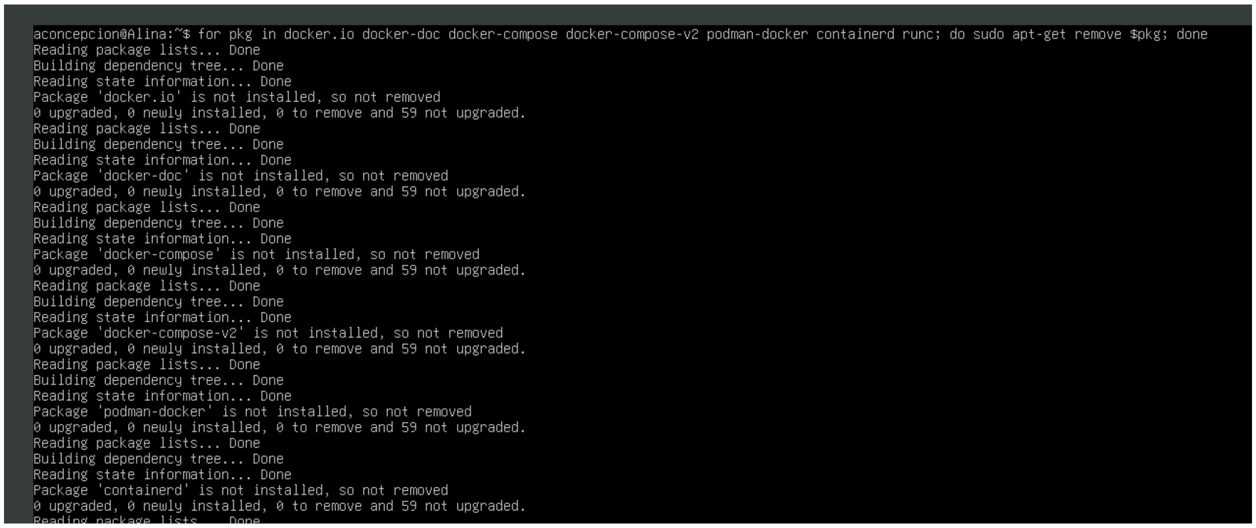
Alina Concepcion
Linux Administration
Professor. Adrianna Holden-Gouveia
1 December 2024

Docker is a tool that allows people to easily deploy their applications in a sandbox (known as containers) to run on the host operating system. The benefit of Dockers is that users are allowed to package an application with all of its dependencies into a standardized unit for software development (dockers-curriculum). Containers are important because they provide most of the isolation of the virtual machines at a fraction of computing power (docker-curriculum).

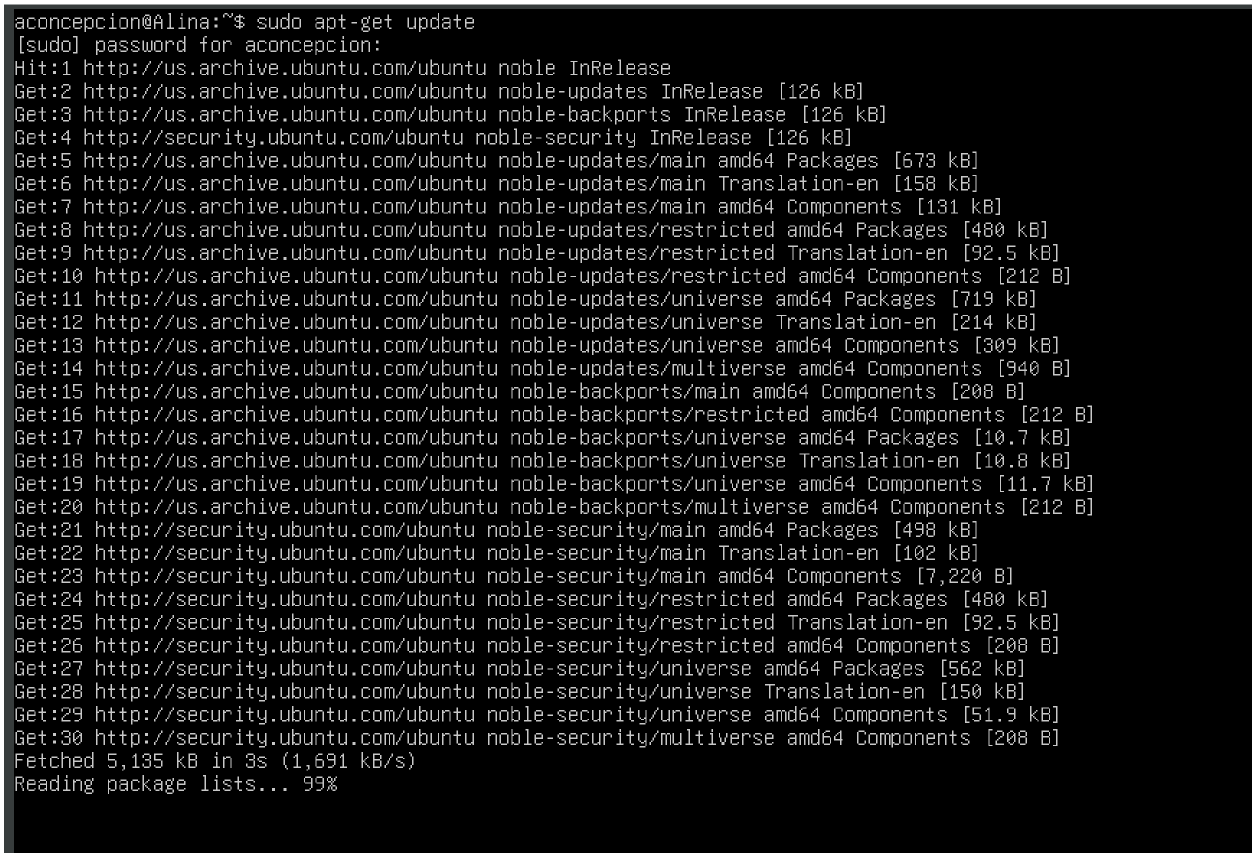
To install Docker on your Linux virtual machine, in this case Ubuntu you can go to this link https://docs.docker.com/engine/install/ubuntu/. You must ensure that you meet all of the requirements, you can click the link at the top of the webpage of docs.docker to view the requirements. For example, to install Docker you need the 64 bit version of Ubuntu. Also, before you install Docker, you need to set up the docker apt repository.

Before officially installing Docker, make sure you didn't install it before in the past (like I did). Use this command ***for pkg in docker.io docker-doc docker-compose docker-compose-v2 podman-docker containerd runc; do sudo apt-get remove $pkg; done*** to remove any Docker packages in case if you needed to remove it for whatever reason.



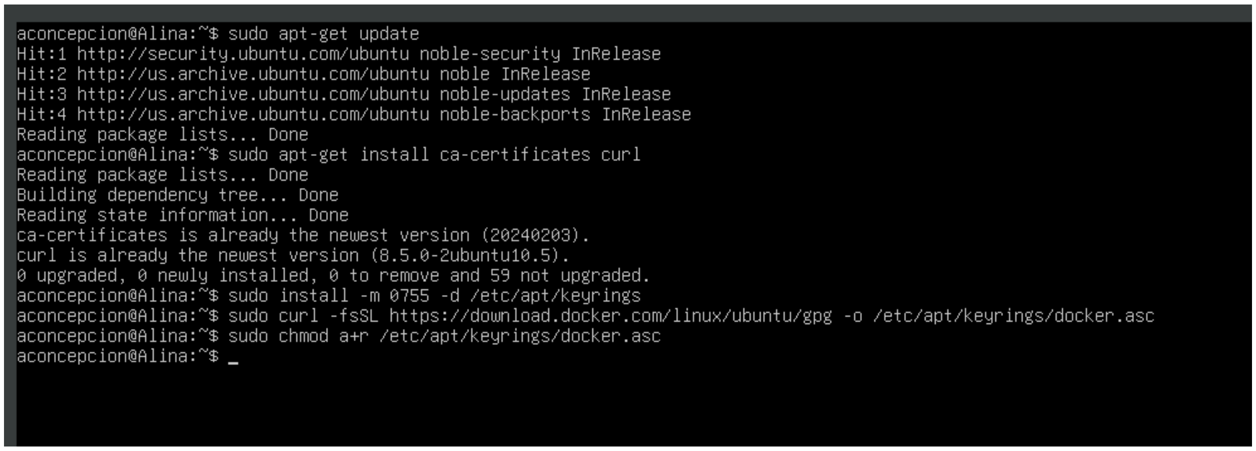
```
aconcepcion@Alina:~$ for pkg in docker.io docker-doc docker-compose docker-compose-v2 podman-docker containerd runc; do sudo apt-get remove $pkg; done
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Package 'docker.io' is not installed, so not removed
0 upgraded, 0 newly installed, 0 to remove and 59 not upgraded.
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Package 'docker-doc' is not installed, so not removed
0 upgraded, 0 newly installed, 0 to remove and 59 not upgraded.
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Package 'docker-compose' is not installed, so not removed
0 upgraded, 0 newly installed, 0 to remove and 59 not upgraded.
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Package 'docker-compose-v2' is not installed, so not removed
0 upgraded, 0 newly installed, 0 to remove and 59 not upgraded.
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Package 'podman-docker' is not installed, so not removed
0 upgraded, 0 newly installed, 0 to remove and 59 not upgraded.
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Package 'containerd' is not installed, so not removed
0 upgraded, 0 newly installed, 0 to remove and 59 not upgraded.
Reading package lists... Done
```

Next, update your server using ***sudo apt-get update***



```
aconcepcion@Alina:~$ sudo apt-get update
[sudo] password for aconcepcion:
Hit:1 http://us.archive.ubuntu.com/ubuntu noble InRelease
Get:2 http://us.archive.ubuntu.com/ubuntu noble-updates InRelease [126 kB]
Get:3 http://us.archive.ubuntu.com/ubuntu noble-backports InRelease [126 kB]
Get:4 http://security.ubuntu.com/ubuntu noble-security InRelease [126 kB]
Get:5 http://us.archive.ubuntu.com/ubuntu noble-updates/main amd64 Packages [673 kB]
Get:6 http://us.archive.ubuntu.com/ubuntu noble-updates/main Translation-en [158 kB]
Get:7 http://us.archive.ubuntu.com/ubuntu noble-updates/main amd64 Components [131 kB]
Get:8 http://us.archive.ubuntu.com/ubuntu noble-updates/restricted amd64 Packages [480 kB]
Get:9 http://us.archive.ubuntu.com/ubuntu noble-updates/restricted Translation-en [92.5 kB]
Get:10 http://us.archive.ubuntu.com/ubuntu noble-updates/restricted amd64 Components [212 B]
Get:11 http://us.archive.ubuntu.com/ubuntu noble-updates/universe amd64 Packages [719 kB]
Get:12 http://us.archive.ubuntu.com/ubuntu noble-updates/universe Translation-en [214 kB]
Get:13 http://us.archive.ubuntu.com/ubuntu noble-updates/universe amd64 Components [309 kB]
Get:14 http://us.archive.ubuntu.com/ubuntu noble-updates/multiverse amd64 Components [940 B]
Get:15 http://us.archive.ubuntu.com/ubuntu noble-backports/main amd64 Components [208 B]
Get:16 http://us.archive.ubuntu.com/ubuntu noble-backports/restricted amd64 Components [212 B]
Get:17 http://us.archive.ubuntu.com/ubuntu noble-backports/universe amd64 Packages [10.7 kB]
Get:18 http://us.archive.ubuntu.com/ubuntu noble-backports/universe Translation-en [10.8 kB]
Get:19 http://us.archive.ubuntu.com/ubuntu noble-backports/universe amd64 Components [11.7 kB]
Get:20 http://us.archive.ubuntu.com/ubuntu noble-backports/multiverse amd64 Components [212 B]
Get:21 http://security.ubuntu.com/ubuntu noble-security/main amd64 Packages [498 kB]
Get:22 http://security.ubuntu.com/ubuntu noble-security/main Translation-en [102 kB]
Get:23 http://security.ubuntu.com/ubuntu noble-security/main amd64 Components [7,220 B]
Get:24 http://security.ubuntu.com/ubuntu noble-security/restricted amd64 Packages [480 kB]
Get:25 http://security.ubuntu.com/ubuntu noble-security/restricted Translation-en [92.5 kB]
Get:26 http://security.ubuntu.com/ubuntu noble-security/restricted amd64 Components [208 B]
Get:27 http://security.ubuntu.com/ubuntu noble-security/universe amd64 Packages [562 kB]
Get:28 http://security.ubuntu.com/ubuntu noble-security/universe Translation-en [150 kB]
Get:29 http://security.ubuntu.com/ubuntu noble-security/universe amd64 Components [51.9 kB]
Get:30 http://security.ubuntu.com/ubuntu noble-security/multiverse amd64 Components [208 B]
Fetched 5,135 kB in 3s (1,691 kB/s)
Reading package lists... 99%
```

