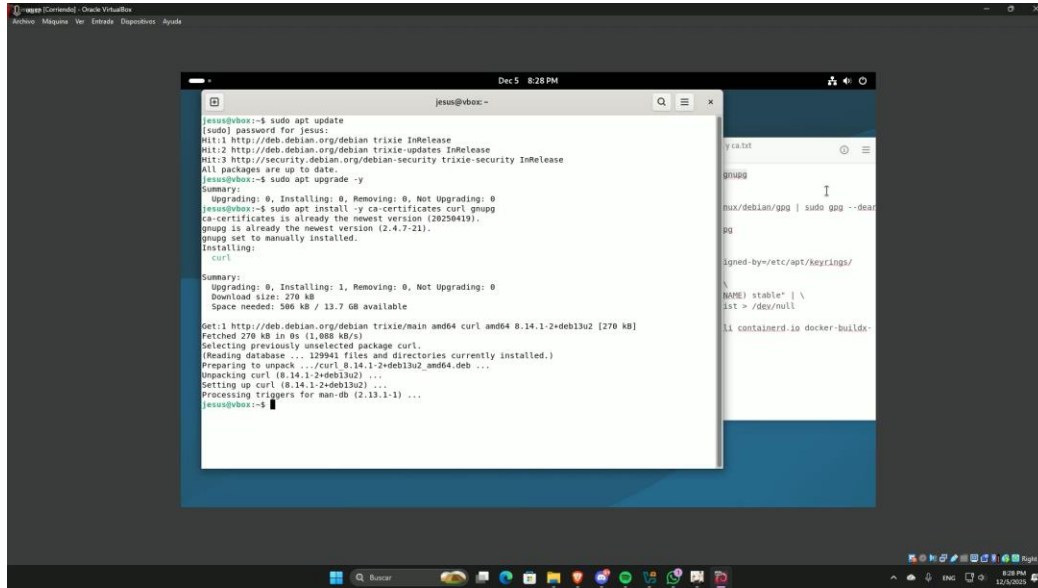


## DOCUMENTO ACTIVIDAD N.4

A continuación, se anexan las capturas con sus respectivas explicaciones

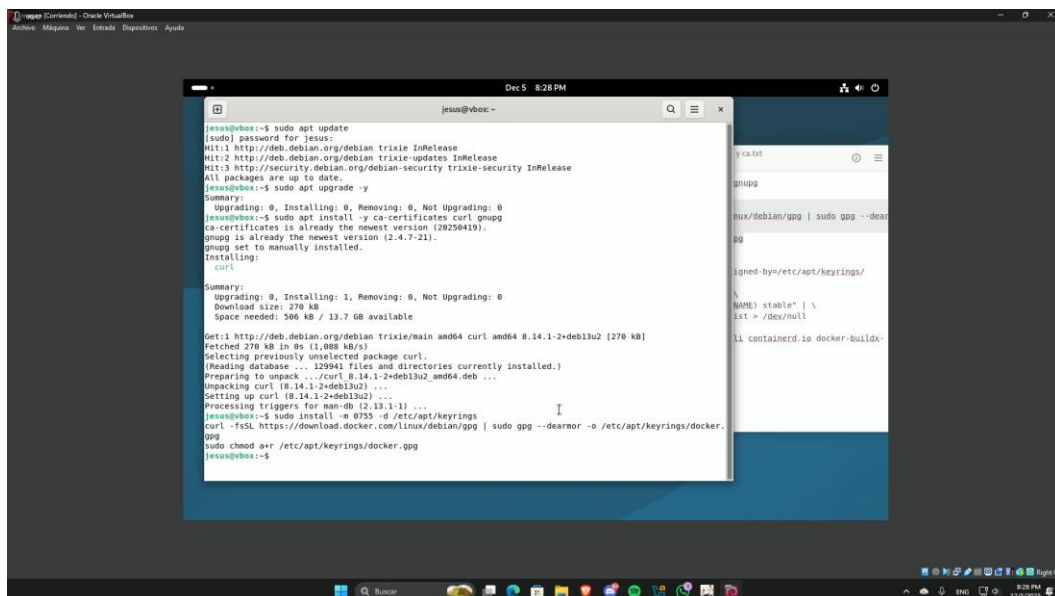
### 4.1 INSTALACION DOCKER ENGINE EN DEBIAN:

1.actualizamos paquetes y versiones, también instalamos las ultimas actualizaciones disponibles



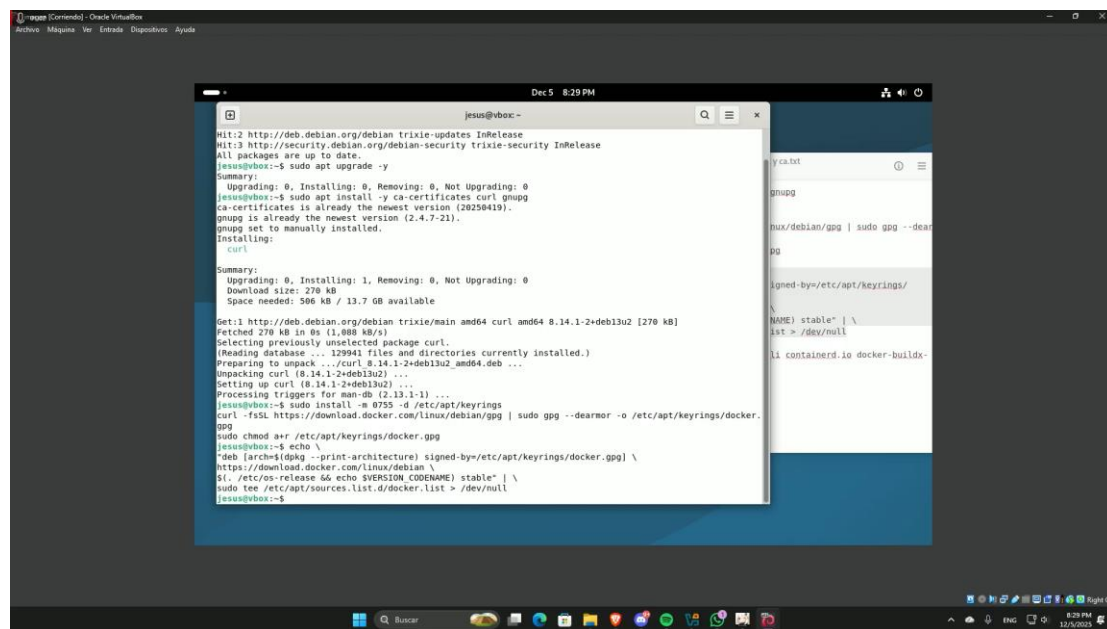
```
jesus@vbox:~$ sudo apt update
[sudo] password for jesus:
Hit:1 http://deb.debian.org/debian trixie InRelease
Hit:2 http://deb.debian.org/debian trixie-updates InRelease
Hit:3 http://security.debian.org/debian-security trixie-security InRelease
All packages are up to date.
jesus@vbox:~$ sudo apt upgrade -y
Summary:
  Upgrading: 0, Installing: 0, Removing: 0, Not Upgrading: 0
jesus@vbox:~$ sudo apt install -y ca-certificates curl gnupg
ca-certificates is already the newest version (20250419).
gnupg is already the newest version (2.4.7-21).
gnupg set to manually installed.
Installing:
  curl
Summary:
  Upgrading: 0, Installing: 1, Removing: 0, Not Upgrading: 0
  Download size: 270 kB
  Space needed: 506 kB / 13.7 GB available
Get:1 http://deb.debian.org/debian trixie/main amd64 curl amd64 8.14.1-2+deb13u2 [270 kB]
Fetched 270 kB in 0s (1,088 kB/s)
Selecting previously unselected package curl.
(Reading database ... 129941 files and directories currently installed.)
Preparing to unpack .../curl_8.14.1-2+deb13u2_amd64.deb ...
Unpacking curl (8.14.1-2+deb13u2) ...
Setting up curl (8.14.1-2+deb13u2) ...
Processing triggers for man-db (2.13.1-1) ...
jesus@vbox:~$ curl -fsSL https://download.docker.com/linux/debian/gpg
jesus@vbox:~$ sudo gpg --dearmor -o /etc/apt/keyrings/docker.gpg
jesus@vbox:~$ sudo chmod a+r /etc/apt/keyrings/docker.gpg
jesus@vbox:~$
```

