

## Actividad de Configuraciones

# Verificación del funcionamiento de Docker, Docker Desktop, Minikube, Kind

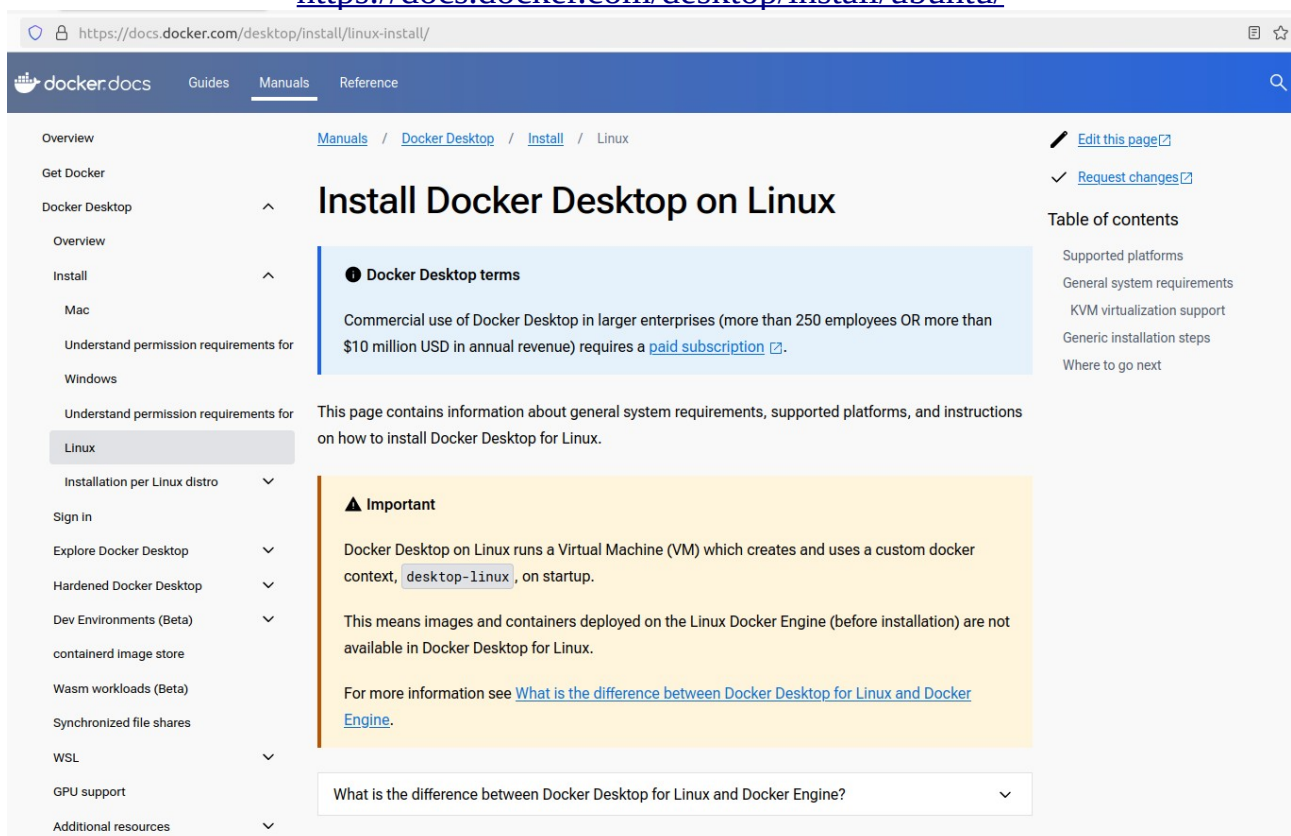
### Objetivos:

- **Comprender las diferencias entre virtualización tradicional y contenerización:** Cómo Docker mejora el desarrollo y la implementación de aplicaciones con entornos ligeros y portátiles.
- **Instalación y configuración de Docker y Docker Desktop en Linux:** Facilitar el desarrollo y la distribución de aplicaciones.
- **Conceptos básicos de Kubernetes:** Usar Minikube y Kind para crear y gestionar clústeres de Kubernetes localmente. Esto incluye entender la arquitectura de Kubernetes y sus objetos (pods, servicios, despliegues, etc.) y manejarlos con estas herramientas.

## Prueba de Docker Engine

1. Descargar e instalar Docker Desktop de:

<https://docs.docker.com/desktop/install/ubuntu/>



The screenshot shows the Docker Docs website page for installing Docker Desktop on Linux. The page title is "Install Docker Desktop on Linux". It includes a sidebar with navigation links like Overview, Get Docker, Docker Desktop, and Linux. The main content area has a section for "Docker Desktop terms" and an "Important" note stating that Docker Desktop on Linux runs a Virtual Machine (VM) and uses a custom docker context, `desktop-linux`, on startup. It also mentions that images and containers deployed on the Linux Docker Engine (before installation) are not available in Docker Desktop for Linux. A search bar at the bottom of the main content area contains the text "What is the difference between Docker Desktop for Linux and Docker Engine?".

instalo el siguiente archivo .deb:



Posteriormente ejecuto el siguiente comando para actualizar mi sistema operativo:

```
david@david-TUF-GAMING-FX504GE-FX80GE:~/Downloads$ sudo apt update
[sudo] password for david:
Hit:1 https://download.docker.com/linux/ubuntu jammy InRelease
Hit:2 https://ppa.launchpadcontent.net/mmk2410/intellij-idea/ubuntu jammy InRelease
Hit:3 https://ftp.postgresql.org/pub/pgadmin/pgadmin4/apt/jammy pgadmin4 InRelease
Get:4 http://security.ubuntu.com/ubuntu jammy-security InRelease [129 kB]
Hit:6 http://pe.archive.ubuntu.com/ubuntu jammy InRelease
Hit:7 https://packages.microsoft.com/repos/edge stable InRelease
Hit:8 http://pe.archive.ubuntu.com/ubuntu jammy-updates InRelease
Hit:5 https://packages.microsoft.com/repos/code stable InRelease
Hit:9 http://pe.archive.ubuntu.com/ubuntu jammy-backports InRelease
Fetched 129 kB in 6s (20,9 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
52 packages can be upgraded. Run 'apt list --upgradable' to see them.
W: https://download.docker.com/linux/ubuntu/dists/jammy/InRelease: Key is stored in legacy trusted.gpg keyring (/etc/apt/trusted.gpg), see the DEPRECATION section in apt-key(8) for details.
```

