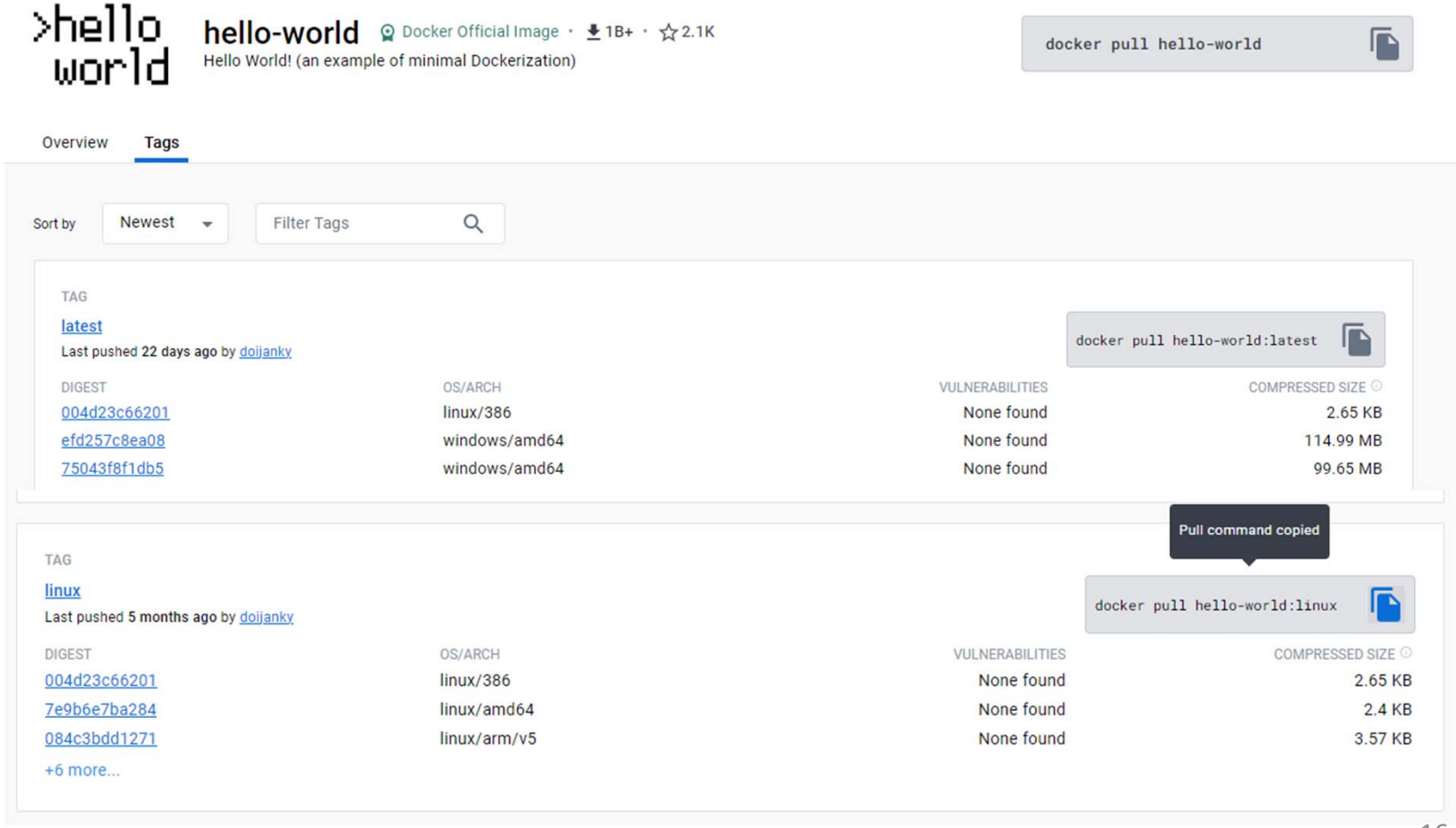
Descarga la imagen correspondiente a ese nombre

`docker rmi id/nombre`

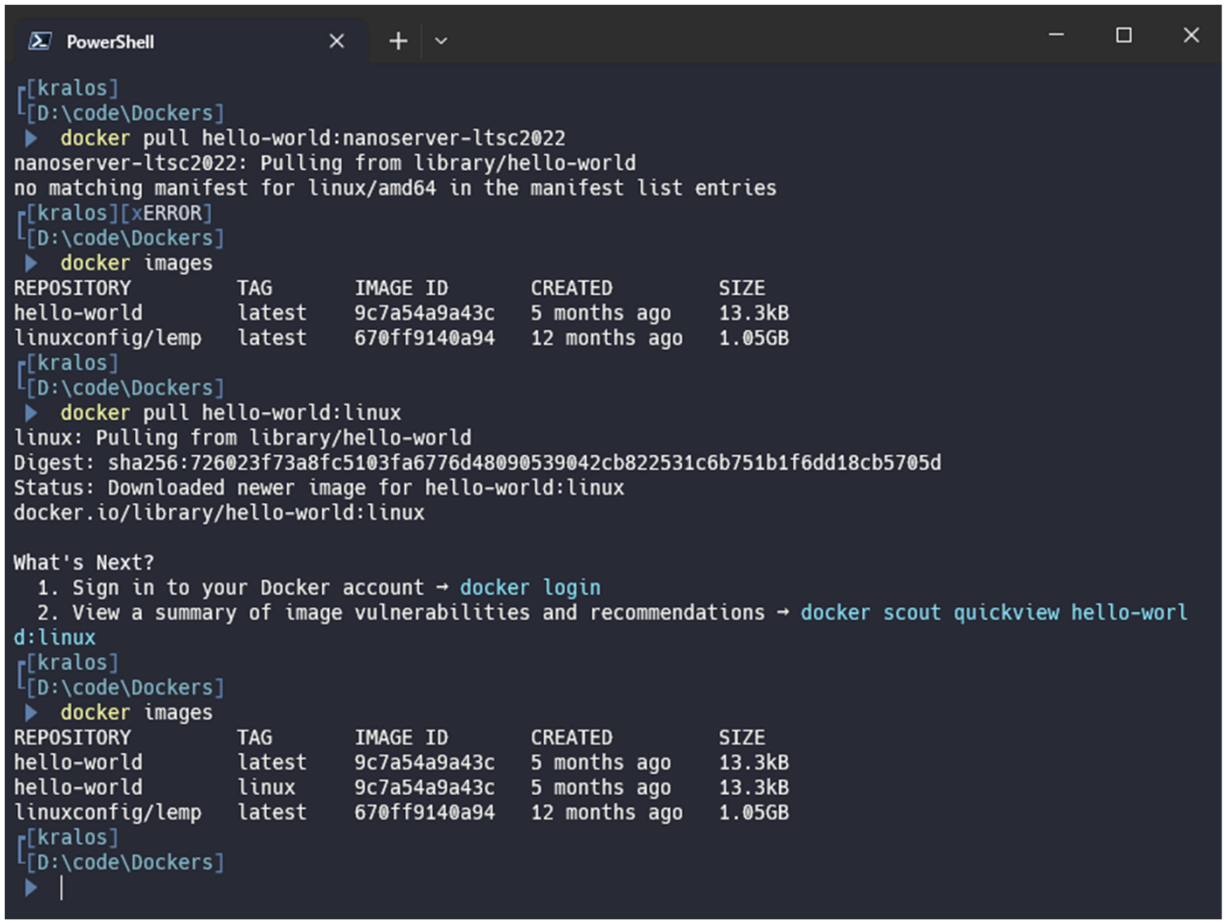
Elimina la imagen correspondiente a ese id o nombre



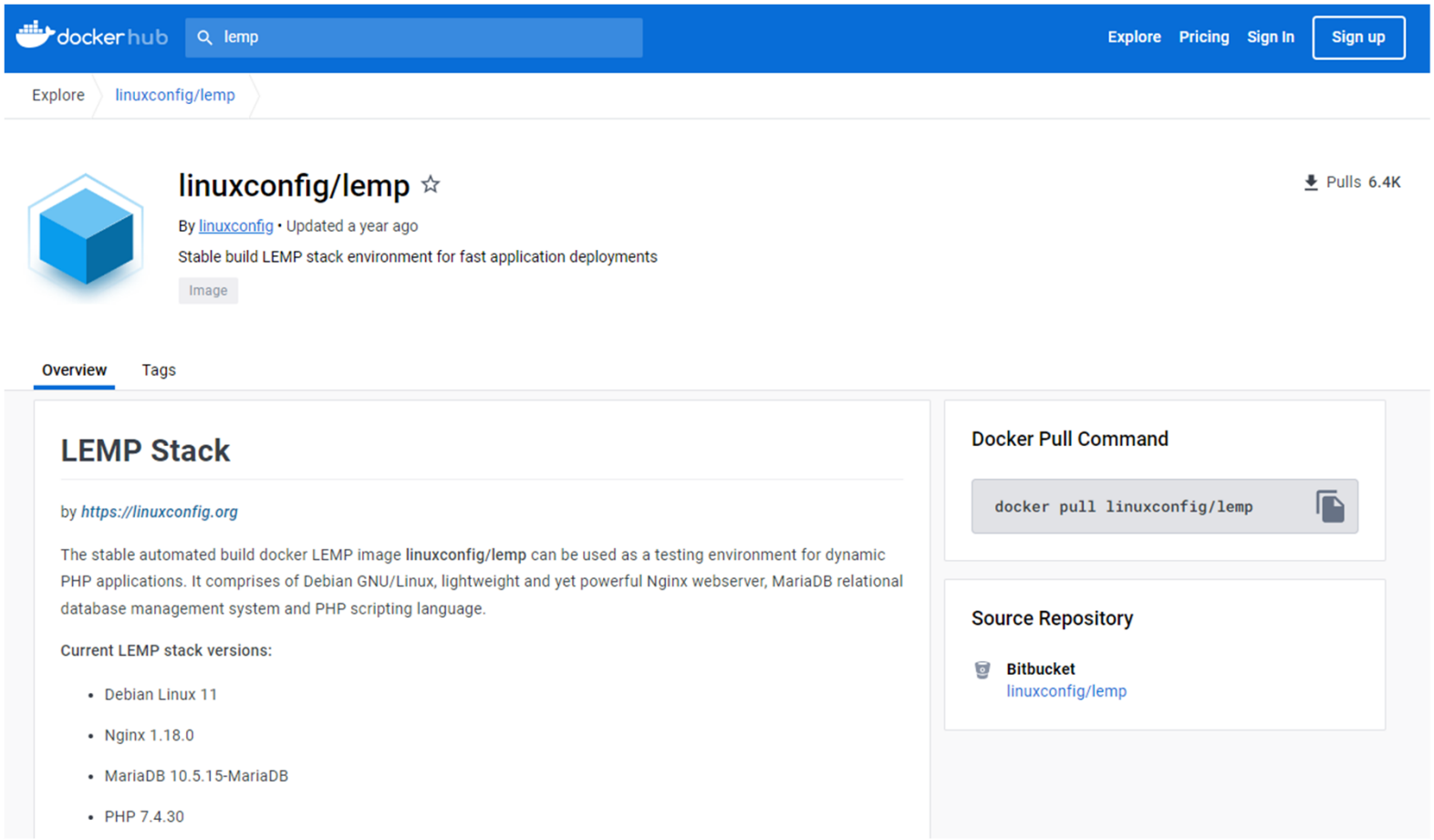
Pull especificando una etiqueta



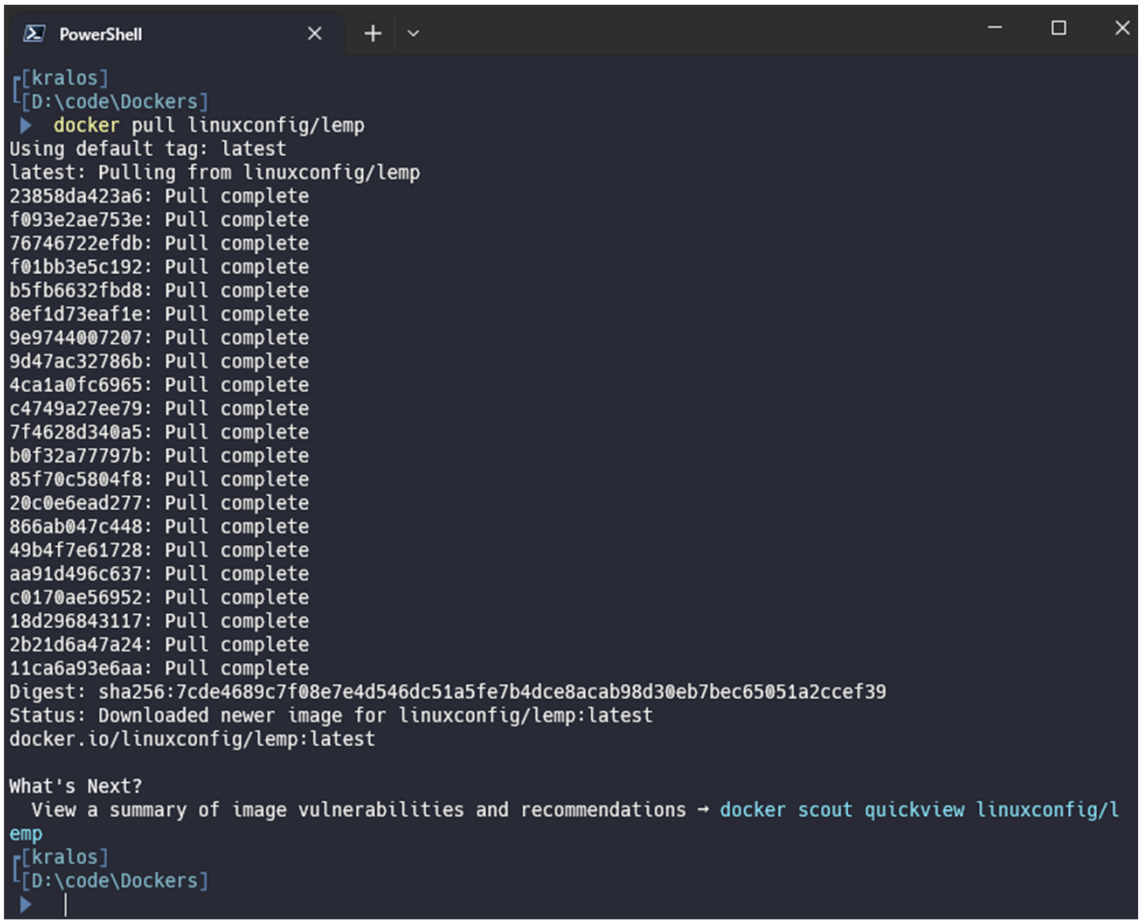
Otro hola mundo



A descargar imágenes



Se tardo creo … no vi :S



Comandos para contenedores

Contenedores

`docker create -it --name nombre imagen`
Crea un contenedor denominado name a partir de imagen