2. verificamos certificados, el curl es la herramienta que permite descargar este contenido, el gnupg verifican la autenticidad del repositorio docker



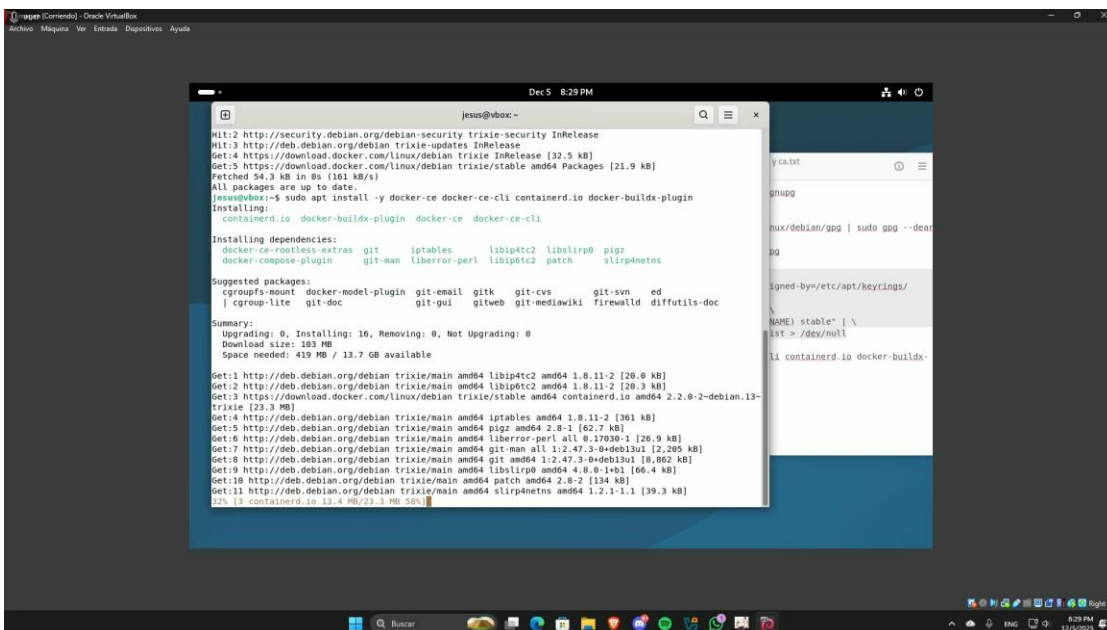
```
jesus@vbox:~$ sudo apt update
[sudo] password for jesus:
Hit:1 http://deb.debian.org/debian trixie InRelease
Hit:2 http://deb.debian.org/debian trixie-updates InRelease
Hit:3 http://security.debian.org/debian-security trixie-security InRelease
All packages are up to date.
jesus@vbox:~$ sudo apt upgrade -y
Summary:
  Upgrading: 0, Installing: 0, Removing: 0, Not Upgrading: 0
jesus@vbox:~$ sudo apt install -y ca-certificates curl gnupg
ca-certificates is already the newest version (20250419).
gnupg is already the newest version (2.4.7-21).
gnupg set to manually installed.
Installing:
  curl
Summary:
  Upgrading: 0, Installing: 1, Removing: 0, Not Upgrading: 0
  Download size: 270 kB
  Space needed: 506 kB / 13.7 GB available
Get:1 http://deb.debian.org/debian trixie/main amd64 curl amd64 8.14.1-2+deb13u2 [270 kB]
Fetched 270 kB in 0s (1,088 kB/s)
Selecting previously unselected package curl.
(Reading database ... 129941 files and directories currently installed.)
Preparing to unpack .../curl_8.14.1-2+deb13u2_amd64.deb ...
Unpacking curl (8.14.1-2+deb13u2) ...
Setting up curl (8.14.1-2+deb13u2) ...
Processing triggers for man-db (2.13.1-1) ...
jesus@vbox:~$ curl -fsSL https://download.docker.com/linux/debian/gpg
jesus@vbox:~$ sudo gpg --dearmor -o /etc/apt/keyrings/docker.gpg
jesus@vbox:~$ sudo chmod a+r /etc/apt/keyrings/docker.gpg
jesus@vbox:~$
```

3. Este bloque de comandos crea la carpeta donde se guardarán las llaves de seguridad de los repositorios (/etc/apt/keyrings), descarga la llave GPG oficial de Docker y la convierte al formato correcto para que el sistema pueda usarla, y finalmente le da permisos de lectura para que apt pueda verificar que los paquetes instalados desde el repositorio de Docker son auténticos y seguros, el echo construye la línea para conectar al Docker para el apt pueda saber dónde buscar los paquetes para actualizar.



