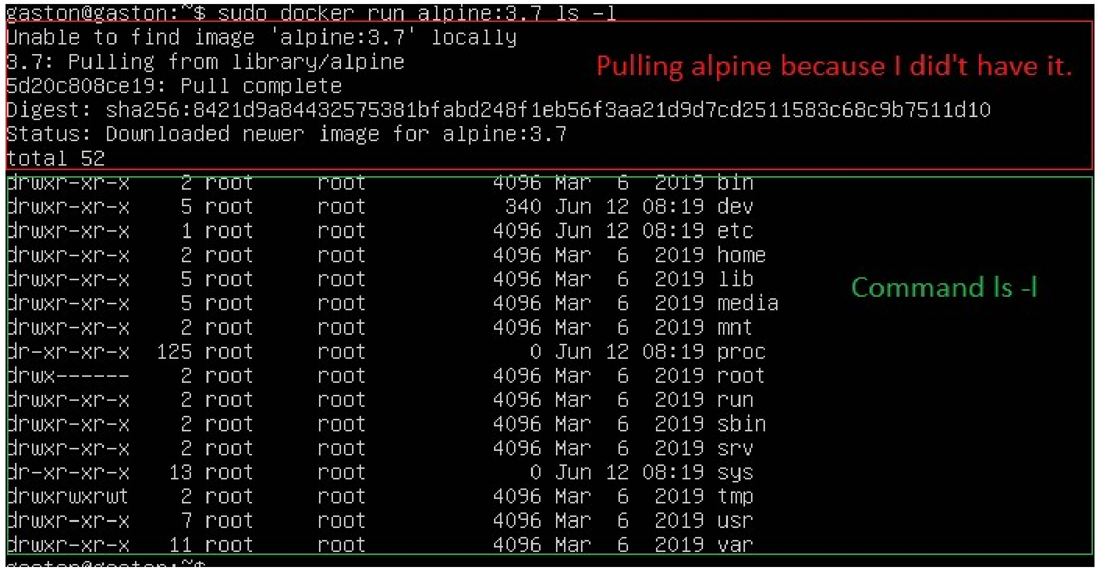
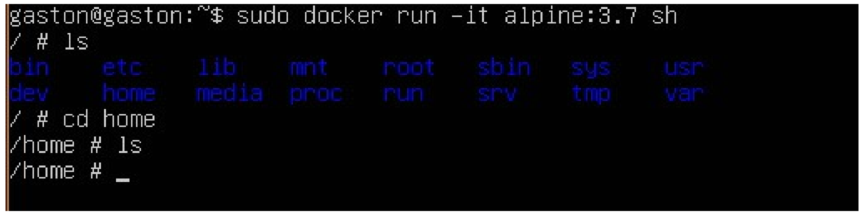
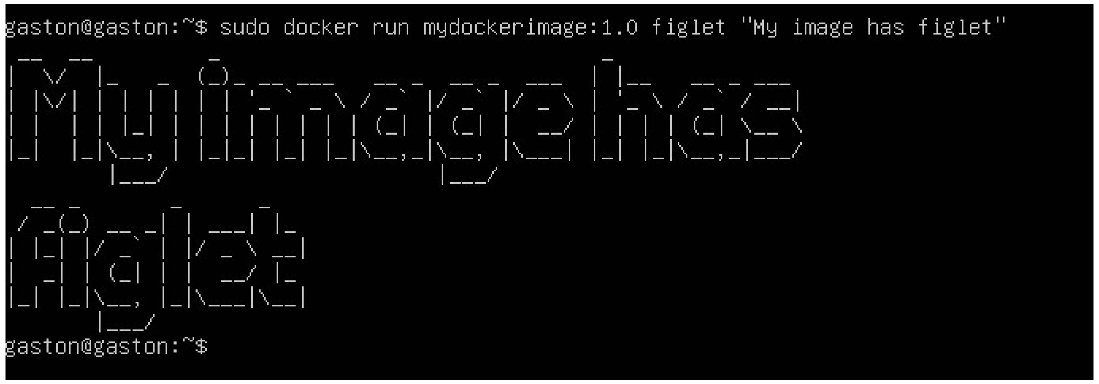
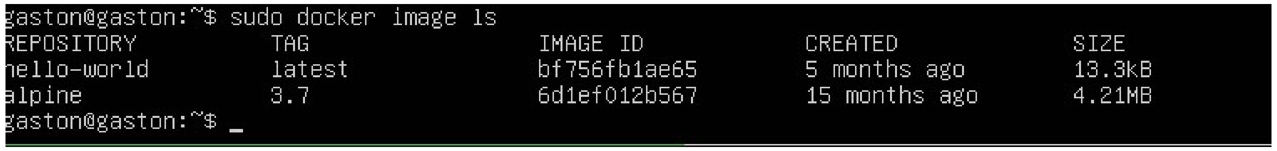
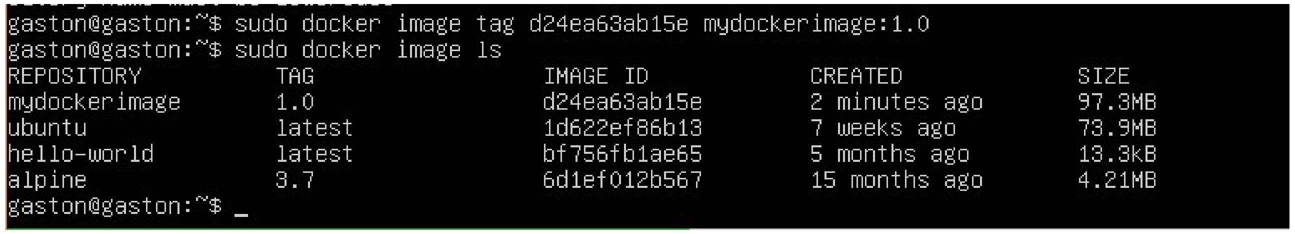
Command	**Description**	**Key word**
docker version	To show docker version	Version
docker ps	To show the conteiners that are running, you can run also docker ps -a | head head = the last ten. a = Also the containers that they are not running.	Show
docker run nameOfTheImage	Run the container, example: docker run alpine:3.7 ls -l and I can see all the files  gaston@gaston:~$ sudo docker run alpine:3.7 ls -l Unable to find image 'alpine:3.7' locally 3.7: Pulling from library/alpine 5d20c808ce19: Pull complete Digest: sha256:8421d9a84432575381bfabd248f1eb56f3aa21d9d7cd2511583c68c9b7511d10 Status: Downloaded newer image for alpine:3.7 total 52 drwxr-xr-x 2 root root 4096 Mar 6 2019 bin drwxr-xr-x 5 root root 340 Jun 12 08:19 dev drwxr-xr-x 1 root root 4096 Jun 12 08:19 etc drwxr-xr-x 2 root root 4096 Mar 6 2019 home drwxr-xr-x 5 root root 4096 Mar 6 2019 lib drwxr-xr-x 5 root root 4096 Mar 6 2019 media drwxr-xr-x 2 root root 4096 Mar 6 2019 mnt dr-xr-xr-x 125 root root 0 Jun 12 08:19 proc drwx------ 2 root root 4096 Mar 6 2019 root drwxr-xr-x 2 root root 4096 Mar 6 2019 run drwxr-xr-x 2 root root 4096 Mar 6 2019 sbin drwxr-xr-x 2 root root 4096 Mar 6 2019 srv dr-xr-xr-x 13 root root 0 Jun 12 08:19 sys drwxrwxrwt 2 root root 4096 Mar 6 2019 tmp drwxr-xr-x 7 root root 4096 Mar 6 2019 usr drwxr-xr-x 11 root root 4096 Mar 6 2019 var gaston@gaston:~$ Pulling alpine because I did't have it. Command ls -l	Run
docker run -d nameOfTheImage	With -d I run the container and it continues running , we use -d when we run for example a nginx service.	Run
docker start idOfCointaner.	To start a container.	Start
docker stop idOfCointaner.	To stop a container.	Stop
docker rm idOfCointaner.	To remove a container.	Remove
docker pull nameOfTheImage	To download a docker image	Pull / Download
docker run -it nameOfTheImage sh docker run -it nameOfTheImage /bin/bash	To run a shell in the container. i = interactive  gaston@gaston:~$ sudo docker run -it alpine:3.7 sh / # ls bin etc lib mnt root sbin sys usr dev home media proc run srv tmp var / # cd home /home # ls /home # _	Run
docker exec -it IdOfTheContainer sh docker exec -it IdOfTheContainer bin/bash	To execute a container that it is running.	Execute