`docker start nombre`
Arranca el contenedor denominado nombre

`docker stop nombre`
Para el contenedor denominado nombre

`docker restart nombre`
Rearranca el contenedor denominado nombre

`docker rm nombre`
Elimina el contenedor denominado nombre

`docker ps -a`
Lista todos los contenedores en ejecución (-a incluye los parados)

`docker run -it imagen comando`
Arranca y ejecuta el comando en un contenedor de esa imagen en modo interactivo

`docker exec -it nombre comando`
Ejecuta el comando en el contenedor nombre

`docker run -d`
Arranca el contenedor en modo daemon

`docker run -P`
Arranca el contenedor y expone los puertos del contenedor en puertos aleatorios del host

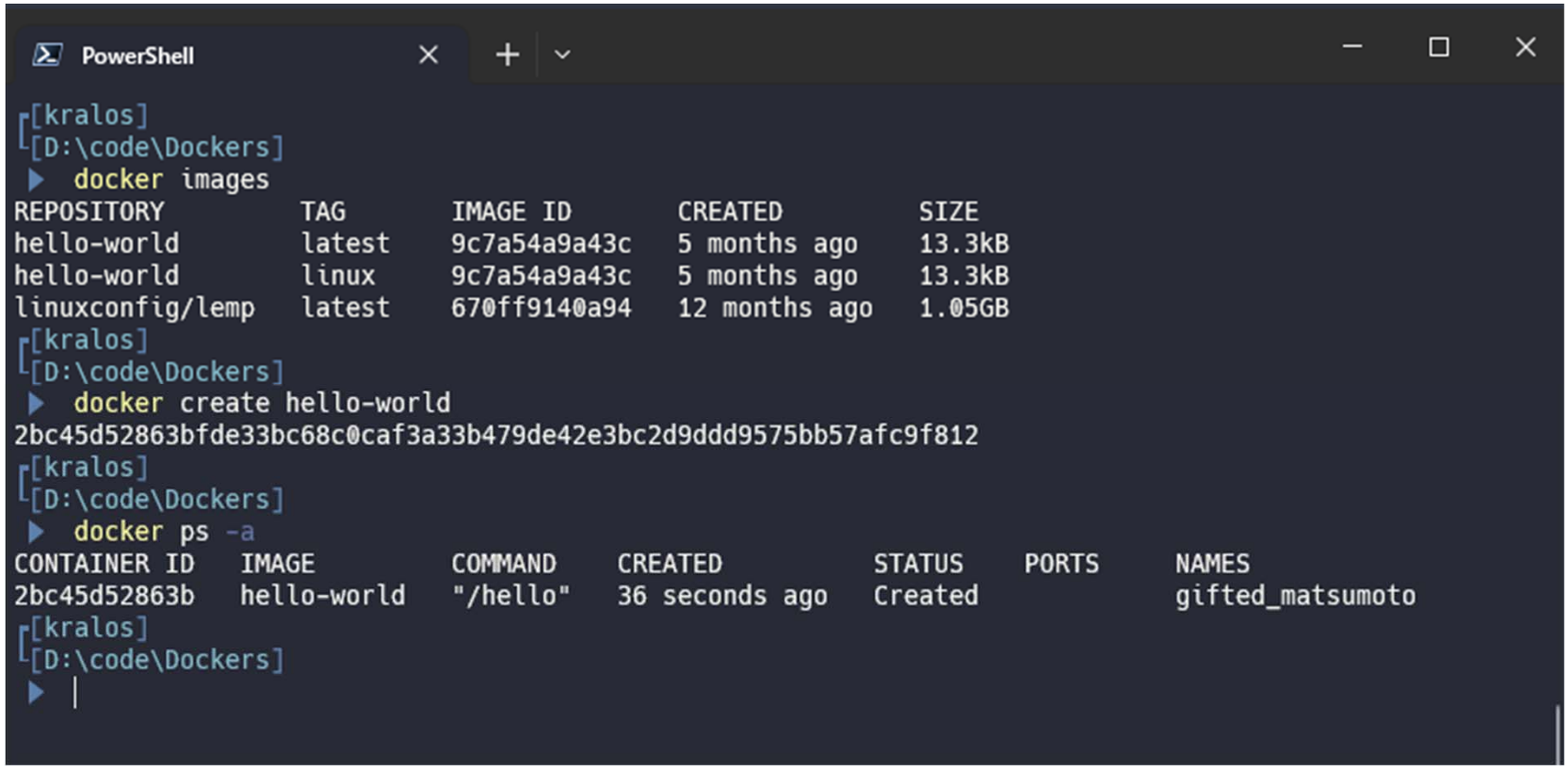
`docker run -p puerto_host:puerto_contenedor`
Arranca el contenedor y expone el puerto_contenedor en el puerto_host

`docker run -v directorio_host:directorio_contenedor`
Asigna el directorio_host para mapearlo en el directorio_contenedor

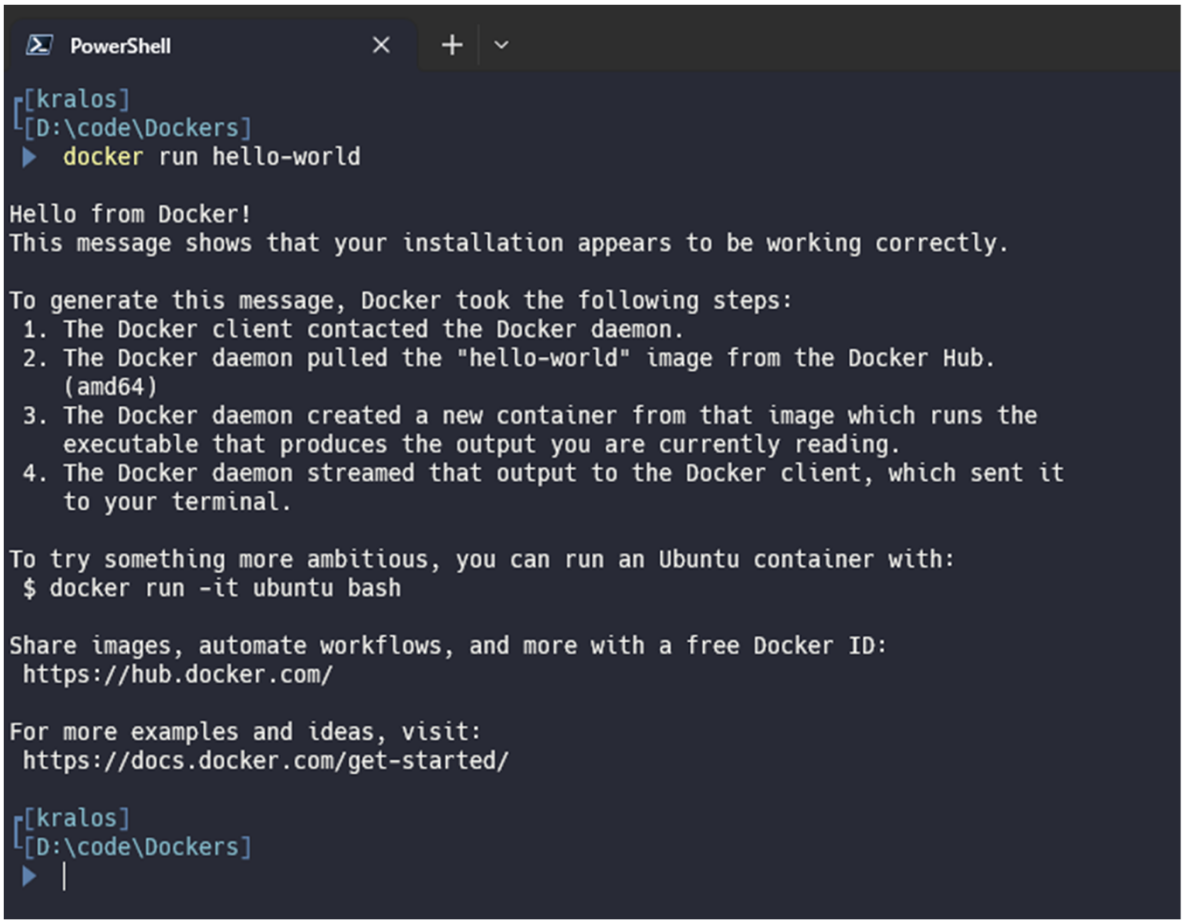
`docker run --hostname nombre_host`
Arranca el contenedor y asigna nombre_host al contenedor

`docker run --net=mired`
Arranca el contenedor y lo asigna en la red denominada mired

“Crear” un contenedor



Ejecutando un contenedor ..

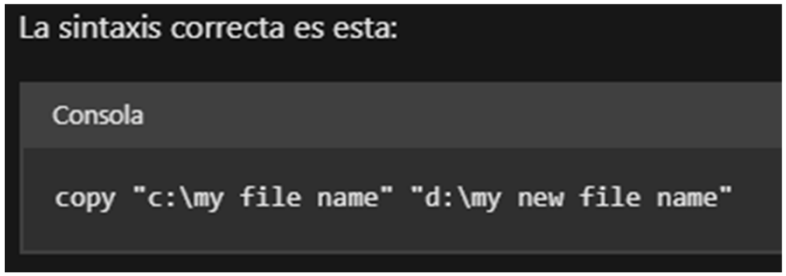


Cuidado con los espacios

- Use comillas al especificar nombres de archivo largos o rutas de acceso con espacios. Por ejemplo, escribir el comando

  copy c:\my file name d:\my new file name

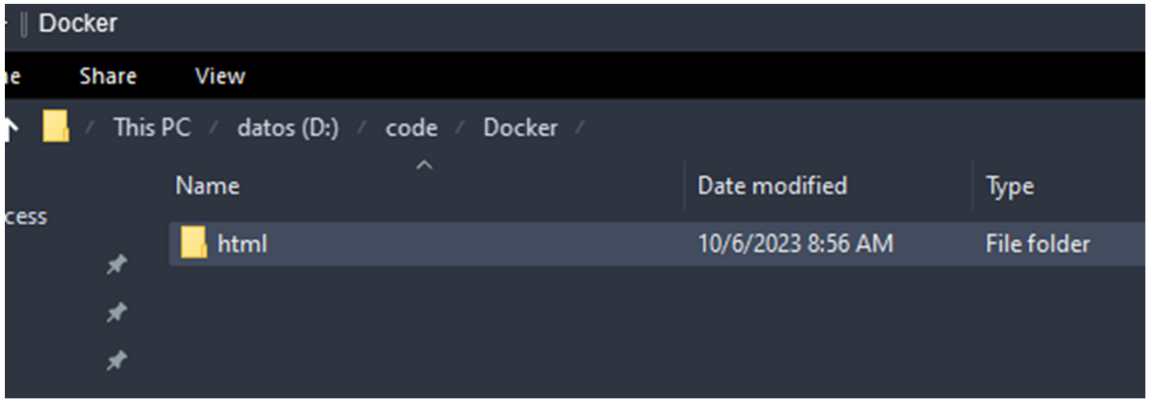
- El símbolo del sistema da como resultado el siguiente mensaje de error: El sistema no puede encontrar el archivo especificado.