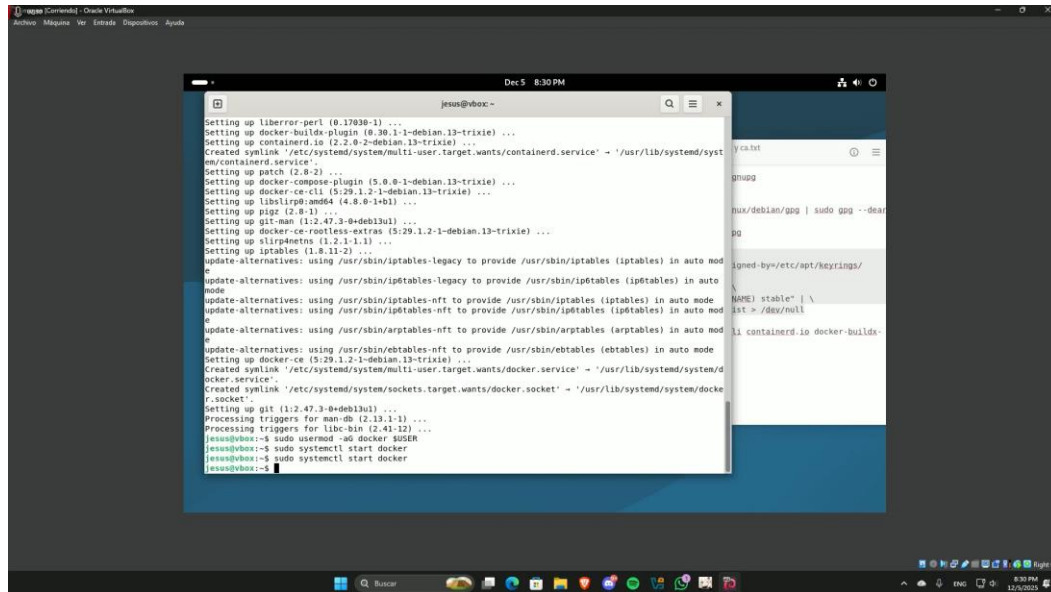
```
jesus@vbox:~$ sudo apt update
Hit:2 http://deb.debian.org/debian trixie-updates InRelease
Hit:3 http://security.debian.org/debian-security trixie-security InRelease
All packages are up to date
jesus@vbox:~$ sudo apt install -y curl
Summary:
  Upgrading: 0, Installing: 1, Removing: 0, Not Upgrading: 0
  Download size: 270 kB
  Space needed: 506 kB / 13.7 GB available
Get:1 http://deb.debian.org/debian trixie/main amd64 curl amd64 8.14.1-2+deb13u2 [270 kB]
Fetched 270 kB in 0s (1,800 kB/s)
Selecting previously unselected package curl.
(Reading database ... 129941 files and directories currently installed.)
Preparing to unpack .../curl_8.14.1-2+deb13u2_amd64.deb ...
Unpacking curl (8.14.1-2+deb13u2) ...
Setting up curl (8.14.1-2+deb13u2) ...
jesus@vbox:~$ sudo install -m 0755 -d /etc/apt/keyrings
curl -fsSL https://download.docker.com/linux/debian/gpg | sudo gpg --dearmor -o /etc/apt/keyrings/docker.gpg
sudo chmod a+r /etc/apt/keyrings/docker.gpg
jesus@vbox:~$ echo \
"deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.gpg] \
https://download.docker.com/linux/debian \
$! /etc/os-release id echo VERSION_CODENAME stable" | \
sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
jesus@vbox:~$
```

4. instalamos el docker engine y sus paquetes relacionados



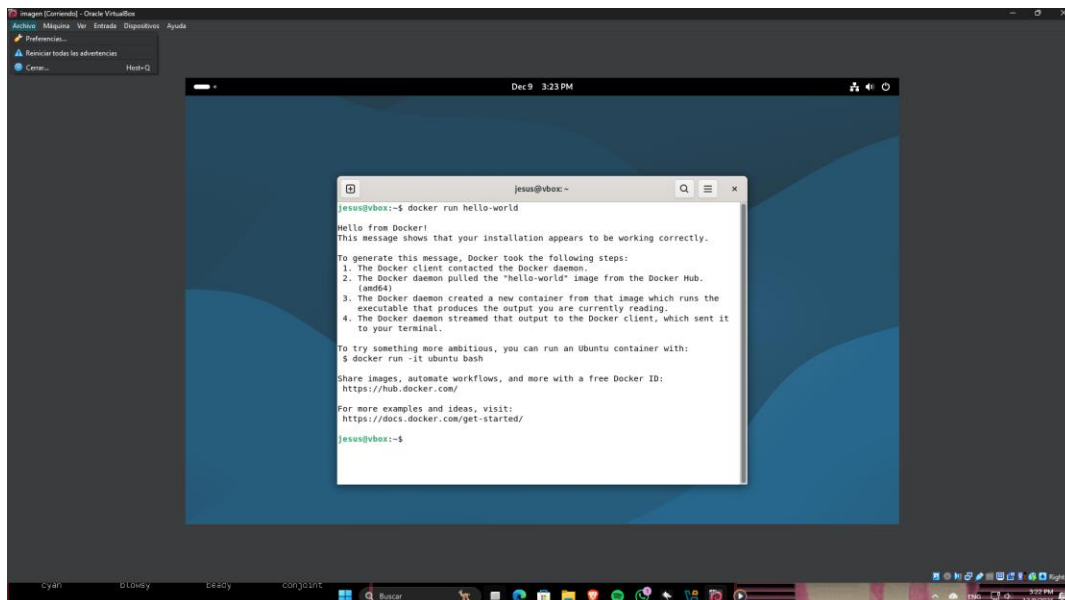
```
jesus@vbox:~$ sudo apt install -y docker-ce docker-ce-cli containerd.io docker-buildx-plugin
Installing:
  containerd.io  docker-buildx-plugin  docker-ce  docker-ce-cli
Installing dependencies:
  docker-ce-rootless-extras  git  iptables  libipset2  liblirp0  pigz
  docker-compose-plugin  git-man  liberror-perl  libipset2  patch  slurp4netns
Suggested packages:
  cgroupfs-mount  docker-model-plugin  git-email  gitk  git-cvs  git-svn  ed
  | cgroup-lite  git-doc  git-gui  gitweb  git-mediawiki  firewallld  diffutils-doc
Summary:
  Upgrading: 0, Installing: 16, Removing: 0, Not Upgrading: 0
  Download size: 183 MB
  Space needed: 419 MB / 13.7 GB available
Get:1 http://deb.debian.org/debian trixie/main amd64 libipset2 amd64 1.8.11-2 [20.0 kB]
Get:2 http://deb.debian.org/debian trixie/main amd64 libipset2 amd64 1.8.11-2 [20.3 kB]
Get:3 http://deb.debian.org/debian trixie/stable amd64 containerd.io amd64 2.2.0-2-debian-13-trixie [23.3 MB]
Get:4 http://deb.debian.org/debian trixie/main amd64 iptables amd64 1.8.11-2 [361 kB]
Get:5 http://deb.debian.org/debian trixie/main amd64 pigz amd64 2.8-1 [62.7 kB]
Get:6 http://deb.debian.org/debian trixie/main amd64 liberror-perl all 0.17030-1 [26.9 kB]
Get:7 http://deb.debian.org/debian trixie/main amd64 git-man all 1:2.47.3-0+deb13u1 [2,295 kB]
Get:8 http://deb.debian.org/debian trixie/main amd64 git amd64 1:2.47.3-0+deb13u1 [8,862 kB]
Get:9 http://deb.debian.org/debian trixie/main amd64 liblirp0 amd64 4.8.0-1+b1 [66.4 kB]
Get:10 http://deb.debian.org/debian trixie/main amd64 patch amd64 2.8-2 [134 kB]
Get:11 http://deb.debian.org/debian trixie/main amd64 slurp4netns amd64 1.2.1-1.1 [39.3 kB]
32% [3 containerd.io 13.4 MB/23.3 MB 58%]
```