Before installing Docker, we need to set up the Docker repository. After setting up our repository we can officially install Docker. I used sudo apt-get update to update my server, then used ***sudo apt-get install ca-certificates curl*** , next ***sudo install -m 0755 -d /etc/apt/keyrings*** . Lastly, sudo ***curl -fsSL https://download.docker.com/linux/ubuntu/gpg -o /etc/apt/keyrings/docker.asc*** and ***sudo chmod a+r /etc/apt/keyrings/docker.asc***. Those commands are used to install Dockers GPG Key.



```
aconcepcion@Alina:~$ sudo apt-get update
Hit:1 http://security.ubuntu.com/ubuntu noble-security InRelease
Hit:2 http://us.archive.ubuntu.com/ubuntu noble InRelease
Hit:3 http://us.archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:4 http://us.archive.ubuntu.com/ubuntu noble-backports InRelease
Reading package lists... Done
aconcepcion@Alina:~$ sudo apt-get install ca-certificates curl
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
ca-certificates is already the newest version (20240203).
curl is already the newest version (8.5.0-2ubuntu10.5).
0 upgraded, 0 newly installed, 0 to remove and 59 not upgraded.
aconcepcion@Alina:~$ sudo install -m 0755 -d /etc/apt/keyrings
aconcepcion@Alina:~$ sudo curl -fsSL https://download.docker.com/linux/ubuntu/gpg -o /etc/apt/keyrings/docker.asc
aconcepcion@Alina:~$ sudo chmod a+r /etc/apt/keyrings/docker.asc
aconcepcion@Alina:~$ _
```

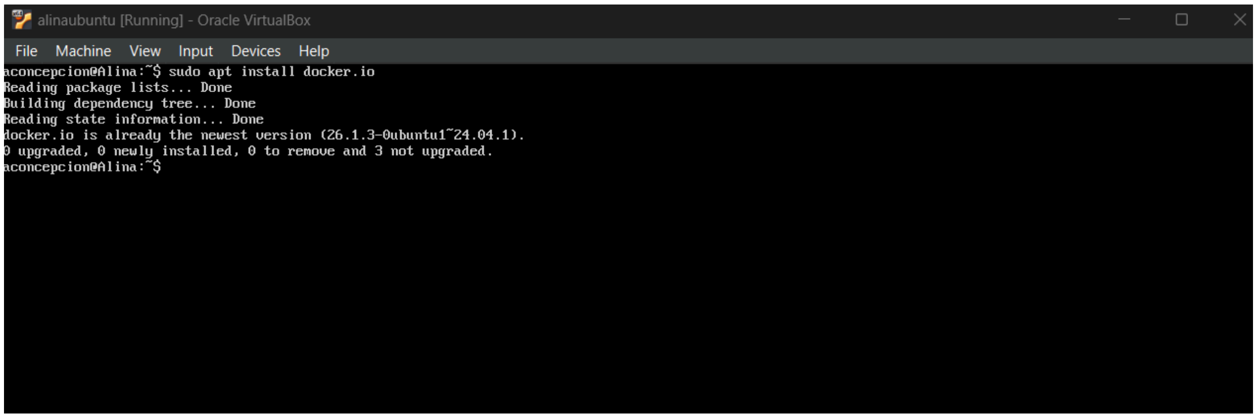
To add the repository(package), you enter this command

```
echo \
  "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.asc]
https://download.docker.com/linux/ubuntu \
  $(. /etc/os-release && echo "$VERSION_CODENAME") stable" | \
  sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
sudo apt-get update
```

Next, you will need to install the latest version, using this command ***sudo apt-get install docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin***

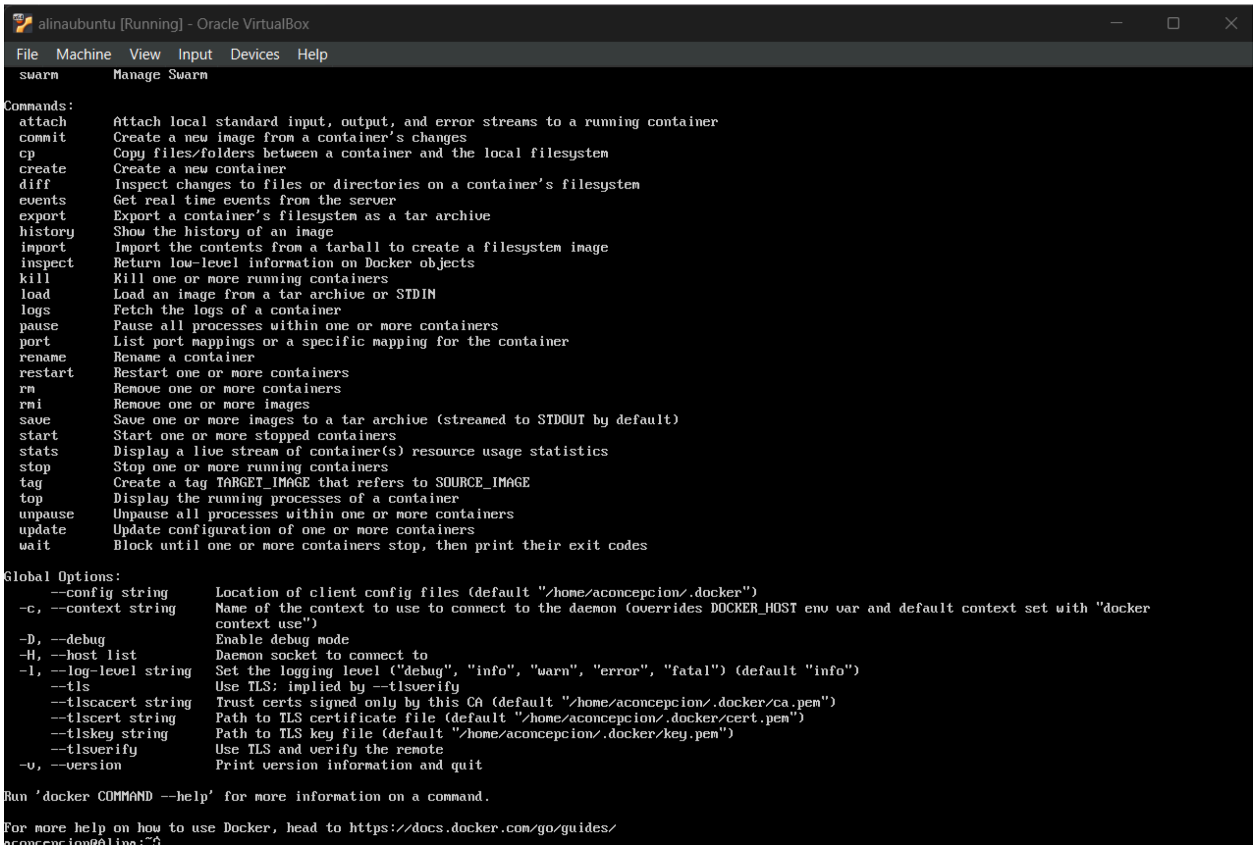
Lastly, if you want to test out Docker you can type ***sudo docker run hello-world***

I had some trouble on the last steps and after some troubleshooting and researching, I found the easiest way to download Docker is to use the command ***sudo apt install docker.io*** . This command is pretty straightforward.


alinaubuntu [Running] - Oracle VirtualBox
File Machine View Input Devices Help

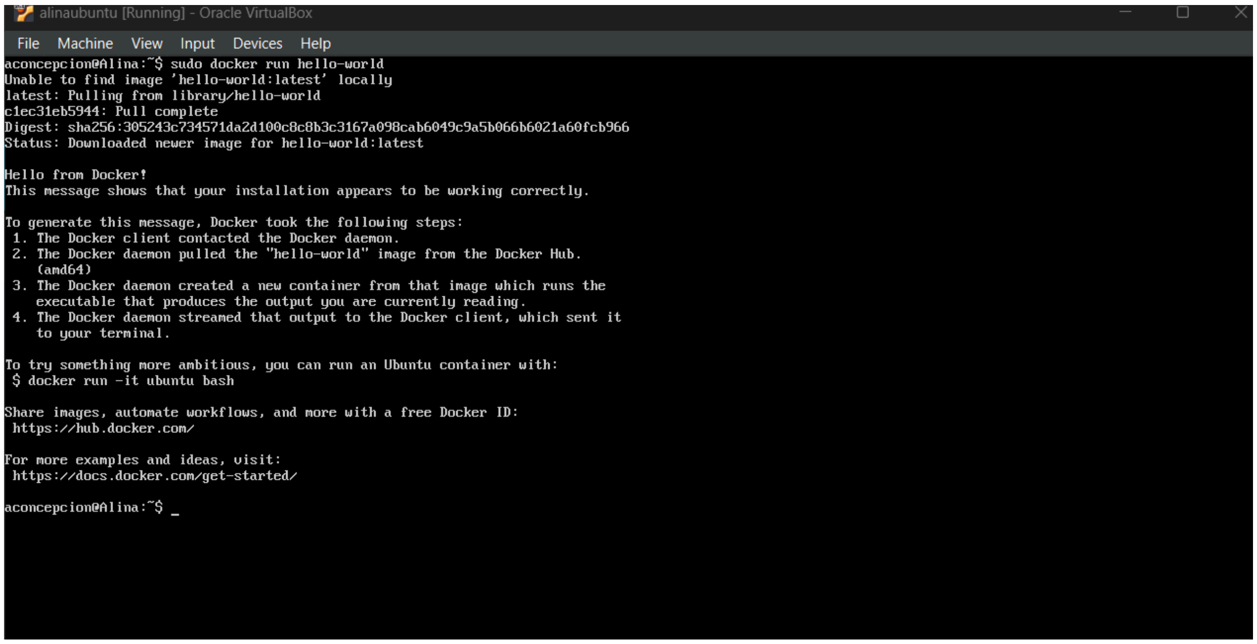
```
aconcepcion@Alina:~$ sudo apt install docker.io
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
docker.io is already the newest version (26.1.3-0ubuntu1~24.04.1).
0 upgraded, 0 newly installed, 0 to remove and 3 not upgraded.
aconcepcion@Alina:~$
```

Next, I tested it out by typing ***docker*** which displays the numerous things you can do with docker.