docker commit IdOfTheContainer	To create a new image with our changes. Example: I ran an alpine container, I installed new packages and then I close the container session, I can see the container that were running (docker ps -a) and create an image with the command commit. Other example, You can install in ubuntu image figlet and then create your own image and run it :  In this example the image is taged with a name and version.	commit
docker image ls docker images	To show our images: 	Show
docker rmi idOfTheImage	To remove an image.	Remove
docker image tag IdOfTheImage	To tag the image, you can run the command docker image tag IdOfTheImage myDockerImage:1.0 or docker image tag IdOfTheImage myDockerImage , if you don’t especify the version the default is latest. 	Tag

```
gaston@gaston:~$ sudo docker run mydockerimage:1.0 figlet "My image has figlet"
 __  __         _                               _
|  \/  |_   _  (_)_ __ ___   __ _  __ _  ___  | |__   __ _ ___
| |\/| | | | | | | '_ ` _ \ / _` |/ _` |/ _ \ | '_ \ / _` / __|
| |  | | |_| | | | | | | | | (_| | (_| |  __/ | | | | (_| \__ \
|_|  |_|\__, | |_|_| |_| |_|\__,_|\__, |\___| |_| |_|\__,_|___/
        |___/                     |___/
  __ _       _      _
 / _(_) __ _| | ___| |_
| |_| |/ _` | |/ _ \ __|
|  _| | (_| | |  __/ |_
|_| |_|\__, |_|\___|\__|
       |___/
gaston@gaston:~$
```