Posteriormente termino de instalar docker con el siguiente comando:

```
david@david-TUF-GAMING-FX504GE-FX80GE:~/Downloads$ sudo apt install docker-ce docker-ce-cli containerd.io
```

2. Ahora que he instalado Docker Desktop con éxito, es momento de probarlo. Para empezar, ejecuta un contenedor Docker simple directamente desde la línea de comando. Abre una ventana de Terminal y ejecuta el siguiente comando:

```
david@david-TUF-GAMING-FX504GE-FX80GE:~/Downloads$ docker version
Client: Docker Engine - Community
 Cloud integration: v1.0.35+desktop.13
 Version: 26.1.4
 API version: 1.45
 Go version: go1.21.11
 Git commit: 5650f9b
 Built: Wed Jun 5 11:28:57 2024
 OS/Arch: linux/amd64
 Context: default

Server: Docker Engine - Community
 Engine:
  Version: 26.1.4
  API version: 1.45 (minimum version 1.24)
  Go version: go1.21.11
  Git commit: de5c9cf
  Built: Wed Jun 5 11:28:57 2024
  OS/Arch: linux/amd64
  Experimental: false
 containerd:
  Version: 1.6.33
  GitCommit: d2d58213f83a351ca8f528a95fbd145f5654e957
 runc:
  Version: 1.1.12
  GitCommit: v1.1.12-0-g51d5e94
 docker-init:
  Version: 0.19.0
  GitCommit: de40ad0
```

3. Para verificar si puedo ejecutar contenedores, debo ingresar el siguiente comando en la ventana de Terminal y presiona Enter:

```
david@david-TUF-GAMING-FX504GE-FX80GE:~/Downloads$ docker container 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.
    (amd64)
 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/
```

En el resultado, debería aparecer que Docker no encontró una imagen llamada `hello-world:latest`, por lo que decidió descargarla desde un registro de imágenes de Docker. Pero esto no sucede debido a que ya tenía instalado la imagen cuando ejecute este comando anteriormente, Docker Engine creó un contenedor a partir de esa imagen y lo ejecutó. La aplicación se ejecuta dentro del contenedor y luego genera todo el texto, comenzando con "Hello from Docker!".

4. Prueba con otra imagen de prueba divertida que se utiliza comúnmente para verificar la instalación de Docker. Ejecuta el siguiente comando:

```
david@david-TUF-GAMING-FX504GE-FX80GE:~/Downloads$ docker container run rancher/cowsay Hello
Unable to find image 'rancher/cowsay:latest' locally
latest: Pulling from rancher/cowsay
cbdbe7a5bc2a: Pull complete
dd05e66d8cea: Pull complete
34d5e986f175: Pull complete
13eefd6dfff68: Pull complete
Digest: sha256:5dab61268bc18daf56febb5a856b618961cd806dbc49a22a636128ca26f0bd94
Status: Downloaded newer image for rancher/cowsay:latest

  < Hello >
  -----
      \      ^__^
       (oo)\_______
          (__)\       )\/\
              ||----w |
              ||     ||

david@david-TUF-GAMING-FX504GE-FX80GE:~/Downloads$
```

¡Genial! Ahora he confirmado que Docker Engine funciona correctamente en mi computadora. Ahora, debo asegurarme de que Docker Desktop también está funcionando.

Nota: para crear el grupo de Docker y agregar tu usuario, sigue estos pasos:

1. Crear el grupo docker: `sudo groupadd docker`

```
david@david-TUF-GAMING-FX504GE-FX80GE:~/Downloads$ sudo groupadd docker
groupadd: group 'docker' already exists
david@david-TUF-GAMING-FX504GE-FX80GE:~/Downloads$
```

2. Agregar mi usuario a mi grupo docker:

```
david@david-TUF-GAMING-FX504GE-FX80GE:~/Downloads$ sudo usermod -aG docker $USER
david@david-TUF-GAMING-FX504GE-FX80GE:~/Downloads$
```

3. Cierra sesión y volver a iniciarla para que se reevalúe tu permanencia en el grupo

4. También puedo ejecutar el siguiente comando para activar los cambios en los grupos:

```
david@david-TUF-GAMING-FX504GE-FX80GE:~/Downloads$ newgrp docker
david@david-TUF-GAMING-FX504GE-FX80GE:~/Downloads$
```

