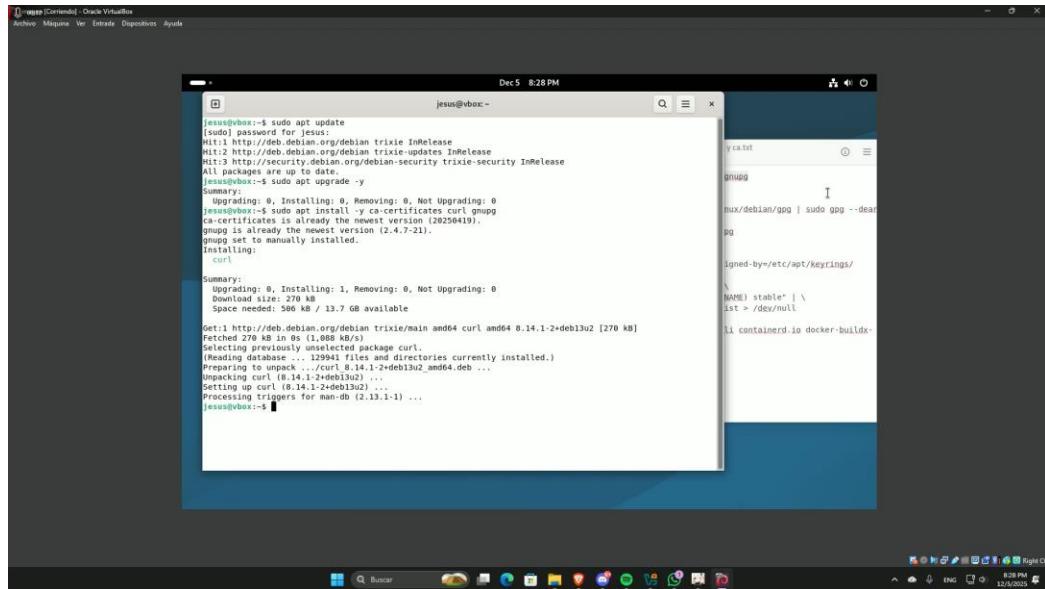


DOCUMENTO ACTIVIDAD N.4

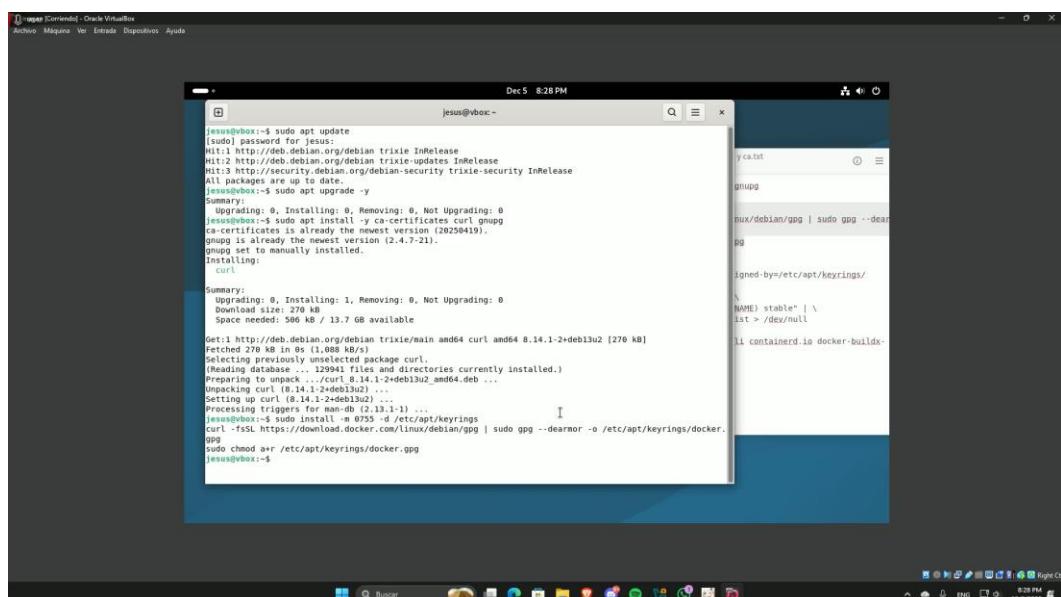
A continuación, se anexan las capturas con sus respectivas explicaciones

4.1 INSTALACION DOCKER ENGINE EN DEBIAN:

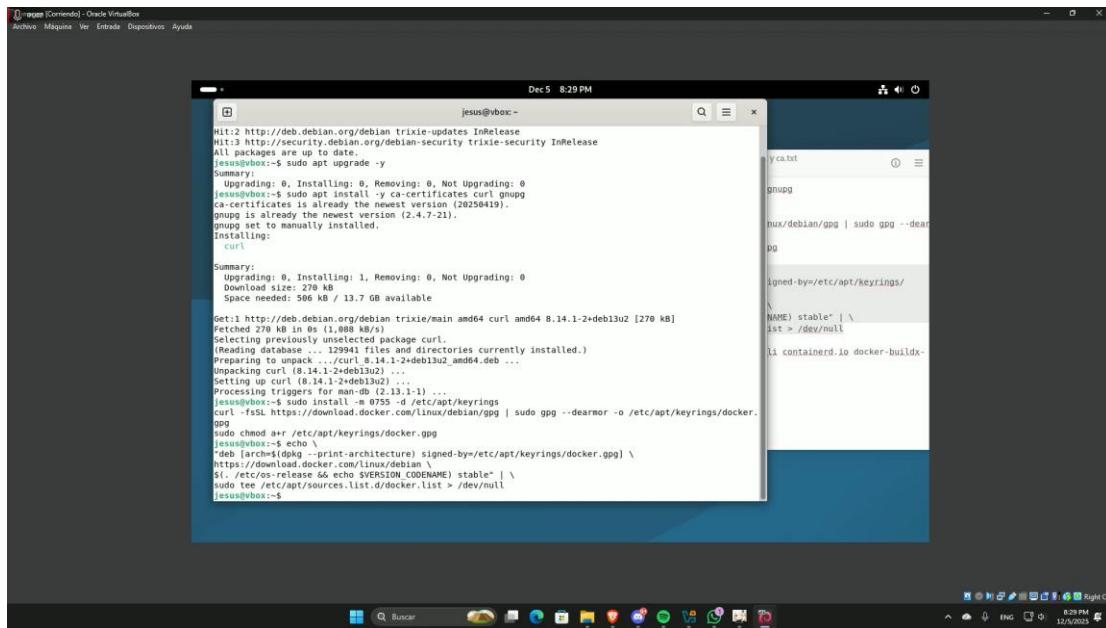
1. actualizamos paquetes y versiones, también instalamos las ultimas actualizaciones disponibles