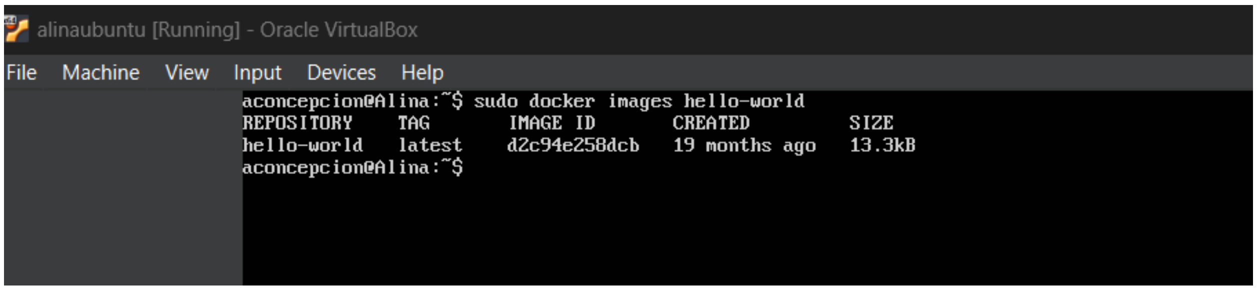

alinaubuntu [Running] - Oracle VirtualBox
File Machine View Input Devices Help

```
  swarm       Manage Swarm

Commands:
  attach      Attach local standard input, output, and error streams to a running container
  commit      Create a new image from a container's changes
  cp          Copy files/folders between a container and the local filesystem
  create      Create a new container
  diff        Inspect changes to files or directories on a container's filesystem
  events      Get real time events from the server
  export      Export a container's filesystem as a tar archive
  history     Show the history of an image
  import      Import the contents from a tarball to create a filesystem image
  inspect     Return low-level information on Docker objects
  kill        Kill one or more running containers
  load        Load an image from a tar archive or STDIN
  logs        Fetch the logs of a container
  pause       Pause all processes within one or more containers
  port        List port mappings or a specific mapping for the container
  rename      Rename a container
  restart     Restart one or more containers
  rm          Remove one or more containers
  rmi         Remove one or more images
  save        Save one or more images to a tar archive (streamed to STDOUT by default)
  start       Start one or more stopped containers
  stats       Display a live stream of container(s) resource usage statistics
  stop        Stop one or more running containers
  tag         Create a tag TARGET_IMAGE that refers to SOURCE_IMAGE
  top         Display the running processes of a container
  unpause     Unpause all processes within one or more containers
  update      Update configuration of one or more containers
  wait        Block until one or more containers stop, then print their exit codes

Global Options:
      --config string      Location of client config files (default "/home/aconcepcion/.docker")
  -c, --context string     Name of the context to use to connect to the daemon (overrides DOCKER_HOST env var and default context set with "docker
                           context use")
  -D, --debug              Enable debug mode
  -H, --host list          Daemon socket to connect to
  -l, --log-level string   Set the logging level ("debug", "info", "warn", "error", "fatal") (default "info")
      --tls                Use TLS; implied by --tlsverify
      --tlscacert string   Trust certs signed only by this CA (default "/home/aconcepcion/.docker/ca.pem")
      --tlscert string     Path to TLS certificate file (default "/home/aconcepcion/.docker/cert.pem")
      --tlskey string      Path to TLS key file (default "/home/aconcepcion/.docker/key.pem")
      --tlsverify          Use TLS and verify the remote
  -v, --version            Print version information and quit

Run 'docker COMMAND --help' for more information on a command.

For more help on how to use Docker, head to https://docs.docker.com/go/guides/
aconcepcion@Alina:~$
```

You can also test docker by running ***sudo docker run hello-world***

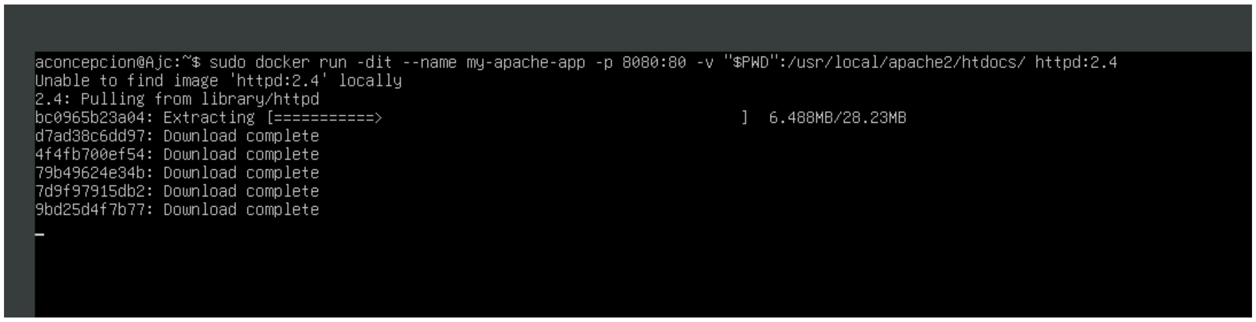

alinaubuntu [Running] - Oracle VirtualBox
File Machine View Input Devices Help

```
aconcepcion@Alina:~$ sudo docker run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
c1ec31eb5944: Pull complete
Digest: sha256:305243c734571da2d100c8c8b3c3167a098cab6049c9a5b066b6021a60fcb966
Status: Downloaded newer image for hello-world:latest

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/

aconcepcion@Alina:~$ _
```


alinaubuntu [Running] - Oracle VirtualBox
File Machine View Input Devices Help

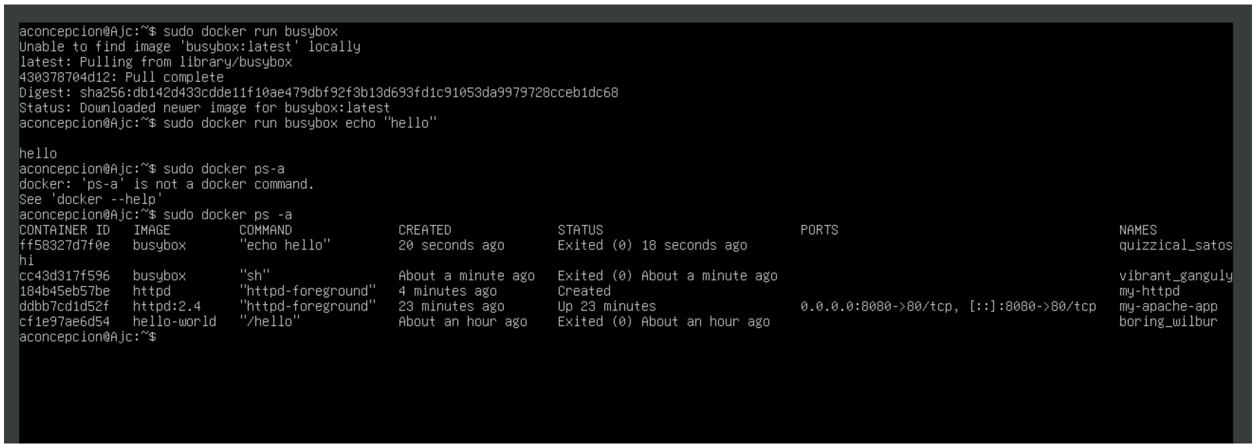
```
aconcepcion@Alina:~$ sudo docker images hello-world
REPOSITORY    TAG       IMAGE ID       CREATED         SIZE
hello-world   latest    d2c94e258dcb   19 months ago   13.3kB
aconcepcion@Alina:~$
```

TO install an image from docker, I used ***sudo docker run -dit –name my-apache-app -p 8080:80 -v "$PWD":/usr/local/apache2/htdocs/ httpd:2.4***



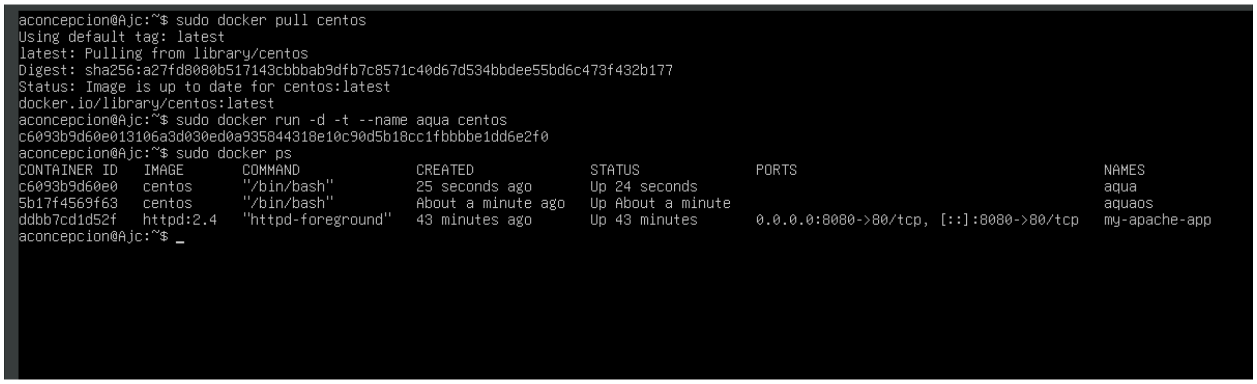
```
aconcepcion@Ajc:~$ sudo docker run -dit --name my-apache-app -p 8080:80 -v "$PWD":/usr/local/apache2/htdocs/ httpd:2.4
Unable to find image 'httpd:2.4' locally
2.4: Pulling from library/httpd
bc0965b23a04: Extracting [===========>                                       ]  6.488MB/28.23MB
d7ad38c6dd97: Download complete
4f4fb700ef54: Download complete
79b49624e34b: Download complete
7d9f97915db2: Download complete
9bd25d4f7b77: Download complete
_
```

I was able to run a few other commands by using sudo. ***Sudo docker run busybox*****,** then I used Docker to echo the word hello and used ***sudo docker ps-a*** to see the containers and information.



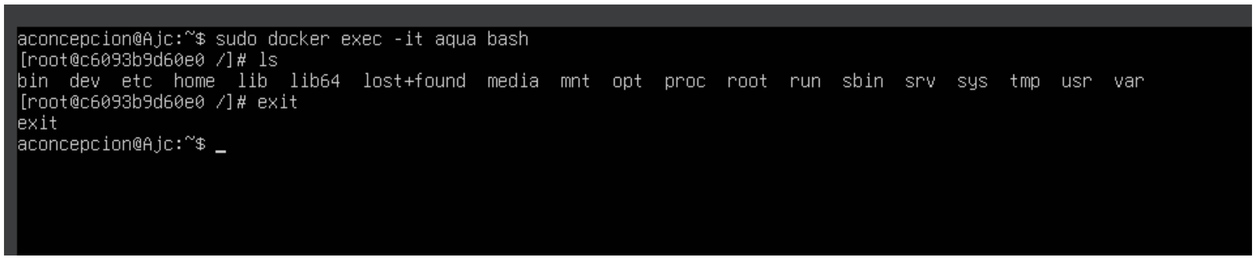
```
aconcepcion@Ajc:~$ sudo docker run busybox
Unable to find image 'busybox:latest' locally
latest: Pulling from library/busybox
430378704d12: Pull complete
Digest: sha256:db142d433cdde11f10ae479dbf92f3b13d693fd1c91053da9979728cceb1dc68
Status: Downloaded newer image for busybox:latest
aconcepcion@Ajc:~$ sudo docker run busybox echo "hello"

hello
aconcepcion@Ajc:~$ sudo docker ps-a
docker: 'ps-a' is not a docker command.
See 'docker --help'
aconcepcion@Ajc:~$ sudo docker ps -a
CONTAINER ID   IMAGE         COMMAND              CREATED              STATUS                          PORTS                                     NAMES
ff58327d7f0e   busybox       "echo hello"         20 seconds ago       Exited (0) 18 seconds ago                                                 quizzical_satos
hi
cc43d317f596   busybox       "sh"                 About a minute ago   Exited (0) About a minute ago                                             vibrant_ganguly
184b45eb57be   httpd         "httpd-foreground"   4 minutes ago        Created                                                                   my-httpd
ddbb7cd1d52f   httpd:2.4     "httpd-foreground"   23 minutes ago       Up 23 minutes                   0.0.0.0:8080->80/tcp, [::]:8080->80/tcp   my-apache-app
cf1e97ae6d54   hello-world   "/hello"             About an hour ago    Exited (0) About an hour ago                                              boring_wilbur
aconcepcion@Ajc:~$
```

I used docker to install the centos container by using the command ***sudo docker pull centos***, then named the container aqua by using ***sudo docker run -d -t –name aqua*** centos. Lastly, I used ***sudo docker ps*** to view the containers.



