

5. INFORME — Creación y Publicación de Imagen Docker Mejorada (Versión 2.0)

1. Crear una nueva carpeta del proyecto

Se crea el directorio donde se almacenará la versión mejorada de la imagen Docker.

Comandos:

```
vboxuser@DEBIAN:~$ mkdir proyecto_v2
vboxuser@DEBIAN:~$ cd proyecto_v2
vboxuser@DEBIAN:~/proyecto_v2$
```

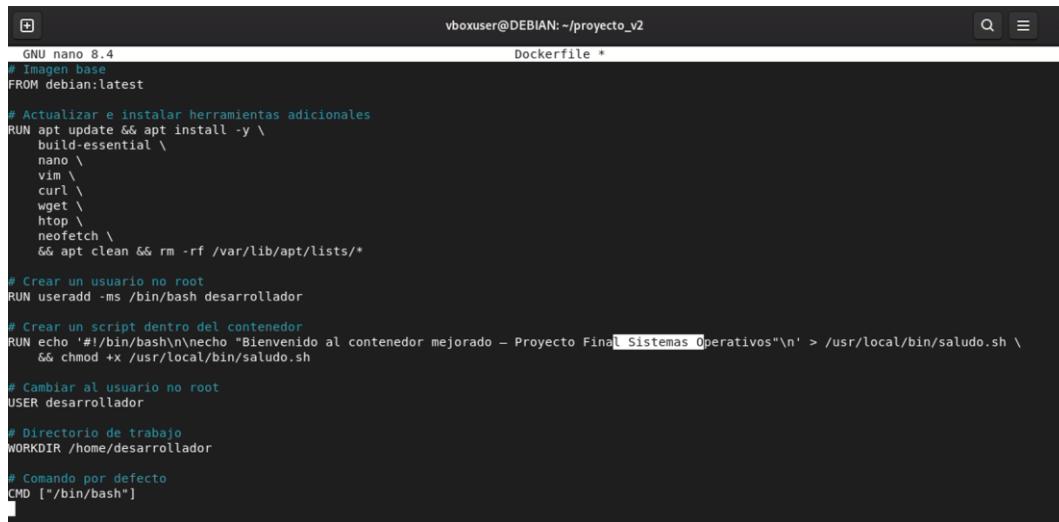
2. Crear el Dockerfile mejorado

Dentro del proyecto abrimos un Dockerfile con nano.

Comando:

```
vboxuser@DEBIAN:~$ mkdir proyecto_v2
vboxuser@DEBIAN:~$ cd proyecto_v2
vboxuser@DEBIAN:~/proyecto_v2$ nano Dockerfile
```

Contenido completo del Dockerfile:



```
GNU nano 8.4
# Imagen base
FROM debian:latest

# Actualizar e instalar herramientas adicionales
RUN apt update && apt install -y \
    build-essential \
    nano \
    vim \
    curl \
    wget \
    htop \
    neofetch \
    && apt clean && rm -rf /var/lib/apt/lists/*

# Crear un usuario no root
RUN useradd -ms /bin/bash desarrollador

# Crear un script dentro del contenedor
RUN echo "#!/bin/bash\n\nnecho \"Bienvenido al contenedor mejorado - Proyecto Final Sistemas Operativos\"\n" > /usr/local/bin/saludo.sh \
    && chmod +x /usr/local/bin/saludo.sh

# Cambiar al usuario no root
USER desarrollador

# Directorio de trabajo
WORKDIR /home/desarrollador

# Comando por defecto
CMD ["/bin/bash"]
```

Descripción breve del Dockerfile:

- Usa **Debian** como base.
- Instala herramientas esenciales: compiladores, editores y utilidades.
- Crea un usuario sin privilegios llamado *desarrollador*.
- Genera el script saludo.sh accesible globalmente desde /usr/local/bin.
- Define el directorio de trabajo y el intérprete Bash como comando inicial.