Ejemplo de contenedor LEMP

1. Crear un directorio llamado “html” en SO anfitrión.

   D:\code\Docker\html

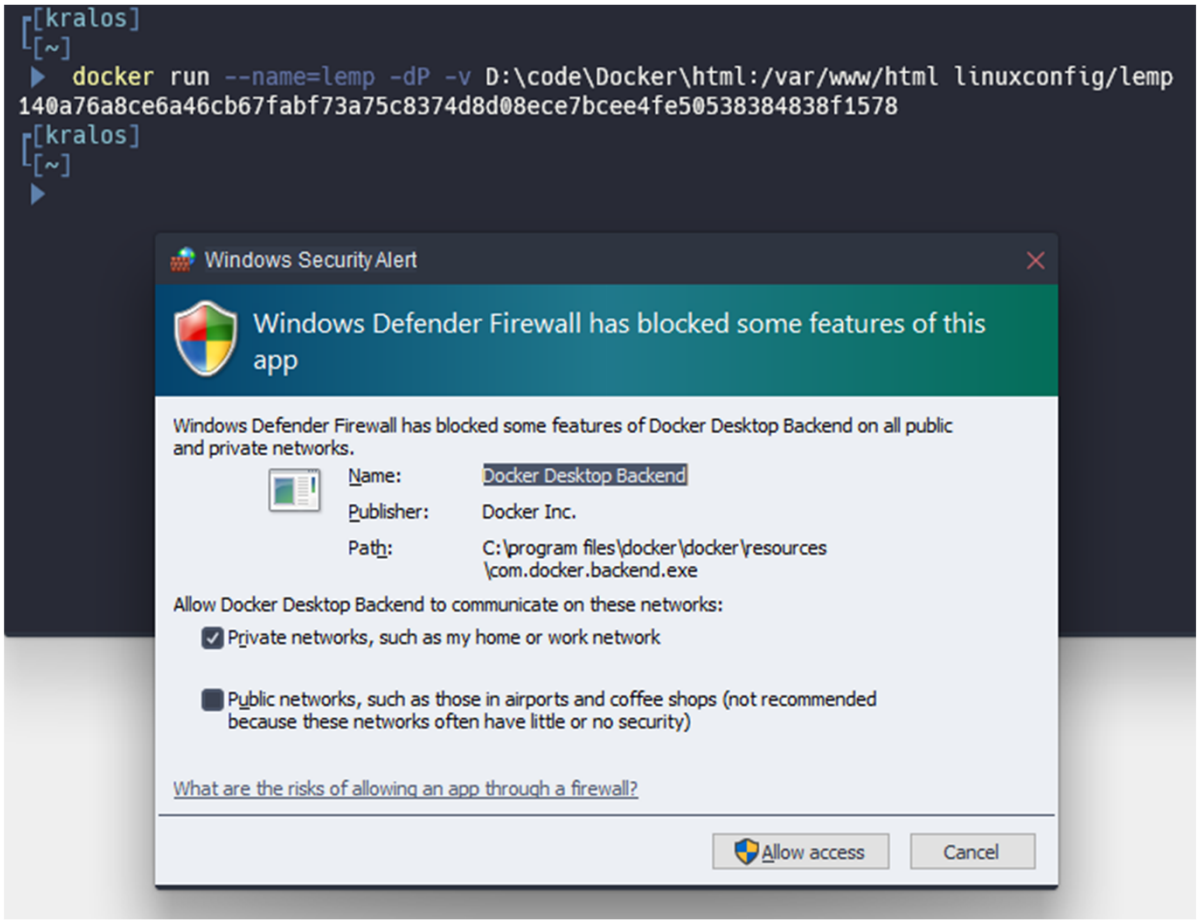


Ejemplo de contenedor LEMP

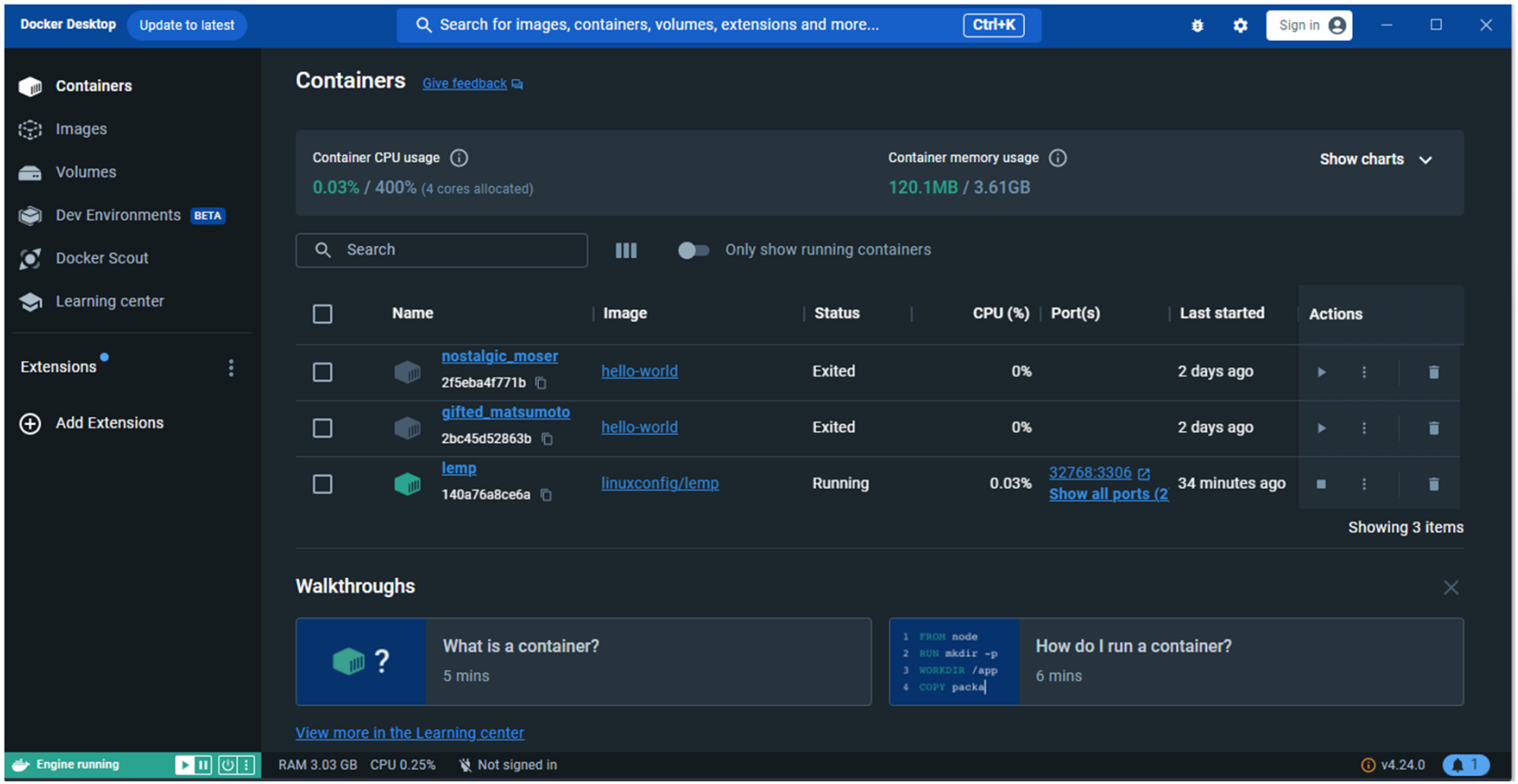
2. Crear y ejecutar un contendor, además se especifica el directorio referenciado entre la PC local y el contendor a crearse.

```
docker run --name=lemp -dP -v
D:\code\Docker\html:/var/www/html linuxconfig/lemp
```

Ejemplo de contenedor LEMP



Ejemplo de contenedor LEMP



Ejemplo de contenedor LEMP

3. Entramos al servidor web, pero antes necesitamos saber en que Puerto esta mapeado.

Vemos que hay dos puertos abiertos:

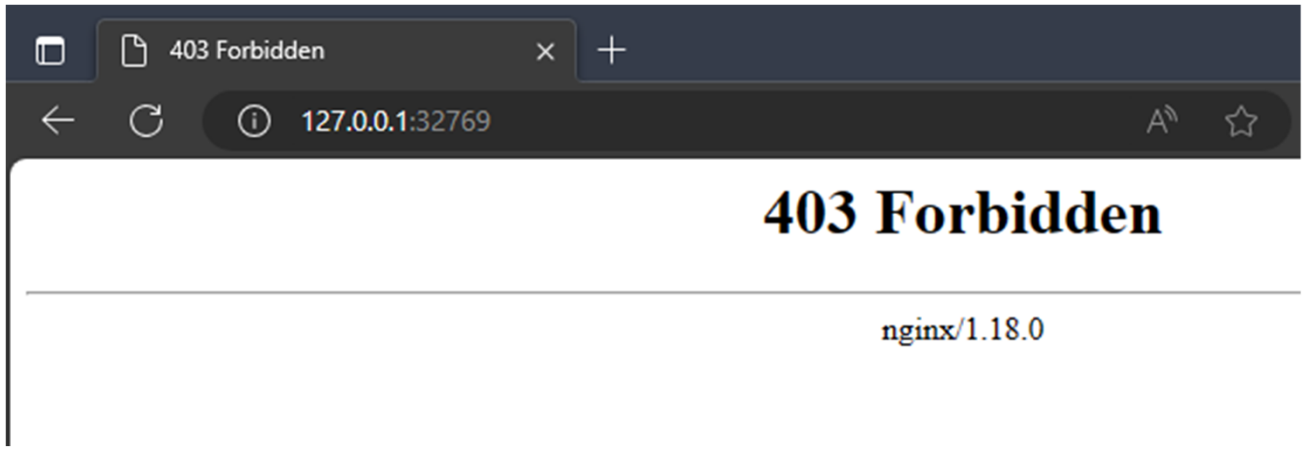
32269 para el servidor web

32768 para el servidor de base de dato

```
docker port lemp
80/tcp -> 0.0.0.0:32769
3306/tcp -> 0.0.0.0:32768
```

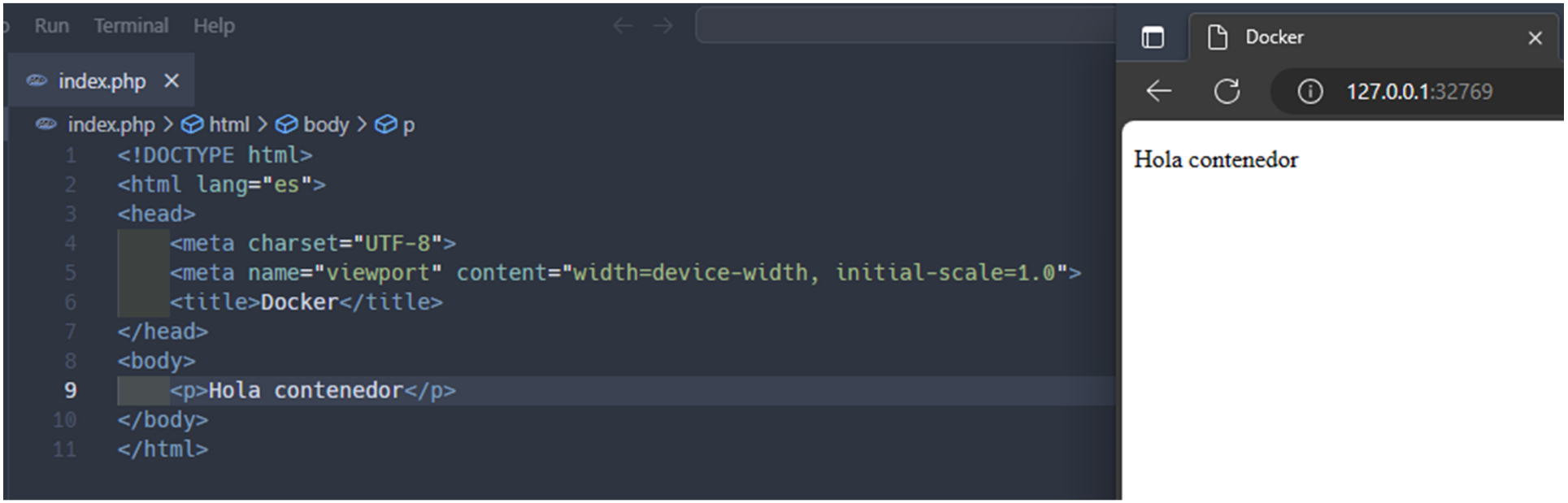
Ejemplo de contenedor LEMP

- El servidor web esta activo pero no tiene nada …



Ejemplo de contenedor LEMP

- El código se debe colocar el directorio “html” creado en el paso 1.
- Podemos usar nuestro editor favorito.

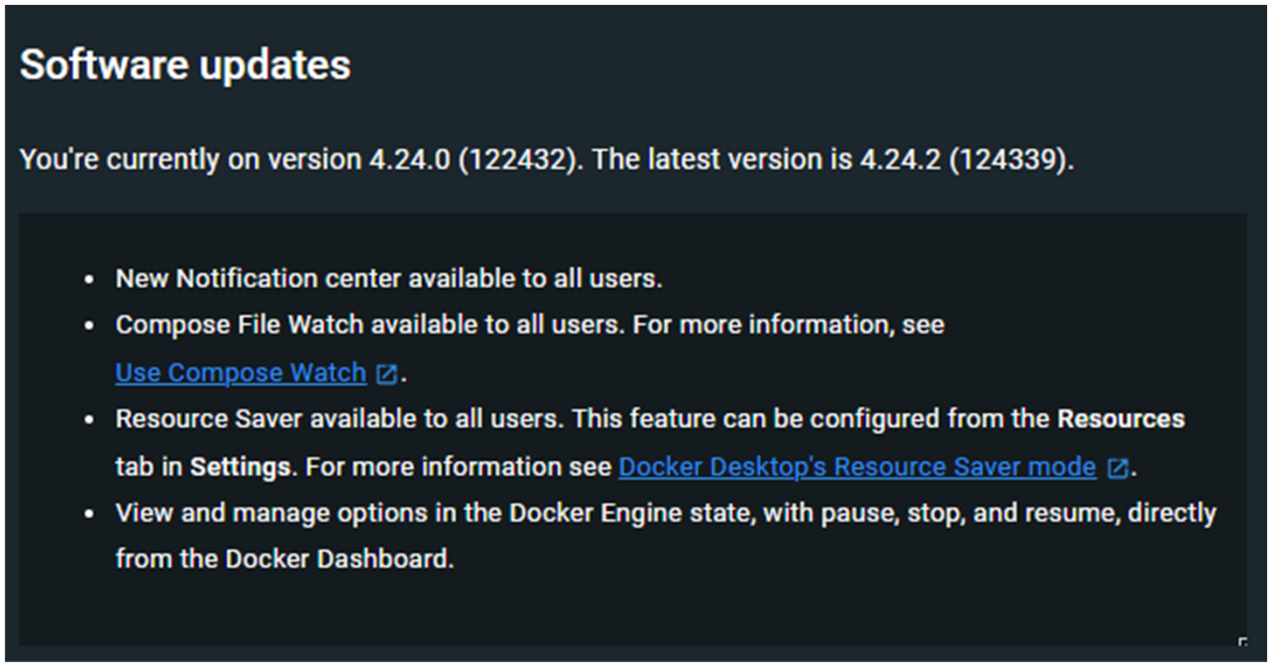


Ejemplo de contenedor LEMP

4. Ingresamos a la línea de comandos del contenedor

```
docker exec -it lemp /bin/bash
root@140a76a8ce6a:/# ls
bin  boot  dev  etc  home  lib  lib64  media  mnt  opt  proc  root  run  sbin
root@140a76a8ce6a:/# pwd
/
root@140a76a8ce6a:/# whoami
root
root@140a76a8ce6a:/# ls /var/www/html/
index.php
root@140a76a8ce6a:/# cat /var/www/html/index.php
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Docker</title>
</head>
<body>
    <p>Hola contenedor</p>
</body>
</html>root@140a76a8ce6a:/#
```

Actualizaciones



Un poco de detalle

```
docker run --name=lemp -dP -v D:\code\Docker\html:/var/www/html linuxconfig/lemp
```

--name=*nombre_de_mi contenedor*

-dP

d hace que el contenedor corra en segundo plano (*daemonizará* el contenedor)

P para decirle a Docker que elija un puerto

-v Cuando queremos montar un volúmen en un contenedor ruta_origen**:**ruta_destino

“linuxconfig/lemp” es el nombre de la imagen a usar

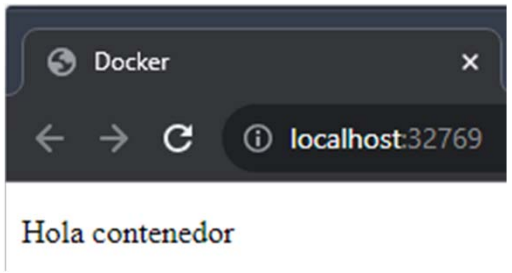
Mapear puertos

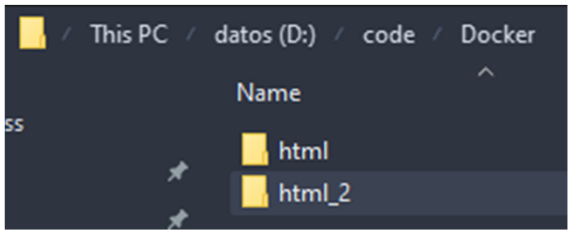
-p puerto_PC_anfitrión: puerto_contendor

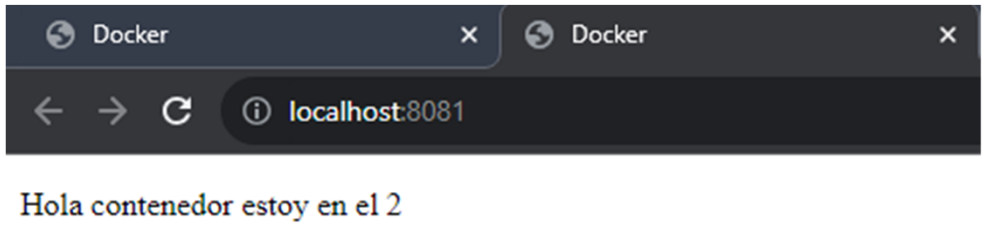
El argumento -p 1234:1234 le indica a Docker que tiene que hacer un port forwarding del contenedor hacia el puerto 1234 de la máquina host.