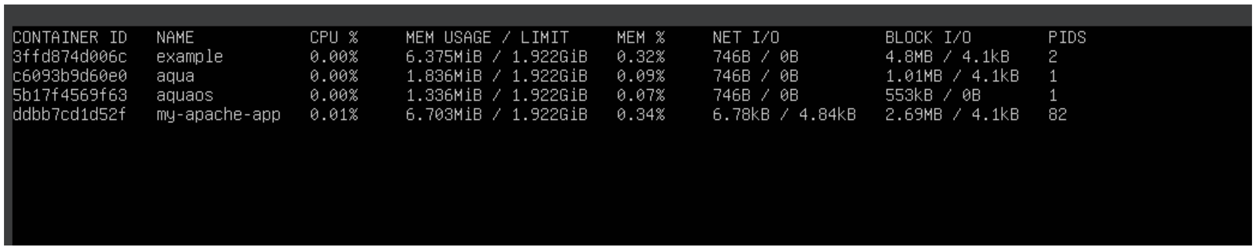
```
aconcepcion@Ajc:~$ sudo docker pull centos
Using default tag: latest
latest: Pulling from library/centos
Digest: sha256:a27fd8080b517143cbbbab9dfb7c8571c40d67d534bbdee55bd6c473f432b177
Status: Image is up to date for centos:latest
docker.io/library/centos:latest
aconcepcion@Ajc:~$ sudo docker run -d -t --name aqua centos
c6093b9d60e013106a3d030ed0a935844318e10c90d5b18cc1fbbbbe1dd6e2f0
aconcepcion@Ajc:~$ sudo docker ps
CONTAINER ID   IMAGE       COMMAND              CREATED              STATUS               PORTS                                     NAMES
c6093b9d60e0   centos      "/bin/bash"          25 seconds ago       Up 24 seconds                                                  aqua
5b17f4569f63   centos      "/bin/bash"          About a minute ago   Up About a minute                                              aquaos
ddbb7cd1d52f   httpd:2.4   "httpd-foreground"   43 minutes ago       Up 43 minutes        0.0.0.0:8080->80/tcp, [::]:8080->80/tcp   my-apache-app
aconcepcion@Ajc:~$ _
```

To open and access the container (centos), I used ***<u>sudo docker exec -it aqua bash.</u>*** As you can see we are root in centos, I used ***<u>ls</u>*** to display the information in the centos container, in my ubuntu vm.
I used the ***<u>exit</u>*** command to exit the container.



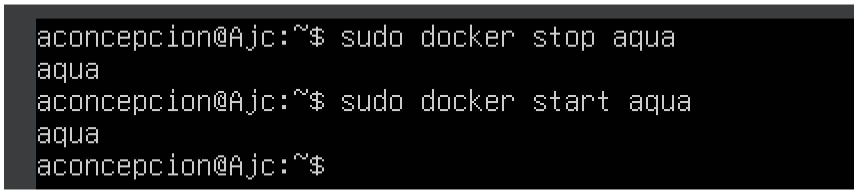
```
aconcepcion@Ajc:~$ sudo docker exec -it aqua bash
[root@c6093b9d60e0 /]# ls
bin  dev  etc  home  lib  lib64  lost+found  media  mnt  opt  proc  root  run  sbin  srv  sys  tmp  usr  var
[root@c6093b9d60e0 /]# exit
exit
aconcepcion@Ajc:~$ _
```

To view the cpu, memory, etc for the running containers, we use ***<u>sudo docker stats</u>*** .



```
CONTAINER ID   NAME            CPU %     MEM USAGE / LIMIT     MEM %     NET I/O           BLOCK I/O         PIDS
3ffd874d006c   example         0.00%     6.375MiB / 1.922GiB   0.32%     746B / 0B         4.8MB / 4.1kB     2
c6093b9d60e0   aqua            0.00%     1.836MiB / 1.922GiB   0.09%     746B / 0B         1.01MB / 4.1kB    1
5b17f4569f63   aquaos          0.00%     1.336MiB / 1.922GiB   0.07%     746B / 0B         553kB / 0B        1
ddbb7cd1d52f   my-apache-app   0.01%     6.703MiB / 1.922GiB   0.34%     6.78kB / 4.84kB   2.69MB / 4.1kB    82
```

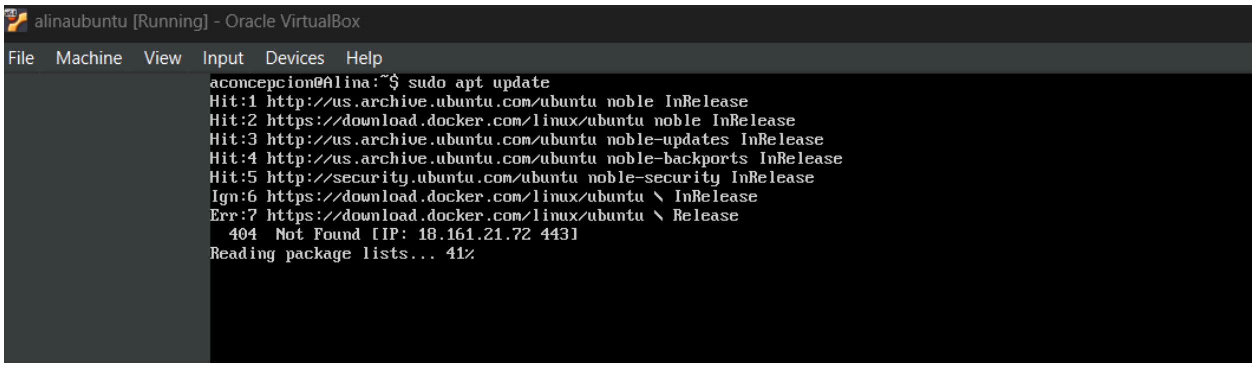
To stop or start a docker we use ***<u>sudo docker stop with the name</u>*** or ***<u>sudo docker start with the name</u>***



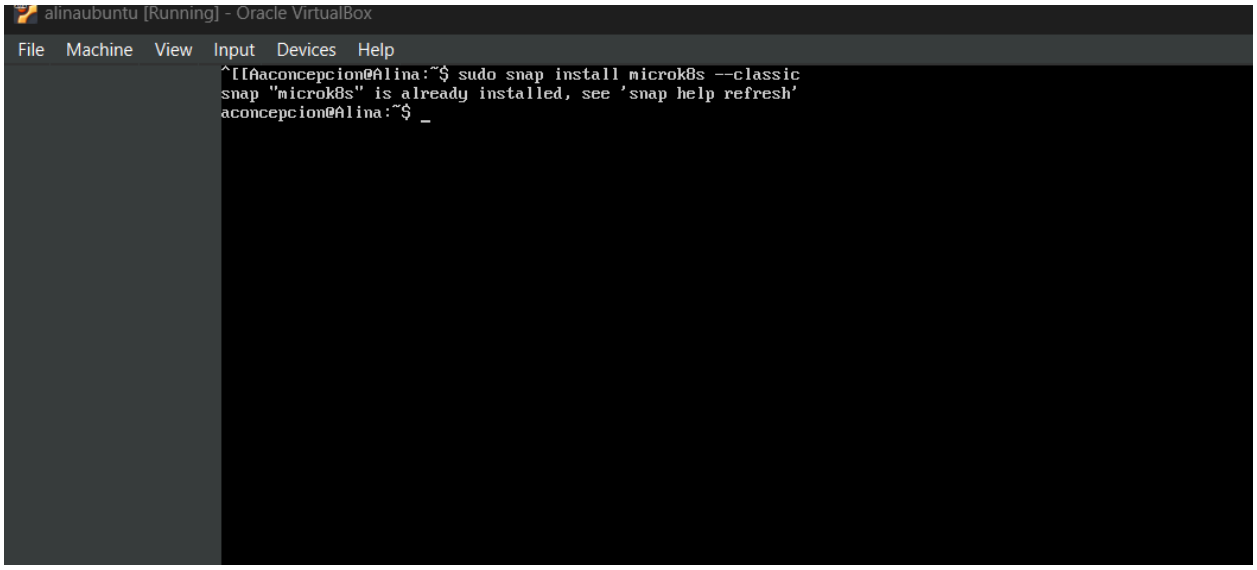
```
aconcepcion@Ajc:~$ sudo docker stop aqua
aqua
aconcepcion@Ajc:~$ sudo docker start aqua
aqua
aconcepcion@Ajc:~$
```

Kubernetes is a portable, extensible, open source platform for managing containerized workloads and services, that facilitates both declarative configuration and automation.(Kubernetes.io)

First, we need to update our server using ***sudo apt update.*** Then we start the installation using ***sudo snap install microk8s – -classic***