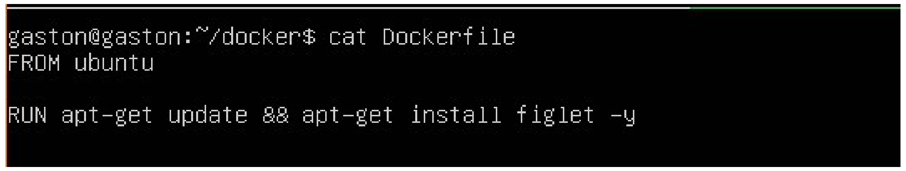
```
gaston@gaston:~$ sudo docker image ls
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
hello-world         latest              bf756fb1ae65        5 months ago        13.3kB
alpine              3.7                 6d1ef012b567        15 months ago       4.21MB
gaston@gaston:~$ _
```

```
gaston@gaston:~$ sudo docker image tag d24ea63ab15e mydockerimage:1.0
gaston@gaston:~$ sudo docker image ls
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
mydockerimage       1.0                 d24ea63ab15e        2 minutes ago       97.3MB
ubuntu              latest              1d622ef86b13        7 weeks ago         73.9MB
hello-world         latest              bf756fb1ae65        5 months ago        13.3kB
alpine              3.7                 6d1ef012b567        15 months ago       4.21MB
gaston@gaston:~$ _
```