Para mas información ejecutar en su terminal el comando siguiente: docker run --help

Practica 2







docker run --name=lemp2 –d –p 8081:80 -v D:\code\Docker\html_2:/var/www/html linuxconfig/lemp

```
PowerShell
┌[kralos]
└[~]
▶ docker run --name=lemp2 -d -p 8081:80 -v D:\code\Docker\html_2:/var/www/html linuxconfig/lemp
f3c176d990da9f34e4177e0a1556500e4ea3900e4928bdf3897e29b915c6c31c
┌[kralos]
└[~]
▶ docker ps -a
CONTAINER ID   IMAGE              COMMAND         CREATED         STATUS                     PORTS                                              NAMES
f3c176d990da   linuxconfig/lemp   "supervisord"   8 seconds ago   Up 4 seconds               3306/tcp, 0.0.0.0:8081->80/tcp                     lemp2
140a76a8ce6a   linuxconfig/lemp   "supervisord"   12 days ago     Up 2 minutes               0.0.0.0:32769->80/tcp, 0.0.0.0:32768->3306/tcp     lemp
2f5eba4f771b   hello-world        "/hello"        13 days ago     Exited (0) 13 days ago                                                        nostalgic_moser
2bc45d52863b   hello-world        "/hello"        13 days ago     Exited (0) 13 days ago                                                        gifted_matsumoto
┌[kralos]
└[~]
▶
```

Comportamiento de los contenedores

docker logs lemp

```
┌[kralos]
└[~]
▶ docker logs lemp
/usr/lib/python3/dist-packages/supervisor/options.py:474: UserWarning: Supervisord is running as root and it is searching for its configuration file in default loca
tions (including its current working directory); you probably want to specify a "-c" argument specifying an absolute path to a configuration file for improved secur
ity.
  self.warnings.warn(
2023-10-06 16:45:54,397 CRIT Supervisor is running as root.  Privileges were not dropped because no user is specified in the config file.  If you intend to run as r
oot, you can set user=root in the config file to avoid this message.
2023-10-06 16:45:54,399 INFO Included extra file "/etc/supervisor/conf.d/supervisor-lemp.conf" during parsing
2023-10-06 16:45:54,421 INFO RPC interface 'supervisor' initialized
2023-10-06 16:45:54,422 CRIT Server 'unix_http_server' running without any HTTP authentication checking
```

```
▶ docker logs lemp2
/usr/lib/python3/dist-packages/supervisor/options.py:474: UserWarning: Supervisord is running as root and it is searching for its configuration file in default loca
tions (including its current working directory); you probably want to specify a "-c" argument specifying an absolute path to a configuration file for improved secur
ity.
  self.warnings.warn(
2023-10-18 18:20:06,537 CRIT Supervisor is running as root.  Privileges were not dropped because no user is specified in the config file.  If you intend to run as r
oot, you can set user=root in the config file to avoid this message.
2023-10-18 18:20:06,537 INFO Included extra file "/etc/supervisor/conf.d/supervisor-lemp.conf" during parsing
2023-10-18 18:20:06,540 INFO RPC interface 'supervisor' initialized
2023-10-18 18:20:06,543 CRIT Server 'unix_http_server' running without any HTTP authentication checking
2023-10-18 18:20:06,543 INFO supervisord started with pid 1
2023-10-18 18:20:07,548 INFO spawned: 'mysqld' with pid 7
2023-10-18 18:20:07,550 INFO spawned: 'nginx' with pid 8
2023-10-18 18:20:07,551 INFO spawned: 'php-fcgi' with pid 9
2023-10-18 18:20:08,693 INFO success: mysqld entered RUNNING state, process has stayed up for > than 1 seconds (startsecs)
2023-10-18 18:20:08,693 INFO success: nginx entered RUNNING state, process has stayed up for > than 1 seconds (startsecs)
2023-10-18 18:20:08,693 INFO success: php-fcgi entered RUNNING state, process has stayed up for > than 1 seconds (startsecs)
┌[kralos]
└[~]
▶
```

Variables de entorno

Las variables de entorno se usan para definir valores que están disponibles para los procesos que se ejecutan dentro del contenedor. Estas variables se definen al construir una imagen o al ejecutar un contenedor.

-e variable_entorno**:**valor_nuestro

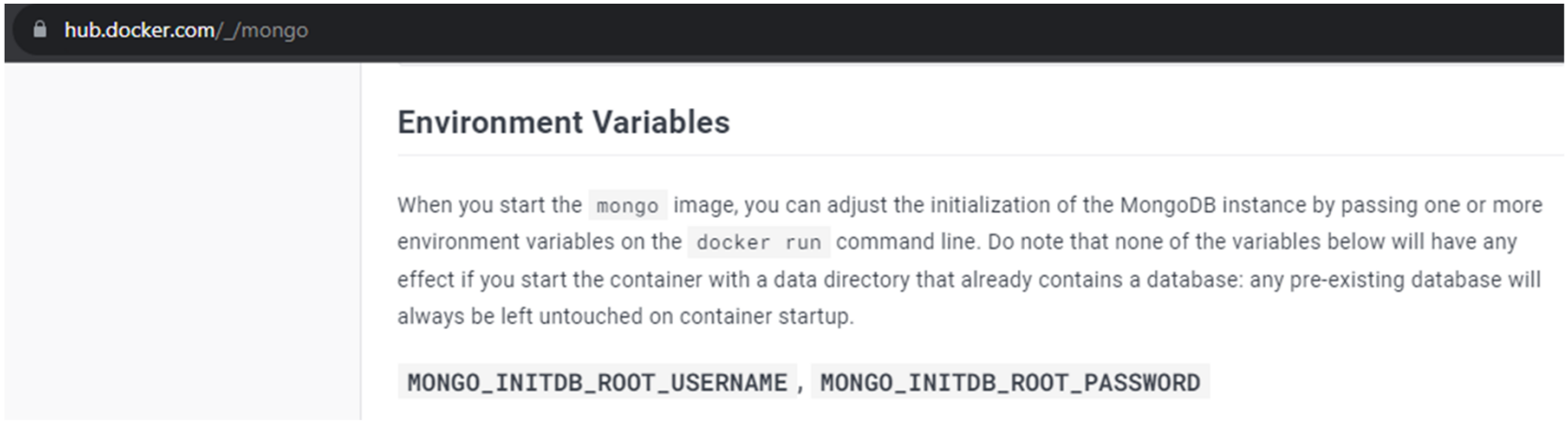
es para asignar valores a las variables de entorno. Esas variables están definidas dentro de la imagen, por lo que nosotros le damos valor, para que el contenedor a ejecutar, use esos datos.

Usaremos la imagen de mongo

- docker pull mongo
- docker run --name=mongo_server –d –p 8082:27017 mongo

```
┌[kralos]
└[~]
 ▶  docker pull mongo
Using default tag: latest
latest: Pulling from library/mongo
43f89b94cd7d: Pull complete
54a7480baa9d: Pull complete
7f9301fbd7df: Pull complete
5e4470f2e90f: Pull complete
40d046ff8fd3: Pull complete
e062d62b861e: Pull complete
72919e34fde8: Pull complete
ab22810dfc64: Pull complete
fb05c29fbdf5: Pull complete
Digest: sha256:d341a86584b96eb665345a8f5b35fba8695ee1d0618fd012ec4696223a3d6c62
Status: Downloaded newer image for mongo:latest
docker.io/library/mongo:latest
```

Practica 3

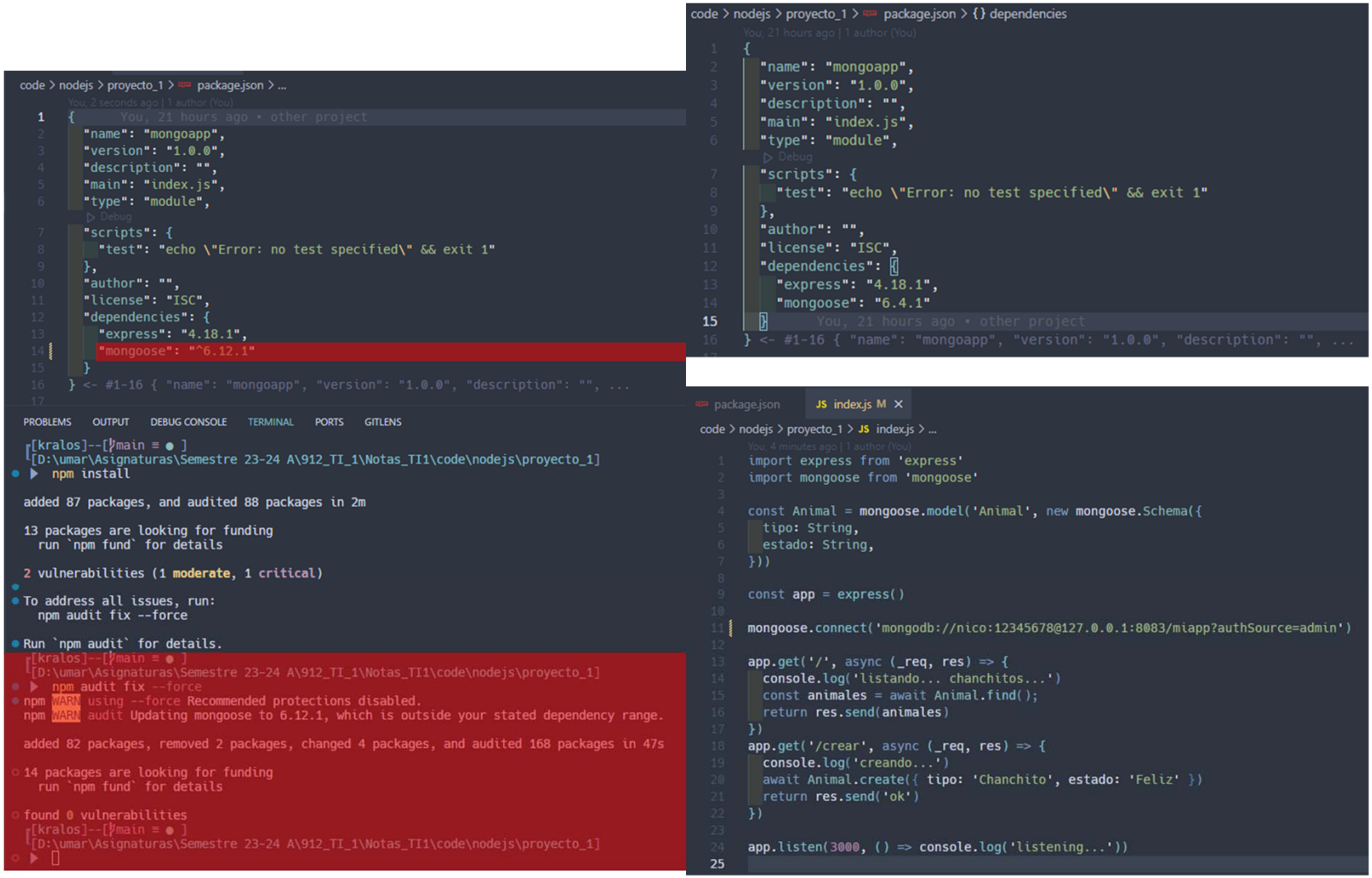



```
docker run --name=monguito –d –p 8083:27017
–e MONGO_INITDB_ROOT_USERNAME=nico
–e MONGO_INITDB_ROOT_PASSWORD=12345678 mongo
```