5. Verificar que puedo correr comandos docker sin usar sudo.

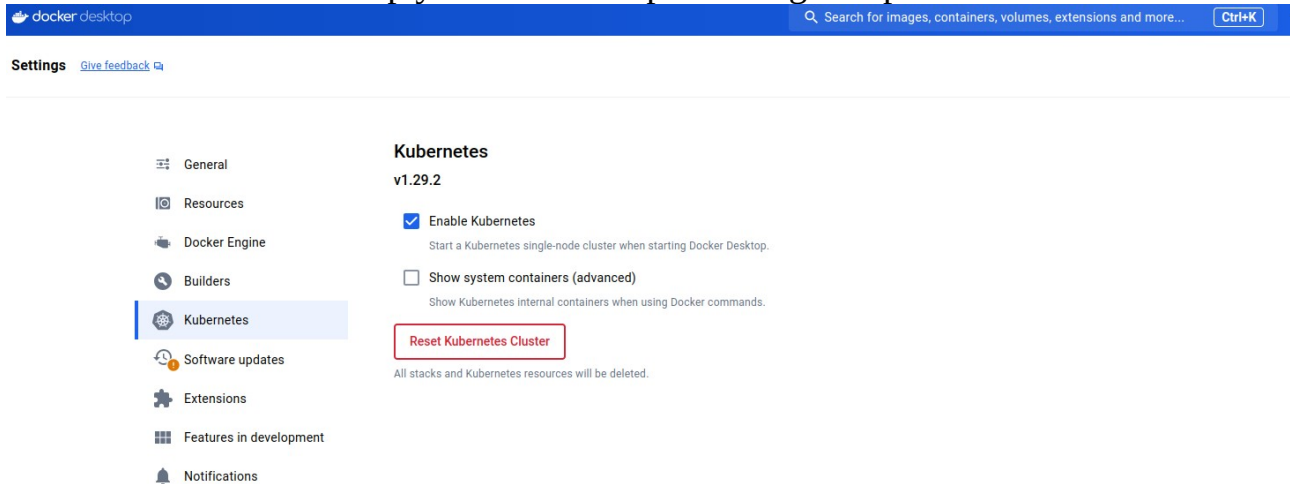
```
david@david-TUF-GAMING-FX504GE-FX80GE:~/Downloads$ docker ps
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS   NAMES
david@david-TUF-GAMING-FX504GE-FX80GE:~/Downloads$
```

# Habilitar Kubernetes en Docker Desktop

Instalo Docker Desktop mediante el siguiente enlace:

<https://docs.docker.com/desktop/install/linux-install/>

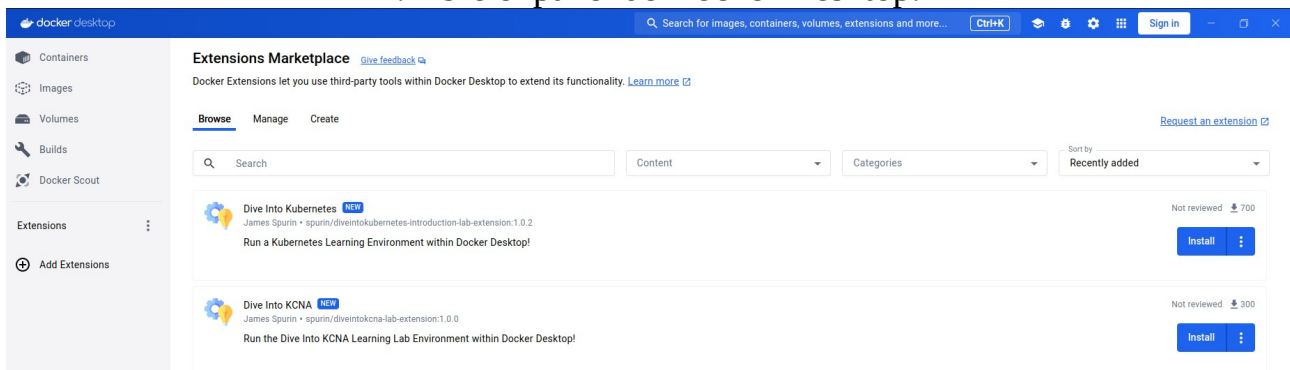
Docker Desktop ya viene con soporte integrado para Kubernetes.



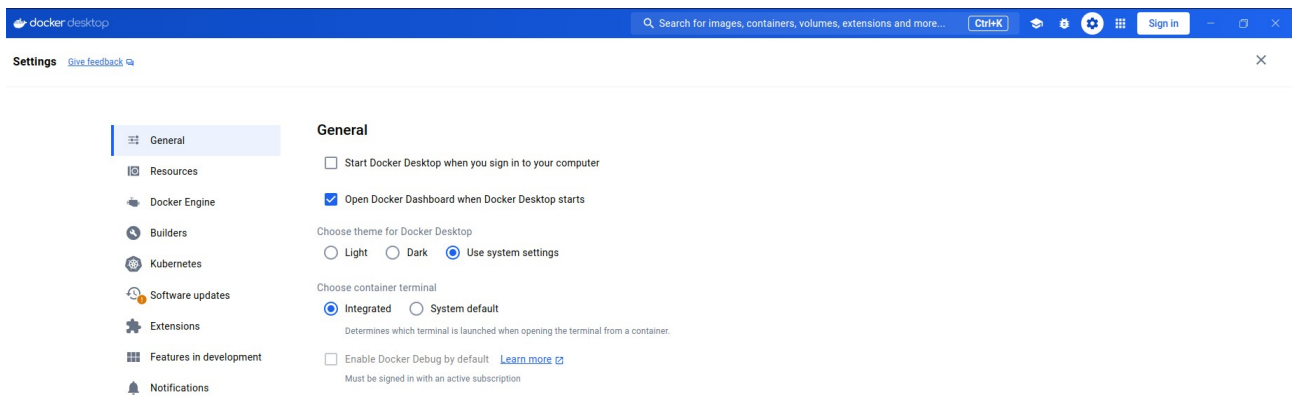
## ¿Qué es Kubernetes?

Kubernetes es una potente plataforma que automatiza la implementación, el escalado y la gestión de aplicaciones en contenedores, proporcionando a desarrolladores, ingenieros de DevOps y administradores de sistemas las herramientas necesarias para gestionar sus aplicaciones de manera escalable y eficiente. Aunque su soporte está desactivado por defecto, es fácil de activar.