<table>
<tr>
<td>Dockerfile</td>
<td>

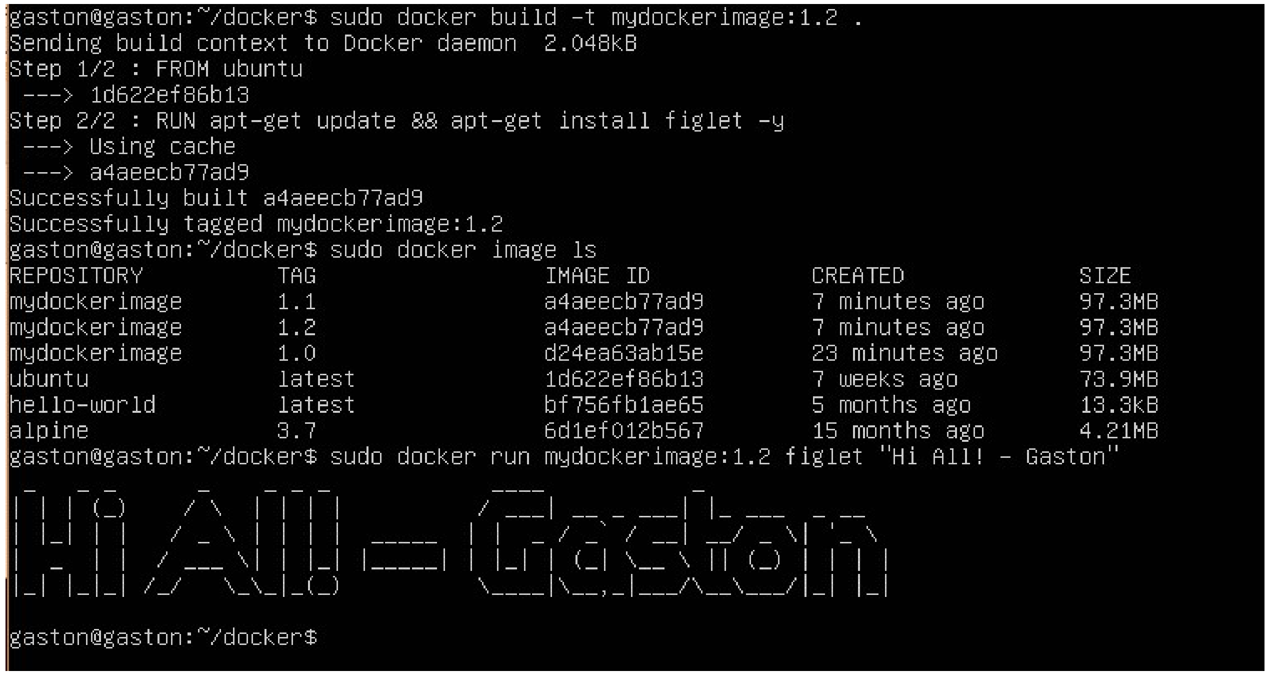
Dockerfile is a set of instuctions to create images.

I create a file with instructions that update ubuntu and the install figlet:



```
gaston@gaston:~/docker$ cat Dockerfile
FROM ubuntu

RUN apt-get update && apt-get install figlet -y
```

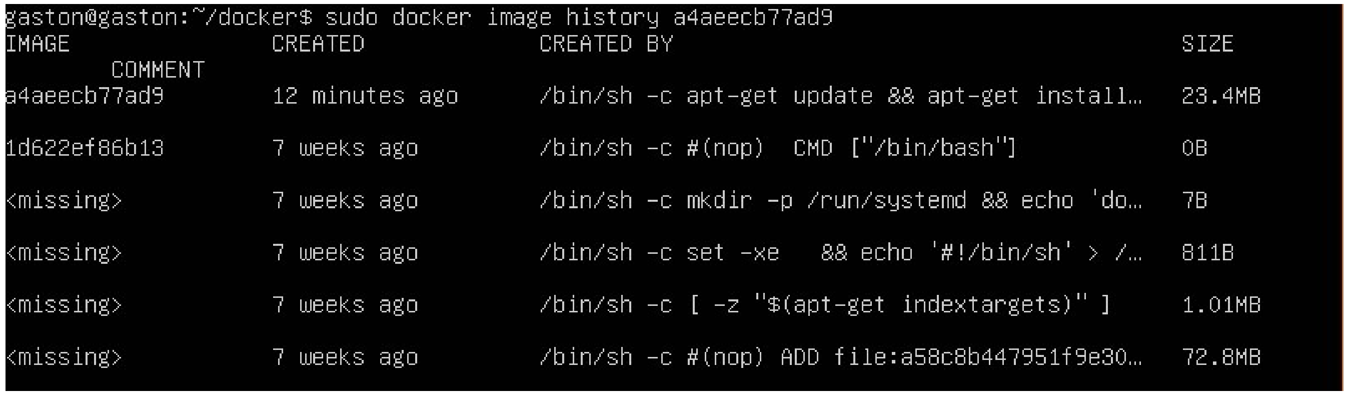
Now I build a new image, check if was created and run to check if it is woking:



```
gaston@gaston:~/docker$ sudo docker build -t mydockerimage:1.2 .
Sending build context to Docker daemon  2.048kB
Step 1/2 : FROM ubuntu
 ---> 1d622ef86b13
Step 2/2 : RUN apt-get update && apt-get install figlet -y
 ---> Using cache
 ---> a4aeecb77ad9
Successfully built a4aeecb77ad9
Successfully tagged mydockerimage:1.2
gaston@gaston:~/docker$ sudo docker image ls
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
mydockerimage       1.1                 a4aeecb77ad9        7 minutes ago       97.3MB
mydockerimage       1.2                 a4aeecb77ad9        7 minutes ago       97.3MB
mydockerimage       1.0                 d24ea63ab15e        23 minutes ago      97.3MB
ubuntu              latest              1d622ef86b13        7 weeks ago         73.9MB
hello-world         latest              bf756fb1ae65        5 months ago        13.3kB
alpine              3.7                 6d1ef012b567        15 months ago       4.21MB
gaston@gaston:~/docker$ sudo docker run mydockerimage:1.2 figlet "Hi All! - Gaston"
 _   _ _      _    _ _ _          ____           _
| | | (_)    / \  | | | |        / ___| __ _ ___| |_ ___  _ __
| |_| | |   / _ \ | | | |  _____| |  _ / _` / __| __/ _ \| '_ \
|  _  | |  / ___ \| | |_| |_____| |_| | (_| \__ \ || (_) | | | |
|_| |_|_| /_/   \_\_|_(_)        \____|\__,_|___/\__\___/|_| |_|

gaston@gaston:~/docker$
```