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



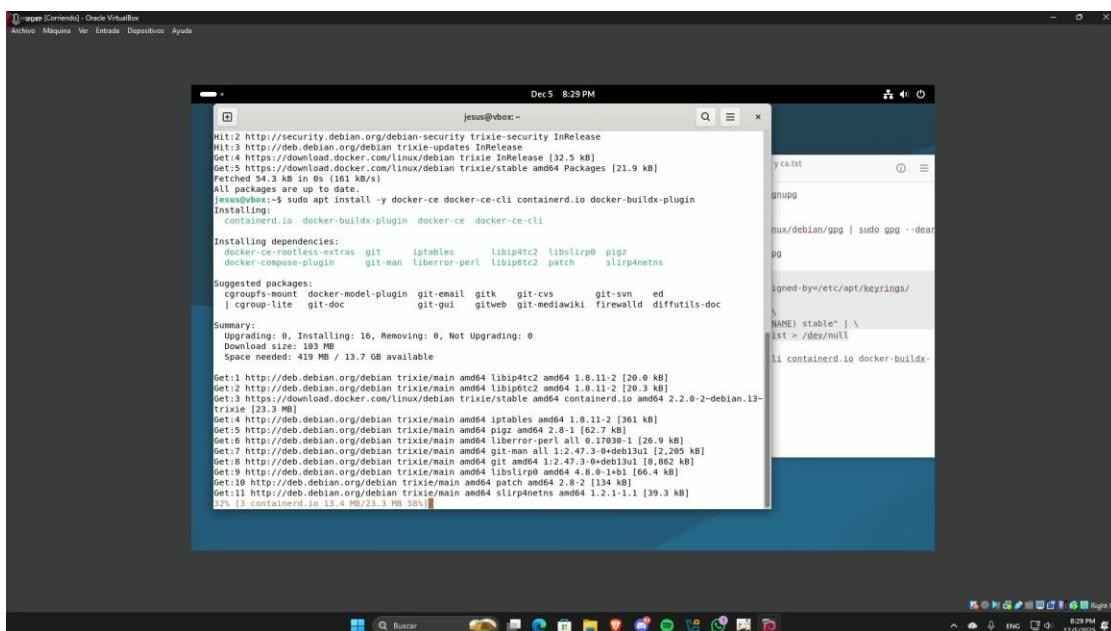
3. Este bloque de comandos crea la carpeta donde se guardarán las llaves de seguridad de los repositorios (/etc/apt/keyrings), descarga la llave GPG oficial de Docker y la convierte al formato correcto para que el sistema pueda usarla, y finalmente le da permisos de lectura para que apt pueda verificar 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:~]
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
  jessie-backports:1:1.0.1-1+deb9u1 all [278 kB] 
  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) ...
  jessie-backports:1:1.0.1-1+deb9u1 all [278 kB] 
  Processing triggers for man-db (2.13.1-1) ...
  curl:amd64 curl / 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=amd64] https://download.docker.com/linux/debian $VERSION_CODENAME stable' > /etc/apt/sources.list.d/docker.list
[jesus@vbox:~]$
```

4. instalamos el docker engine y sus paquetes relacionados



```

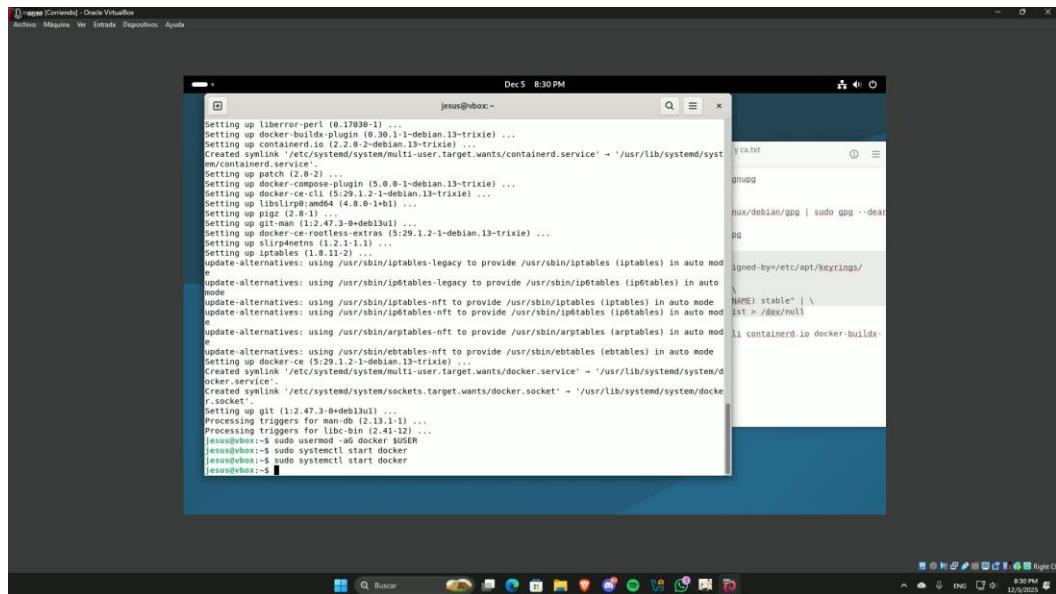
[jesus@vbox:~]
Hit:1 http://security.debian.org/debian-security trixie-security InRelease
Hit:2 http://deb.debian.org/debian trixie-updates InRelease
Get:3 https://download.docker.com/linux/debian trixie InRelease [32.5 kB]
Get:5 https://download.docker.com/linux/debian trixie/stable amd64 Packages [21.9 kB]
All packages are up to date.
[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-cl
  docker-ce-rootless-extras git iptables libipset2 libslirp0 pigz
  docker-compose-plugin git-man liberror-perl libipset2 patch slirp4netns

