Assignment I

Git repository link:

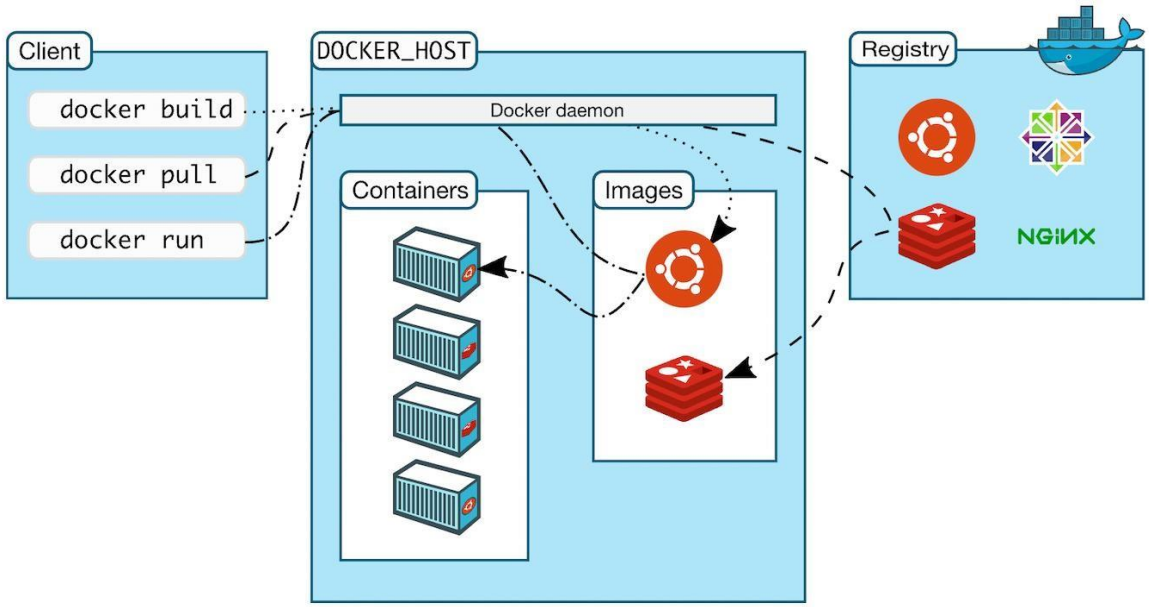
https://github.com/jucollas/operating_systems.git

Activity No. 5: Docker practical session [10%]

Docker is a command line-based software allowing users to manipulate images and create application containers.

As presented in the course, Docker consists of two elements:

- a client, to receive commands from the user
- a server, to execute commands and manage images and containers



Docker commands architecture

Typing this command will give the Client and Server versions available on your computer. ***Paste a screenshot of result***

docker *version*

```
❯ docker version
Client:
 Version:           27.0.3
 API version:       1.46
 Go version:        go1.21.11
 Git commit:        7d4bcd8
 Built:             Sat Jun 29 00:03:32 2024
 OS/Arch:           windows/amd64
 Context:           desktop-linux

Server: Docker Desktop 4.32.0 (157355)
 Engine:
  Version:          27.0.3
  API version:      1.46 (minimum version 1.24)
  Go version:       go1.21.11
  Git commit:       662f78c
  Built:            Sat Jun 29 00:02:50 2024
  OS/Arch:          linux/amd64
  Experimental:     false
 containerd:
  Version:          1.7.18
  GitCommit:        ae71819c4f5e67bb4d5ae76a6b735f29cc25774e
 runc:
  Version:          1.7.18
  GitCommit:        v1.1.13-0-g58aa920
 docker-init:
  Version:          0.19.0
  GitCommit:        de40ad0
```

Usage, options and a full list of available commands can be accessed through the command line in a terminal.Type the following command

```
docker --help
```

The general usage of a Docker command line is as follows:

```
docker [OPTIONS] COMMAND [arg...]
```