3. Construcción de la imagen mejorada

- Desde la carpeta del proyecto se ejecutó:

```
vboxuser@DEBIAN:/proyecto_v2$ sudo docker build -t miimage:2.0 .
[sudo] contraseña para vboxuser:
[+] Building 3.0s (4/8)
--> [internal] load build definition from Dockerfile
--> >> transferring dockerfile: 726B
--> [internal] load metadata for docker.io/library/debian:latest
--> [internal] load .dockerignore
--> >> transferring context: 2B
--> CACHED [/S1] FROM docker.io/library/debian:latest@sha256:0d01188e8dd0ac63bf1f155900fad49279131a876a1ea7fac917c62e87ccb2732d
--> >> resolve docker.io/library/debian:latest@sha256:0d01188e8dd0ac63bf1f155900fad49279131a876a1ea7fac917c62e87ccb2732d
--> [2/5] RUN apt update && apt install -y build-essential nano vim curl wget htop neofetch && apt clean
--> >> # WARNING: apt does not have a stable CLI interface. Use with caution in scripts.
--> >> # Get:1 http://deb.debian.org/debian trixie InRelease [140 kB]
--> >> # Get:2 http://deb.debian.org/debian trixie-updates InRelease [47.3 kB]
--> >> # Get:3 http://deb.debian.org/debian-security trixie-security InRelease [43.4 kB]
--> >> # Get:4 http://deb.debian.org/debian trixie/main amd64 Packages [9670 kB]
```

- sudo docker build -t miimage:2.0 .
- El sistema descargó paquetes, configuró usuarios y generó la imagen final.
- La línea final del proceso muestra:

```
-----
Dockerfile:5
-----
4 |      # Actualizar e instalar herramientas adicionales
5 | >>> RUN apt update && apt install -y \
6 | >>>      build-essential \
7 | >>>      nano \
8 | >>>      vim \
9 | >>>      curl \
10 | >>>      wget \
11 | >>>      htop \
12 | >>>      neofetch \
13 | >>>      && apt clean && rm -rf /var/lib/apt/lists/*
14 |
```

4. Comparación de tamaños

Se listaron las imágenes instaladas y se obtuvo una captura donde se observan los tamaños de cada una. Esto evidencia que la nueva versión incluye más herramientas y archivos:

IMAGE	ID	DISK USAGE	CONTENT SIZE	EXTRA
hello-world:latest	d4aaab6242e0	25.9kB	9.52kB	U
miimage:1.0	792561e5975c	620MB	171MB	U
miimage:2.0	1b659656105c	3.96GB	1.1GB	
miimagen:1.0	1ed5a9b4aae5	620MB	171MB	U

5. Ejecución de la imagen mejorada

Para iniciar el contenedor se ejecutó. Esto abre una sesión interactiva dentro del contenedor :

```
vboxuser@DEBIAN:~/proyecto_v2$ sudo docker run -it --name contenedor_mejorado miimage:2.0 /bin/bash  
desarrollador@c55a50c7c242:~$
```

Para validar el Script:

```
desarrollador@c55a50c7c242:~$ saludo.sh  
Bienvenido al contenedor mejorado – Proyecto Final Sistemas Operativos  
desarrollador@c55a50c7c242:~$
```

CONCLUSIÓN

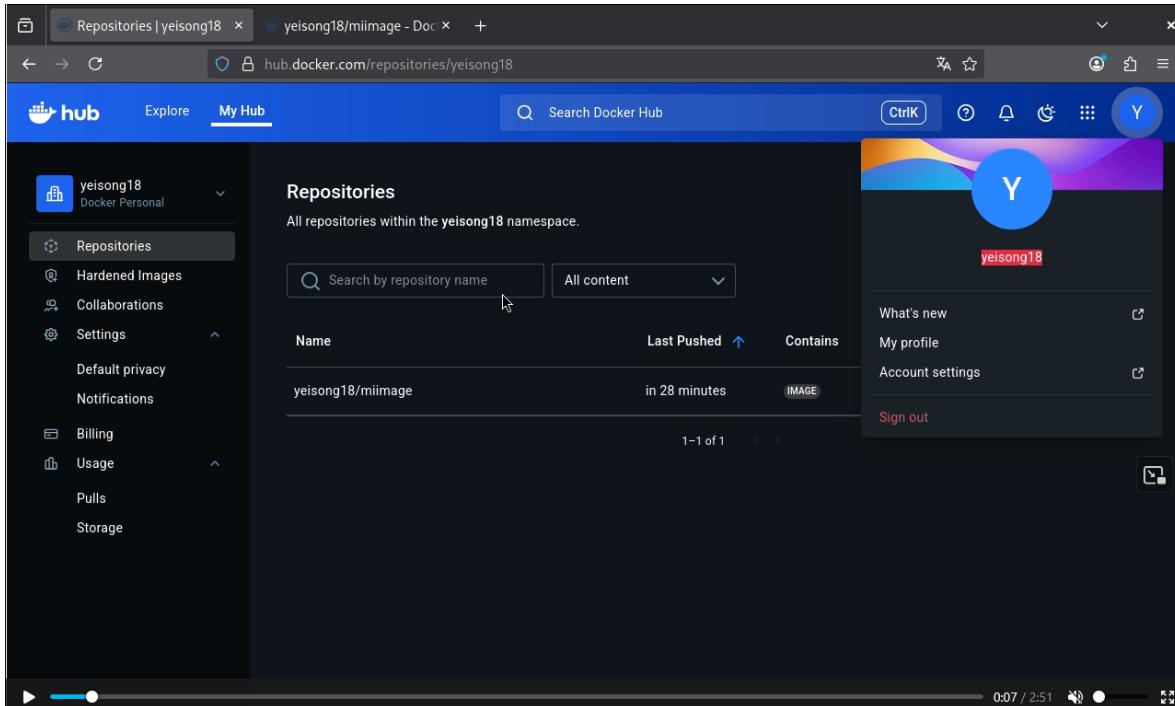
En esta actividad se modificó la imagen Docker previamente creada en el proyecto para agregar funcionalidades adicionales, optimizar su utilidad y posteriormente publicar la versión mejorada en Docker Hub.