5. agregamos al usuario al grupo docker, para no poner sudo cada vez, después habilitamos el servidor docker y iniciamos el Docker



```
jesus@vbox:~$ adduser jesus
Setting up liberror-perl (0.17030-1) ...
Setting up docker-buildx-plugin (0.30.1-1-debian.13-trixie) ...
Setting up containerd.io (1.2.20-2-debian.13-trixie) ...
Created symlink '/etc/systemd/system/multi-user.target.wants/containerd.service' -> '/usr/lib/systemd/system/containerd.service'.
Setting up patch (2.8-2) ...
Setting up docker-compose-plugin (5.0.0-1-debian.13-trixie) ...
Setting up docker-ce-cli (5:29.1.2-1-debian.13-trixie) ...
Setting up libslirp0:amd64 (4.8.0-1+b1) ...
Setting up pigz (2.8-1) ...
Setting up git-man (1:2.47.3-0-deb13u1) ...
Setting up docker-ce-rootless-extras (5:29.1.2-1-debian.13-trixie) ...
Setting up slipstreams (1.2.1-1.1) ...
Setting up iptables (1.8.11-2) ...
update-alternatives: using /usr/sbin/iptables-legacy to provide /usr/sbin/iptables (iptables) in auto mode
update-alternatives: using /usr/sbin/iptables-legacy to provide /usr/sbin/iptables (iptables) in auto mode
update-alternatives: using /usr/sbin/iptables-nft to provide /usr/sbin/iptables (iptables) in auto mode
update-alternatives: using /usr/sbin/iptables-nft to provide /usr/sbin/iptables (iptables) in auto mode
update-alternatives: using /usr/sbin/arptables-nft to provide /usr/sbin/arptables (arptables) in auto mode
update-alternatives: using /usr/sbin/ebtables-nft to provide /usr/sbin/ebtables (ebtables) in auto mode
Setting up docker-ce (5:29.1.2-1-debian.13-trixie) ...
Created symlink '/etc/systemd/system/multi-user.target.wants/docker.service' -> '/usr/lib/systemd/system/docker.service'.
Created symlink '/etc/systemd/system/sockets.target.wants/docker.socket' -> '/usr/lib/systemd/system/docker.socket'.
Setting up git (1:2.47.3-0-deb13u1) ...
Processing triggers for man-db (2.13.1-1) ...
Processing triggers for libc-bin (2.41-12) ...
jesus@vbox:~$ sudo usermod -s0 docker jesus
jesus@vbox:~$ sudo systemctl start docker
jesus@vbox:~$
```

6. descargamos una imagen de prueba



```
jesus@vbox:~$ docker run hello-world

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
1. The Docker client contacted the Docker daemon.
2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
   (and44)
3. The Docker daemon created a new container from that image which runs the
   executable that produces the output you are currently reading.
4. The Docker daemon streamed that output to the Docker client, which sent it
   to your terminal.

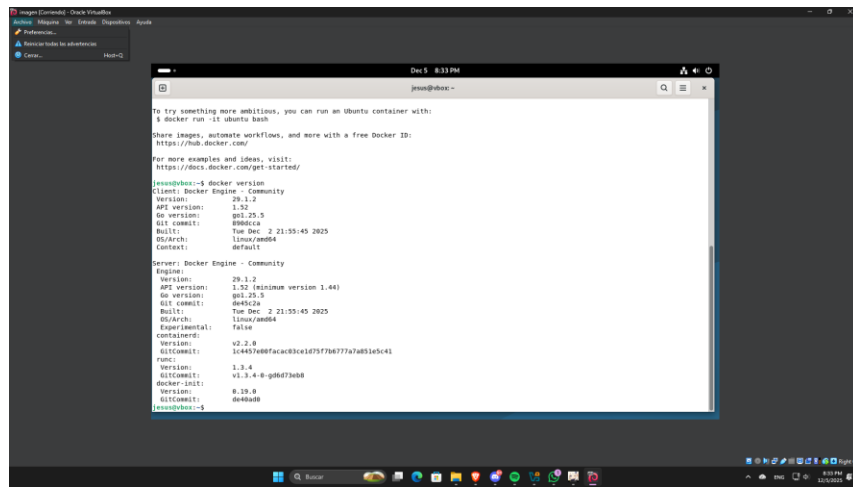
To try something more ambitious, you can run an Ubuntu container with:
$ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
https://hub.docker.com/

For more examples and ideas, visit:
https://docs.docker.com/get-started/

jesus@vbox:~$
```

## 7. Docker info y Docker versión



The screenshot shows a terminal window titled 'jessu@vbox: ~' with the following output from the 'docker version' command:

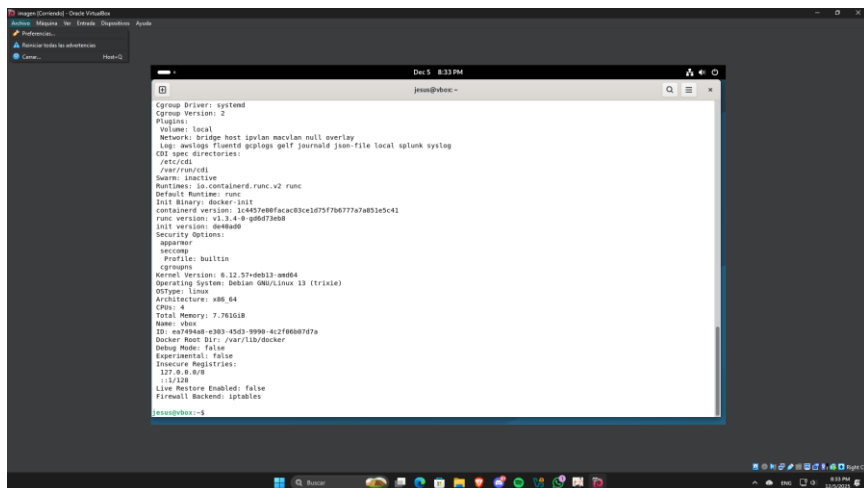
```
To try something more ambitious, you can run an Ubuntu container with:
$ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
https://hub.docker.com/

For more examples and ideas, visit:
https://docs.docker.com/get-started/

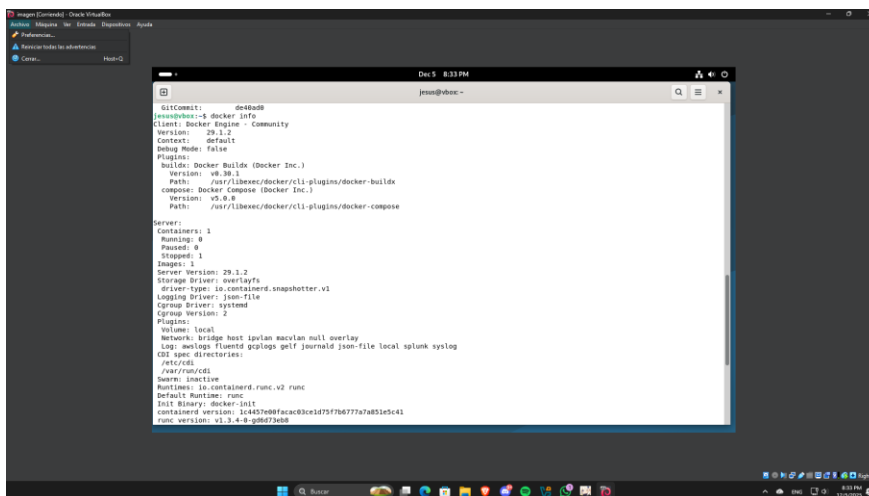
jessu@vbox:~$ docker version
Client: Docker Engine - Community
 Version: 29.1.2
 API version: 1.42
 Go version: go1.25.5
 Git commit: 098f22a
 Built: Tue Dec 2 21:05:45 2025
 OS/Arch: linux/amd64
 Context: default

Server: Docker Engine - Community
 Engine:
  Version: 29.1.2
  API version: 1.42 (minimum version 1.40)
  Go version: go1.25.5
  Git commit: 098f22a
  Built: Tue Dec 2 21:05:45 2025
  OS/Arch: linux/amd64
  Experimental: false
 Containers:
  Version: v2.2.0
  GitCommit: 1c483760fac83c3e3d7577b777a7853e5c41
  Plugins:
  runc:
    Version: 1.3.4
    GitCommit: v1.3.4-0-g06d7360b
  docker-init:
    Version: 0.19.0
    GitCommit: de40a0b
jessu@vbox:~$
```