Questions

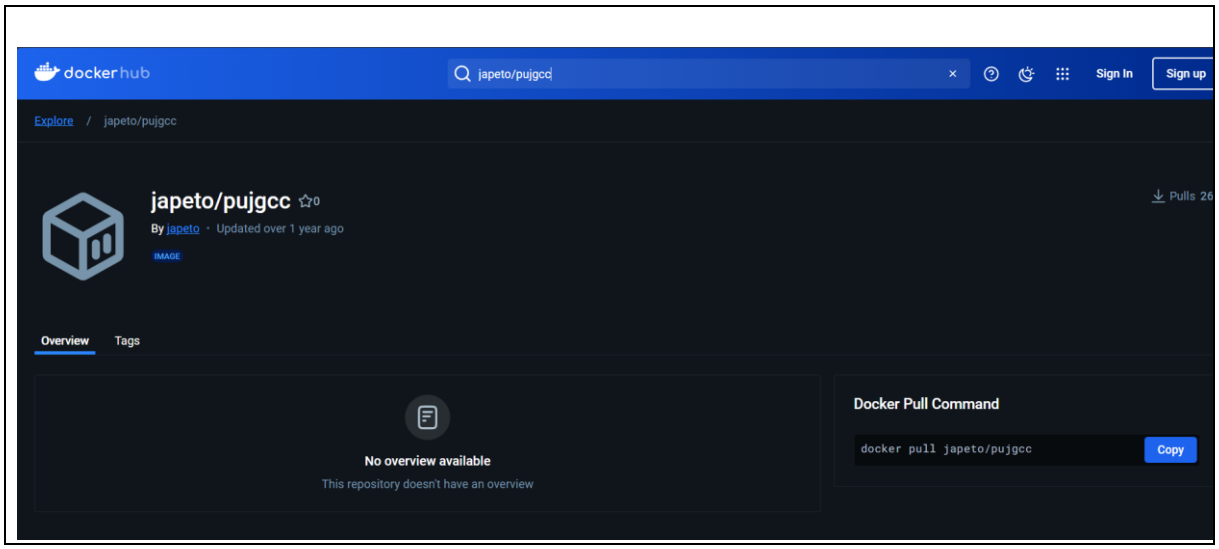
1. How many arguments are absolutely required by the command 'docker pull' ?
   R// El comando docker pull requiere **un argumento** absolutamente necesario, que es el nombre de la **imagen** que deseas descargar, a esto se le puede adicionar la etiqueta (tag) y el idenfidicador (digest).

   ```
   docker pull [OPTIONS] NAME[:TAG|@DIGEST]
   ```

2. Do you remember what a registry is?
   R// Un registro (**registry**) en Docker es un servicio donde se almacenan y distribuyen las imágenes de Docker.

Download a predefined image available on the DockerHub

In a web browser, navigate to the DockerHub : https://hub.docker.com/ In the top search bar, type : japeto/pujgcc and paste a screenshot



Our course has images available for the development of practical sessions.

Questions

1. How many times was the *japeto* image downloaded ?
   R// 262

Execute the command inside a terminal.
Paste a screenshot of result

docker pull japeto/pujgcc:v0.12

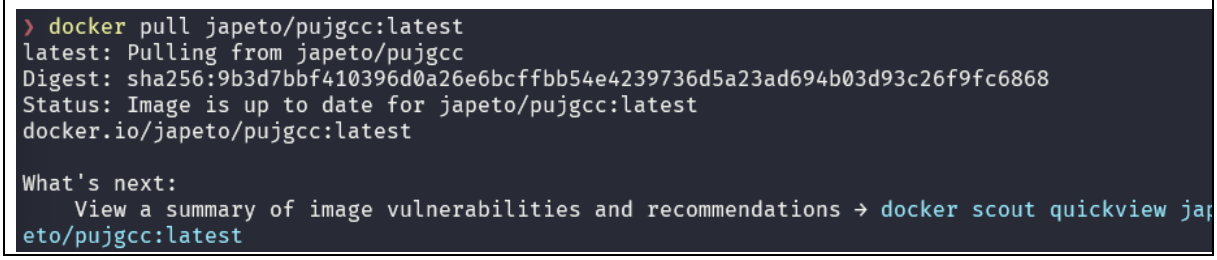
```
❯ docker pull japeto/pujgcc:v0.12

What's next:
    View a summary of image vulnerabilities and recommendations → docker scout quickview ja
eto/pujgcc:v0.12
Error response from daemon: manifest for japeto/pujgcc:v0.12 not found: manifest unknown: m
nifest unknown
```

You will get an error as this image has no default tag (“latest”). So we need to specify one in the command line.