El objetivo es demostrar el proceso completo: creación de una imagen avanzada, ejecución de scripts dentro del contenedor, edición de un Dockerfile mejorado y publicación final en un repositorio público.

La nueva imagen miimage:2.0 incluye herramientas de administración, edición, compilación y un script automatizado accesible desde cualquier parte del contenedor.

5.2 INFORME — Descripción detallada de cada paso y comando - Publicación en Docker Hub

1. Tener una cuenta en Docker Hun



2. Iniciar sesión en Docker Hub

Comando

```
vboxuser@DEBIAN: $ sudo docker login
USING WEB-BASED LOGIN
Info → To sign in with credentials on the command line, use 'docker login -u <username>'

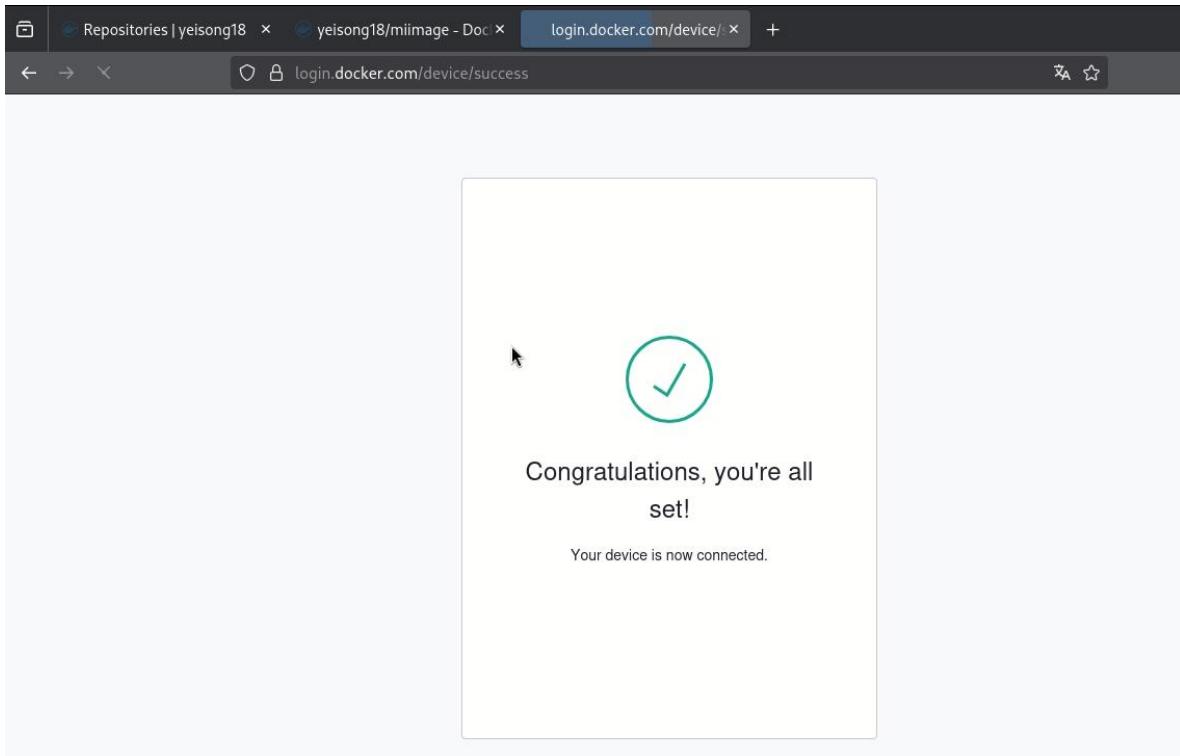
Your one-time device confirmation code is: XCFX-LXVQ
Press ENTER to open your browser or submit your device code here: https://login.docker.com/activate
Waiting for authentication in the browser...
```

Descripción

Este comando abre una sesión desde tu computador hacia Docker Hub.