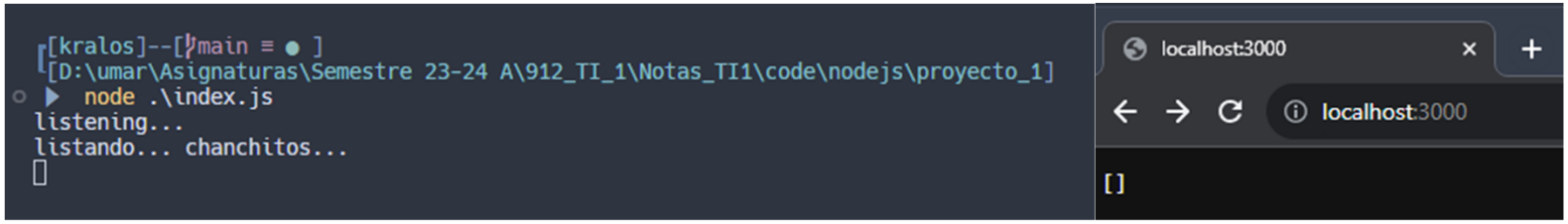
Practica 3

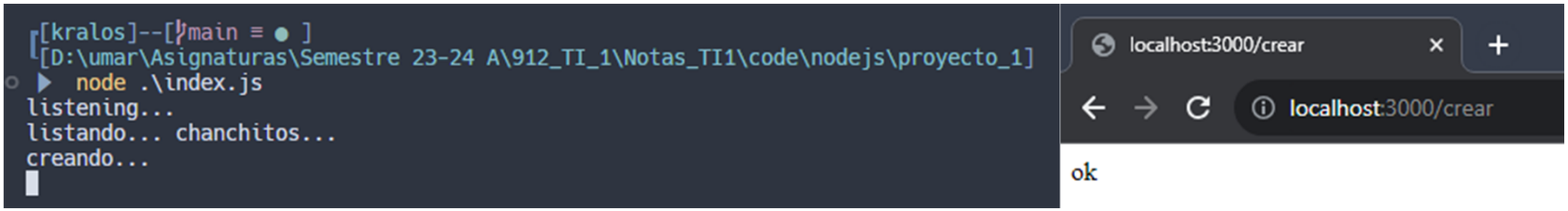
```
┌[kralos]--[ main ≡ ● ]
└[D:\umar\Asignaturas\Semestre 23-24 A\912_TI_1\Notas_TI1\code\nodejs\proyecto_1]
● ▶  docker ps -a
CONTAINER ID   IMAGE              COMMAND         CREATED        STATUS                      PORTS     NAMES
f3c176d990da   linuxconfig/lemp   "supervisord"   23 hours ago   Exited (137) 23 hours ago             lemp2
140a76a8ce6a   linuxconfig/lemp   "supervisord"   13 days ago    Exited (137) 23 hours ago             lemp
2f5eba4f771b   hello-world        "/hello"        2 weeks ago    Exited (0) 2 weeks ago                nostalgic_moser
2bc45d52863b   hello-world        "/hello"        2 weeks ago    Exited (0) 2 weeks ago                gifted_matsumoto
┌[kralos]--[ main ≡ ● ]
└[D:\umar\Asignaturas\Semestre 23-24 A\912_TI_1\Notas_TI1\code\nodejs\proyecto_1]
● ▶  docker ps
● CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
┌[kralos]--[ main ≡ ● ]
└[D:\umar\Asignaturas\Semestre 23-24 A\912_TI_1\Notas_TI1\code\nodejs\proyecto_1]
● ▶  docker run --name=monguito -d -p 8083:27017 -e MONGO_INITDB_ROOT_USERNAME=nico -e MONGO_INITDB_ROOT_PASSWORD=12345678 mongo
29293400974ad156801a24f8cc0da3ff3158116250386c3bc7d692daa3cb9e5a
┌[kralos]--[ main ≡ ● ]
└[D:\umar\Asignaturas\Semestre 23-24 A\912_TI_1\Notas_TI1\code\nodejs\proyecto_1]
● ▶  docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED              STATUS              PORTS                     NAMES
29293400974a   mongo     "docker-entrypoint.s…"   About a minute ago   Up About a minute   0.0.0.0:8083->27017/tcp   monguito
┌[kralos]--[ main ≡ ● ]
└[D:\umar\Asignaturas\Semestre 23-24 A\912_TI_1\Notas_TI1\code\nodejs\proyecto_1]
○ ▶
```

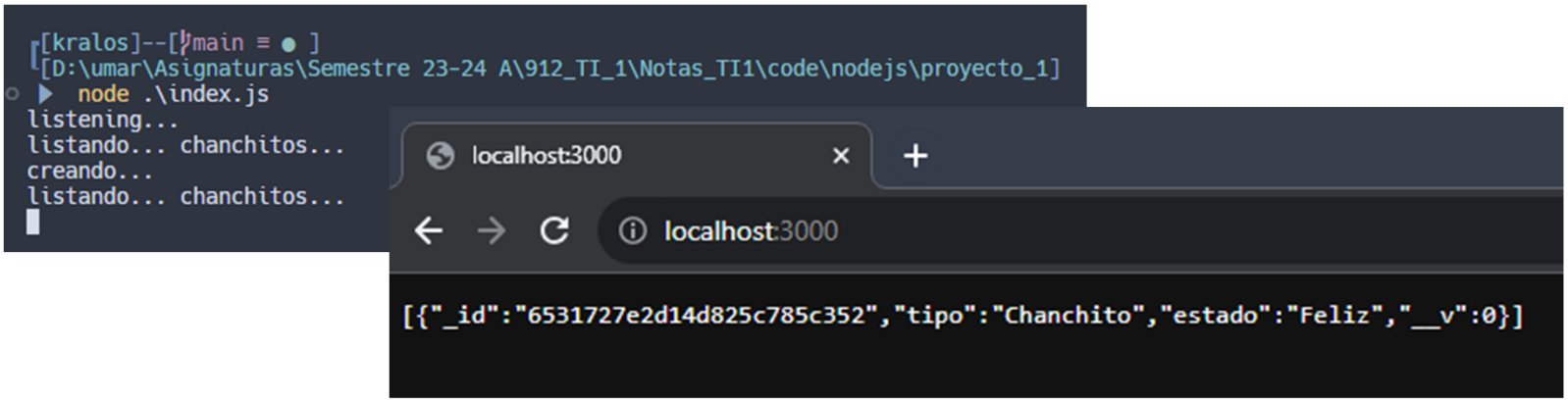
Practica 3


code > nodejs > proyecto_1 > package.json > ...
{
"name": "mongoapp",
"version": "1.0.0",
"description": "",
"main": "index.js",
"type": "module",
"scripts": {
"test": "echo \"Error: no test specified\" && exit 1"
},
"author": "",
"license": "ISC",
"dependencies": {
"express": "4.18.1",
"mongoose": "^6.12.1"
}
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS GITLENS
[kralos]--[main ≡ ●]
[D:\umar\Asignaturas\Semestre 23-24 A\912_TI_1\Notas_TI1\code\nodejs\proyecto_1]
npm install
added 87 packages, and audited 88 packages in 2m
13 packages are looking for funding
run `npm fund` for details
2 vulnerabilities (1 moderate, 1 critical)
To address all issues, run:
npm audit fix --force
Run `npm audit` for details.
[kralos]--[main ≡ ●]
[D:\umar\Asignaturas\Semestre 23-24 A\912_TI_1\Notas_TI1\code\nodejs\proyecto_1]
npm audit fix --force
npm WARN using --force Recommended protections disabled.
npm WARN audit Updating mongoose to 6.12.1, which is outside your stated dependency range.
added 82 packages, removed 2 packages, changed 4 packages, and audited 168 packages in 47s
14 packages are looking for funding
run `npm fund` for details
found 0 vulnerabilities
[kralos]--[main ≡ ●]
[D:\umar\Asignaturas\Semestre 23-24 A\912_TI_1\Notas_TI1\code\nodejs\proyecto_1]
code > nodejs > proyecto_1 > package.json > {} dependencies
{
"name": "mongoapp",
"version": "1.0.0",
"description": "",
"main": "index.js",
"type": "module",
"scripts": {
"test": "echo \"Error: no test specified\" && exit 1"
},
"author": "",
"license": "ISC",
"dependencies": {
"express": "4.18.1",
"mongoose": "6.4.1"
}
package.json
index.js M
code > nodejs > proyecto_1 > index.js > ...
import express from 'express'
import mongoose from 'mongoose'
const Animal = mongoose.model('Animal', new mongoose.Schema({
tipo: String,
estado: String,
}))
const app = express()
mongoose.connect('mongodb://nico:12345678@127.0.0.1:8083/miapp?authSource=admin')
app.get('/', async (_req, res) => {
console.log('listando... chanchitos...')
const animales = await Animal.find();
return res.send(animales)
})
app.get('/crear', async (_req, res) => {
console.log('creando...')
await Animal.create({ tipo: 'Chanchito', estado: 'Feliz' })
return res.send('ok')
})
app.listen(3000, () => console.log('listening...'))

Practica 3


[kralos]--[main ≡ ●]
[D:\umar\Asignaturas\Semestre 23-24 A\912_TI_1\Notas_TI1\code\nodejs\proyecto_1]
node .\index.js
listening...
listando... chanchitos...
localhost:3000
localhost:3000
[]

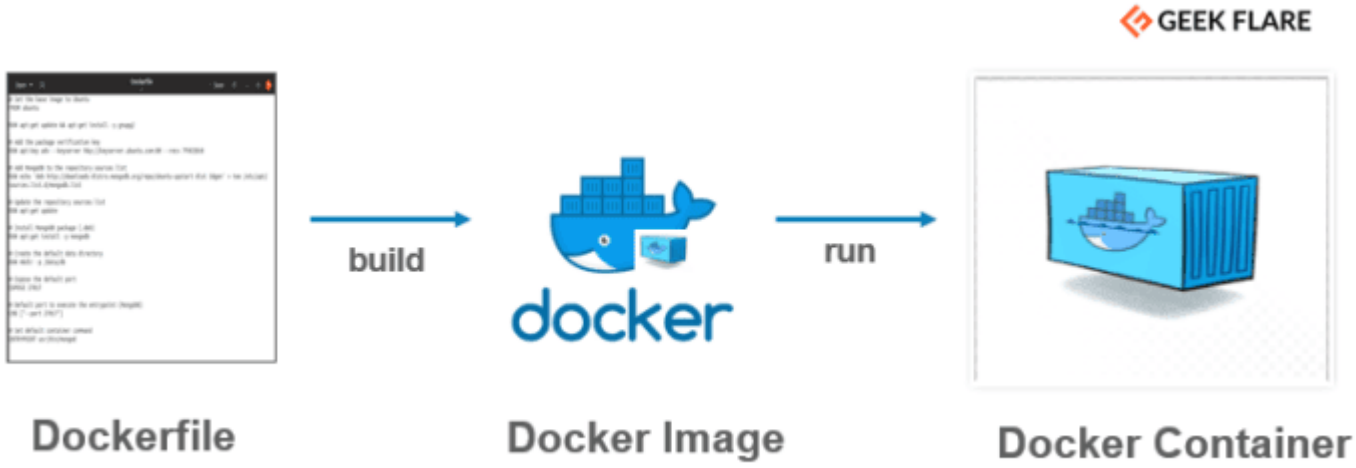

[kralos]--[main ≡ ●]
[D:\umar\Asignaturas\Semestre 23-24 A\912_TI_1\Notas_TI1\code\nodejs\proyecto_1]
node .\index.js
listening...
listando... chanchitos...
creando...
localhost:3000/crear
localhost:3000/crear
ok


[kralos]--[main ≡ ●]
[D:\umar\Asignaturas\Semestre 23-24 A\912_TI_1\Notas_TI1\code\nodejs\proyecto_1]
node .\index.js
listening...
listando... chanchitos...
creando...
listando... chanchitos...
localhost:3000
localhost:3000
[{"_id":"6531727e2d14d825c785c352","tipo":"Chanchito","estado":"Feliz","__v":0}]

Dockerfile

- Es un archivo de texto simple que incluye una serie de instrucciones que se necesitan ejecutar de manera consecutiva para cumplir con los procesos necesarios para la creación de una nueva imagen.
- Este conjunto de instrucciones serán los encargados de indicar los pasos a seguir para el ensamblaje de los elementos necesarios para el desarrollo de un contenedor en Docker.

Dockerfile


GEEK FLARE
build
docker
run
Dockerfile
Docker Image
Docker Container

Parámetros en Dockerfile

- **FROM:** es la primera instrucción. Establecer la imagen sobre la que los pasos e imágenes siguientes se desarrollan en el sistema.
- **ENV:** Indica las variables de entorno que se necesitan en el proceso de construcción de una imagen en Docker y permite la ejecución de los contenedores y sus labores en el sistema.
- **RUN:** Ejecutar una instrucción incluida en la línea de comandos de la imagen durante su proceso de construcción. El run dockerfile puede escribirse en formato SHELL o bajo la opción de escritura EXEC.

Parámetros en Dockerfile

- **ADD:** Copia de archivos, directorios y una imagen en Dockerfile. Crea una nueva capa de imagen.
- **EXPOSE:** Definición de las asignaciones referentes a los puertos para los contenedores de la plataforma que se encuentren en su etapa de ejecución.
- **CMD:** Función similar al comando RUN, pero se ejecuta sólo después de instanciar el contenedor.
- **ENTRYPOINT:** Se dirige a su aplicación por defecto en la imagen cuando se crea el contenedor.