Go to the “Tags” tab and copy the pull command of version latest

Paste a screenshot of result

docker pull japeto/pujgcc:latest


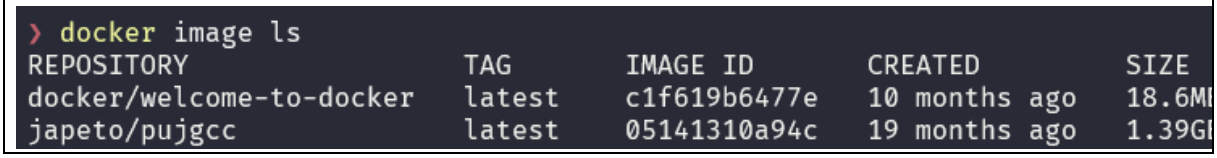
```
❯ docker pull japeto/pujgcc:latest
latest: Pulling from japeto/pujgcc
Digest: sha256:9b3d7bbf410396d0a26e6bcffbb54e4239736d5a23ad694b03d93c26f9fc6868
Status: Image is up to date for japeto/pujgcc:latest
docker.io/japeto/pujgcc:latest

What's next:
    View a summary of image vulnerabilities and recommendations → docker scout quickview jap
eto/pujgcc:latest
```

Question:

1. How many times do you see 'Pull complete' displayed ? Why ?
   R// Cuando Docker descarga una imagen, a menudo está compuesta por varias capas. Cada capa representa una parte del sistema de archivos que ha sido creada o modificada. El proceso de descarga de una imagen implica obtener cada una de estas capas desde el registro. No pudimos determinar cuántas veces aparece "pull complete" ya que se había descargado con anterioridad la imagen.

Now, to be sure that the image was correctly pulled, let's see the list of all available downloaded images inside our workspace. ***Paste a screenshot of result***

docker image ls


```
❯ docker image ls
REPOSITORY                TAG       IMAGE ID       CREATED         SIZE
docker/welcome-to-docker  latest    c1f619b6477e   10 months ago   18.6MB
japeto/pujgcc             latest    05141310a94c   19 months ago   1.39GB
```

Question:

1. What is the size of the japeto/pujgcc image ?
   R// el tamaño de imagen japeto/pujgcc es de 1.39GB

Perform a task using a pulled image

Among the Docker commands, we will now use the 'run' command.

Question:

1. What are the options and parameters of the 'run' command ?
   R// Los parámetros son:
   "docker run [OPTIONS] IMAGE [COMMAND] [ARG...]"

docker run: Comando principal para crear y ejecutar un contenedor.