Suggested packages:
  cgroupsfs-mount docker-model-plugin git-email gitk git-cvs git-svn ed
  | cgroupfs-lite git-doc git-gui gitweb git-mediawiki firewalld diffutils-doc

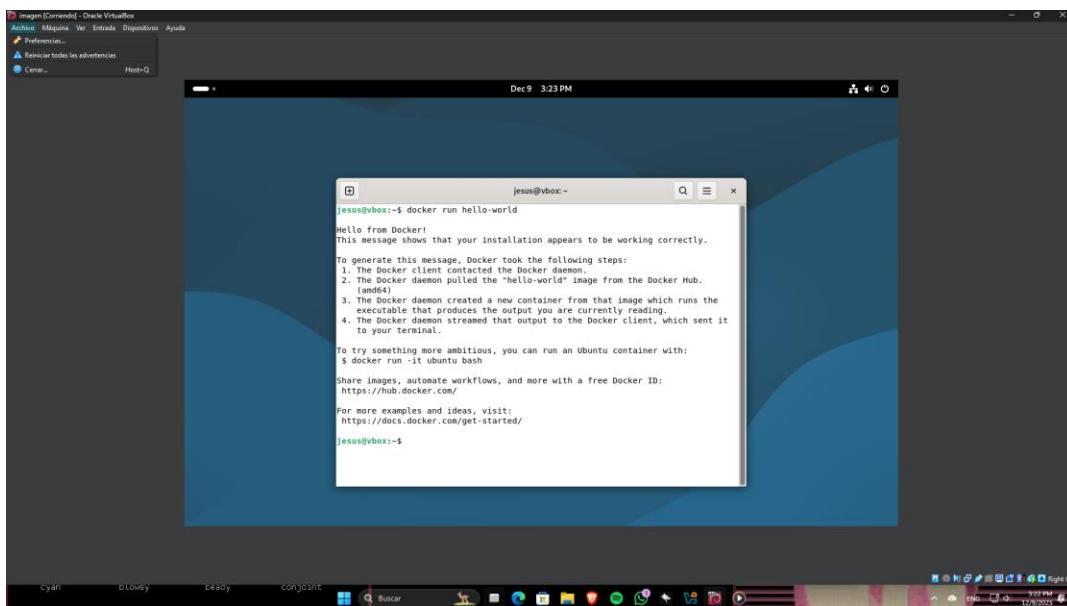
Summary:
  Upgrading: 0, Installing: 16, Removing: 0, Not Upgrading: 0
  Download size: 183 MB
  Space needed: 419 MB / 13.7 GB available