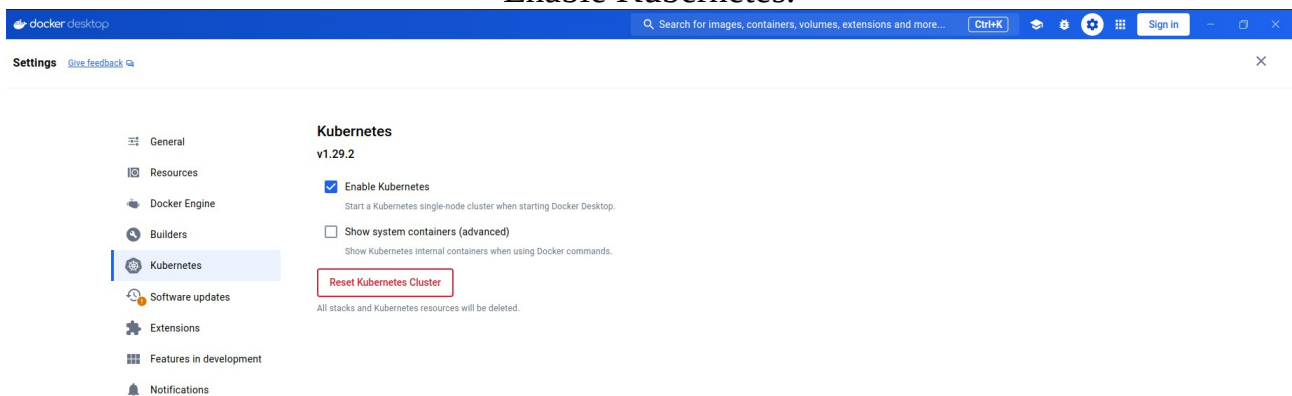
### 1. Abre el panel de Docker Desktop.



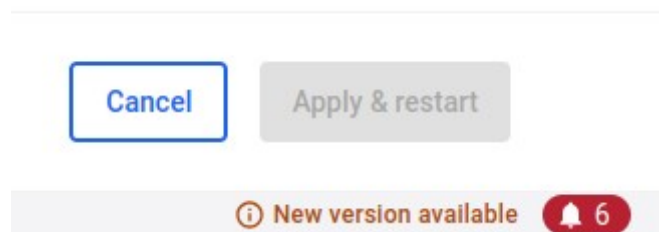
2. En la esquina superior izquierda, selecciona el ícono de la rueda dentada. Esto abrirá la página de configuración (setting).



3. En el lado izquierdo, selecciona la pestaña Kubernetes y luego marca la casilla Enable Kubernetes.



4. Haz clic en el botón Apply & restart.





Para probar Minikube y kubectl debo descargar e instalar [Minikube](https://minikube.sigs.k8s.io/docs/start/) y [kubectl](https://kubernetes.io/docs/tasks/tools/install-kubectl-linux/). Luego, iniciar el clúster por defecto con el comando minikube start. La primera vez que lo haga, tomará un tiempo debido a la descarga de todos los binarios de Kubernetes.

```
david@david-TUF-GAMING-FX504GE-FX80GE:~$ minikube start
🐳 minikube v1.33.1 on Ubuntu 22.04
🌟 Using the docker driver based on existing profile

❌ Requested memory allocation (1864MB) is less than the recommended minimum 1900MB. Deployments may fail.

👍 Starting "minikube" primary control-plane node in "minikube" cluster
🔄 Pulling base image v0.0.44 ...
🔄 Restarting existing docker container for "minikube" ...
🔄 Preparing Kubernetes v1.30.0 on Docker 26.1.1 ...
🔍 Verifying Kubernetes components...
   ▪ Using image gcr.io/k8s-minikube/storage-provisioner:v5
🌟 Enabled addons: storage-provisioner, default-storageclass
🎉 Done! kubectl is now configured to use "minikube" cluster and "default" namespace by default
david@david-TUF-GAMING-FX504GE-FX80GE:~$
```

1. Para acceder al clúster usando kubectl, asegúrate de tener seleccionado el contexto correcto. Si instalaste Docker Desktop y luego Minikube, usa `kubectl config get-contexts` para verificar el contexto actual. Un asterisco junto al contexto llamado minikube indica que este es el contexto activo, permitiendo que kubectl trabaje con el nuevo clúster creado por Minikube.

```
david@david-TUF-GAMING-FX504GE-FX80GE:~$ kubectl config get-contexts
CURRENT      NAME           CLUSTER        AUTHINFO        NAMESPACE
*            minikube       minikube       minikube        default
david@david-TUF-GAMING-FX504GE-FX80GE:~$
```

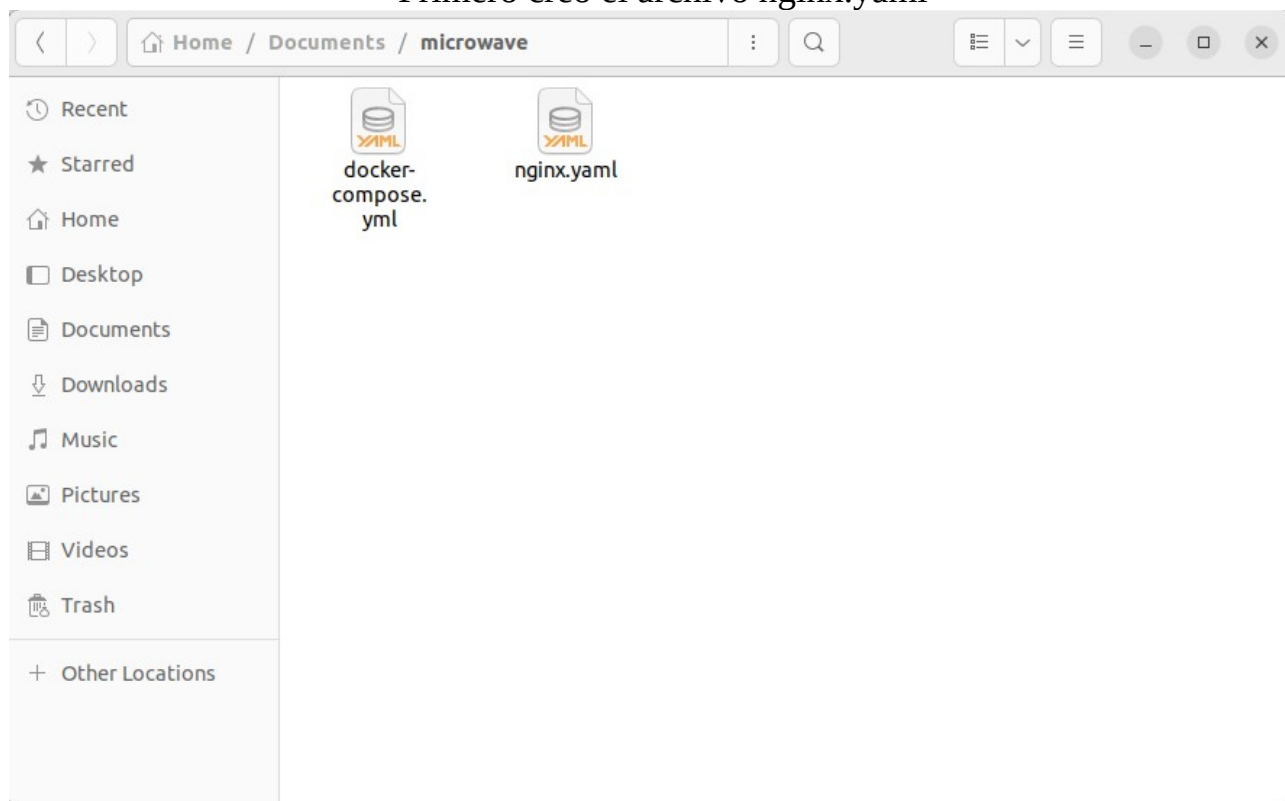
2. Para ver cuántos nodos tiene tu clúster, debo usar el comando `kubectl get nodes`.