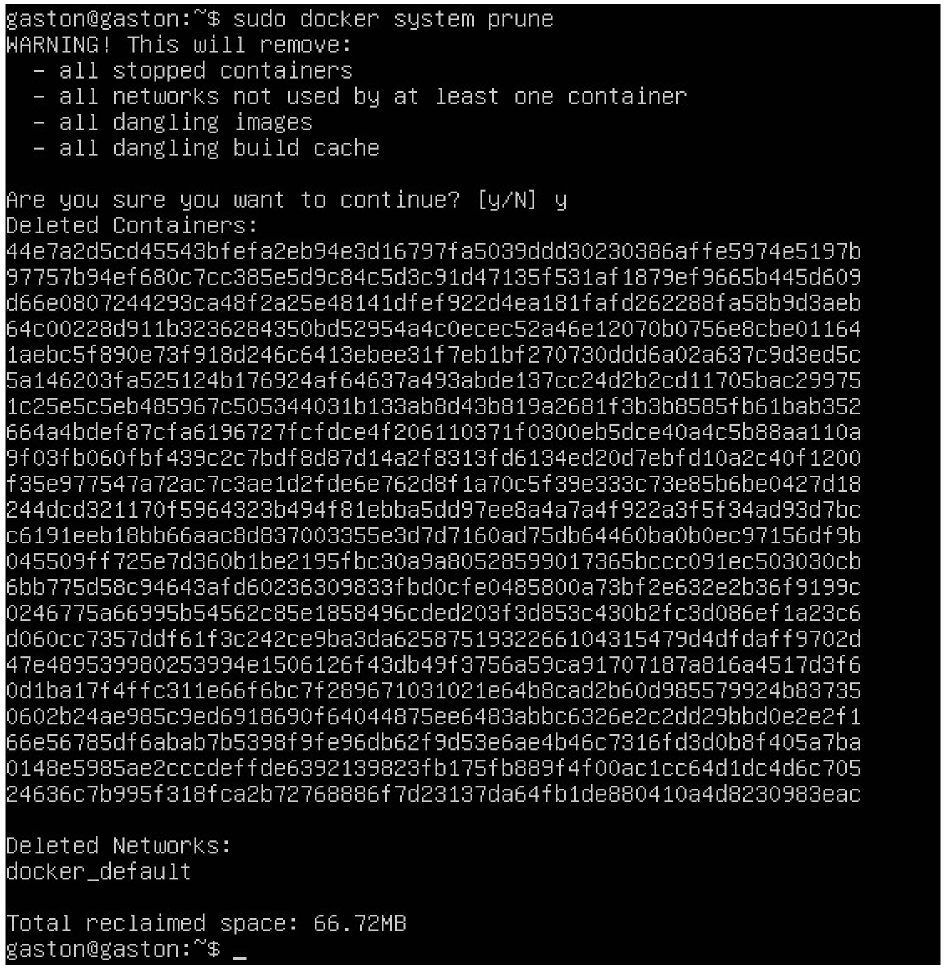
</td>
<td>Dockerfile / Build</td>
</tr>
<tr>
<td>docker image history IdOfTheImage</td>
<td>