The screenshot shows a terminal window titled 'jessu@vbox: ~' with the following output from the 'docker info' command:

```
Cgroup Driver: systemd
Cgroup Version: 2
Plugins: local
Network: bridge host ipvlan macvlan null overlay
Log: awslogs fluentd gcplogs gelf journald json-file local splunk syslog
CDI spec directories:
  runcmd
  /var/run/cdi
Swarm: inactive
Runtimes: io.containerd.runc.v2 runc
Default Runtime: runc
Init Binary: docker-init
containerd version: 1c483760fac83c3e3d7577b777a7853e5c41
runc version: v1.3.4-0-g06d7360b
init version: de40a0b
Security Options:
  apparmor
  seccomp
   Profile: builtin
cgroupns:
  cgroupns:
    Kernel Version: 6.12.37-dm33-amd64
    Operating System: Debian GNU/Linux 13 (trixie)
    OSType: Linux
    Architecture: x86_64
    CPUs: 4
    Total Memory: 7.761GiB
Name: vbox
ID: ea7f8ead-e383-45d3-9990-4c2f60807d7a
Docker Root Dir: /var/lib/docker
Debug Mode: false
Experimental: false
Insecure Registries:
  127.0.0.0/8
  ::/0
Live Restore Enabled: false
Firewall: Docker iptables
jessu@vbox:~$
```



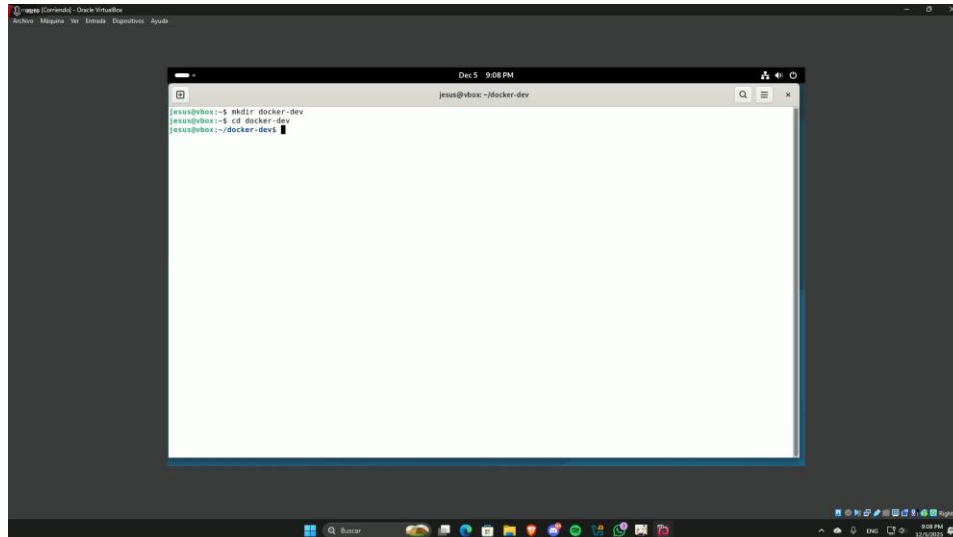
The screenshot shows a terminal window titled 'jessu@vbox: ~' with the following output from the 'docker info' command:

```
GitCommit: de40a0b
jessu@vbox:~$ docker info
Client: Docker Engine - Community
 Version: 29.1.2
 Context: default
 Debug Mode: false
 Plugins:
  buildx: Docker Buildx (Docker Inc.)
    Version: v0.39.1
    Path: /usr/libexec/docker/cli-plugins/docker-buildx
  compose: Docker Compose (Docker Inc.)
    Version: v5.0.8
    Path: /usr/libexec/docker/cli-plugins/docker-compose

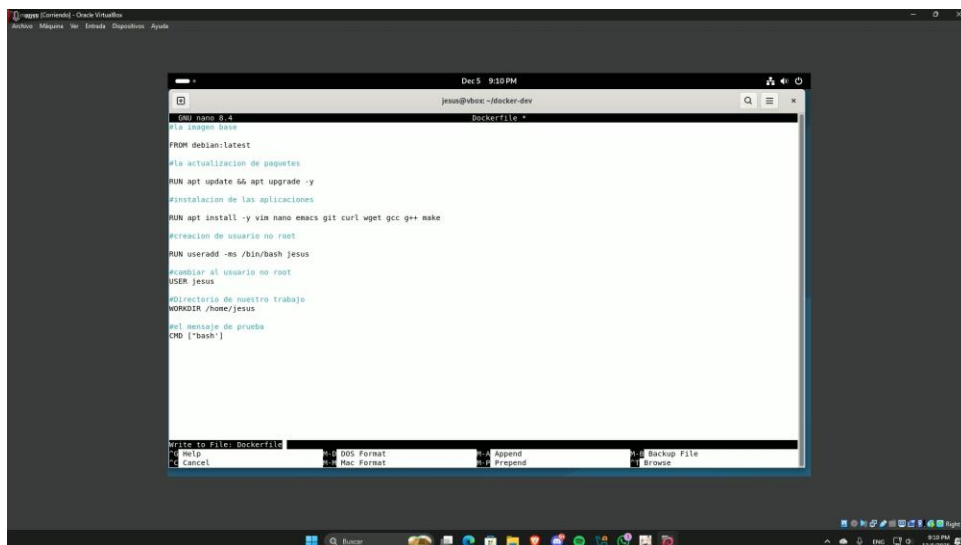
Server:
 Containers: 1
  Running: 0
  Paused: 0
  Stopped: 1
 Images: 1
 Server Version: 29.1.2
 Storage Driver: overlayfs
 driver-type: io.containerd.snapshotter.v1
 Logging Driver: json-file
 Cgroup Driver: systemd
 Cgroup Version: 2
 Plugins:
  Volume: local
 Network: bridge host ipvlan macvlan null overlay
 Log: awslogs fluentd gcplogs gelf journald json-file local splunk syslog
 CDI spec directories:
  runcmd
  /var/run/cdi
Swarm: inactive
Runtimes: io.containerd.runc.v2 runc
Default Runtime: runc
Init Binary: docker-init
containerd version: 1c483760fac83c3e3d7577b777a7853e5c41
runc version: v1.3.4-0-g06d7360b
jessu@vbox:~$
```