[OPTIONS]: Opciones adicionales que puedes proporcionar para configurar el contenedor. Estas opciones pueden incluir configuraciones como puertos expuestos, variables de entorno, volúmenes montados, entre otros. Estas son todas las opciones:

```
Options:
      --add-host list                  Add a custom host-to-IP mapping
                                       (host:ip)
      --annotation map                 Add an annotation to the
                                       container (passed through to the
                                       OCI runtime) (default map[])
  -a, --attach list                    Attach to STDIN, STDOUT or STDERR
      --blkio-weight uint16            Block IO (relative weight),
                                       between 10 and 1000, or 0 to
                                       disable (default 0)
      --blkio-weight-device list       Block IO weight (relative device
                                       weight) (default [])
      --cap-add list                   Add Linux capabilities
      --cap-drop list                  Drop Linux capabilities
      --cgroup-parent string           Optional parent cgroup for the
                                       container
      --cgroupns string                Cgroup namespace to use
                                       (host|private)
                                       'host':    Run the container in
                                       the Docker host's cgroup
                                       namespace
                                       'private': Run the container in
                                       its own private cgroup namespace
                                       '':        Use the cgroup
                                       namespace as configured by the
                                                  default-cgroupns-mode
                                       option on the daemon (default)
      --cidfile string                 Write the container ID to the file
      --cpu-period int                 Limit CPU CFS (Completely Fair
                                       Scheduler) period
      --cpu-quota int                  Limit CPU CFS (Completely Fair
                                       Scheduler) quota
      --cpu-rt-period int              Limit CPU real-time period in
                                       microseconds
      --cpu-rt-runtime int             Limit CPU real-time runtime in
                                       microseconds
  -c, --cpu-shares int                 CPU shares (relative weight)
      --cpus decimal                   Number of CPUs
      --cpuset-cpus string             CPUs in which to allow execution
                                       (0-3, 0,1)
      --cpuset-mems string             MEMs in which to allow execution
                                       (0-3, 0,1)
  -d, --detach                         Run container in background and
                                       print container ID
      --detach-keys string             Override the key sequence for
                                       detaching a container
      --device list                    Add a host device to the container
      --device-cgroup-rule list        Add a rule to the cgroup allowed
                                       devices list
      --device-read-bps list           Limit read rate (bytes per
                                       second) from a device (default [])
      --device-read-iops list          Limit read rate (IO per second)
                                       from a device (default [])
      --device-write-bps list          Limit write rate (bytes per
                                       second) to a device (default [])
      --device-write-iops list         Limit write rate (IO per second)
                                       to a device (default [])
      --disable-content-trust          Skip image verification (default
                                       true)
      --dns list                       Set custom DNS servers
      --dns-option list                Set DNS options
      --dns-search list                Set custom DNS search domains
      --domainname string              Container NIS domain name
      --entrypoint string              Overwrite the default ENTRYPOINT
                                       of the image
  -e, --env list                       Set environment variables
```

```
    --env-file list                  Read in a file of environment
                                     variables
    --expose list                    Expose a port or a range of ports
    --gpus gpu-request               GPU devices to add to the
                                     container ('all' to pass all GPUs)
    --group-add list                 Add additional groups to join
    --health-cmd string              Command to run to check health
    --health-interval duration       Time between running the check
                                     (ms|s|m|h) (default 0s)
    --health-retries int             Consecutive failures needed to
                                     report unhealthy
    --health-start-interval duration Time between running the check
                                     during the start period
                                     (ms|s|m|h) (default 0s)
    --health-start-period duration   Start period for the container
                                     to initialize before starting
                                     health-retries countdown
                                     (ms|s|m|h) (default 0s)
    --health-timeout duration        Maximum time to allow one check
                                     to run (ms|s|m|h) (default 0s)
    --help                           Print usage
-h, --hostname string                Container host name
    --init                           Run an init inside the container
                                     that forwards signals and reaps
                                     processes
-i, --interactive                    Keep STDIN open even if not attached
    --ip string                      IPv4 address (e.g., 172.30.100.104)
    --ip6 string                     IPv6 address (e.g., 2001:db8::33)
    --ipc string                     IPC mode to use
    --isolation string               Container isolation technology
    --kernel-memory bytes            Kernel memory limit
-l, --label list                     Set meta data on a container
    --label-file list                Read in a line delimited file of
                                     labels
    --link list                      Add link to another container
    --link-local-ip list             Container IPv4/IPv6 link-local
                                     addresses
    --log-driver string              Logging driver for the container
    --log-opt list                   Log driver options
    --mac-address string             Container MAC address (e.g.,
                                     92:d0:c6:0a:29:33)
-m, --memory bytes                   Memory limit
    --memory-reservation bytes       Memory soft limit
    --memory-swap bytes              Swap limit equal to memory plus
                                     swap: '-1' to enable unlimited swap
    --memory-swappiness int          Tune container memory swappiness
                                     (0 to 100) (default -1)
    --mount mount                    Attach a filesystem mount to the
                                     container
    --name string                    Assign a name to the container
    --network network                Connect a container to a network
    --network-alias list             Add network-scoped alias for the
                                     container
    --no-healthcheck                 Disable any container-specified
                                     HEALTHCHECK
    --oom-kill-disable               Disable OOM Killer
    --oom-score-adj int              Tune host's OOM preferences
                                     (-1000 to 1000)
    --pid string                     PID namespace to use
    --pids-limit int                 Tune container pids limit (set
                                     -1 for unlimited)
    --platform string                Set platform if server is
                                     multi-platform capable
    --privileged                     Give extended privileges to this
                                     container
-p, --publish list                   Publish a container's port(s) to
                                     the host
-P, --publish-all                    Publish all exposed ports to
                                     random ports
    --pull string                    Pull image before running
                                     ("always", "missing", "never")
                                     (default "missing")
```