To check the history of the image:



```
gaston@gaston:~/docker$ sudo docker image history a4aeecb77ad9
IMAGE               CREATED             CREATED BY                                      SIZE
        COMMENT
a4aeecb77ad9        12 minutes ago      /bin/sh -c apt-get update && apt-get install…   23.4MB

1d622ef86b13        7 weeks ago         /bin/sh -c #(nop)  CMD ["/bin/bash"]            0B

<missing>           7 weeks ago         /bin/sh -c mkdir -p /run/systemd && echo 'do…   7B

<missing>           7 weeks ago         /bin/sh -c set -xe   && echo '#!/bin/sh' > /…   811B

<missing>           7 weeks ago         /bin/sh -c [ -z "$(apt-get indextargets)" ]     1.01MB

<missing>           7 weeks ago         /bin/sh -c #(nop) ADD file:a58c8b447951f9e30…   72.8MB
```

</td>
<td>History</td>
</tr>
<tr>
<td>docker logs IdOfTheContainer</td>
<td>To check the logs of the container. To add the time you can run docker logs -t idOfTheContainer</td>
<td>Logs</td>
</tr>
<tr>
<td>docker stats IdOfTheContainer</td>
<td>To check the stats of the container.</td>
<td>Stats</td>
</tr>
</table>

<table>
<tr><td>docker system prune</td><td>To clean all what you are not using:

<pre>
gaston@gaston:~$ sudo docker system prune
WARNING! This will remove:
  - all stopped containers
  - all networks not used by at least one container
  - all dangling images
  - all dangling build cache

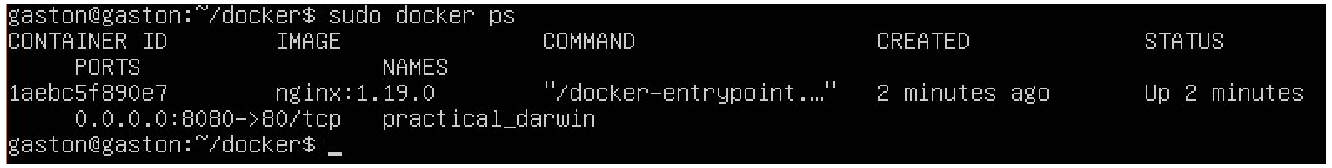
Are you sure you want to continue? [y/N] y
Deleted Containers:
44e7a2d5cd45543bfefa2eb94e3d16797fa5039ddd30230386affe5974e5197b
97757b94ef680c7cc385e5d9c84c5d3c91d47135f531af1879ef9665b445d609
d66e0807244293ca48f2a25e48141dfef922d4ea181fafd262288fa58b9d3aeb
64c00228d911b3236284350bd52954a4c0ecec52a46e12070b0756e8cbe01164
1aebc5f890e73f918d246c6413ebee31f7eb1bf270730ddd6a02a637c9d3ed5c
5a146203fa525124b176924af64637a493abde137cc24d2b2cd11705bac29975
1c25e5c5eb485967c505344031b133ab8d43b819a2681f3b3b8585fb61bab352
664a4bdef87cfa6196727fcfdce4f206110371f0300eb5dce40a4c5b88aa110a
9f03fb060fbf439c2c7bdf8d87d14a2f8313fd6134ed20d7ebfd10a2c40f1200
f35e977547a72ac7c3ae1d2fde6e762d8f1a70c5f39e333c73e85b6be0427d18
244dcd321170f5964323b494f81ebba5dd97ee8a4a7a4f922a3f5f34ad93d7bc
c6191eeb18bb66aac8d837003355e3d7d7160ad75db64460ba0b0ec97156df9b
045509ff725e7d360b1be2195fbc30a9a80528599017365bccc091ec503030cb
6bb775d58c94643afd60236309833fbd0cfe0485800a73bf2e632e2b36f9199c
0246775a66995b54562c85e1858496cded203f3d853c430b2fc3d086ef1a23c6
d060cc7357ddf61f3c242ce9ba3da625875193226610431547 9d4dfdaff9702d
47e489539980253994e1506126f43db49f3756a59ca91707187a816a4517d3f6
0d1ba17f4ffc311e66f6bc7f289671031021e64b8cad2b60d985579924b83735
0602b24ae985c9ed6918690f64044875ee6483abbc6326e2c2dd29bbd0e2e2f1
66e56785df6abab7b5398f9fe96db62f9d53e6ae4b46c7316fd3d0b8f405a7ba
0148e5985ae2cccdeffde6392139823fb175fb889f4f00ac1cc64d1dc4d6c705
24636c7b995f318fca2b72768886f7d23137da64fb1de880410a4d8230983eac