## 4.2 CREACION DE IMAGEN DOCKER

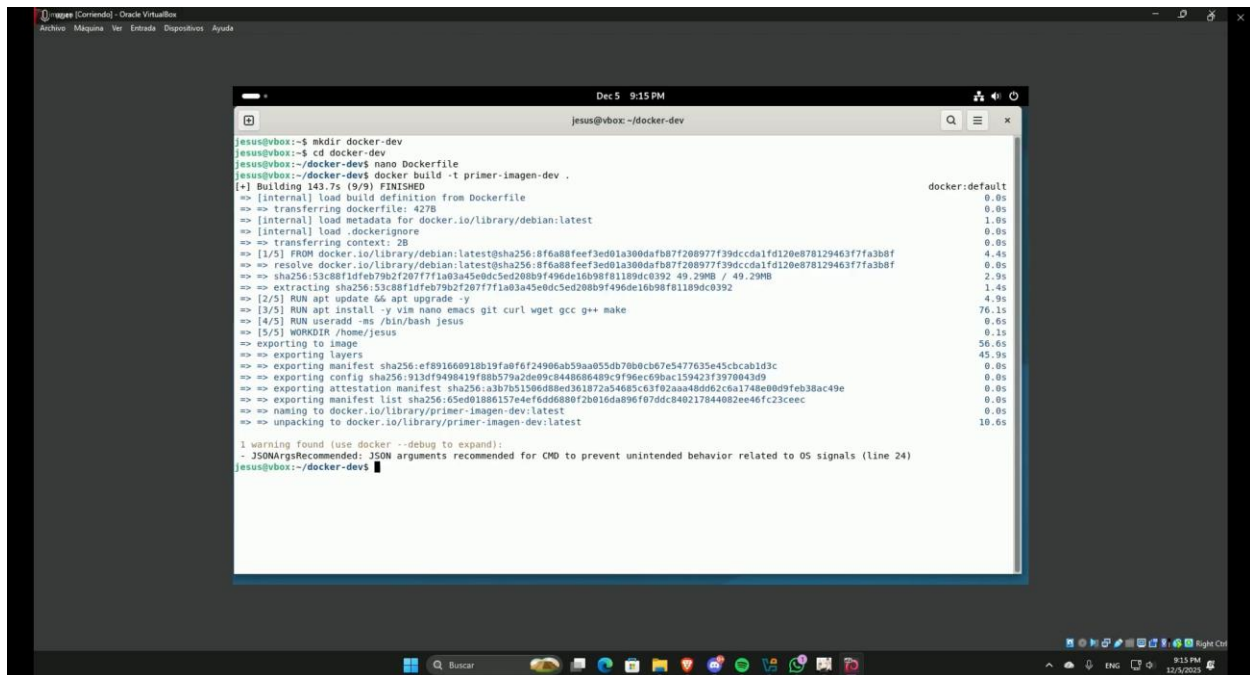
### 1. Creación de la carpeta donde se almacenará la imagen creada



2. Configuración del dockerfile, en la primera linea de FROM debian, nos indica la imagen base donde trabajaremos el dockerfile, la segunda RUN apt update && apt upgrade -y nos ayuda a que los paquetes esten actualizados en cada momento que se abra el dockerfile, RUN apt install nos descarga las aplicaciones de desarrollo, RUN useradd crea el usuario no root, con el shell /bin/bash y crea su carpeta home -ms, el USER jesus nos ayuda a que las instrucciones se ejecuten como usuario jesus, para mayor seguridad, el WORKDIR establece el directorio donde se trabajara y el CMD es el comando por defecto que se ejecuta al abrir el contenedor.



3. con `docker build -t` construimos la imagen y el punto al final indica que esta en la carpeta actual, el mensaje de advertencia nos indica que varias opciones mas recientes no estaran disponibles, porque estamos usando tipo `string ["bash"]`, es una recomendacion de sistema, pero varias de las opciones que necesitamos si se permiten en este formato.