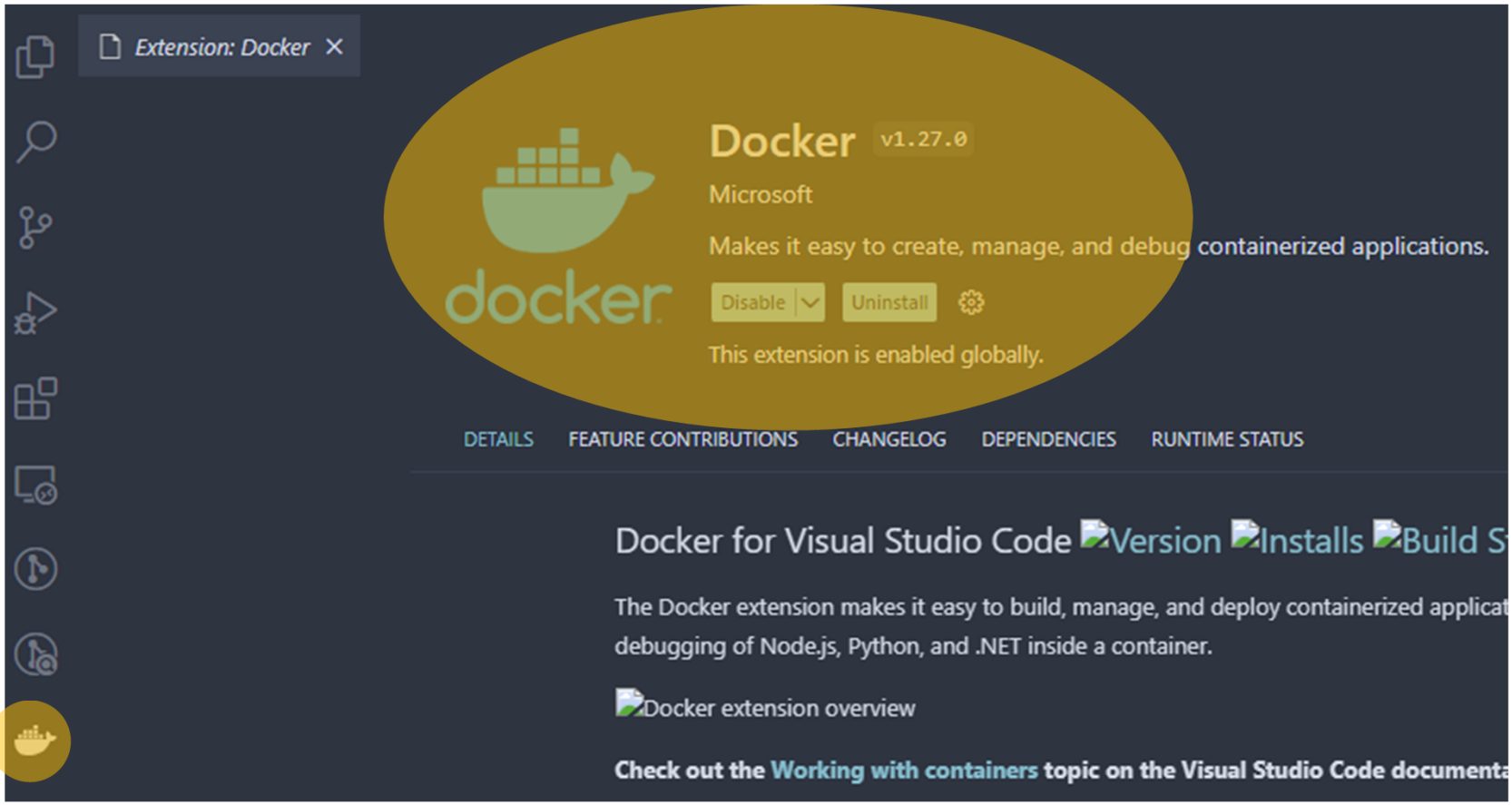
Parámetros en Dockerfile

Nota: Se puede cambiar ADD por **COPY** ya que ADD es para operaciones de copiado más complejas, y COPY copiamos de manera simple.

Así para poner comentarios

Sin comentarios

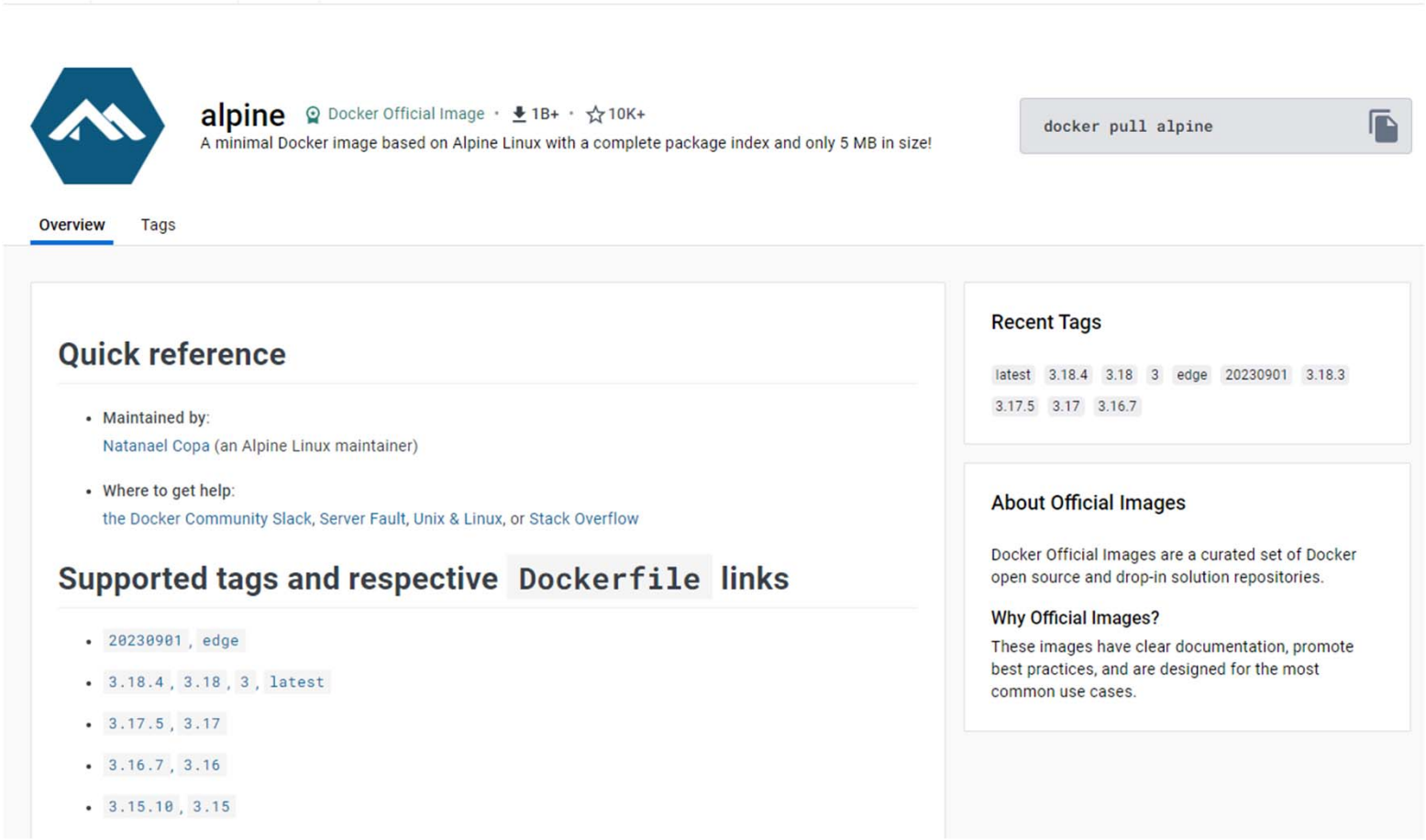
Dockerfile en VScode



docker build

```
$ docker build -h

Usage:  docker build [OPTIONS] PATH | URL | -

Build an image from a Dockerfile

  --build-arg=[]                  Set build-time variables
  --cpu-shares=0                  CPU shares (relative weight)
  --cgroup-parent=                Optional parent cgroup for the container
  --cpu-period=0                  Limit the CPU CFS (Completely Fair Scheduler) period
  --cpu-quota=0                   Limit the CPU CFS (Completely Fair Scheduler) quota
  --cpuset-cpus=                  CPUs in which to allow execution (0-3, 0,1)
  --cpuset-mems=                  MEMs in which to allow execution (0-3, 0,1)
  --disable-content-trust=true    Skip image verification
  -f, --file=                     Name of the Dockerfile (Default is 'PATH/Dockerfile')
  --force-rm=false                Always remove intermediate containers
  --help=false                    Print usage
  -m, --memory=                   Memory limit
  --memory-swap=                  Total memory (memory + swap), '-1' to disable swap
  --no-cache=false                Do not use cache when building the image
  --pull=false                    Always attempt to pull a newer version of the image
  -q, --quiet=false               Suppress the verbose output generated by the containers
  --rm=true                       Remove intermediate containers after a successful build
  -t, --tag=                      Repository name (and optionally a tag) for the image
  --ulimit=[]                     Ulimit options
```

Dockerfile {ejemplo}



alpine Docker Official Image · 1B+ · 10K+

A minimal Docker image based on Alpine Linux with a complete package index and only 5 MB in size!

```
docker pull alpine
```

Overview Tags

Quick reference

- Maintained by:
  Natanael Copa (an Alpine Linux maintainer)
- Where to get help:
  the Docker Community Slack, Server Fault, Unix & Linux, or Stack Overflow

Supported tags and respective `Dockerfile` links

- `20230901`, `edge`
- `3.18.4`, `3.18`, `3`, `latest`
- `3.17.5`, `3.17`
- `3.16.7`, `3.16`
- `3.15.10`, `3.15`

Recent Tags

`latest` `3.18.4` `3.18` `3` `edge` `20230901` `3.18.3` `3.17.5` `3.17` `3.16.7`

About Official Images

Docker Official Images are a curated set of Docker open source and drop-in solution repositories.

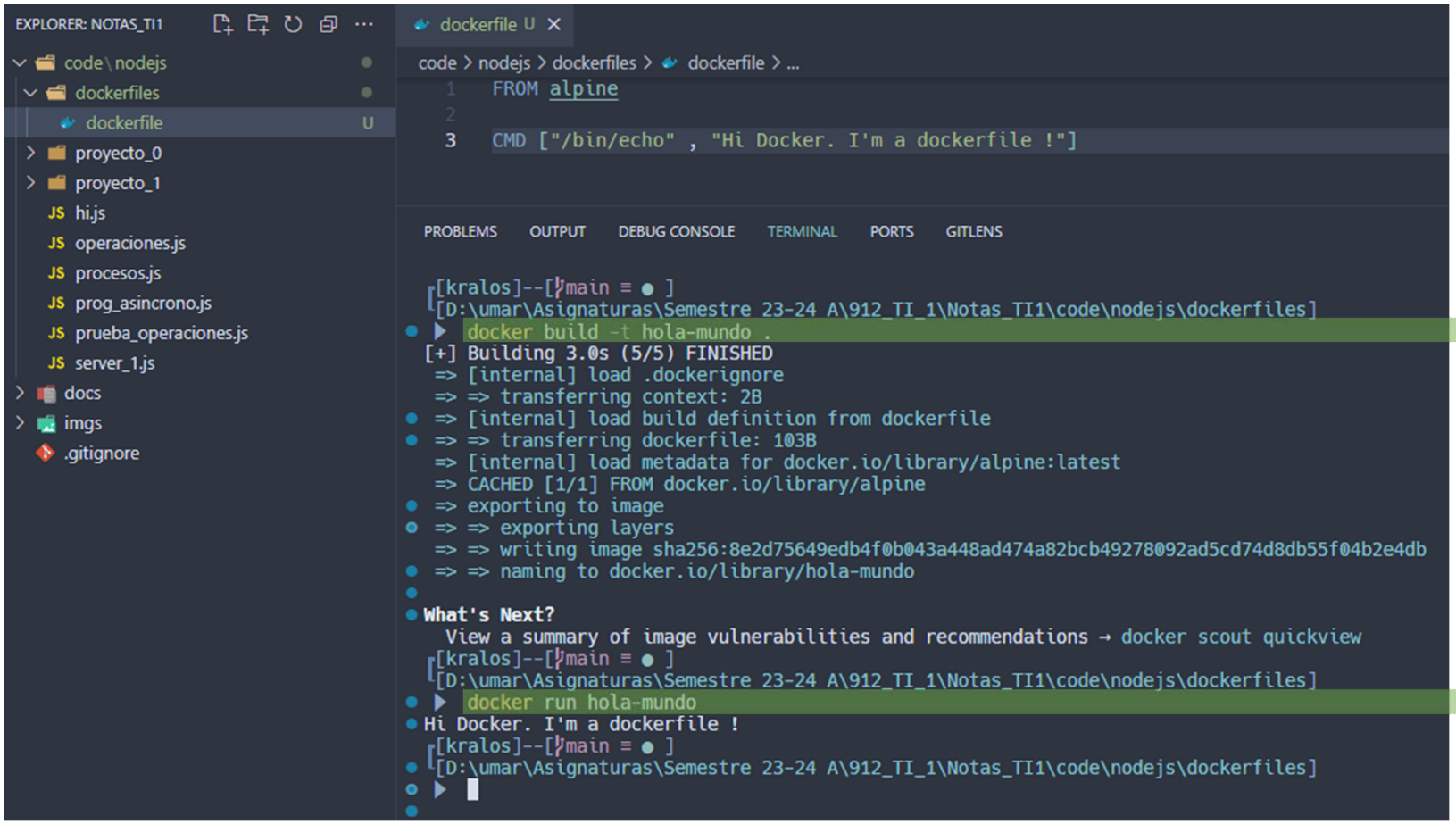
Why Official Images?

These images have clear documentation, promote best practices, and are designed for the most common use cases.