- Permite autenticarte usando:
- Usuario: yeisong18
- Contraseña: la de tu cuenta de Docker Hub

Sin esta autenticación, no puedes subir imágenes al repositorio.



3. Subir la imagen al repositorio en Docker Hub

Comando:

```
vboxuser@DEBIAN:~$ sudo docker tag miimage:2.0 yeisong18/miimage:2.0
vboxuser@DEBIAN:~$
```

Descripción

Este comando renombra (etiqueta) la imagen local para que tenga el formato correcto requerido por Docker Hub:

usuario/imagen:tag

En mi caso queda:

- Usuario: yeisong18
- Imagen: miimage
- Etiqueta: 2.0

Esta etiqueta indica la versión que vamos a publicar.

4. Subir la imagen al repositorio en Docker Hub

Comando:

```
vboxuser@DEBIAN:~$ sudo docker push yeisong18/miimage:2.0
The push refers to repository [docker.io/yeisong18/miimage]
85636c73425a: Already exists
4f4fb700ef54: Layer already exists
61de855609c5: Layer already exists
4f58d5466508: Layer already exists
2981f7e8980b: Layer already exists
5da71eb08337: Layer already exists
2.0: digest: sha256:1b659656105c6d597fd5d8f9bffe950c3c5579f58f110a3cac53424784466702 size: 856
vboxuser@DEBIAN:~$
```

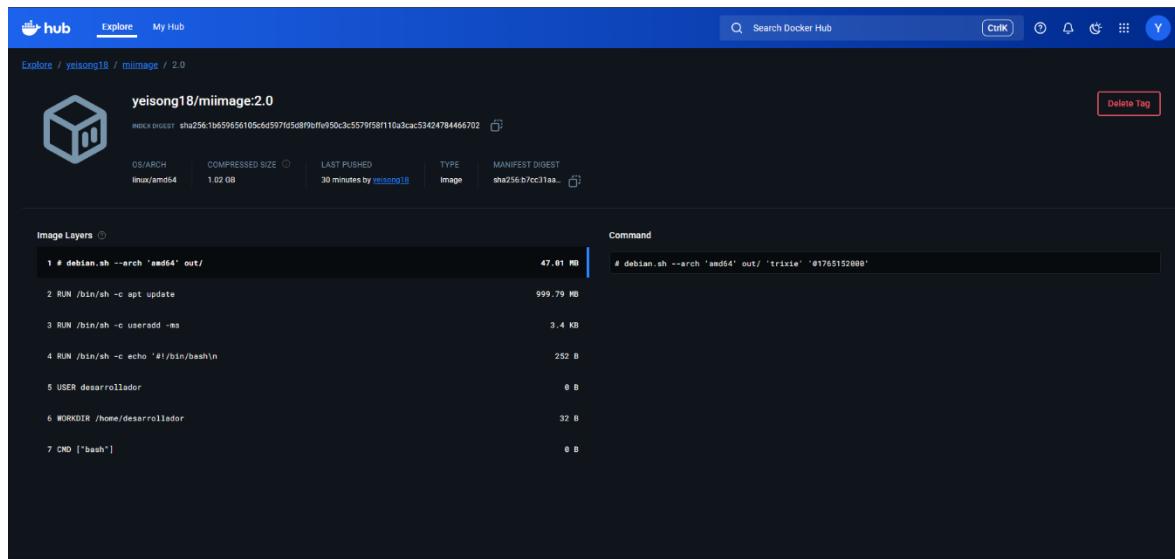
Descripción

- Este comando envía (sube) la imagen desde tu máquina hacia tu repositorio público en Docker Hub.
- Durante la subida verás que Docker envía varias capas (layers) comprimidas.
- Cuando finaliza, tu imagen queda disponible públicamente.

5. Verificar la subida en Docker Hub

URL

👉 <https://hub.docker.com/r/yeisong18/miimage>



Descripción

Al abrir esa página deberías ver:

- La imagen miimage

- La versión publicada 2.0
- Las capas que contiene
- El tamaño total
- El botón de "Pull" para descargarla