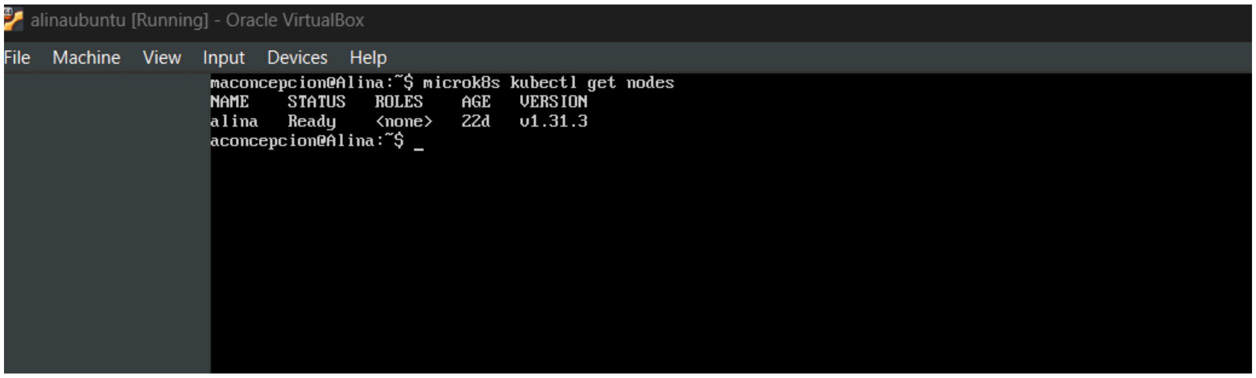

alinaubuntu [Running] - Oracle VirtualBox
File Machine View Input Devices Help

```
aconcepcion@Alina:~$ sudo apt update
Hit:1 http://us.archive.ubuntu.com/ubuntu noble InRelease
Hit:2 https://download.docker.com/linux/ubuntu noble InRelease
Hit:3 http://us.archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:4 http://us.archive.ubuntu.com/ubuntu noble-backports InRelease
Hit:5 http://security.ubuntu.com/ubuntu noble-security InRelease
Ign:6 https://download.docker.com/linux/ubuntu \ InRelease
Err:7 https://download.docker.com/linux/ubuntu \ Release
  404  Not Found [IP: 18.161.21.72 443]
Reading package lists... 41%
```


alinaubuntu [Running] - Oracle VirtualBox
File Machine View Input Devices Help

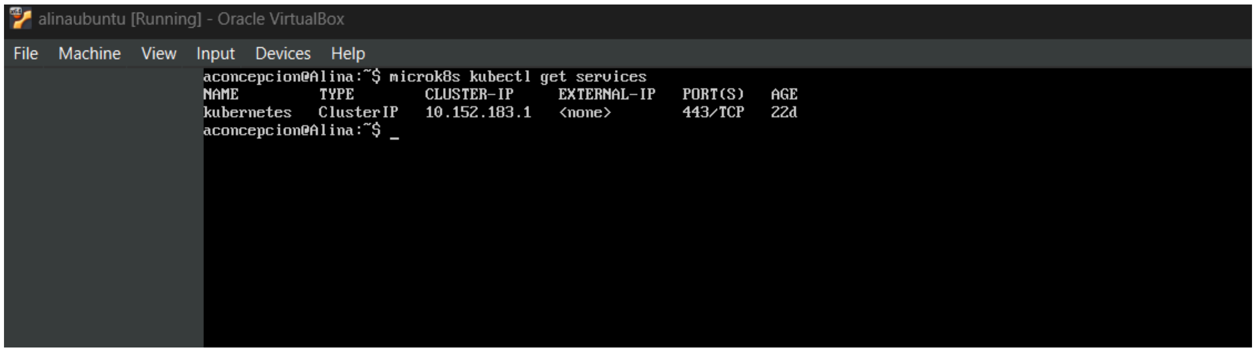
```
^[[Aaconcepcion@Alina:~$ sudo snap install microk8s --classic
snap "microk8s" is already installed, see 'snap help refresh'
aconcepcion@Alina:~$ _
```

To retrieve the information about kubernetes, we use ***microk8s kubectl get nodes*** . As you can see we have the name of the server, status, roles, age and the version type.


alinaubuntu [Running] - Oracle VirtualBox
File Machine View Input Devices Help

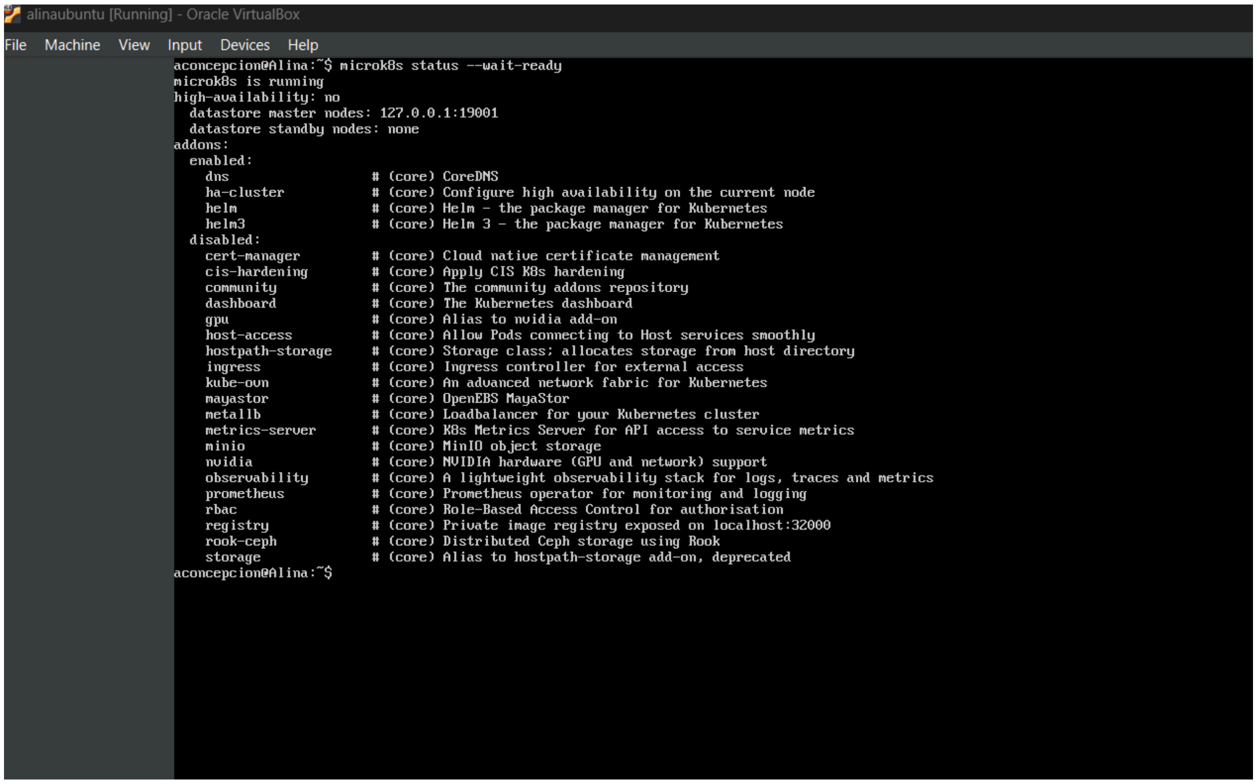
```
maconcepcion@Alina:~$ microk8s kubectl get nodes
NAME    STATUS   ROLES    AGE   VERSION
alina   Ready    <none>   22d   v1.31.3
aconcepcion@Alina:~$ _
```

We can also find out other information such as the type, cluster and external Ip, the ports and age by typing ***microk8s kubectl get services***.


alinaubuntu [Running] - Oracle VirtualBox
File Machine View Input Devices Help

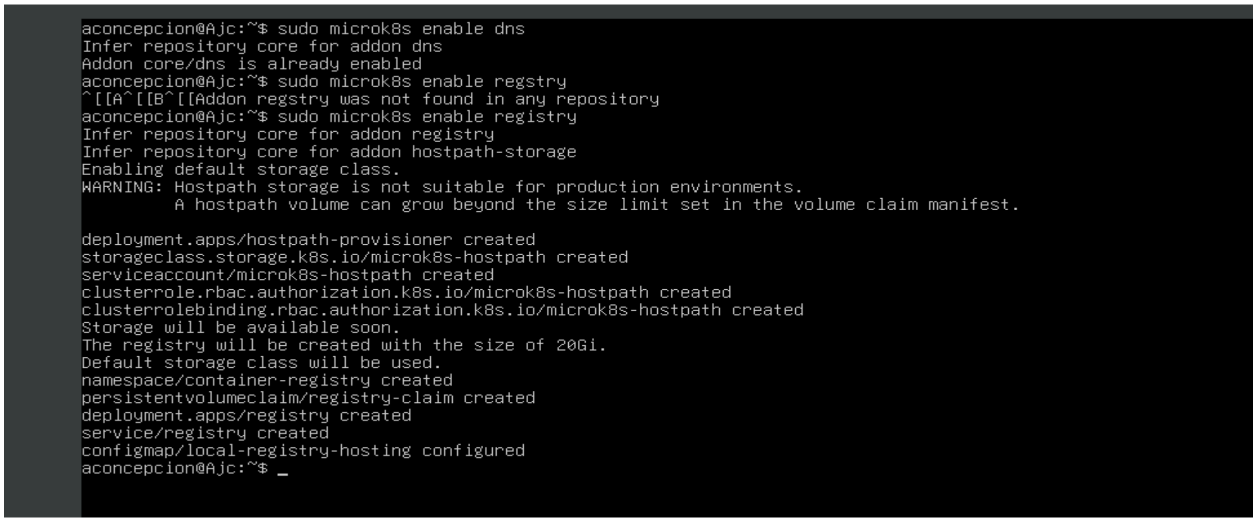
```
aconcepcion@Alina:~$ microk8s kubectl get services
NAME         TYPE        CLUSTER-IP     EXTERNAL-IP   PORT(S)   AGE
kubernetes   ClusterIP   10.152.183.1   <none>        443/TCP   22d
aconcepcion@Alina:~$ _
```

If we want to check the status of Kubernetes, we type ***microk8s status - - wait ready.***