Dockerfile {ejemplo}

```
┌[kralos]--[ main ≡ ● ]
└[D:\umar\Asignaturas\Semestre 23-24 A\912_TI_1\Notas_TI1\code\nodejs\dockerfiles]
 ▶  docker images
REPOSITORY         TAG       IMAGE ID       CREATED         SIZE
mongo              latest    ee3b4d1239f1   7 days ago      748MB
hello-world        latest    9c7a54a9a43c   5 months ago    13.3kB
hello-world        linux     9c7a54a9a43c   5 months ago    13.3kB
linuxconfig/lemp   latest    670ff9140a94   13 months ago   1.05GB
┌[kralos]--[ main ≡ ● ]
└[D:\umar\Asignaturas\Semestre 23-24 A\912_TI_1\Notas_TI1\code\nodejs\dockerfiles]
 ▶  docker pull alpine
Using default tag: latest
latest: Pulling from library/alpine
96526aa774ef: Pull complete
Digest: sha256:eece025e432126ce23f223450a0326fbebde39cdf496a85d8c016293fc851978
Status: Downloaded newer image for alpine:latest
docker.io/library/alpine:latest

What's Next?
  View a summary of image vulnerabilities and recommendations → docker scout quickview alpine
┌[kralos]--[ main ≡ ● ]
└[D:\umar\Asignaturas\Semestre 23-24 A\912_TI_1\Notas_TI1\code\nodejs\dockerfiles]
 ▶  docker images
REPOSITORY         TAG       IMAGE ID       CREATED         SIZE
mongo              latest    ee3b4d1239f1   7 days ago      748MB
alpine             latest    8ca4688f4f35   3 weeks ago     7.34MB
hello-world        latest    9c7a54a9a43c   5 months ago    13.3kB
hello-world        linux     9c7a54a9a43c   5 months ago    13.3kB
linuxconfig/lemp   latest    670ff9140a94   13 months ago   1.05GB
┌[kralos]--[ main ≡ ● ]
└[D:\umar\Asignaturas\Semestre 23-24 A\912_TI_1\Notas_TI1\code\nodejs\dockerfiles]
 ▶
```

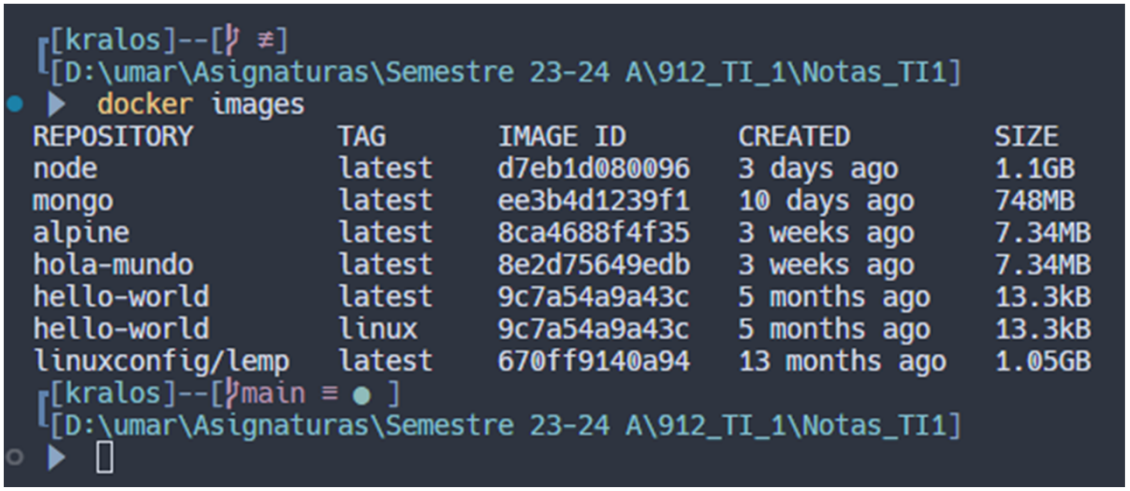
Dockerfile {ejemplo}



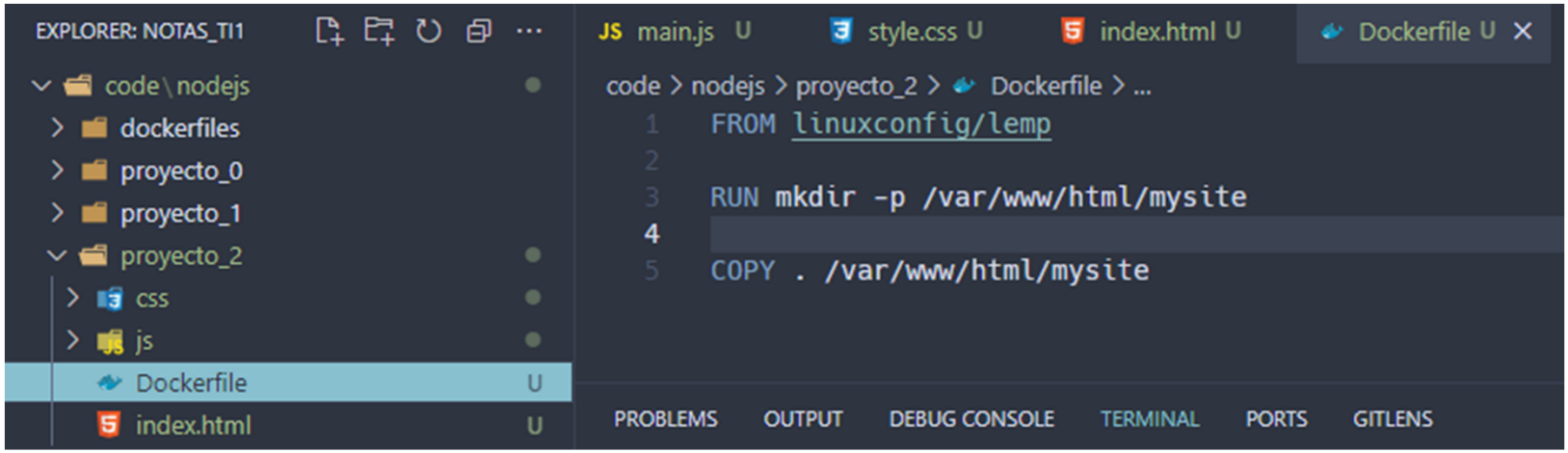
Dockerfile {ejemplo}

```
▶ docker run hola-mundo
Hi Docker. I'm a dockerfile !
┌[kralos]--[ main ≡ ● ]
└[D:\umar\Asignaturas\Semestre 23-24 A\912_TI_1\Notas_TI1\code\nodejs\dockerfiles]
▶ docker images
REPOSITORY          TAG       IMAGE ID       CREATED         SIZE
mongo               latest    ee3b4d1239f1   7 days ago      748MB
alpine              latest    8ca4688f4f35   3 weeks ago     7.34MB
hola-mundo          latest    8e2d75649edb   3 weeks ago     7.34MB
hello-world         latest    9c7a54a9a43c   5 months ago    13.3kB
hello-world         linux     9c7a54a9a43c   5 months ago    13.3kB
linuxconfig/lemp    latest    670ff9140a94   13 months ago   1.05GB
┌[kralos]--[ main ≡ ● ]
└[D:\umar\Asignaturas\Semestre 23-24 A\912_TI_1\Notas_TI1\code\nodejs\dockerfiles]
▶ docker ps -a
CONTAINER ID   IMAGE              COMMAND                  CREATED         STATUS                          PORTS     NAMES
071feda489ee   hola-mundo         "/bin/echo 'Hi Docke…"   2 minutes ago   Exited (0) About a minute ago             sharp_jemison
29293400974a   mongo              "docker-entrypoint.s…"   25 hours ago    Exited (0) 24 hours ago                   monguito
f3c176d990da   linuxconfig/lemp   "supervisord"            2 days ago      Exited (137) 2 days ago                   lemp2
140a76a8ce6a   linuxconfig/lemp   "supervisord"            2 weeks ago     Exited (137) 2 days ago                   lemp
┌[kralos]--[ main ≡ ● ]
└[D:\umar\Asignaturas\Semestre 23-24 A\912_TI_1\Notas_TI1\code\nodejs\dockerfiles]
▶
```

Actualización de imágenes docker



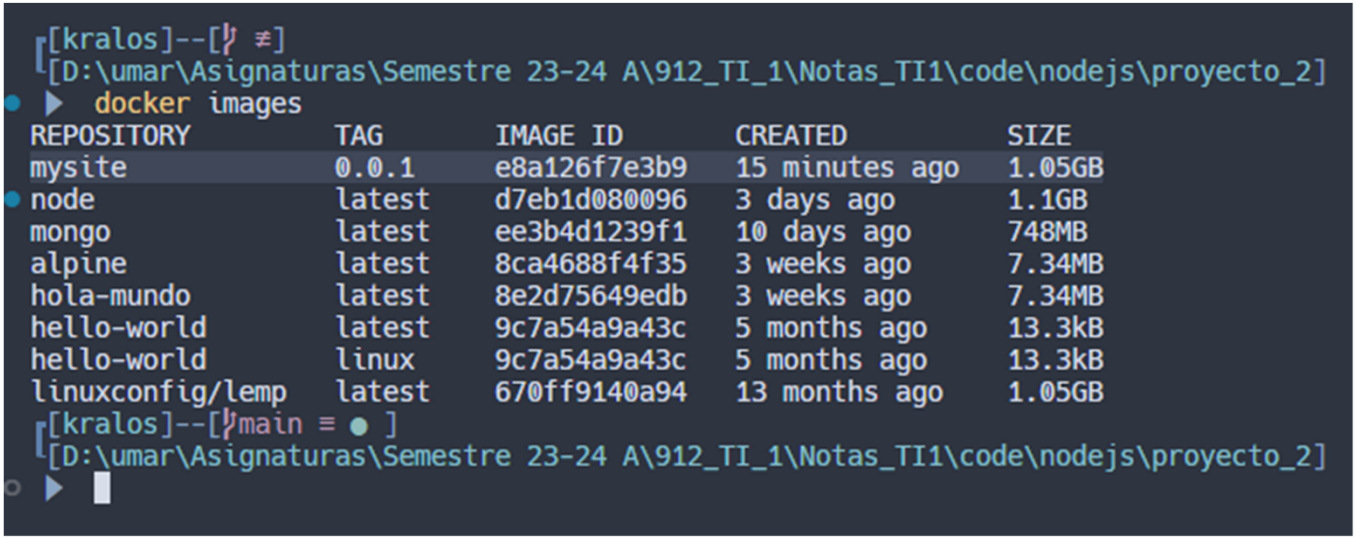
Practica 4


EXPLORER: NOTAS_TI1
code \ nodejs
dockerfiles
proyecto_0
proyecto_1
proyecto_2
css
js
Dockerfile U
index.html U
main.js U
style.css U
index.html U
Dockerfile U
code > nodejs > proyecto_2 > Dockerfile > ...
1 FROM linuxconfig/lemp
2
3 RUN mkdir -p /var/www/html/mysite
4
5 COPY . /var/www/html/mysite
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
GITLENS

Practica 4

```
┌[kralos]--[ main ≡ ● ]
└[D:\umar\Asignaturas\Semestre 23-24 A\912_TI_1\Notas_TI1]
 ▶ cd .\code\nodejs\proyecto_2\
┌[kralos]--[ main ≡ ● ]
└[D:\umar\Asignaturas\Semestre 23-24 A\912_TI_1\Notas_TI1\code\nodejs\proyecto_2]
● ▶ ls

        Directory: D:\umar\Asignaturas\Semestre 23-24 A\912_TI_1\Notas_TI1\code\nodejs\proyecto_2


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d----         10/23/2023  11:07 AM                css
d----         10/23/2023  11:07 AM                js
-a---         10/23/2023  11:41 AM             89 Dockerfile
-a---         10/23/2023  11:10 AM            346 index.html

┌[kralos]--[ main ≡ ● ]
└[D:\umar\Asignaturas\Semestre 23-24 A\912_TI_1\Notas_TI1\code\nodejs\proyecto_2]
● ▶ docker build -t mysite:0.0.1 .
[+] Building 0.0s (0/0)
[+] Building 46.3s (8/8) FINISHED
 => [internal] load build definition from Dockerfile
 => => transferring dockerfile: 126B
 => [internal] load .dockerignore
 => => transferring context: 2B
 => [internal] load metadata for docker.io/linuxconfig/lemp:latest
 => [1/3] FROM docker.io/linuxconfig/lemp
 => [internal] load build context
 => => transferring context: 623B
 => [2/3] RUN mkdir -p /var/www/html/mysite
 => [3/3] COPY . /var/www/html/mysite
 => exporting to image
 => => exporting layers
 => => writing image sha256:e8a126f7e3b911f911b32daf95225f2bcf3c6d07ea296a78873681f17e0d675d
 => => naming to docker.io/library/mysite:0.0.1

What's Next?
  View a summary of image vulnerabilities and recommendations → docker scout quickview
┌[kralos]--[ main ≡ ● ]
└[D:\umar\Asignaturas\Semestre 23-24 A\912_TI_1\Notas_TI1\code\nodejs\proyecto_2]
○ ▶
```

Practica 4


[kralos]--[≠]
[D:\umar\Asignaturas\Semestre 23-24 A\912_TI_1\Notas_TI1\code\nodejs\proyecto_2]
docker images
REPOSITORY TAG IMAGE ID CREATED SIZE
mysite 0.0.1 e8a126f7e3b9 15 minutes ago 1.05GB
node latest d7eb1d080096 3 days ago 1.1GB
mongo latest ee3b4d1239f1 10 days ago 748MB
alpine latest 8ca4688f4f35 3 weeks ago 7.34MB
hola-mundo latest 8e2d75649edb 3 weeks ago 7.34MB
hello-world latest 9c7a54a9a43c 5 months ago 13.3kB
hello-world linux 9c7a54a9a43c 5 months ago 13.3kB
linuxconfig/lemp latest 670ff9140a94 13 months ago 1.05GB
[kralos]--[main ≡ ●]
[D:\umar\Asignaturas\Semestre 23-24 A\912_TI_1\Notas_TI1\code\nodejs\proyecto_2]

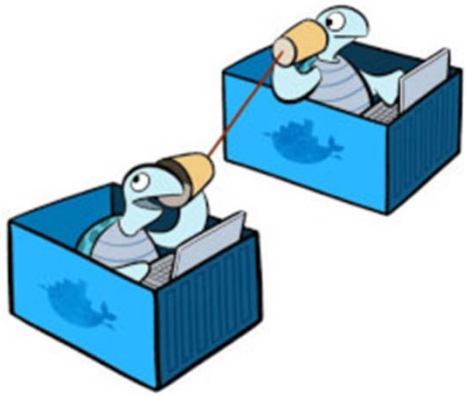
Practica 4

```
[kralos]--[main ≡ ● ][x125]
[D:\umar\Asignaturas\Semestre 23-24 A\912_TI_1\Notas_TI1\code\nodejs\proyecto_2]
▶ docker run --name=mi_sitio -dP mysite:0.0.1
f05bd8066b4e5a56c691c3f961cacc8b103975608ebd2a0eda57d80a7589743e
[kralos]--[main ≡ ● ]
[D:\umar\Asignaturas\Semestre 23-24 A\912_TI_1\Notas_TI1\code\nodejs\proyecto_2]
▶ docker port mi_sitio
80/tcp -> 0.0.0.0:32769
3306/tcp -> 0.0.0.0:32768
[kralos]--[main ≡ ● ]
[D:\umar\Asignaturas\Semestre 23-24 A\912_TI_1\Notas_TI1\code\nodejs\proyecto_2]
▶ docker exec -it mi_sitio /bin/bash
root@f05bd8066b4e:/# ls /var/www/html/
index.nginx-debian.html  index.php                mysite/
root@f05bd8066b4e:/# ls /var/www/html/mysite/
Dockerfile  css  index.html  js
root@f05bd8066b4e:/# cat /var/www/html/mysite/index.html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>In_Docker</title>
    <link rel="stylesheet" href="./css/style.css">
</head>
<body>
    <h1>Hola estoy en un contendeor ...</h1>
</body>
<script src="./js/main.js"></script>
</html>root@f05bd8066b4e:/#
```

Practica 1er Parcial

1. Crear app con nodejs {express}
2. Crear un dockerfile
   - Usar la imagen node
   - Copiar el proyecto creado en el paso 1
   - Exponer un puerto
3. Generar una nueva imagen con algunas versión.
4. Crear un contenedor con la imagen del paso 3
5. Verificar el lanzamiento de la app desde un navegador.