En este ejemplo, el clúster es de un solo nodo, que actúa como el nodo maestro (plano de control). Un clúster típico de Kubernetes tiene varios nodos maestros y muchos nodos trabajadores. La versión de Kubernetes utilizada aquí es la v1.28.3, aunque puede variar en tu caso.

```
david@david-TUF-GAMING-FX504GE-FX80GE:~$ kubectl get nodes
NAME        STATUS    ROLES           AGE    VERSION
minikube    Ready     control-plane   114s   v1.30.0
david@david-TUF-GAMING-FX504GE-FX80GE:~$
```

3. Intentar ejecutar algo en el cluster. Para ello utilizaremos Ngix, un servidor web famoso para estas tareas. Utilizaremos el archivo .yaml, que acompaña a la actividad para utilizar esta prueba:

- Primero creo el archivo nginx.yaml



escribo lo siguiente en este archivo:





Posteriormente abro una nueva terminal y creo un pod que ejecute Nginx con el siguiente comando:

```
hadoop@david-TUF-GAMING-FX504GE-FX80GE:~/Documents/microwave$ kubectl apply -f nginx.yaml
pod/nginx created
hadoop@david-TUF-GAMING-FX504GE-FX80GE:~/Documents/microwave$
```

4. Puedo verificar si el pod esta ejecutando con kubectl:

```
hadoop@david-TUF-GAMING-FX504GE-FX80GE:~/Documents/microwave$ kubectl get pods
NAME      READY   STATUS    RESTARTS   AGE
nginx     1/1     Running   0           114s
hadoop@david-TUF-GAMING-FX504GE-FX80GE:~/Documents/microwave$
```

5. Para acceder al servidor Nginx, necesitas exponer la aplicación en el pod usando el comando:

```
$ kubectl expose pod nginx --type=NodePort --port=80
```

Esto crea un servicio de Kubernetes que permite acceder a Nginx desde tu computadora, por ejemplo, a través de un navegador.

```
hadoop@david-TUF-GAMING-FX504GE-FX80GE:~/Documents/microwave$ kubectl expose pod nginx --type=NodePort --port=80
service/nginx exposed
hadoop@david-TUF-GAMING-FX504GE-FX80GE:~/Documents/microwave$
```

6. Podemos usar kubectl para listar todos los servicios en el clúster con:

```
$ kubectl get services
```

Esto mostrará el servicio Nginx que creamos, el cual es de tipo NodePort. El puerto 80 del pod está asignado al puerto 30432 del nodo del clúster en Minikube.

```
hadoop@david-TUF-GAMING-FX504GE-FX80GE:~/Documents/microwave$ kubectl get services
NAME          TYPE        CLUSTER-IP      EXTERNAL-IP      PORT(S)          AGE
kubernetes    ClusterIP   10.96.0.1        <none>            443/TCP          7m5s
nginx         NodePort    10.108.248.144   <none>            80:30127/TCP     2m2s
hadoop@david-TUF-GAMING-FX504GE-FX80GE:~/Documents/microwave$
```

7. Ahora puedes usar Minikube para crear un túnel hacia el clúster y acceder al servidor web Nginx en un navegador. Usa este comando:

## \$ minikube service nginx

Esto abrirá la URL correcta en tu navegador para acceder a Nginx.

```
hadoop@david-TUF-GAMING-FX504GE-FX80GE:~/Documents/microwave$ minikube service nginx
```

NAMESPACE	NAME	TARGET PORT	URL
default	nginx	80	http://192.168.49.2:30127

```
Starting tunnel for service nginx.
```

NAMESPACE	NAME	TARGET PORT	URL
default	nginx		http://127.0.0.1:35365

```
Opening service default/nginx in default browser...
! Because you are using a Docker driver on linux, the terminal needs to be open to run it.
Gtk-Message: 16:39:58.628: Not loading module "atk-bridge": The functionality is provided by GTK natively. Please try to not load it.
[8128, Main Thread] WARNING: GTK+ module /snap/firefox/4336/gnome-platform/usr/lib/gtk-2.0/modules/libcanberra-gtk-module.so cannot be loaded.
GTK+ 2.x symbols detected. Using GTK+ 2.x and GTK+ 3 in the same process is not supported.: 'glib warning', file /build/firefox/parts/firefox/build/toolkit/xre/nsSigHandlers.cpp:187

(firefox:8128): Gtk-WARNING **: 16:39:58.964: GTK+ module /snap/firefox/4336/gnome-platform/usr/lib/gtk-2.0/modules/libcanberra-gtk-module.so cannot be loaded.
GTK+ 2.x symbols detected. Using GTK+ 2.x and GTK+ 3 in the same process is not supported.
Gtk-Message: 16:39:58.964: Failed to load module "canberra-gtk-module"
[8128, Main Thread] WARNING: GTK+ module /snap/firefox/4336/gnome-platform/usr/lib/gtk-2.0/modules/libcanberra-gtk-module.so cannot be loaded.
GTK+ 2.x symbols detected. Using GTK+ 2.x and GTK+ 3 in the same process is not supported.: 'glib warning', file /build/firefox/parts/firefox/build/toolkit/xre/nsSigHandlers.cpp:187

(firefox:8128): Gtk-WARNING **: 16:39:58.974: GTK+ module /snap/firefox/4336/gnome-platform/usr/lib/gtk-2.0/modules/libcanberra-gtk-module.so cannot be loaded.
GTK+ 2.x symbols detected. Using GTK+ 2.x and GTK+ 3 in the same process is not supported.
Gtk-Message: 16:39:58.974: Failed to load module "canberra-gtk-module"
```