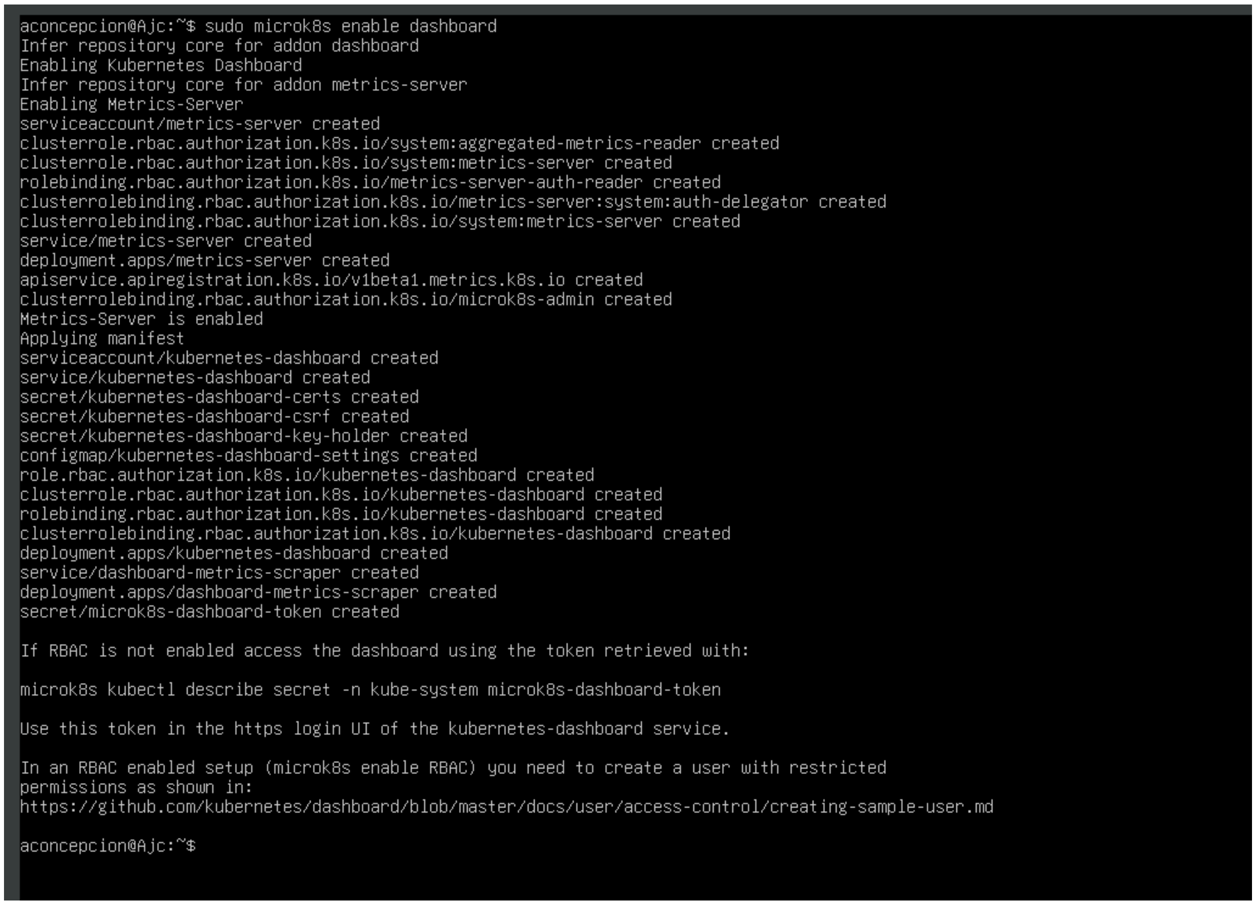

alinaubuntu [Running] - Oracle VirtualBox
File Machine View Input Devices Help

```
aconcepcion@Alina:~$ microk8s status --wait-ready
microk8s is running
high-availability: no
  datastore master nodes: 127.0.0.1:19001
  datastore standby nodes: none
addons:
  enabled:
    dns                  # (core) CoreDNS
    ha-cluster           # (core) Configure high availability on the current node
    helm                 # (core) Helm - the package manager for Kubernetes
    helm3                # (core) Helm 3 - the package manager for Kubernetes
  disabled:
    cert-manager         # (core) Cloud native certificate management
    cis-hardening        # (core) Apply CIS K8s hardening
    community            # (core) The community addons repository
    dashboard            # (core) The Kubernetes dashboard
    gpu                  # (core) Alias to nvidia add-on
    host-access          # (core) Allow Pods connecting to Host services smoothly
    hostpath-storage     # (core) Storage class; allocates storage from host directory
    ingress              # (core) Ingress controller for external access
    kube-ovn             # (core) An advanced network fabric for Kubernetes
    mayastor             # (core) OpenEBS MayaStor
    metallb              # (core) Loadbalancer for your Kubernetes cluster
    metrics-server       # (core) K8s Metrics Server for API access to service metrics
    minio                # (core) MinIO object storage
    nvidia               # (core) NVIDIA hardware (GPU and network) support
    observability        # (core) A lightweight observability stack for logs, traces and metrics
    prometheus           # (core) Prometheus operator for monitoring and logging
    rbac                 # (core) Role-Based Access Control for authorisation
    registry             # (core) Private image registry exposed on localhost:32000
    rook-ceph            # (core) Distributed Ceph storage using Rook
    storage              # (core) Alias to hostpath-storage add-on, deprecated
aconcepcion@Alina:~$
```

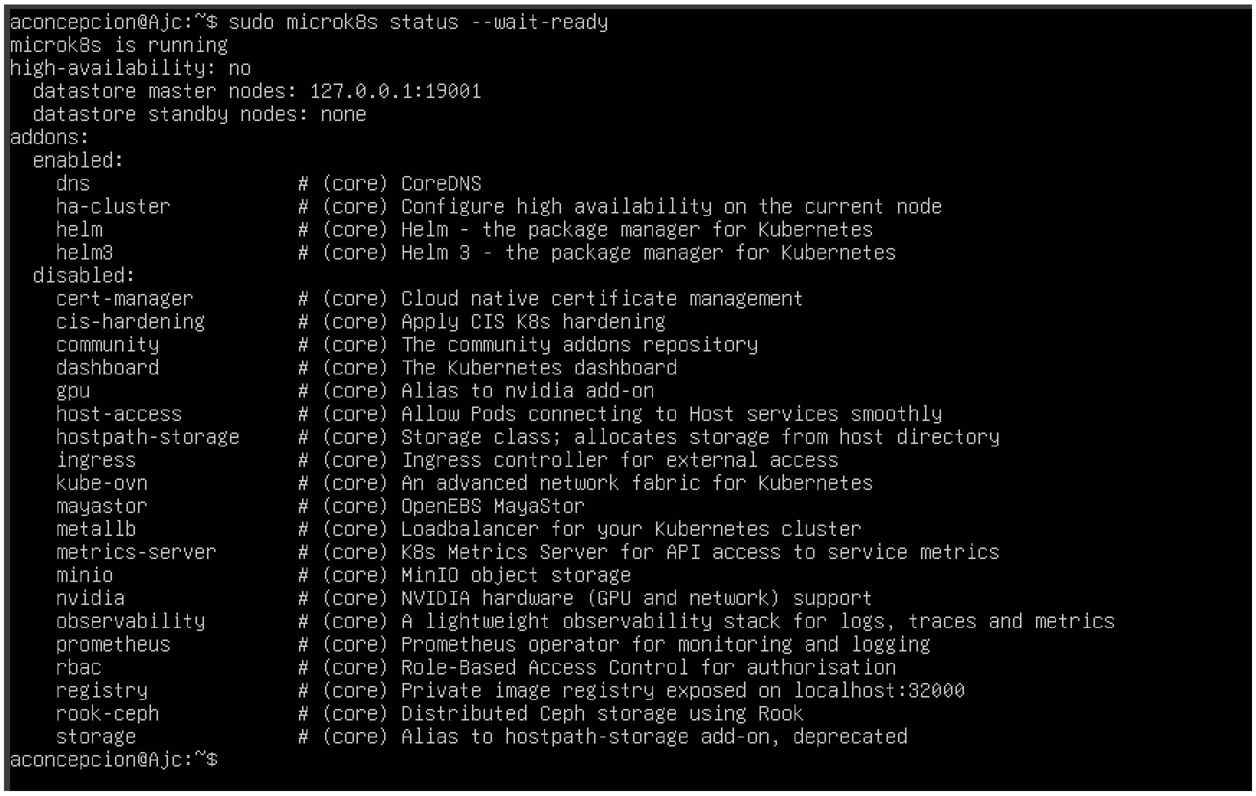
There's other commands that we can use. An example is ***<u>sudo microk8s enable dns</u>.***



```
aconcepcion@Ajc:~$ sudo microk8s enable dns
Infer repository core for addon dns
Addon core/dns is already enabled
aconcepcion@Ajc:~$ sudo microk8s enable regstry
^[[A^[[B^[[Addon regstry was not found in any repository
aconcepcion@Ajc:~$ sudo microk8s enable registry
Infer repository core for addon registry
Infer repository core for addon hostpath-storage
Enabling default storage class.
WARNING: Hostpath storage is not suitable for production environments.
         A hostpath volume can grow beyond the size limit set in the volume claim manifest.

deployment.apps/hostpath-provisioner created
storageclass.storage.k8s.io/microk8s-hostpath created
serviceaccount/microk8s-hostpath created
clusterrole.rbac.authorization.k8s.io/microk8s-hostpath created
clusterrolebinding.rbac.authorization.k8s.io/microk8s-hostpath created
Storage will be available soon.
The registry will be created with the size of 20Gi.
Default storage class will be used.
namespace/container-registry created
persistentvolumeclaim/registry-claim created
deployment.apps/registry created
service/registry created
configmap/local-registry-hosting configured
aconcepcion@Ajc:~$ _
```



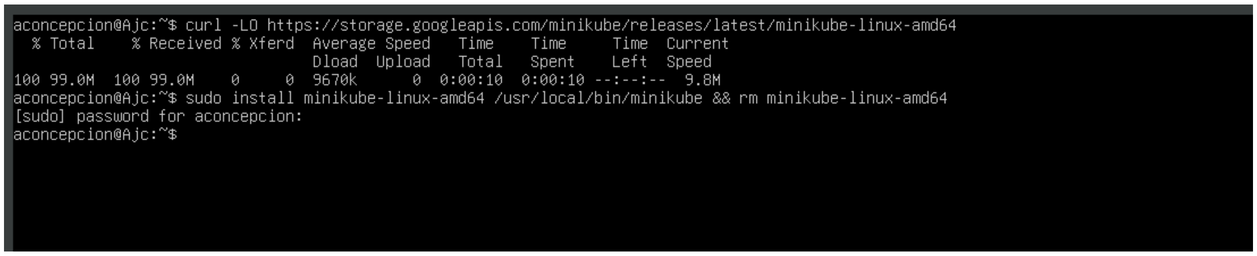
```
aconcepcion@Ajc:~$ sudo microk8s enable dashboard
Infer repository core for addon dashboard
Enabling Kubernetes Dashboard
Infer repository core for addon metrics-server
Enabling Metrics-Server
serviceaccount/metrics-server created
clusterrole.rbac.authorization.k8s.io/system:aggregated-metrics-reader created
clusterrole.rbac.authorization.k8s.io/system:metrics-server created
rolebinding.rbac.authorization.k8s.io/metrics-server-auth-reader created
clusterrolebinding.rbac.authorization.k8s.io/metrics-server:system:auth-delegator created
clusterrolebinding.rbac.authorization.k8s.io/system:metrics-server created
service/metrics-server created
deployment.apps/metrics-server created
apiservice.apiregistration.k8s.io/v1beta1.metrics.k8s.io created
clusterrolebinding.rbac.authorization.k8s.io/microk8s-admin created
Metrics-Server is enabled
Applying manifest
serviceaccount/kubernetes-dashboard created
service/kubernetes-dashboard created
secret/kubernetes-dashboard-certs created
secret/kubernetes-dashboard-csrf created
secret/kubernetes-dashboard-key-holder created
configmap/kubernetes-dashboard-settings created
role.rbac.authorization.k8s.io/kubernetes-dashboard created
clusterrole.rbac.authorization.k8s.io/kubernetes-dashboard created
rolebinding.rbac.authorization.k8s.io/kubernetes-dashboard created
clusterrolebinding.rbac.authorization.k8s.io/kubernetes-dashboard created
deployment.apps/kubernetes-dashboard created
service/dashboard-metrics-scraper created
deployment.apps/dashboard-metrics-scraper created
secret/microk8s-dashboard-token created

If RBAC is not enabled access the dashboard using the token retrieved with:

microk8s kubectl describe secret -n kube-system microk8s-dashboard-token

Use this token in the https login UI of the kubernetes-dashboard service.

In an RBAC enabled setup (microk8s enable RBAC) you need to create a user with restricted
permissions as shown in:
https://github.com/kubernetes/dashboard/blob/master/docs/user/access-control/creating-sample-user.md

aconcepcion@Ajc:~$
```



```
aconcepcion@Ajc:~$ sudo microk8s status --wait-ready
microk8s is running
high-availability: no
  datastore master nodes: 127.0.0.1:19001
  datastore standby nodes: none
addons:
  enabled:
    dns                  # (core) CoreDNS
    ha-cluster           # (core) Configure high availability on the current node
    helm                 # (core) Helm - the package manager for Kubernetes
    helm3                # (core) Helm 3 - the package manager for Kubernetes
  disabled:
    cert-manager         # (core) Cloud native certificate management
    cis-hardening        # (core) Apply CIS K8s hardening
    community            # (core) The community addons repository
    dashboard            # (core) The Kubernetes dashboard
    gpu                  # (core) Alias to nvidia add-on
    host-access          # (core) Allow Pods connecting to Host services smoothly
    hostpath-storage     # (core) Storage class; allocates storage from host directory
    ingress              # (core) Ingress controller for external access
    kube-ovn             # (core) An advanced network fabric for Kubernetes
    mayastor             # (core) OpenEBS MayaStor
    metallb              # (core) Loadbalancer for your Kubernetes cluster
    metrics-server       # (core) K8s Metrics Server for API access to service metrics
    minio                # (core) MinIO object storage
    nvidia               # (core) NVIDIA hardware (GPU and network) support
    observability        # (core) A lightweight observability stack for logs, traces and metrics
    prometheus           # (core) Prometheus operator for monitoring and logging
    rbac                 # (core) Role-Based Access Control for authorisation
    registry             # (core) Private image registry exposed on localhost:32000
    rook-ceph            # (core) Distributed Ceph storage using Rook
    storage              # (core) Alias to hostpath-storage add-on, deprecated
aconcepcion@Ajc:~$
```