Redes en Docker



Redes en Docker

- Por defecto los contenedores que creamos se conectan a la red de tipo bridge llamada **bridge** (por defecto el direccionamiento de esta red es 172.17.0.0/16). Los contenedores conectados a esta red que quieren exponer algún puerto al exterior tienen que usar la opción -p para mapear puertos.

```
▶ docker network ls
NETWORK ID     NAME      DRIVER    SCOPE
c42d81defb27   bridge    bridge    local
6ae7b1eda800   host      host      local
193e4faae928   none      null      local
```

Redes en Docker

```
[kralos]
[~]
 ▶ docker network inspect bridge
[
    {
        "Name": "bridge",
        "Id": "c42d81defb27afc6f15350402006b61fd2bd7d529e4e42d2c4978b69e80e577a",
        "Created": "2023-11-06T17:00:53.2462976Z",
        "Scope": "local",
        "Driver": "bridge",
        "EnableIPv6": false,
        "IPAM": {
            "Driver": "default",
            "Options": null,
            "Config": [
                {
                    "Subnet": "172.17.0.0/16",
                    "Gateway": "172.17.0.1"
                }
            ]
        },
        "Internal": false,
        "Attachable": false,
        "Ingress": false,
        "ConfigFrom": {
            "Network": ""
        },
        "ConfigOnly": false,
        "Containers": {},
        "Options": {
            "com.docker.network.bridge.default_bridge": "true",
            "com.docker.network.bridge.enable_icc": "true",
            "com.docker.network.bridge.enable_ip_masquerade": "true",
            "com.docker.network.bridge.host_binding_ipv4": "0.0.0.0",
            "com.docker.network.bridge.name": "docker0",
            "com.docker.network.driver.mtu": "1500"
        },
        "Labels": {}
    }
]
```

Redes en Docker

--network nombre_red

- Para conectar un contenedor a una red.

- Si conecto un contenedor a la red **host**, el contenedor será accesible usando la misma IP que tu máquina.
- La red **none** no configurará ninguna IP para el contenedor y no tiene acceso a la red externa ni a otros contenedores. Tiene la dirección loopback y se puede usar para ejecutar trabajos por lotes.

Redes en Docker

- Crea una red dentro de Docker

  docker network create nombre-red

- Crea una red con rango dentro de Docker

```
docker network create -d macvlan \
--subnet=172.16.86.0/24 \
--gateway=172.16.86.1 \
-o parent=eth0  name-red
```

- Borrar una red dentro de Docker

  docker network rm nombre-red

Redes en Docker

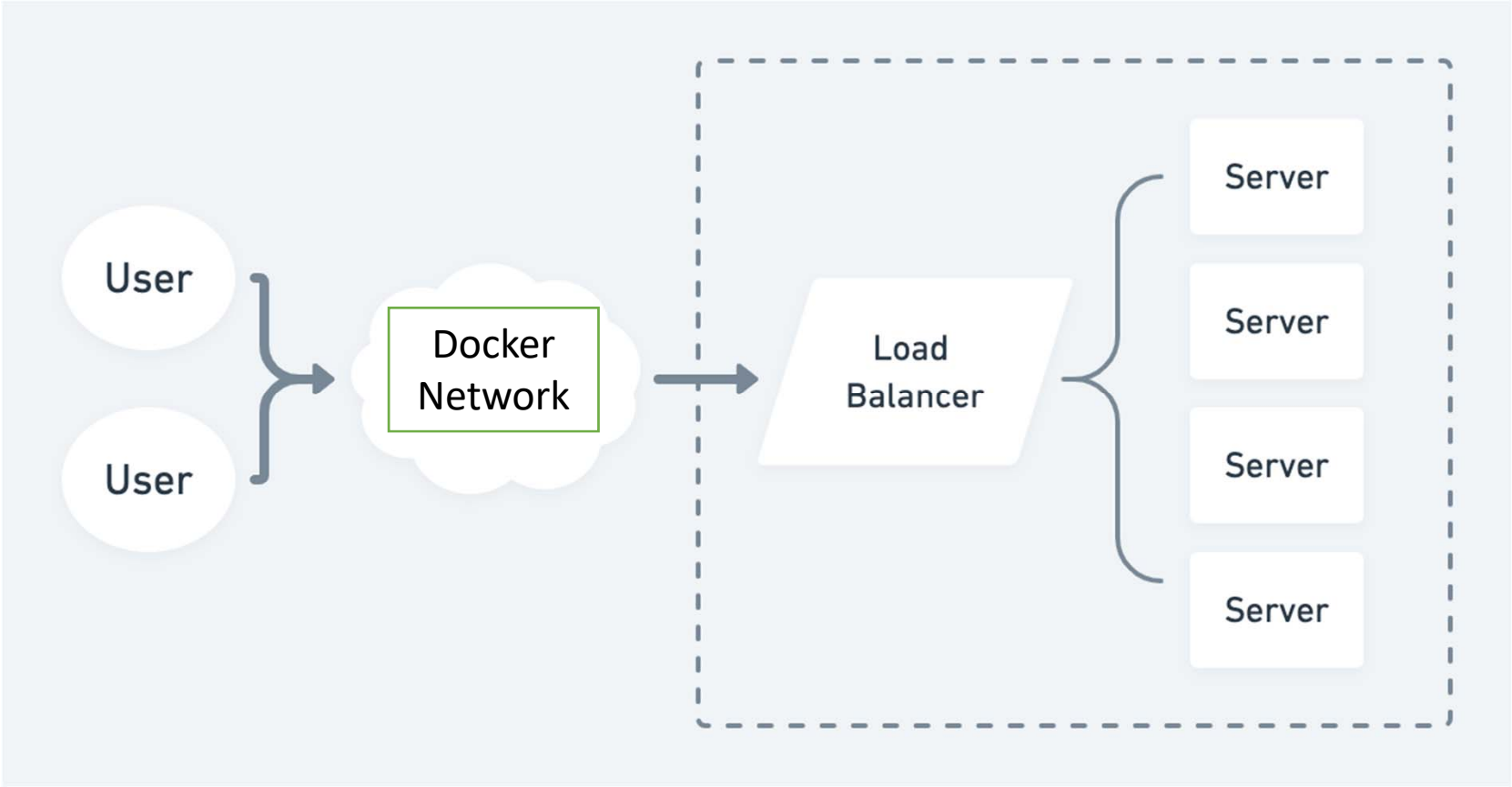
```
 ▶ docker network create -d macvlan --subnet=172.16.86.0/24 --gateway=172.16.86.1 -o parent=eth0 my-macvlan-net
6f61c1ecf426d808f39be8436de6c2de4e868f0f80eb3776acdcb4f4471dfaf1
┌[kralos]
└[~]
 ▶ docker network ls
NETWORK ID     NAME            DRIVER    SCOPE
c42d81defb27   bridge          bridge    local
6ae7b1eda800   host            host      local
6f61c1ecf426   my-macvlan-net  macvlan   local
193e4faae928   none            null      local
┌[kralos]
└[~]
 ▶ docker network inspect my-macvlan-net
[
    {
        "Name": "my-macvlan-net",
        "Id": "6f61c1ecf426d808f39be8436de6c2de4e868f0f80eb3776acdcb4f4471dfaf1",
        "Created": "2023-11-06T17:32:47.8301725Z",
        "Scope": "local",
        "Driver": "macvlan",
        "EnableIPv6": false,
        "IPAM": {
            "Driver": "default",
            "Options": {},
            "Config": [
                {
                    "Subnet": "172.16.86.0/24",
                    "Gateway": "172.16.86.1"
                }
            ]
        },
        "Internal": false,
        "Attachable": false,
        "Ingress": false,
        "ConfigFrom": {
            "Network": ""
        },
        "ConfigOnly": false,
        "Containers": {},
        "Options": {
            "parent": "eth0"
        },
        "Labels": {}
    }
]
```

Practica 5

- Enrutamiento y Balanceo

```
┌[kralos]--[ main ≡ ● ]
└[D:\umar\Asignaturas\Semestre 23-24 A\912_TI_1\Notas_TI1]
● ▶ docker network create mynet
36367c36d4688ca5153e9562adf9edcfc4e79002589df15655b311139f738e29
┌[kralos]--[ main ≡ ● ]
└[D:\umar\Asignaturas\Semestre 23-24 A\912_TI_1\Notas_TI1]
● ▶ docker network ls
NETWORK ID     NAME              DRIVER    SCOPE
500fcdf0c8d4   bridge            bridge    local
6ae7b1eda800   host              host      local
6f61c1ecf426   my-macvlan-net    macvlan   local
36367c36d468   mynet             bridge    local
193e4faae928   none              null      local
┌[kralos]--[ main ≡ ● ]
└[D:\umar\Asignaturas\Semestre 23-24 A\912_TI_1\Notas_TI1]
● ▶ docker images
REPOSITORY          TAG                        IMAGE ID       CREATED          SIZE
nginx               mainline-alpine3.18-slim   ab7c6a6862ef   2 weeks ago      17MB
nginx               latest                     593aee2afb64   2 weeks ago      187MB
mysite              0.0.1                      e8a126f7e3b9   2 weeks ago      1.05GB
node                latest                     d7eb1d080096   2 weeks ago      1.1GB
mongo               latest                     ee3b4d1239f1   3 weeks ago      748MB
alpine              latest                     8ca4688f4f35   5 weeks ago      7.34MB
hola-mundo          latest                     8e2d75649edb   5 weeks ago      7.34MB
hello-world         latest                     9c7a54a9a43c   6 months ago     13.3kB
hello-world         linux                      9c7a54a9a43c   6 months ago     13.3kB
linuxconfig/lemp    latest                     670ff9140a94   14 months ago    1.05GB
```

Practica 5


User
User
Docker Network
Load Balancer
Server
Server
Server
Server

Practica 5

```
[kralos]--[main ≡ ● ][x125]
[D:\umar\Asignaturas\Semestre 23-24 A\912_TI_1\Notas_TI1\code\networking\Balanceo]
  docker network inspect mynet
[
    {
        "Name": "mynet",
        "Id": "36367c36d4688ca5153e9562adf9edcfc4e79002589df15655b311139f738e29",
        "Created": "2023-11-08T16:44:07.691363211Z",
        "Scope": "local",
        "Driver": "bridge",
        "EnableIPv6": false,
        "IPAM": {
            "Driver": "default",
            "Options": {},
            "Config": [
                {
                    "Subnet": "172.18.0.0/16",
                    "Gateway": "172.18.0.1"
                }
            ]
        },
        "Internal": false,
        "Attachable": false,
        "Ingress": false,
        "ConfigFrom": {
            "Network": ""
        },
        "ConfigOnly": false,
        "Containers": {},
        "Options": {},
        "Labels": {}
    }
]
```

Practica 5

YAML
{YAML Ain't Markup Language}

- YAML es un lenguaje de serialización de datos y suele utilizarse en el diseño de archivos de configuración, no para los documentos.
- YAML utiliza una extensión de archivos .yml o .yaml y sigue reglas de sintaxis específicas.
- Tiene características que provienen de Perl, C, XML, HTML y otros lenguajes de programación. También se basa en JSON, por lo que los archivos JSON son compatibles con YAML.

Docker Compose

- Es una herramienta de Docker que orquesta contenedores en un mismo cliente. Consiste en un archivo de texto en formato YAML, que define de forma declarativa los contenedores que se van a desplegar, así como las dependencias entre ellos.
- Este archivo normalmente tiene el nombre docker-compose.yml pero puede tener cualquier otro nombre, siempre que se especifique al ejecutar el comando docker-compose con el argumento -f. La utilidad más habitual para esta herramienta es el despliegue de aplicaciones en entornos locales, para el desarrollo y pruebas.

Docker Compose

- Lo primero es definir los servicios que vamos a desplegar.
- Estos servicios son los que conforman nuestra aplicación y, normalmente, cada uno despliega un contenedor, asociado a una imagen Docker. Dentro de cada servicio podemos definir las características de cada contenedor, como el nombre, la imagen, los puertos que expone, volúmenes y redes a las que se conecta, etc.

Docker Compose

Dockerfile 73 Bytes

```
1  FROM node:12
2  WORKDIR /home/nodeApp
3  EXPOSE 8080
4  CMD [ "node", "index.js" ]
```

docker-compose.yml 162 Bytes

```
1  version: "3.3"
2  services:
3   web:
4    container_name: nodeDockerApp
5    build: .
6
7
8    ports:
9      - "8080:8080"
10   volumes:
11
12     - ./nodeApp:/home/nodeApp
13
```

Referencias

- *Comandos más usados en Docker*. (n.d.). Tutoriales.online. Retrieved October 4, 2023, from https://tutoriales.online/chuletas/docker
- *Docker*. (n.d.). Docker.com. Retrieved October 4, 2023, from https://hub.docker.com/
- *Docker*. (n.d.). Docker.com. Retrieved October 4, 2023, from https://hub.docker.com/
- Avi. (2023, September 14). *¿Qué es Dockerfile y cómo crear una imagen Docker?* Geekflare. https://geekflare.com/es/dockerfile-tutorial/
- Muñoz, J. D. (2020, February 18). *Introducción a las redes en docker. Enlazando contenedores docker*. PLEDIN 3.0. https://www.josedomingo.org/pledin/2020/02/redes-en-docker/
- Torres, G. (2020, December 23). *Cómo funcionan las redes en Docker - return(GiS);* return(GiS); Gisela Torres. https://www.returngis.net/2020/12/como-funcionan-las-redes-en-docker/
- *¿Qué es YAML?* (n.d.). Redhat.com. Retrieved November 6, 2023, from https://www.redhat.com/es/topics/automation/what-is-yaml

Referencias {Código}

- https://github.com/nschurmann/mongoapp-curso-docker