Get:1 http://security.debian.org/debian-trixie/main amd64 libipset2 amd64 1:0.11-2 [20.6 kB]
Get:2 http://deb.debian.org/debian-trixie/main amd64 libipset2 amd64 1:0.11-2 [20.6 kB]
Get:3 https://download.docker.com/linux/debian/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 libslirp0 amd64 1:0.11-1 [361 kB]
Get:5 http://deb.debian.org/debian-trixie/main amd64 patch amd64 2.8-2 [16.4 kB]
Get:6 http://deb.debian.org/debian-trixie/main amd64 liberror-perl all 0.17.89-1 [26.9 kB]
Get:7 http://deb.debian.org/debian-trixie/main amd64 git-man all 1:2.47.3-6+deb13u1 [26.9 kB]
Get:8 http://deb.debian.org/debian-trixie/main amd64 git amd64 1:2.47.3-6+deb13u1 [8,862 kB]
Get:9 http://deb.debian.org/debian-trixie/main amd64 slirp4netns amd64 1:2.1.1-1 [166.4 kB]
Get:10 http://deb.debian.org/debian-trixie/main amd64 patch amd64 2.8-2 [13.8 kB]
Get:11 http://deb.debian.org/debian-trixie/main amd64 slirp4netns amd64 1:2.1.1-1 [39.3 kB]
32% 13 containerd.io 123.9 MB/23.3 MB 595
```

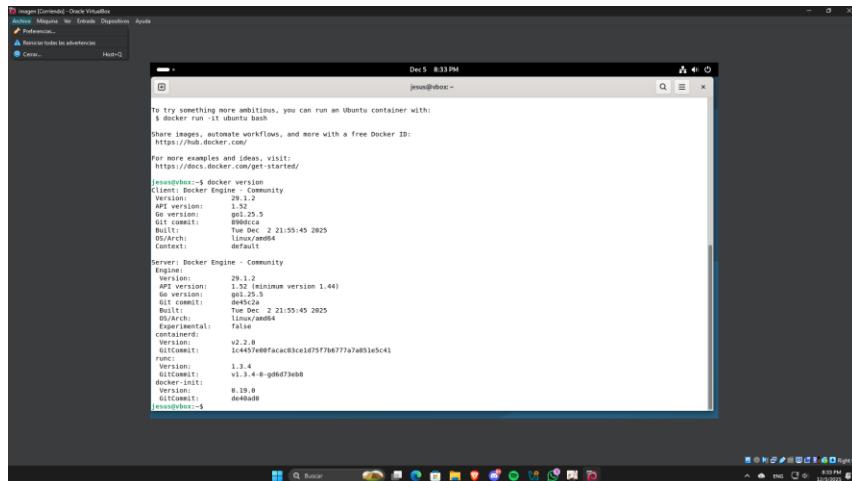
5. agregamos al usuario al grupo docker, para no poner sudo cada vez, después habilitamos el servidor docker y iniciamos el Docker



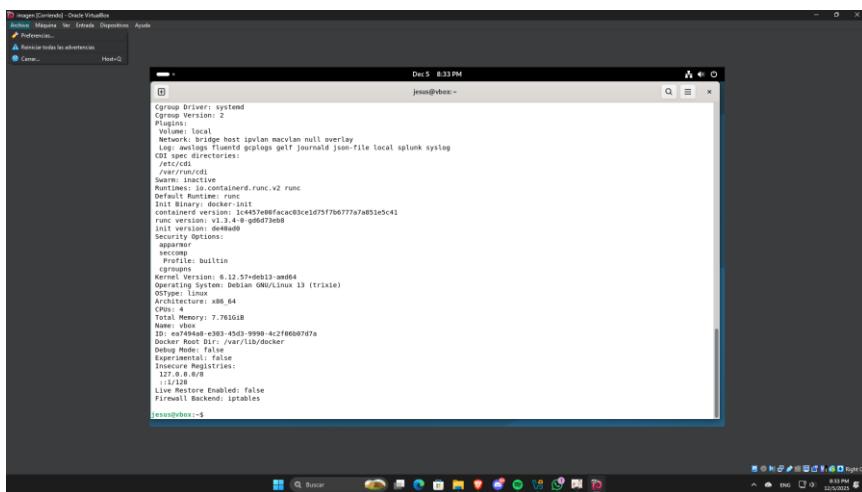
6. descargamos una imagen de prueba



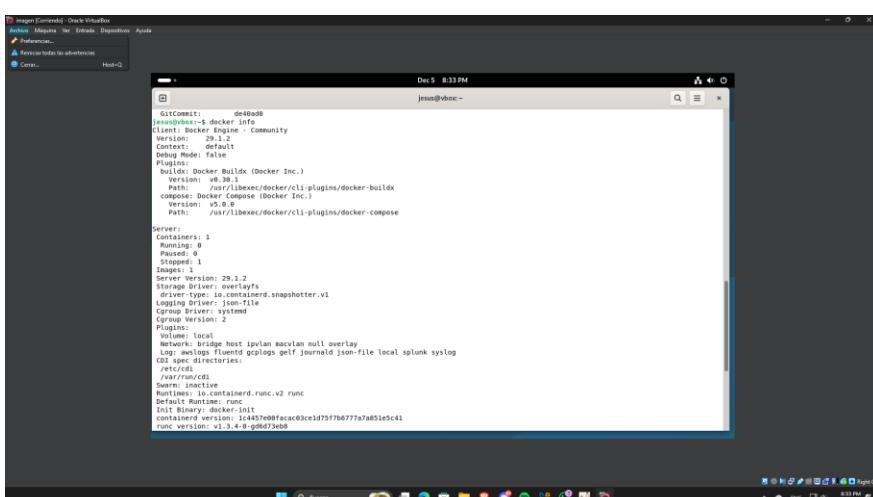
7. Docker info y Docker versión