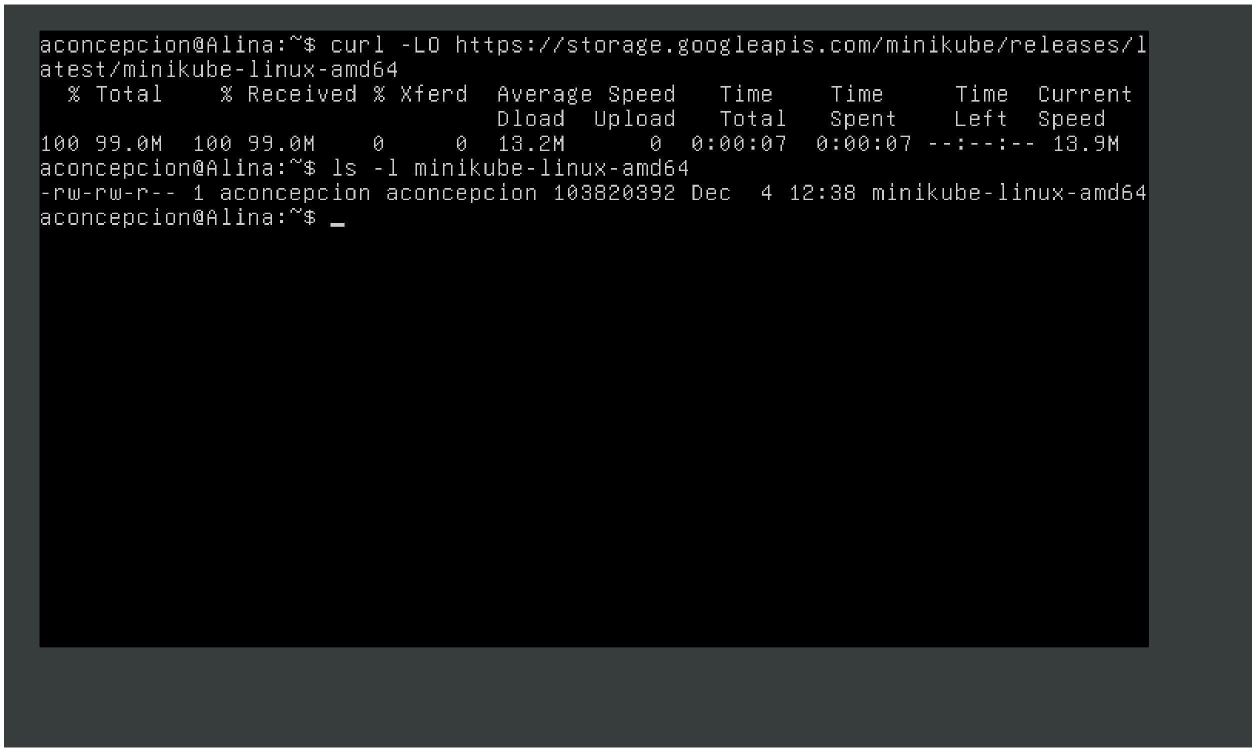
Minikube is a free, open-source tool that allows users to set up a Kubernetes environment on their local computer (minikube.sigs.k8s.io).

To install minicube, you type curl -LO ***https://storage.googlepis.com/minikube/release/latest/minikube-linux-amd64*** . Next, you type ***sudo install minikube-linux-amd64 /usr/local/bin/minikube && rm minikube-linux-amd64.***



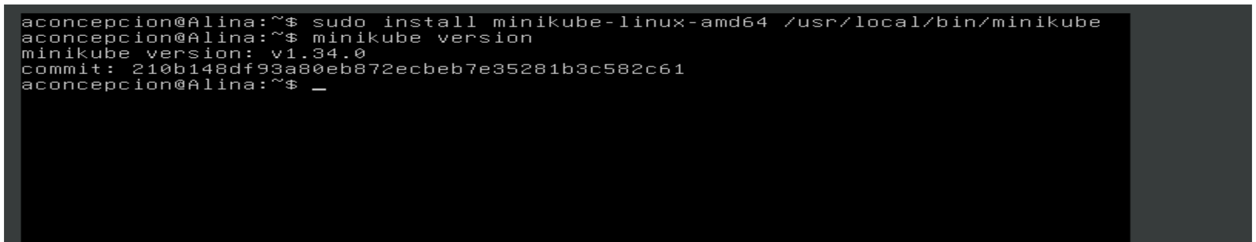
```
aconcepcion@Ajc:~$ curl -LO https://storage.googleapis.com/minikube/releases/latest/minikube-linux-amd64
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100 99.0M  100 99.0M    0     0  9670k      0  0:00:10  0:00:10 --:--:--  9.8M
aconcepcion@Ajc:~$ sudo install minikube-linux-amd64 /usr/local/bin/minikube && rm minikube-linux-amd64
[sudo] password for aconcepcion:
aconcepcion@Ajc:~$
```

I used ls to show the successful installation of minikube.



```
aconcepcion@Alina:~$ curl -LO https://storage.googleapis.com/minikube/releases/l
atest/minikube-linux-amd64
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100 99.0M  100 99.0M    0     0  13.2M      0  0:00:07  0:00:07 --:--:-- 13.9M
aconcepcion@Alina:~$ ls -l minikube-linux-amd64
-rw-rw-r-- 1 aconcepcion aconcepcion 103820392 Dec  4 12:38 minikube-linux-amd64
aconcepcion@Alina:~$ _
```

Use the ***minikube version*** command to display the minikube version that you have

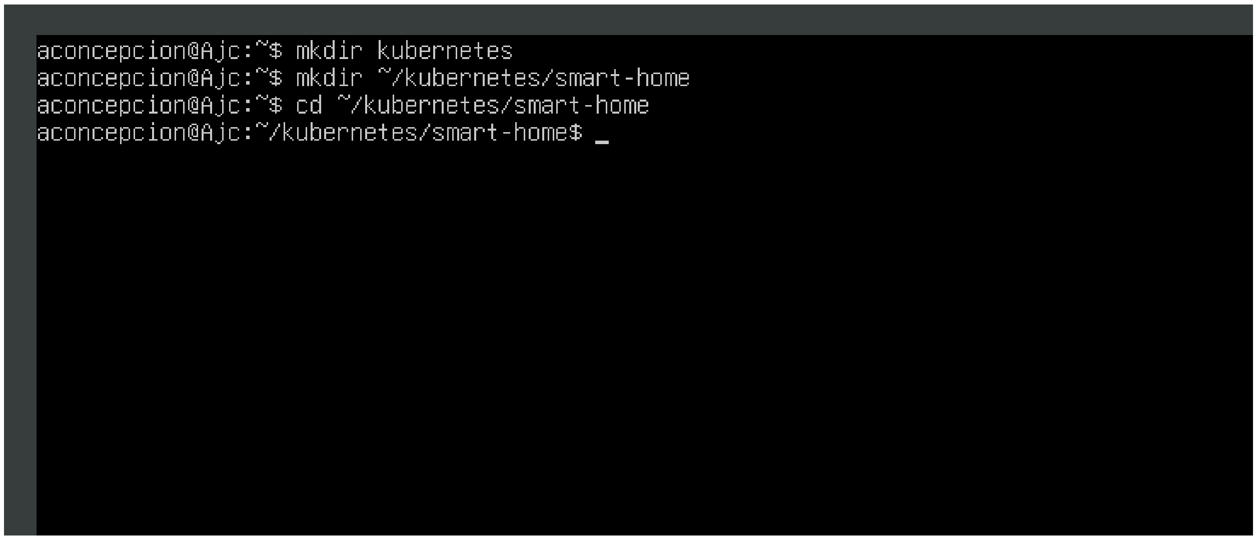


```
aconcepcion@Alina:~$ sudo install minikube-linux-amd64 /usr/local/bin/minikube
aconcepcion@Alina:~$ minikube version
minikube version: v1.34.0
commit: 210b148df93a80eb872ecbeb7e35281b3c582c61
aconcepcion@Alina:~$ _
```

Kubernetes and YAML

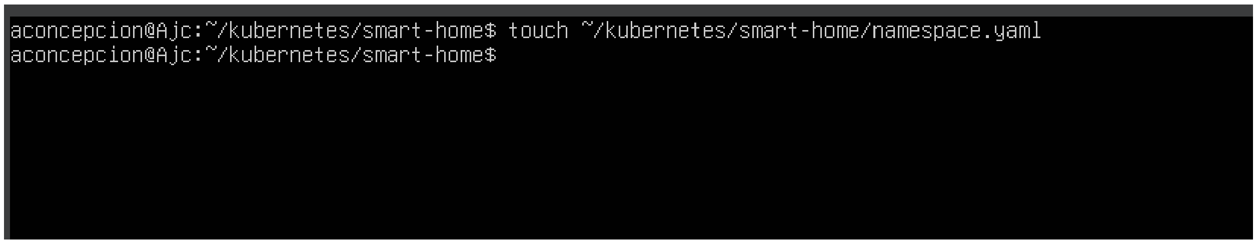
"Kubectl is a tool that Kubernetes uses to be able to communicate with the cluster." it is like a walkie talkie. Kubectl will be used on any computer that can talk to your server that is running Kubernetes.

We can see if it is working by typing the ***kubectl cluster-info*** command. After doing that we will be setting up a namespace,which is a way to organize our things. We will call the namespace smart-home. To create our namespace we use the ***mkdir command.*** In this screenshot, I used mkdir and made a directory called kubernetes then made another directory inside kubernetes, called smart-home, then I used cd to change into that directory.