Deleted Networks:
docker_default

Total reclaimed space: 66.72MB
gaston@gaston:~$ _
</pre></td><td>Clean / Remove</td></tr>
<tr><td>docker inspect IdOfTheContainer</td><td>Give you information about the container (variables, ip , ports and more)</td><td></td></tr>
<tr><td>docker cp</td><td>To copy files from the container to the host or from the host to the container.

docker cp idOfTheContainer:/folderContainer/Subfolder/file.txt /home/gaston/folderhost/
docker cp /home/gaston/folderhost/file.txt idOfTheContainer:/folderContainer/Subfolder/</td><td>Copy</td></tr>
<tr><td>docker add</td><td>To add files from a url to the container.

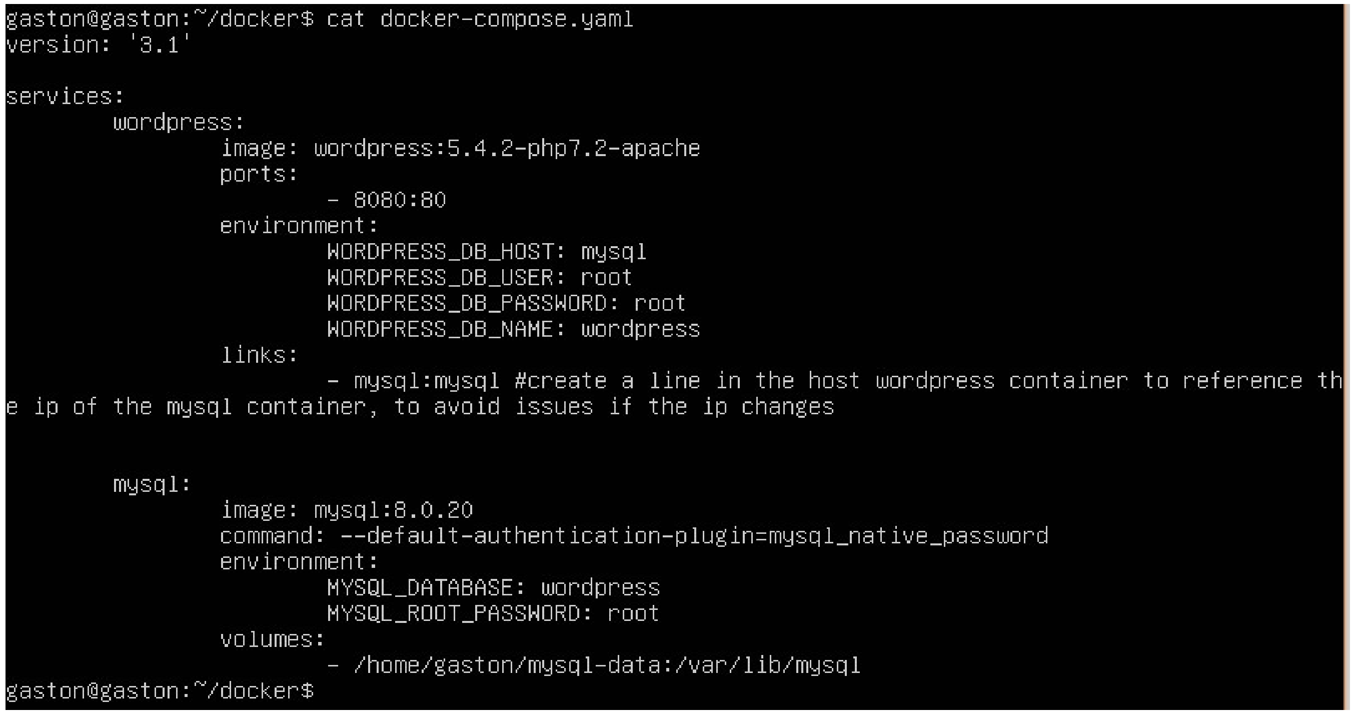
docker add mysite.com/file /files/</td><td>Add</td></tr>
</table>