```
Dec 5 8:33PM jesus@vbox:~$  
To try something more ambitious, you can run an Ubuntu container with:  
$ docker run -it ubuntu bash  
You can also run automatic workflows, and more with a free Docker ID:  
https://hub.docker.com/  
For more examples and ideas, visit:  
https://docs.docker.com/get-started/  
jessievbox:~$ docker version  
Client: Docker Engine - Community  
Version: 29.1.2  
API version: 1.25  
Go version: go1.25.5  
Git commit: 9806ca  
Build time: Tue Dec 2 21:55:45 2025  
OS/Arch: linux/amd64  
Experimental: false  
Server: Docker Engine - Community  
Version: 29.1.2  
API version: 1.25 (minimum version 1.44)  
Go version: go1.25.5  
Git commit: d467a5  
Build time: Tue Dec 2 21:55:45 2025  
OS/Arch: linux/amd64  
Experimental: false  
containers:  
Version: 29.1.0  
GitCommit: 1c4457e0facac03ce1d75f7b6777a7a851e5c41  
runc:  
Version: 1.3.4  
GitCommit: v1.3.4-0-gd6d73eb8  
dumb-init:  
Version: 0.19.0  
GitCommit: 4a649d8  
jessievbox:~$
```



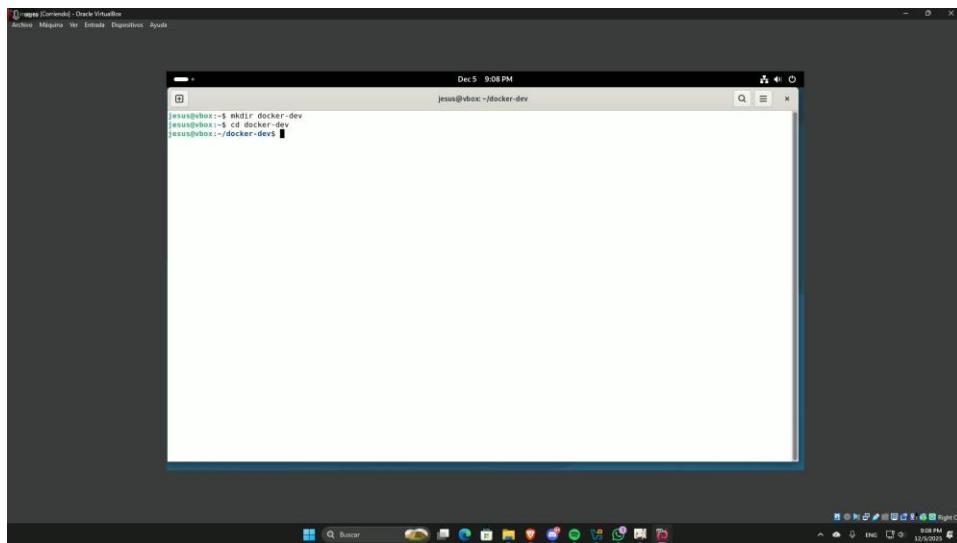
```
Dec 5 8:33PM jesus@vbox:~$  
Cgroup Driver: systemd  
Cgroup Version: 2  
Plugins:  
nvidia local  
Networking: bridge host ipvlan macvlan null overlay  
Log: fluentd gelf logs gelf journald json-file local syslog syslog  
OCI spec directories:  
/etc/docker  
/var/run/docker  
wants: initrd  
Mount Points: /var/lib/docker  
Default Runtime: runc  
Init Binary: docker-init  
Containerd version: 1c4457e0facac03ce1d75f7b6777a7a851e5c41  
Runc version: v1.3.4-0-gd6d73eb8  
Init version: dedead0  
Security Options:  
seccomp  
apparmor  
seccomp  
Filesystem: built-in  
cgroups: native  
Kernel Version: 6.12.9+deb13-amd64  
Operating System: debian GNU/Linux 13 (trixie)  
OSType: linux  
Architecture: amd64  
CPUs: 4  
Total Memory: 7.76GiB  
Name: vbox  
ID: 4591-e981-45d1-9998-4c2f406697d8  
Docker Root Dir: /var/lib/docker  