127.0.0.1:35365

Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to [nginx.org](http://nginx.org).  
Commercial support is available at [nginx.com](http://nginx.com).

Thank you for using nginx.

El resultado muestra que Minikube creó un túnel para el servicio Nginx, escuchando en el puerto 30432 de tu computadora portátil. Has accedido con éxito a un servidor web Nginx en un clúster de Kubernetes de un solo nodo en Minikube. Para limpiar:

- Detén el túnel presionando Ctrl + C en la terminal.

```
(firefox:8128): Gtk-WARNING **: 16:39:58.974: GTK+ module /snap/firefox/4336/gnome-platform/usr/lib/gtk-2.0/modules/libcanberra-gtk-module.so cannot be loaded.  
GTK+ 2.x symbols detected. Using GTK+ 2.x and GTK+ 3 in the same process is not supported.  
Gtk-Message: 16:39:58.974: Failed to load module "canberra-gtk-module"  
^C 🖱 Stopping tunnel for service nginx.  
hadoop@david-TUF-GAMING-FX504GE-FX80GE:~/Documents/microwave$
```

- Elimina el servicio y el pod Nginx:

```
kubectl delete service nginx  
kubectl delete pod nginx
```

```
hadoop@david-TUF-GAMING-FX504GE-FX80GE:~/Documents/microwave$ kubectl delete service nginx  
service "nginx" deleted  
hadoop@david-TUF-GAMING-FX504GE-FX80GE:~/Documents/microwave$ kubectl delete pod nginx  
pod "nginx" deleted  
hadoop@david-TUF-GAMING-FX504GE-FX80GE:~/Documents/microwave$
```

- Detén el clúster con:

```
minikube stop
```

```
hadoop@david-TUF-GAMING-FX504GE-FX80GE:~/Documents/microwave$ minikube stop  
🖱 Stopping node "minikube" ...  
🔴 Powering off "minikube" via SSH ...  
🔴 1 node stopped.  
hadoop@david-TUF-GAMING-FX504GE-FX80GE:~/Documents/microwave$
```

## Ejercicios:

A veces, probar con un clúster de un solo nodo no es suficiente. Minikube lo resuelve. Sigue estas instrucciones para crear un verdadero clúster de Kubernetes de múltiples nodos en Minikube:

1. Si quieres trabajar con un clúster que consta de varios nodos en Minikube, podemos usar este comando:

```
$ minikube start --nodes 3 -p demo
```

El comando anterior crea un clúster con tres nodos y lo llamas "demo".

```
hadoop@david-TUF-GAMING-FX504GE-FX80GE:~/Documents/microwave$ minikube start --nodes 3 -p demo
🐳 minikube v1.33.1 on Ubuntu 22.04
🌟 Using the docker driver based on existing profile
⚠️ You cannot change the number of nodes for an existing minikube cluster. Please use 'minikube node
add' to add nodes to an existing cluster.

❌ Requested memory allocation (1864MB) is less than the recommended minimum 1900MB. Deployments may
fail.

👍 Starting "minikube" primary control-plane node in "minikube" cluster
🔄 Pulling base image v0.0.44 ...
🔄 Restarting existing docker container for "minikube" ...
🔄 Preparing Kubernetes v1.30.0 on Docker 26.1.1 ...
🔍 Verifying Kubernetes components...
   ■ Using image gcr.io/k8s-minikube/storage-provisioner:v5
🌟 Enabled addons: default-storageclass, storage-provisioner
🌟 Done! kubectl is now configured to use "minikube" cluster and "default" namespace by default
hadoop@david-TUF-GAMING-FX504GE-FX80GE:~/Documents/microwave$
```

2. Utiliza kubectl para enumerar todos los nodos de tu clúster:

```
$ kubectl get nodes
```

Tenemos un clúster de 3 nodos donde el nodo "demo" es un nodo maestro y los dos nodos restantes son nodos de trabajo.

```
hadoop@david-TUF-GAMING-FX504GE-FX80GE:~/Documents/microwave$ kubectl get nodes
NAME          STATUS    ROLES          AGE    VERSION
minikube      Ready     control-plane  25m    v1.30.0
hadoop@david-TUF-GAMING-FX504GE-FX80GE:~/Documents/microwave$
```

3. No vamos a continuar con este ejemplo aquí, así que usa el siguiente comando para detener el clúster:

```
$ minikube stop
```

```
hadoop@david-TUF-GAMING-FX504GE-FX80GE:~/Documents/microwave$ minikube stop
🔥 Stopping node "minikube" ...
🔥 Powering off "minikube" via SSH ...
🔥 1 node stopped.
hadoop@david-TUF-GAMING-FX504GE-FX80GE:~/Documents/microwave$
```

4. Eliminar todos los clusteres de mi sistema con el siguiente comando:

```
$ minikube delete -all
```

```
hadoop@david-TUF-GAMING-FX504GE-FX80GE:~/Documents/microwave$ minikube delete --all
🔥 Deleting "minikube" in docker ...
🔥 Removing /home/hadoop/.minikube/machines/minikube ...
💀 Removed all traces of the "minikube" cluster.
🔥 Successfully deleted all profiles
hadoop@david-TUF-GAMING-FX504GE-FX80GE:~/Documents/microwave$
```



Esto eliminará el clúster predeterminado (llamado minikube) y el clúster demo en nuestro caso.