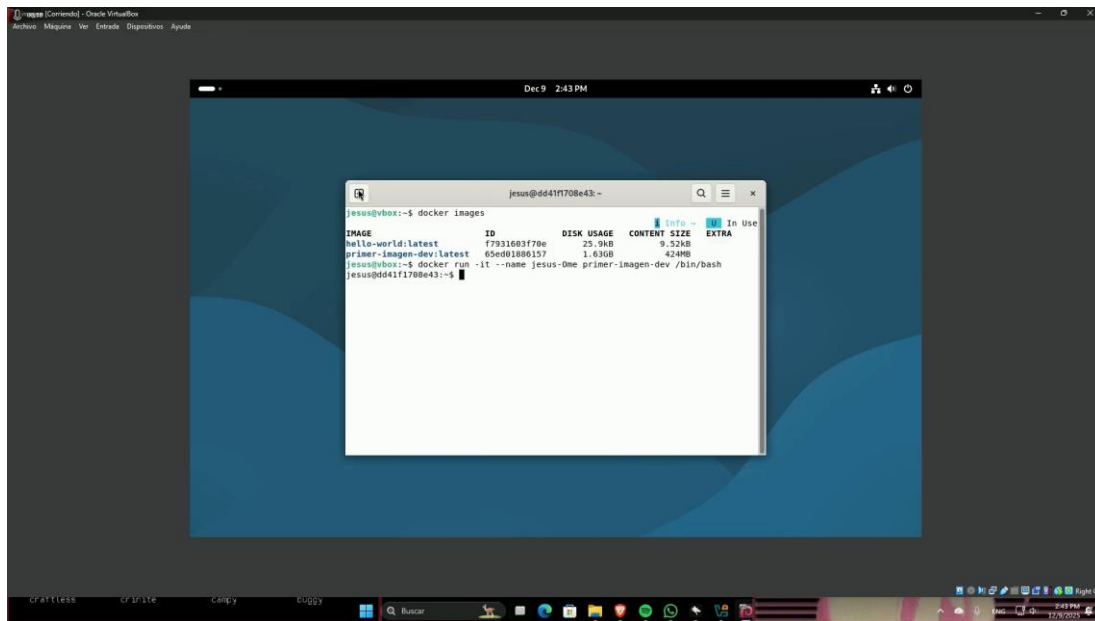
```
jesus@vbox:~$ mkdir docker-dev
jesus@vbox:~$ cd docker-dev
jesus@vbox:~/docker-dev$ nano Dockerfile
jesus@vbox:~/docker-dev$ docker build -t primer-imagen-dev .
[*] Building 143.7s (9/9) FINISHED

=> [internal] load build definition from Dockerfile
=> == transferring dockerfile: 427B
=> [internal] load metadata for docker.io/library/debian:latest
=> [internal] load .dockerignore
=> == transferring context: 2B
=> [1/5] FROM docker.io/library/debian:latest@sha256:8f6a88feef3ed01a380dafb87f208977f39dcdaf120e87812946377fa3b8f
=> == resolve docker.io/library/debian:latest@sha256:8f6a88feef3ed01a380dafb87f208977f39dcdaf120e87812946377fa3b8f
=> == sha256:53c88f1dfb79b2f2077f1a03a45e0dc5ed208b9f496de16b98f81189dc0392 49.29MB / 49.29MB
=> == extracting sha256:53c88f1dfb79b2f2077f1a03a45e0dc5ed208b9f496de16b98f81189dc0392
=> [2/5] RUN apt update && apt upgrade -y
=> [3/5] RUN apt install -y vim nano emacs git curl wget gcc g++ make
=> [4/5] RUN useradd -ms /bin/bash jesus
=> [5/5] WORKDIR /home/jesus
=> exporting image
=> == exporting layers
=> == exporting manifest sha256:ef89166a918b19fa0f6f24906ab59aa055db78b0cb67e5477635e45cabcab1d3c
=> == exporting config sha256:913df9498419f88b579a2de09c8448686489c9f96ecc9bac159423f3970043d9
=> == exporting attestation manifest sha256:a3b705156ed88ed361872a54685c03f02aaa48d602c6a1748e00d9feb38ac49e
=> == exporting manifest list sha256:65ed01886157e4ef6d6880f2b016da896f67ddc840217844082ee46fc23ceec
=> == naming to docker.io/library/primer-imagen-dev:latest
=> == unpacking to docker.io/library/primer-imagen-dev:latest

1 warning found (use docker --debug to expand):
- JSOArgsRecommended: JSON arguments recommended for CMD to prevent unintended behavior related to OS signals (line 24)
jesus@vbox:~/docker-dev$
```

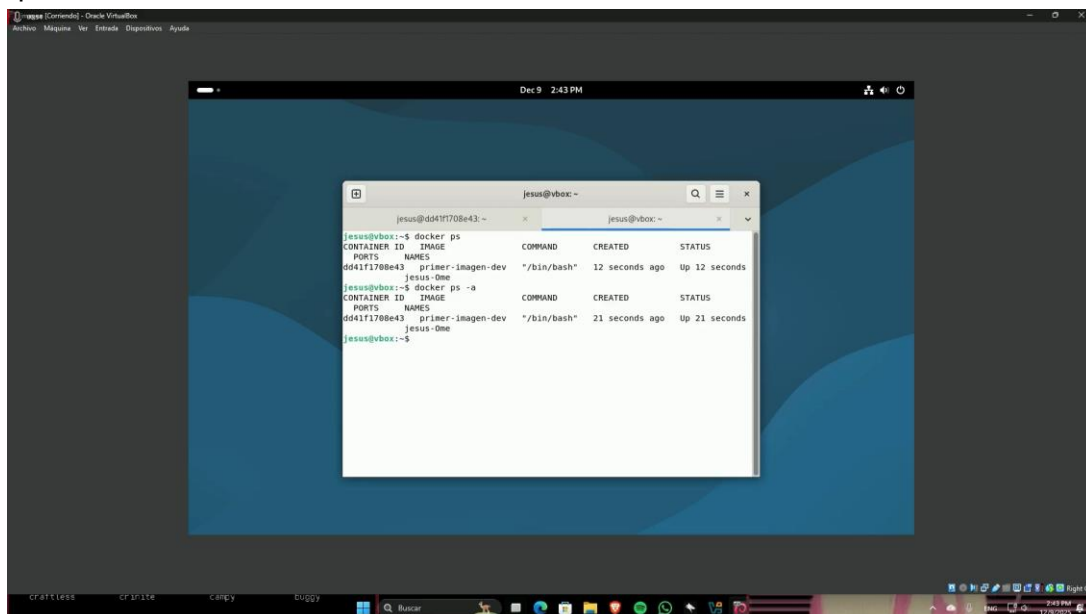
## 4.3 CREACION Y GESTION DE CONTENEDOR

1. hacemos listado de imagenes que tenemos disponible y seleccionamos la que vamos a trabajar en modo iterativo el contenedor, le asignamos el nombre, el nombre la imagen, con /bin/bash asignamos la ruta, y como se muestra ya estamos dentro



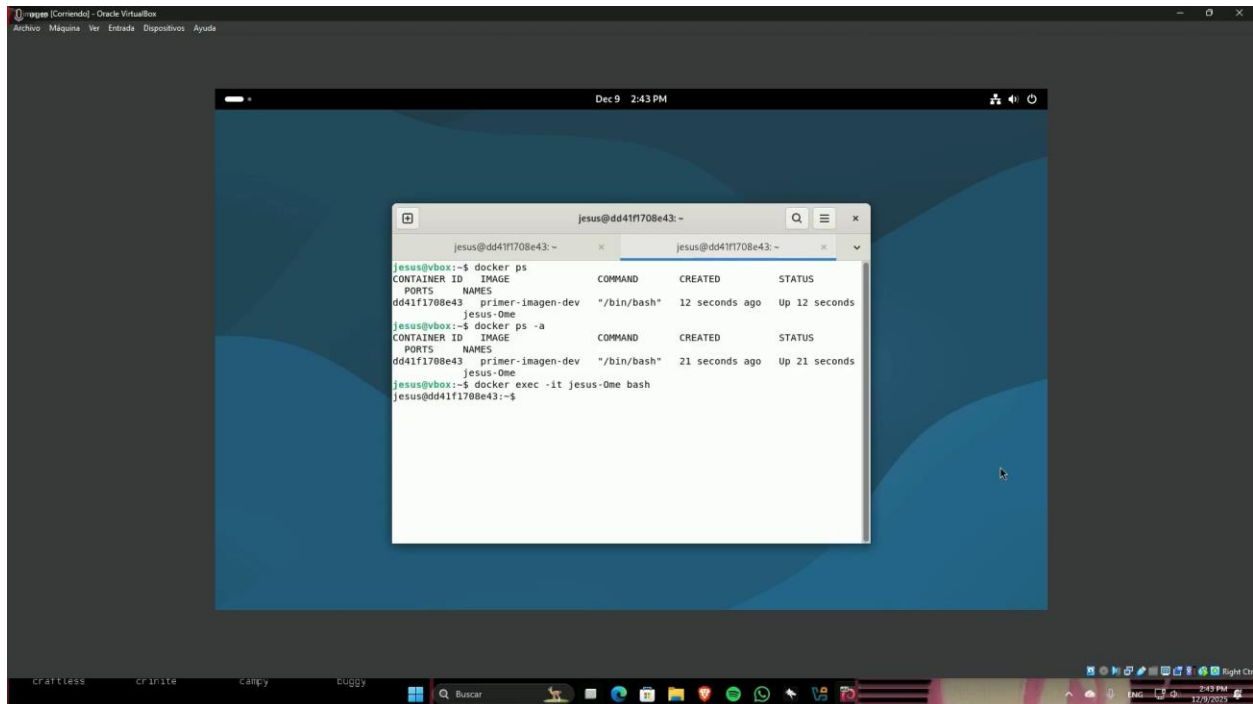
```
jesus@vbox:~$ docker images
IMAGE                ID                                DISK USAGE  CONTENT SIZE  EXTRA
hello-world:latest   f7931603f79e                     25.9kB      9.52kB
primer-imagen-dev:latest  65ed01886157                     1.63GB      424MB
jesus@dd41f1708e43:~$
```