Debug Mode: False  
Experimental: False  
Insecure Registries:  
127.0.0.0/8  
:/var/run/docker.sock  
Live Restore Enabled: False  
Firewall Backend: iptables  
jessievbox:~$
```



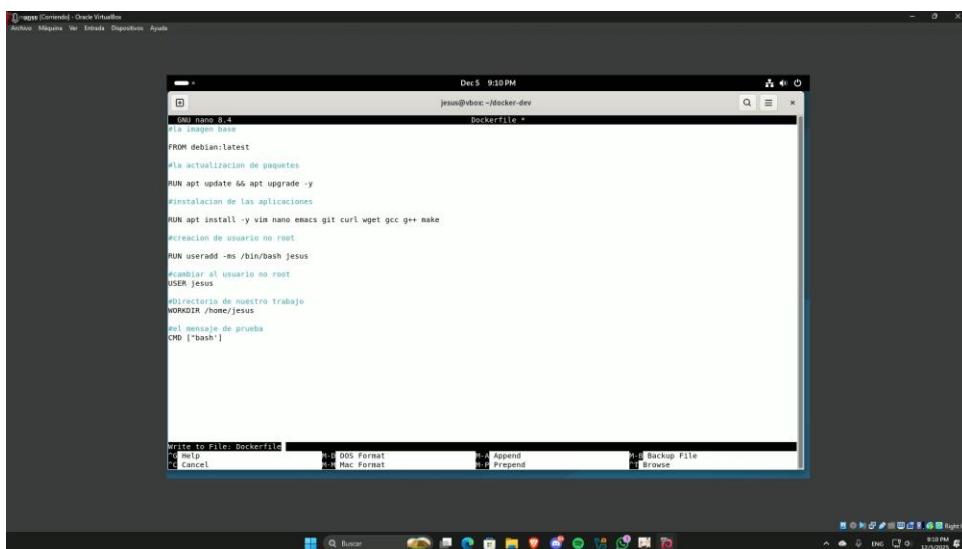
```
Dec 5 8:33PM jesus@vbox:~$  
GitCommit: de4bad8  
Version: 29.1.2  
Client: Docker Engine - Community  
Version: 29.1.2  
API version: 1.25  
Go version: go1.25.5  
Build time: Tue Dec 2 21:55:45 2025  
OS/Arch: linux/amd64  
Experimental: false  
Plugins:  
buildx: Docker Buildx (Docker Inc.)  
Version: 1.12.0  
Path: /usr/libexec/docker/cli-plugins/docker-buildx  
compose: Docker Compose (Docker Inc.)  
Version: 2.13.0  
Path: /usr/libexec/docker/cli-plugins/docker-compose  
Server:  
Containers: 1  
Running: 0  
Paused: 0  
Images: 1  
Images Total: 1  
Server Version: 29.1.2  
Storage Driver: overlayfs  
  Driver Type: in-containerd.snapshottor.v1  
  Supports d_type: true  
Cgroup Driver: systemd  
Cgroup Version: 2  
Plugins:  
  nvidia  
  Networking: bridge host ipvlan macvlan null overlay  
  Log: fluentd gelf logs gelf journald json-file local syslog syslog  
  OCI spec directories:  
  /etc/docker  
  /var/run/docker  
  wants: initrd  
Mount Points: /var/lib/docker  
  Runtimes: in-containerd.runc.v2 runc  
  Default Runtime: runc  
  Init Binary: docker-init  
  Containerd version: 1c4457e0facac03ce1d75f7b6777a7a851e5c41  
  Runc version: v1.3.4-0-gd6d73eb8  
jessievbox:~$
```

4.2 CREACION DE IMAGEN DOCKER

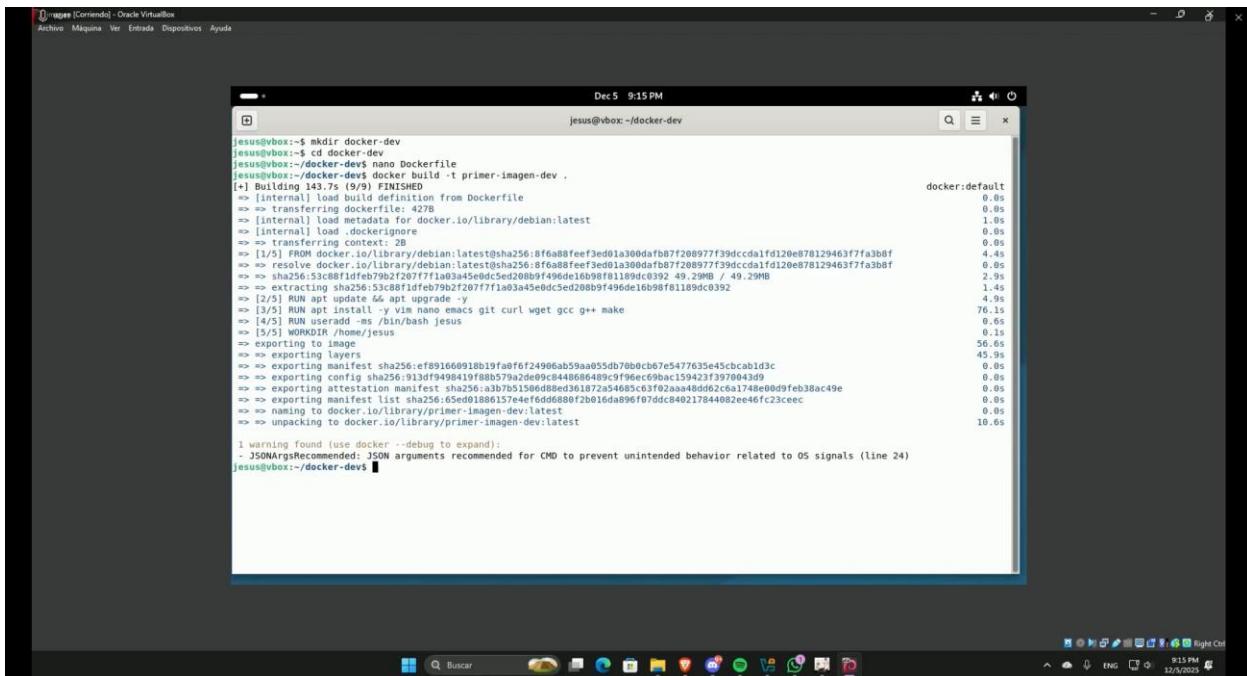
1. Creación de la carpeta donde se almacenará la imagen creada



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.



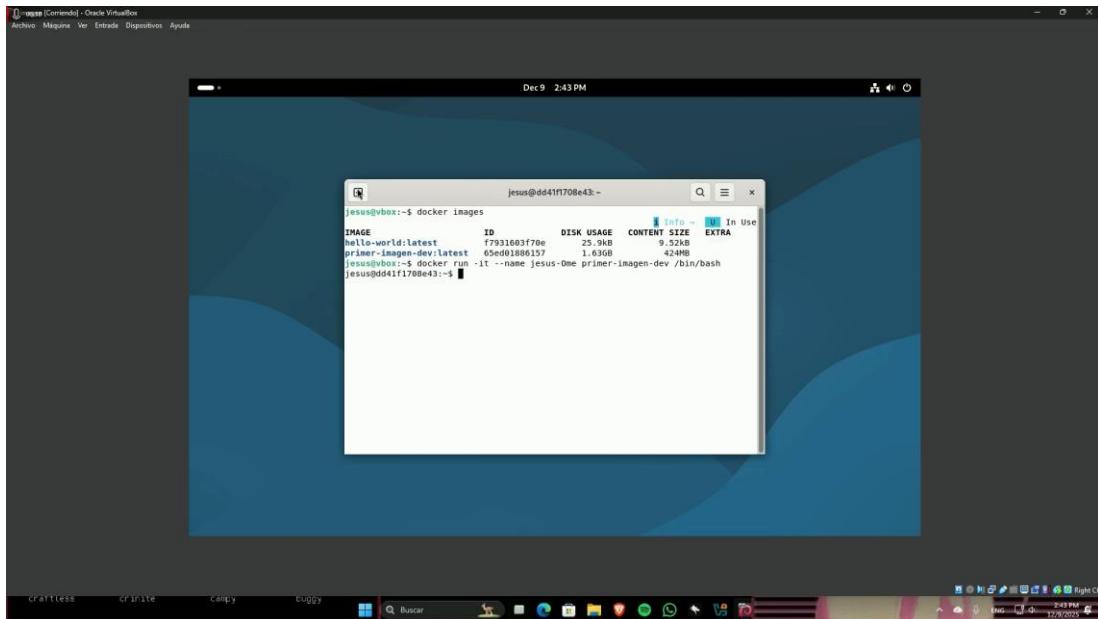
The screenshot shows a terminal window titled "jesus@vbox: ~/docker-dev" running on a Linux desktop environment. The terminal displays the following command and its output:

```
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 .dockerignore
   => [internal] transfer Dockerfile: 42B
   => [internal] load metadata for docker.io/library/debian:latest
   => [internal] load .dockercfg
   => [internal] transfer context: 2B
   => [internal] resolve docker.io/library/debian:latest@sha256:8f6aa8ffef3ed01a300daffb7f20897f39dcda1fd120e870129463fffa3b8f
   => sha256:53c8ff1dfeb79b2f207f71a03a45e0d5ed20809f1496de1609ff01189dc0392 49.29MB / 49.29MB
   => sha256:53c8ff1dfeb79b2f207f71a03a45e0d5ed20809f1496de1609ff01189dc0392
   => extracting sha256:53c8ff1dfeb79b2f207f71a03a45e0d5ed20809f1496de1609ff01189dc0392
[2/5] RUN apt update && apt upgrade -y
   => [internal] export image: vim name emacs git curl wget gcc g++ make
   => [4/5] WORKDIR /home/jesus
   => [5/5] NOMECKER /home/jesus
   => exporting to image
   => exporting layers
   => sha256:ef891666910b19fa8f6f24906ab59aa055db70b0cb67e5477635e45ccb1d3c
   => exporting config sha256:913df9498419ff8b579a2de099c8446864949cf96ec09bac15942313970843d9
   => exporting attestation manifest sha256:a3b7b51596d88bed361b7a54685c63f02aaa48dd62c6a1748e0d9feb38ac49e
   => exporting manifest list sha256:65e0d1886157e4ef6dd6880f2016da896f07ddc840217844082ee46fc23ceec
   => 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):
- JSONArgsRecommended: JSON arguments recommended for CMD to prevent unintended behavior related to OS signals (line 24)
jesus@vbox:~/docker-dev$
```

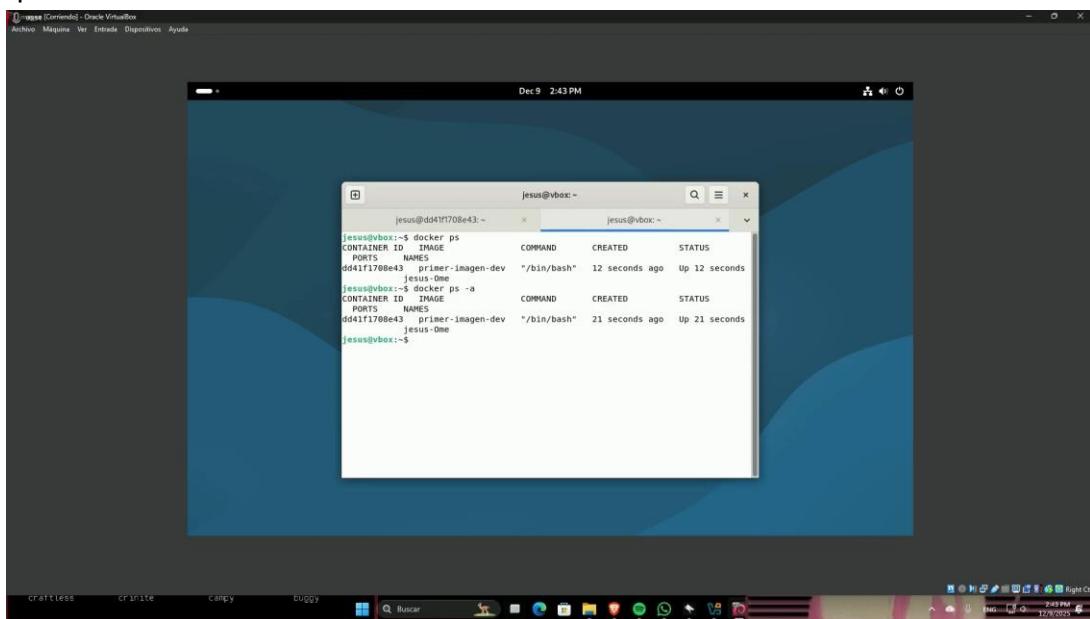
The terminal window is part of a desktop environment with a taskbar at the bottom containing various application icons. The desktop background is dark, and the overall interface is typical of a Linux desktop like Ubuntu.

4.3 CREACION Y GESTION DE CONTENEDOR

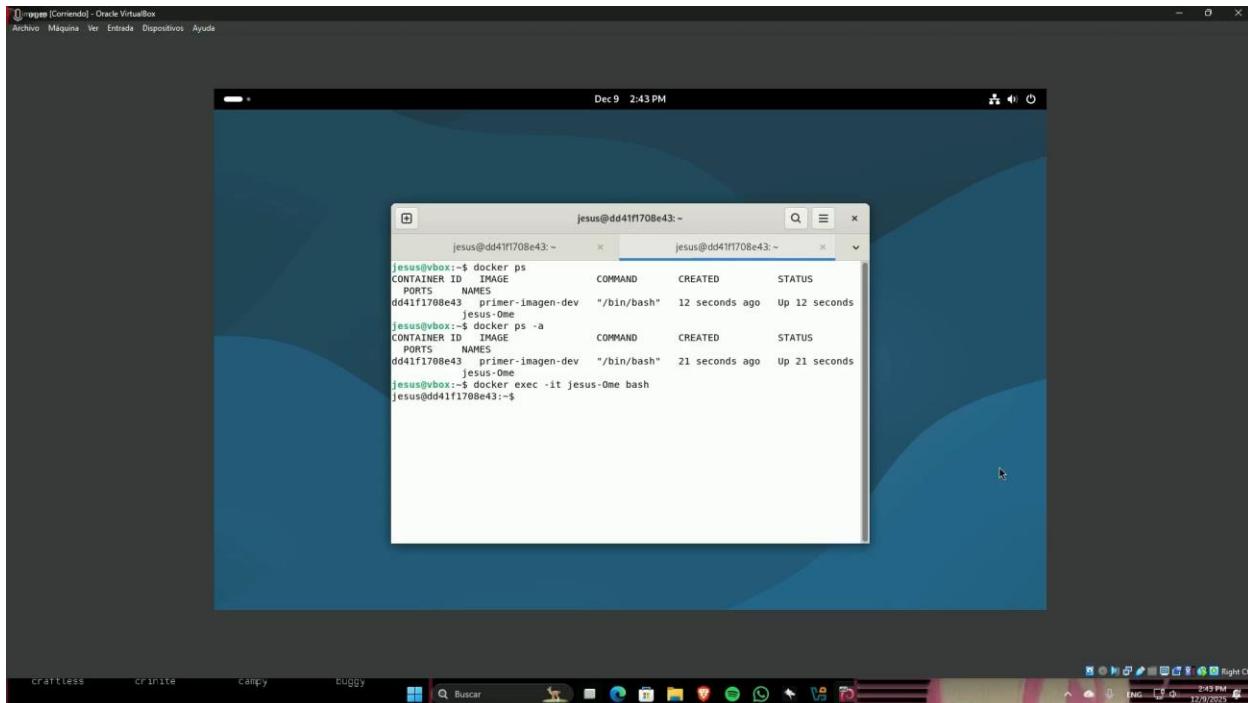
1. hacemos listado de imagenes que tenemos disponible y seleccionamos la que vamos a trabajar en modo iterativo el contenedor, le asignamos el nombre, el nombre la imagen, con /bin/bash asignamos la ruta, y como se muestra ya estamos dentro



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



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

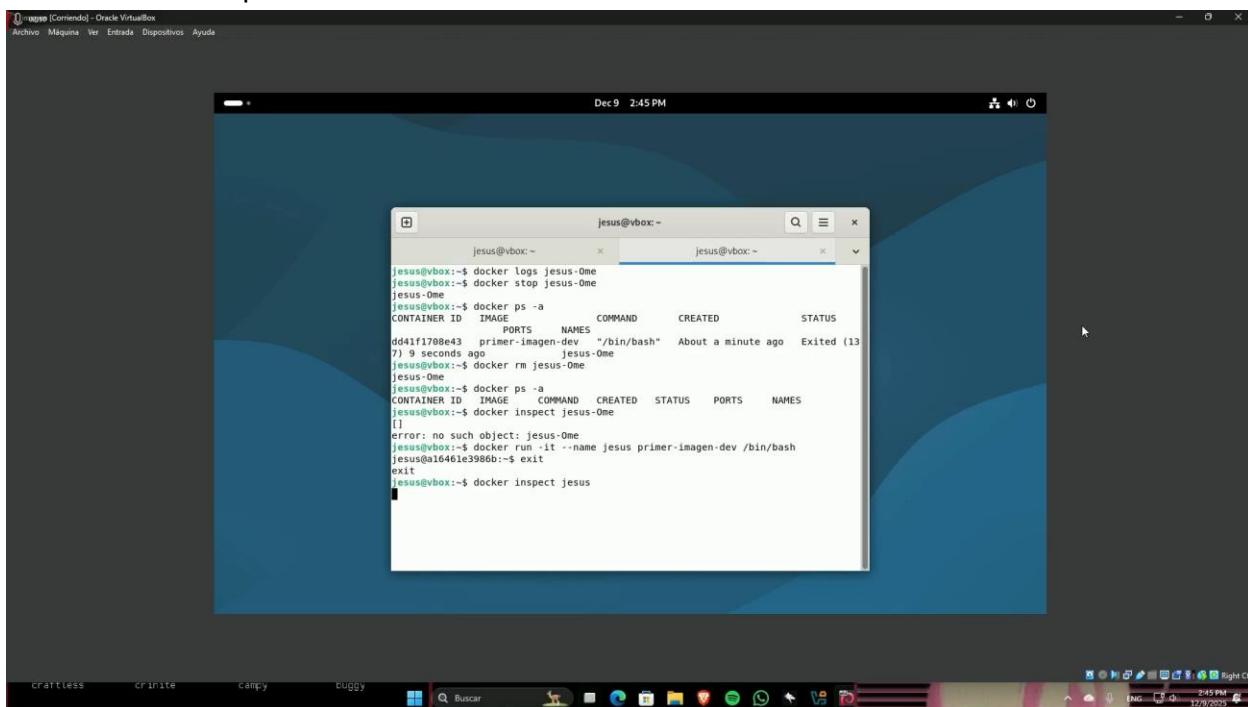