Con esto, pasaremos a la siguiente herramienta interesante y útil a la hora de trabajar con contenedores y Kubernetes. Deberías tenerlo instalado y disponible en la computadora de tu trabajo.

## Kind

Kind (<https://kind.sigs.k8s.io/docs/user/quick-start>) es otra herramienta popular que se puede utilizar para ejecutar un clúster de Kubernetes de múltiples nodos localmente en tu máquina. Es muy fácil de instalar y usar.

Vamos:

1. En una máquina Linux, puedes usar el siguiente script para instalar Kind desde sus archivos binarios:

```
curl -Lo ./kind https://kind.sigs.k8s.io/dl/v0.22.0/kind-linux-amd64  
chmod +x ./kind
```

```
sudo mv ./kind /usr/local/bin/kind
```

```
hadoop@david-TUF-GAMING-FX504GE-FX80GE:~/Documents/microwave$ curl -Lo ./kind https://kind.sigs.k8s.io/dl/v0.22.0/kind-linux-amd64  
% Total    % Received % Xferd  Average Speed   Time    Time     Time  Current  
             Dload  Upload  Total   Spent    Left   Speed  
100    97  100    97    0     0    105      0  --:--:-- --:--:-- --:--:--   105  
0     0    0     0    0     0     0      0  --:--:-- 0:00:01 --:--:--    0  
100 6245k  100 6245k    0     0  1118k      0  0:00:05 0:00:05 --:--:-- 2007k  
hadoop@david-TUF-GAMING-FX504GE-FX80GE:~/Documents/microwave$ chmod +x ./kind  
hadoop@david-TUF-GAMING-FX504GE-FX80GE:~/Documents/microwave$ sudo mv ./kind /usr/local/bin/kind  
[sudo] password for hadoop:  
hadoop@david-TUF-GAMING-FX504GE-FX80GE:~/Documents/microwave$
```

2. Una vez instalado Kind, pruébalo con el siguiente comando:

kind version

```
hadoop@david-TUF-GAMING-FX504GE-FX80GE:~/Documents/microwave$ kind version  
kind v0.22.0 go1.20.13 linux/amd64  
hadoop@david-TUF-GAMING-FX504GE-FX80GE:~/Documents/microwave$
```

3. Ahora, intenta crear un clúster de Kubernetes simple que consta de un nodo maestro y dos nodos trabajadores. Utiliza este comando para lograr esto:

kind create cluster

```
hadoop@david-TUF-GAMING-FX504GE-FX80GE:~/Documents/microwave$ kind create cluster
Creating cluster "kind" ...
 ✓ Ensuring node image (kindest/node:v1.29.2) 🚢
 ✓ Preparing nodes 📦
 ✓ Writing configuration 📄
 ✓ Starting control-plane 🔥
 ✓ Installing CNI 🖱️
 ✓ Installing StorageClass 💾
Set kubect context to "kind-kind"
You can now use your cluster with:

kubectl cluster-info --context kind-kind

Have a nice day! 🙌
hadoop@david-TUF-GAMING-FX504GE-FX80GE:~/Documents/microwave$
```

4. Para verificar que se ha creado un clúster, utiliza este comando:

kind get clusters

```
hadoop@david-TUF-GAMING-FX504GE-FX80GE:~/Documents/microwave$ kind get clusters
kind
hadoop@david-TUF-GAMING-FX504GE-FX80GE:~/Documents/microwave$
```

5. Podemos crear un clúster adicional con un nombre diferente usando el parámetro --name, así:

kind create cluster --name demo

```
hadoop@david-TUF-GAMING-FX504GE-FX80GE:~/Documents/microwave$ kind create cluster --name demo
Creating cluster "demo" ...
 ✓ Ensuring node image (kindest/node:v1.29.2) 🚢
 ✓ Preparing nodes 📦
 ✓ Writing configuration 📄
 ✓ Starting control-plane 🔥
 ✓ Installing CNI 🖱️
 ✓ Installing StorageClass 💾
Set kubect context to "kind-demo"
You can now use your cluster with:

kubectl cluster-info --context kind-demo

Thanks for using kind! 😊
hadoop@david-TUF-GAMING-FX504GE-FX80GE:~/Documents/microwave$
```



6. Enumera los clústeres:

\$ kind get clusters

```
hadoop@david-TUF-GAMING-FX504GE-FX80GE:~/Documents/microwave$ kind get clusters
demo
kind
hadoop@david-TUF-GAMING-FX504GE-FX80GE:~/Documents/microwave$
```

Ahora podemos usar kubectl para acceder y trabajar con los clústeres que acabamos de crear. Kind actualiza automáticamente el archivo de configuración de kubectl mientras se crea el clúster. Puedes verificar los contextos disponibles con:

\$ kubectl config get-contexts

```
hadoop@david-TUF-GAMING-FX504GE-FX80GE:~/Documents/microwave$ kubectl config get-contexts
CURRENT  NAME           CLUSTER      AUTHINFO      NAMESPACE
*         kind-demo      kind-demo    kind-demo
kind-kind kind-kind      kind-kind    kind-kind
hadoop@david-TUF-GAMING-FX504GE-FX80GE:~/Documents/microwave$
```

Selecciona el contexto del clúster "demo" si no es el contexto actual:

\$ kubectl config use-context kind-demo

```
hadoop@david-TUF-GAMING-FX504GE-FX80GE:~/Documents/microwave$ kubectl config use-context kind-demo
Switched to context "kind-demo".
hadoop@david-TUF-GAMING-FX504GE-FX80GE:~/Documents/microwave$
```