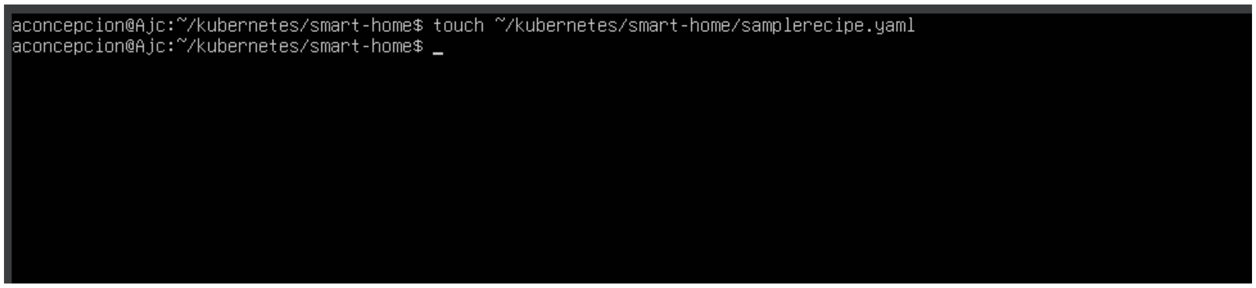
```
aconcepcion@Ajc:~$ mkdir kubernetes
aconcepcion@Ajc:~$ mkdir ~/kubernetes/smart-home
aconcepcion@Ajc:~$ cd ~/kubernetes/smart-home
aconcepcion@Ajc:~/kubernetes/smart-home$ _
```

After changing into the smart-home directory, I made a yaml file called namespace.yaml, using the touch command***. Touch ~/kubernetes/smart-home/namespace.yaml***



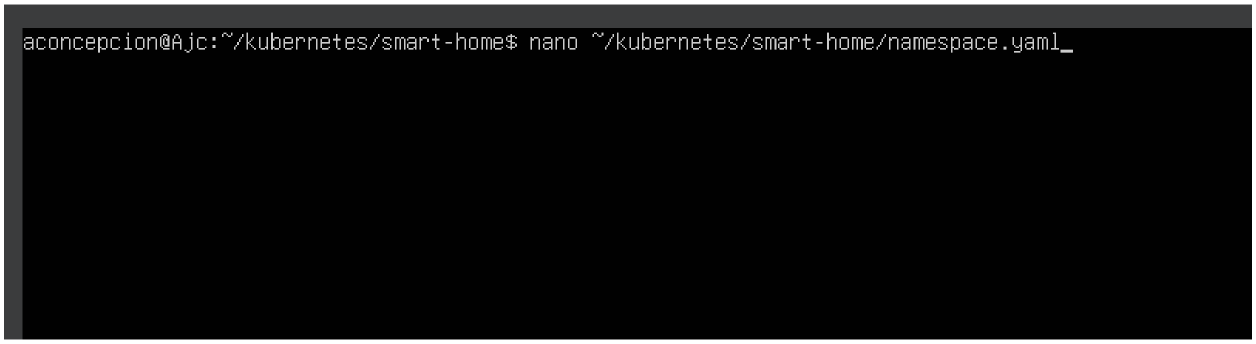
```
aconcepcion@Ajc:~/kubernetes/smart-home$ touch ~/kubernetes/smart-home/namespace.yaml
aconcepcion@Ajc:~/kubernetes/smart-home$
```

Next, I used the touch command to make another yaml file, called samplerecipe.yaml. I used touch ~/kubernetes/smart-home/samplerecipe.yaml



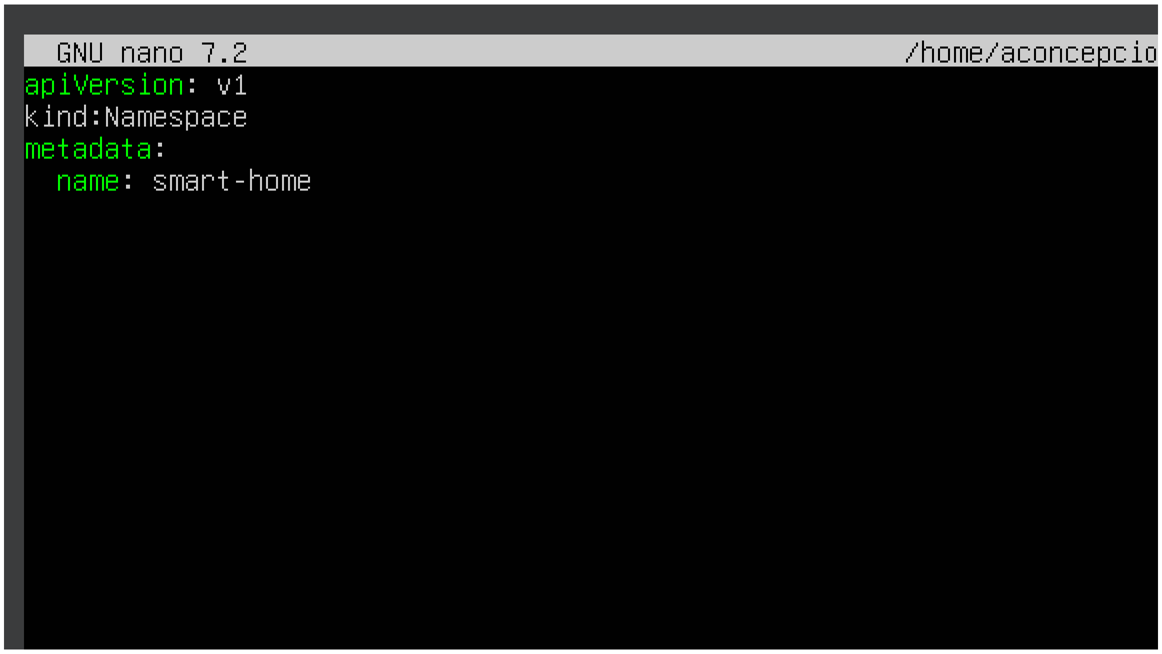
```
aconcepcion@Ajc:~/kubernetes/smart-home$ touch ~/kubernetes/smart-home/samplerecipe.yaml
aconcepcion@Ajc:~/kubernetes/smart-home$ _
```

For the name-space.yaml, I used nano and the script at
https://www.aholdengouveia.name/SmartHome/namespace.yaml .



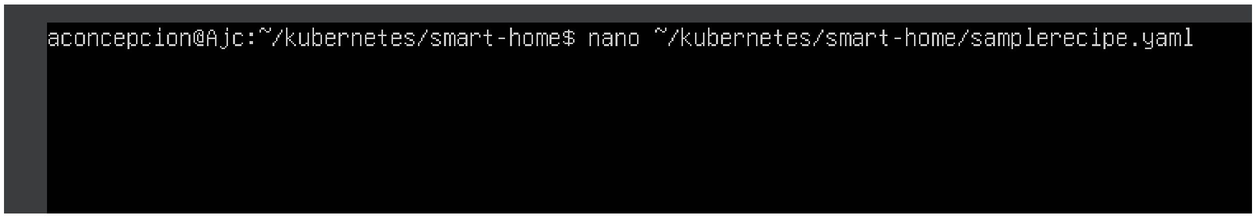
```
aconcepcion@Ajc:~/kubernetes/smart-home$ nano ~/kubernetes/smart-home/namespace.yaml_
```

Here's what the script should look like when you put it in the yaml file.



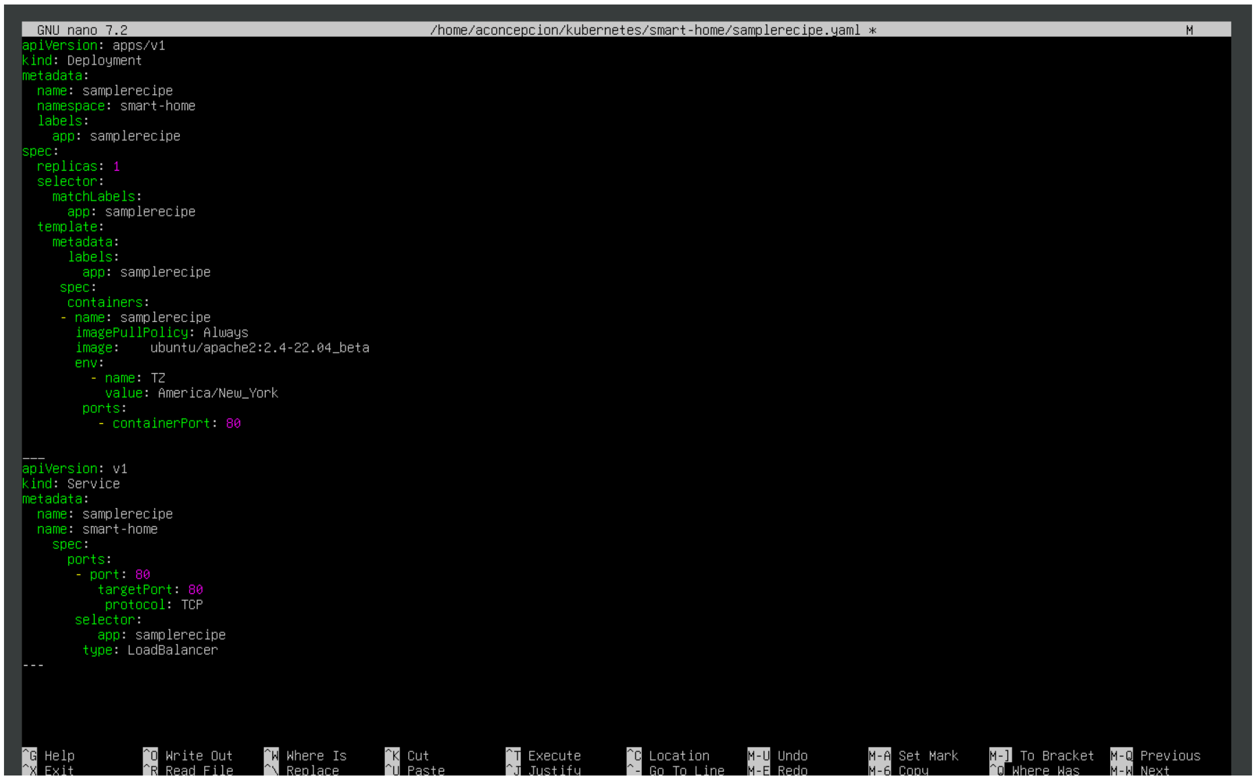
```
  GNU nano 7.2                                               /home/aconcepcio
apiVersion: v1
kind:Namespace
metadata:
  name: smart-home
```

Next, is the samplerecipe yaml file.



```
aconcepcion@Ajc:~/kubernetes/smart-home$ nano ~/kubernetes/smart-home/samplerecipe.yaml
```

Here is what we want the file to look like. Next we hit control x to save it then the letter y.



```
  GNU nano 7.2                          /home/aconcepcion/kubernetes/smart-home/samplerecipe.yaml *                          M
apiVersion: apps/v1
kind: Deployment
metadata:
  name: samplerecipe
  namespace: smart-home
  labels:
    app: samplerecipe
spec:
  replicas: 1
  selector:
    matchLabels:
      app: samplerecipe
  template:
    metadata:
      labels:
        app: samplerecipe
     spec:
      containers:
     - name: samplerecipe
       imagePullPolicy: Always
       image:    ubuntu/apache2:2.4-22.04_beta
       env:
         - name: TZ
           value: America/New_York
        ports:
          - containerPort: 80

---
apiVersion: v1
kind: Service
metadata:
  name: samplerecipe
  name: smart-home
    spec:
      ports:
       - port: 80
          targetPort: 80
           protocol: TCP
       selector:
          app: samplerecipe
        type: LoadBalancer
---




^G Help        ^O Write Out   ^W Where Is    ^K Cut         ^T Execute     ^C Location    M-U Undo       M-A Set Mark   M-] To Bracket M-Q Previous
^X Exit        ^R Read File   ^\ Replace     ^U Paste       ^J Justify     ^- Go To Line  M-E Redo       M-6 Copy       ^Q Where Was   M-W Next
```

I ran into some errors after this part. I believe it's a syntax error which I've tried debugging .. But to get our yaml files up and running, we use ***kubectl apply -f ~/kubernetes/smart-home/namespace.yaml*** . A message saying "home created" should appear indicating that it is successful. To confirm that the namespace creation is successful we use ***kubectl get namespace*** which will tell us the name of the name space, age and status. To finish up the setup, we will use the ***kubectl apply -f ~/kubernetes/smart-home/samplerecipe.yaml*** Command which sends the instructions to create our container yaml.To confirm that this is successful we use the ***kubectl get pods -n smart-home*** command, then we use ***kubectl get services -n smart-home*** which will show the name, type, eternal and internal ips, cluster type, posts and age.

Sources:
Docker Tutorial - Running A Web Server

Minikube and Kubectl explained | Setup for Beginners | Kubernetes Tutorial 17

https://docker-curriculum.com/
The intro to Docker I wish I had when I started
Learn Docker in 7 Easy Steps - Full Beginner's Tutorial
https://microk8s.io/
https://www.aholdengouveia.name/SmartHome/Virtualizationsetup.html