```
Dec 9 2:43 PM

jubuntu@dd41ff708e43:~$ docker ps
CONTAINER ID        IMAGE               COMMAND      CREATED     STATUS
dd41ff708e43        primer-Imagen-dev   "/bin/bash"  12 seconds ago   Up 12 seconds
jubuntu@vbox:~$ docker ps -a
CONTAINER ID        IMAGE               COMMAND      CREATED     STATUS
dd41ff708e43        primer-Imagen-dev   "/bin/bash"  21 seconds ago   Up 21 seconds
jubuntu@vbox:~$ docker exec -it jesus-Ome bash
jubuntu@dd41ff708e43:~$ docker inspect jesus-Ome
jubuntu@vbox:~$ docker logs jesus-Ome
jubuntu@vbox:~$ docker stop jesus-Ome
jubuntu@vbox:~$ docker ps -a
CONTAINER ID        IMAGE               COMMAND      CREATED     STATUS
dd41ff708e43        primer-Imagen-dev   "/bin/bash"  About a minute ago   Exited (13
7) 9 seconds ago
jubuntu@vbox:~$ docker rm jesus-Ome
jubuntu@vbox:~$ docker inspect jesus-Ome
[{"Container": "dd41ff708e43", "Image": "primer-Imagen-dev", "Command": "\u2022/bin/bash", "Created": "2013-12-09T14:43:43.000000000Z", "Status": "exited", "Ports": [{"Port": "0.0.0.0:2375"}], "Names": ["jesus-Ome"]}

jubuntu@vbox:~$ docker inspect jesus
jubuntu@vbox:~$
```

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



```
Dec 9 2:45 PM

jubuntu@vbox:~$ docker logs jesus-Ome
jubuntu@vbox:~$ docker stop jesus-Ome
jubuntu@vbox:~$ docker ps -a
CONTAINER ID        IMAGE               COMMAND      CREATED     STATUS
dd41ff708e43        primer-Imagen-dev   "/bin/bash"  About a minute ago   Exited (13
7) 9 seconds ago
jubuntu@vbox:~$ docker rm jesus-Ome
jubuntu@vbox:~$ docker inspect jesus-Ome
[{"Container": "dd41ff708e43", "Image": "primer-Imagen-dev", "Command": "\u2022/bin/bash", "Created": "2013-12-09T14:43:43.000000000Z", "Status": "exited", "Ports": [{"Port": "0.0.0.0:2375"}], "Names": ["jesus-Ome"]}

jubuntu@vbox:~$ docker logs jesus
jubuntu@vbox:~$ docker stop jesus
jubuntu@vbox:~$ docker ps -a
CONTAINER ID        IMAGE               COMMAND      CREATED     STATUS
dd41ff708e43        primer-Imagen-dev   "/bin/bash"  About a minute ago   Exited (13
7) 9 seconds ago
jubuntu@vbox:~$ docker rm jesus
jubuntu@vbox:~$ docker inspect jesus-Ome
[{"Container": "dd41ff708e43", "Image": "primer-Imagen-dev", "Command": "\u2022/bin/bash", "Created": "2013-12-09T14:43:43.000000000Z", "Status": "exited", "Ports": [{"Port": "0.0.0.0:2375"}], "Names": ["jesus-Ome"]}

jubuntu@vbox:~$ docker inspect jesus
jubuntu@vbox:~$
```