7. Enumera todos los nodos del clúster de demo:

\$ kubectl get nodes

```
hadoop@david-TUF-GAMING-FX504GE-FX80GE:~/Documents/microwave$ kubectl get nodes
NAME                STATUS    ROLES          AGE    VERSION
demo-control-plane  Ready    control-plane  12m    v1.29.2
hadoop@david-TUF-GAMING-FX504GE-FX80GE:~/Documents/microwave$
```

8. Intenta ejecutar un contenedor en el clúster utilizando Nginx. Usa el siguiente comando para aplicar la configuración desde un archivo YAML:

\$ kubectl apply -f nginx.yaml

```
hadoop@david-TUF-GAMING-FX504GE-FX80GE:~/Documents/microwave$ kubectl apply -f nginx.yaml
pod/nginx created
hadoop@david-TUF-GAMING-FX504GE-FX80GE:~/Documents/microwave$
```

Verifica que el pod se haya creado correctamente:

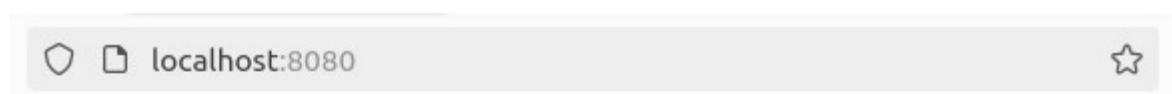
pod/nginx created

9. Para acceder al servidor Nginx, realiza el reenvío de puertos utilizando kubectl. Usa este comando para redirigir el puerto local al puerto del contenedor:

\$ kubectl port-forward nginx 8080:80

```
hadoop@david-TUF-GAMING-FX504GE-FX80GE:~/Documents/microwave$ kubectl port-forward nginx 8080:80
Forwarding from 127.0.0.1:8080 -> 80
Forwarding from [::1]:8080 -> 80
Handling connection for 8080
█
```

Abre una nueva pestaña del navegador y navega hasta <http://localhost:8080> para ver la pantalla de bienvenida de Nginx.



## Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to [nginx.org](http://nginx.org).  
Commercial support is available at [nginx.com](http://nginx.com).

*Thank you for using nginx.*

10. Una vez que hayas terminado de usar Nginx, elimina el pod del clúster con el siguiente comando:

\$ kubectl delete -f nginx.yaml

```
hadoop@david-TUF-GAMING-FX504GE-FX80GE:~/Documents/microwave$ kubectl delete -f nginx.yaml
pod "nginx" deleted
hadoop@david-TUF-GAMING-FX504GE-FX80GE:~/Documents/microwave$
```

Antes de continuar, limpia y elimina los clústeres que hemos creado:

```
$ kind delete cluster --name kind  
$ kind delete cluster --name demo
```

```
hadoop@david-TUF-GAMING-FX504GE-FX80GE:~/Documents/microwave$ kind delete cluster --name kind  
Deleting cluster "kind" ...  
Deleted nodes: ["kind-control-plane"]  
hadoop@david-TUF-GAMING-FX504GE-FX80GE:~/Documents/microwave$ kind delete cluster --name demo  
Deleting cluster "demo" ...  
Deleted nodes: ["demo-control-plane"]  
hadoop@david-TUF-GAMING-FX504GE-FX80GE:~/Documents/microwave$
```

Con esto, has instalado y configurado las herramientas necesarias para trabajar con contenedores en tu máquina local de manera efectiva.

### Preguntas:

**1. En tus propias palabras, usando analogías, explica qué es un contenedor.**

Un contenedor es como una caja que contiene todo lo necesario para ejecutar una aplicación, incluyendo su código, bibliotecas y configuraciones, asegurando que funcione igual en cualquier entorno.

**2. ¿Por qué se considera que los contenedores cambian las reglas del juego en IT? Menciona tres o cuatro razones.**

- **Portabilidad:** Funcionan en cualquier entorno.
- **Eficiencia:** Usan menos recursos que las máquinas virtuales.
- **Escalabilidad:** Facilitan la expansión de aplicaciones.
- **Aislamiento:** Mejoran la seguridad y evitan conflictos entre aplicaciones.

**3. ¿Qué significa cuando afirmamos que, si un contenedor se ejecuta en una plataforma determinada, entonces se ejecutará en cualquier lugar? Menciona dos o tres razones por las que esto es cierto.**

- **Consistencia:** Incluyen todas las dependencias.
- **Estándares:** Son compatibles entre sistemas operativos.
- **Aislamiento:** No dependen de la configuración del sistema anfitrión.

**4. ¿Es verdadera o falsa la siguiente afirmación: los contenedores Docker solo son útiles para aplicaciones modernas y totalmente nuevas basadas en microservicios? Por favor justifica tu respuesta.**

Falsa. Los contenedores también benefician a las aplicaciones tradicionales, facilitando su portabilidad, escalabilidad y gestión.

**5. ¿Por qué nos importaría instalar y usar un administrador de paquetes en nuestra computadora local?**

Simplifica la instalación, actualización y gestión del software, ahorrando tiempo y evitando conflictos de dependencias.

**6. ¿Con Docker Desktop, puede desarrollar y ejecutar contenedores de Linux?**

Sí, Docker Desktop permite desarrollar y ejecutar contenedores de Linux en Windows o macOS.

**7. ¿Por qué son esenciales buenas habilidades de programación (como Bash o PowerShell) para el uso productivo de los contenedores?**

Porque permiten automatizar tareas, gestionar contenedores eficientemente y escribir scripts para despliegue y monitoreo.

**8. Nombra tres o cuatro distribuciones de Linux en las que Docker esté certificado para ejecutarse.**

- Ubuntu
- Debian
- CentOS
- Red Hat Enterprise Linux (RHEL)

**9. Instalaste minikube en tu sistema. ¿Para qué tipo de tareas utilizarás esta herramienta?**

Para desplegar y probar clústeres de Kubernetes localmente, desarrollar y depurar aplicaciones en contenedores, y aprender sobre Kubernetes en un entorno controlado.