2. con docker ps vemos los contenedores activos y con ps -a vemos todos los contenedores que se encuentran



```
jesus@vbox:~$ docker ps
CONTAINER ID   IMAGE                COMMAND                  CREATED        STATUS
dd41f1708e43   primer-imagen-dev   "/bin/bash"             12 seconds ago Up 12 seconds
jesus-one
jesus@vbox:~$ docker ps -a
CONTAINER ID   IMAGE                COMMAND                  CREATED        STATUS
dd41f1708e43   primer-imagen-dev   "/bin/bash"             21 seconds ago Up 21 seconds
jesus-one
jesus@vbox:~$
```

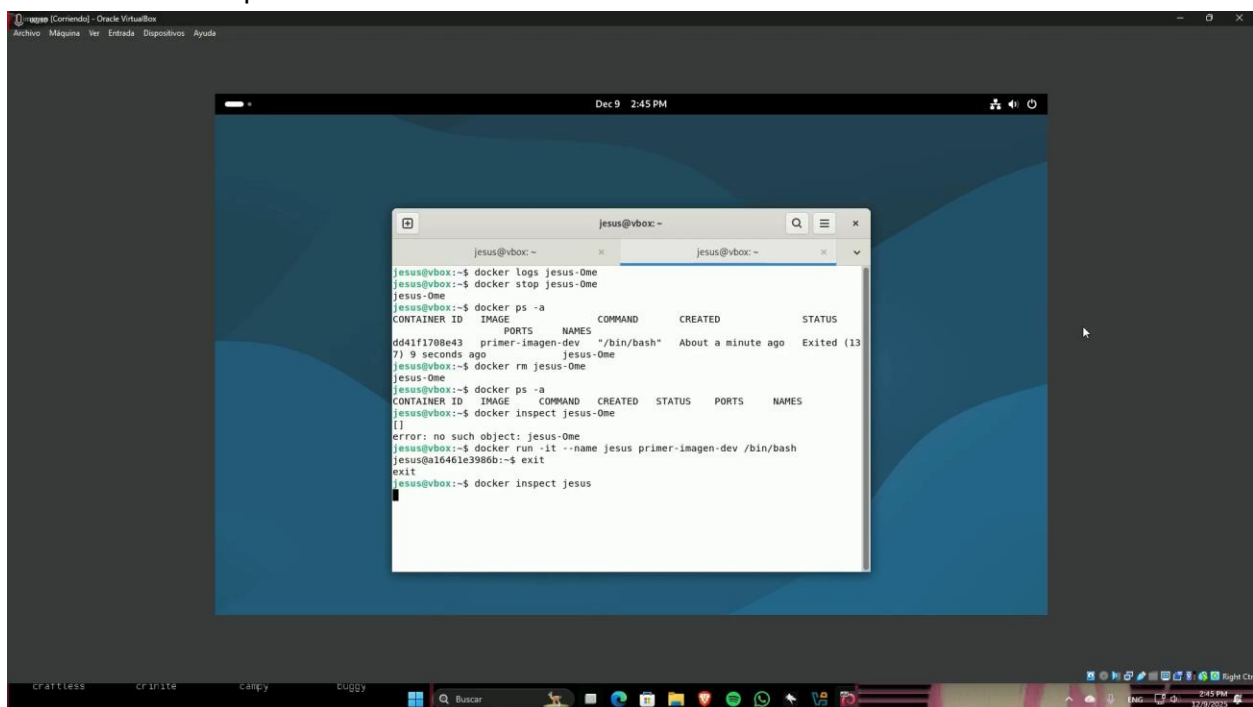
3. con docker exec entramos al contenedor, indicando que es iterativo y el bash al final indicar que es de tipo string, que es donde estamos trabajando



The screenshot shows a terminal window titled 'jesus@dd41f1708e43: ~' with the following commands and output:

```
jesus@vbox:~$ docker ps
CONTAINER ID   IMAGE      COMMAND                  CREATED        STATUS
dd41f1708e43   primer-imagen-dev   "/bin/bash"        12 seconds ago   Up 12 seconds
jesus@vbox:~$ docker ps -a
CONTAINER ID   IMAGE      COMMAND                  CREATED        STATUS
dd41f1708e43   primer-imagen-dev   "/bin/bash"        21 seconds ago   Up 21 seconds
jesus@vbox:~$ docker exec -it jesus-0me bash
jesus@dd41f1708e43:~$
```

4. con docker inspect vemos toda informacion mas detallada del contenedor



The screenshot shows a terminal window titled 'jesus@vbox: ~' with the following commands and output:

```
jesus@vbox:~$ docker logs jesus-0me
jesus@vbox:~$ docker stop jesus-0me
jesus@vbox:~$ docker ps -a
CONTAINER ID   IMAGE      COMMAND                  CREATED        STATUS
dd41f1708e43   primer-imagen-dev   "/bin/bash"        About a minute ago   Exited (13
7) 9 seconds ago
jesus@vbox:~$ docker rm jesus-0me
jesus@vbox:~$ docker ps -a
CONTAINER ID   IMAGE      COMMAND                  CREATED        STATUS        PORTS      NAMES
[]
jesus@vbox:~$ docker inspect jesus-0me
[]
error: no such object: jesus-0me
jesus@vbox:~$ docker run -it --name jesus primer-imagen-dev /bin/bash
jesus@a10401e3986b:~$ exit
exit
jesus@vbox:~$ docker inspect jesus
```