Volume	When you run a container, you can create a volume to save files in your host server, for example: docker run -v home/gaston/docker/index.html:/usr/share/nginx/html/index.html:ro -d nginx:1.19.0 docker run -v path_inside_the_host:path_inside_the_container:xx -d web_server_image:version -v = volume -d = run and continue running. ro = read only	Volume
Ports	To open a port you can add in the docker run a new parameter: docker run -v home/gaston/docker/index.html:/usr/share/nginx/html/index.html:ro -p 8080:80 -d nginx:1.19.0 -p host_port:container_port A now you can check from your host server your site in the container. 	Ports

```
gaston@gaston:~/docker$ sudo docker ps
CONTAINER ID        IMAGE               COMMAND                  CREATED             STATUS
     PORTS                  NAMES
1aebc5f890e7        nginx:1.19.0        "/docker-entrypoint.…"   2 minutes ago       Up 2 minutes
     0.0.0.0:8080->80/tcp   practical_darwin
gaston@gaston:~/docker$ _
```

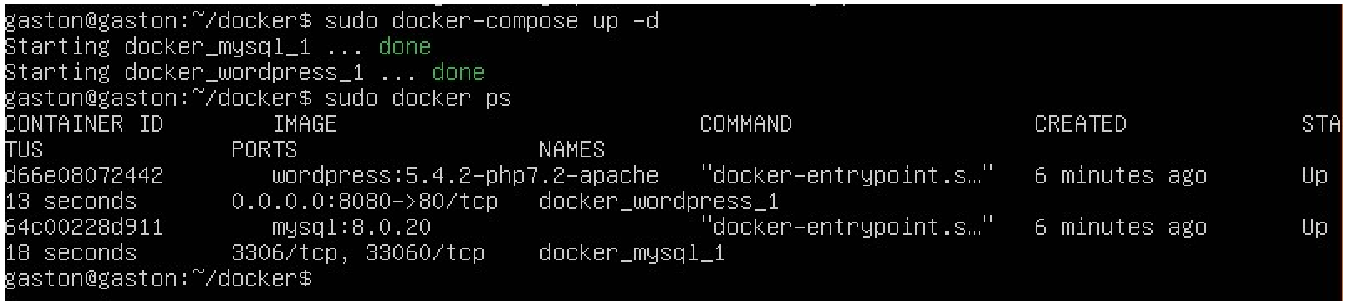
<table>
<tr>
<td>Docker compose</td>
<td>

You can create a yaml script to work in a more organized way and then run the yaml to create the container:



```
gaston@gaston:~/docker$ cat docker-compose.yaml
version: '3.1'

services:
        wordpress:
                image: wordpress:5.4.2-php7.2-apache
                ports:
                        - 8080:80
                environment:
                        WORDPRESS_DB_HOST: mysql
                        WORDPRESS_DB_USER: root
                        WORDPRESS_DB_PASSWORD: root
                        WORDPRESS_DB_NAME: wordpress
                links:
                        - mysql:mysql #create a line in the host wordpress container to reference th
e ip of the mysql container, to avoid issues if the ip changes


        mysql:
                image: mysql:8.0.20
                command: --default-authentication-plugin=mysql_native_password
                environment:
                        MYSQL_DATABASE: wordpress
                        MYSQL_ROOT_PASSWORD: root
                volumes:
                        - /home/gaston/mysql-data:/var/lib/mysql
gaston@gaston:~/docker$
```

And then run your yaml script with the command docker-compose up -d



```
gaston@gaston:~/docker$ sudo docker-compose up -d
Starting docker_mysql_1 ... done
Starting docker_wordpress_1 ... done
gaston@gaston:~/docker$ sudo docker ps
CONTAINER ID        IMAGE                          COMMAND                  CREATED             STA
TUS             PORTS                  NAMES
d66e08072442        wordpress:5.4.2-php7.2-apache   "docker-entrypoint.s…"   6 minutes ago       Up
13 seconds      0.0.0.0:8080->80/tcp   docker_wordpress_1
64c00228d911        mysql:8.0.20                    "docker-entrypoint.s…"   6 minutes ago       Up
18 seconds      3306/tcp, 33060/tcp    docker_mysql_1
gaston@gaston:~/docker$
```

As you can see , you have 2 containers, one with mysql and the other with wordpress.

</td>
<td>Docker compose</td>
</tr>
</table>