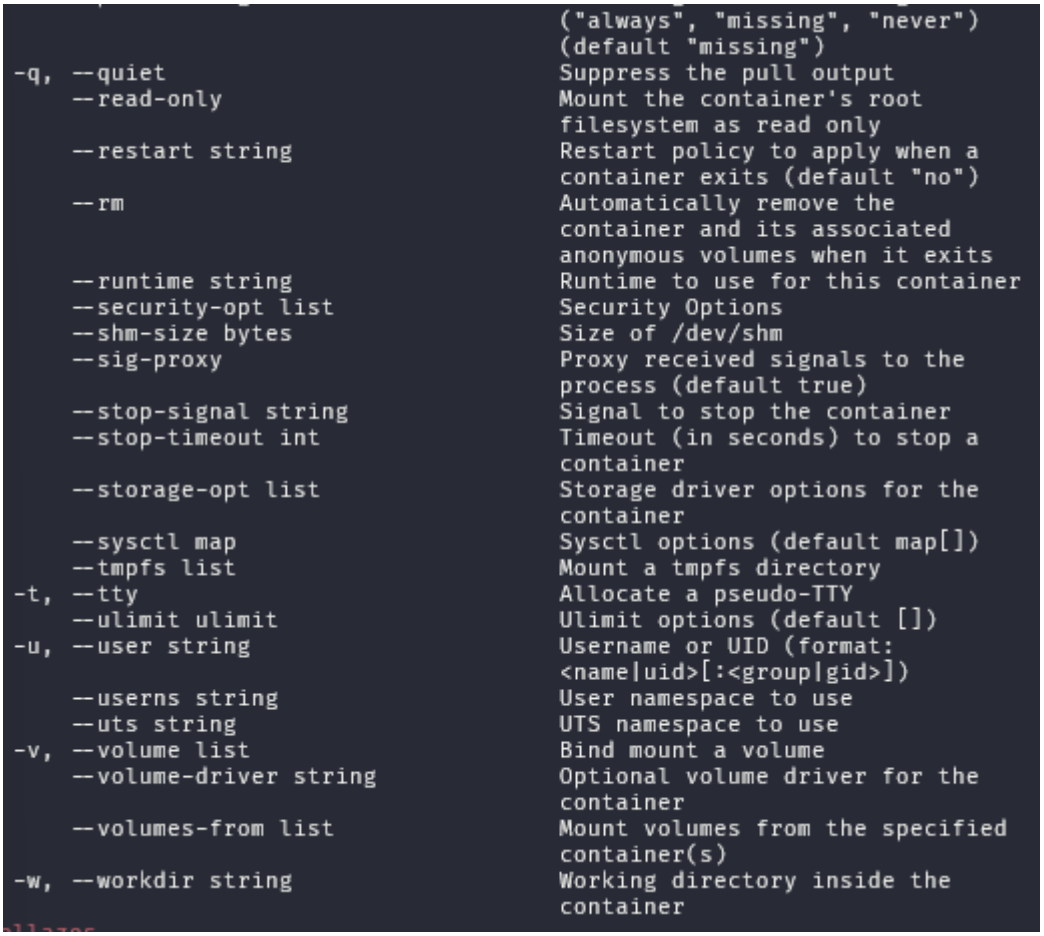


IMAGE: El nombre de la imagen de Docker que se utilizará para crear el contenedor. Puedes especificar una etiqueta opcional para usar una versión específica de la imagen.

[COMMAND]: (Opcional) El comando que deseas ejecutar en el contenedor. Si no se especifica, Docker ejecutará el comando predeterminado definido en la imagen.

[ARG...]: (Opcional) Argumentos que se pasan al comando especificado.

docker run --help

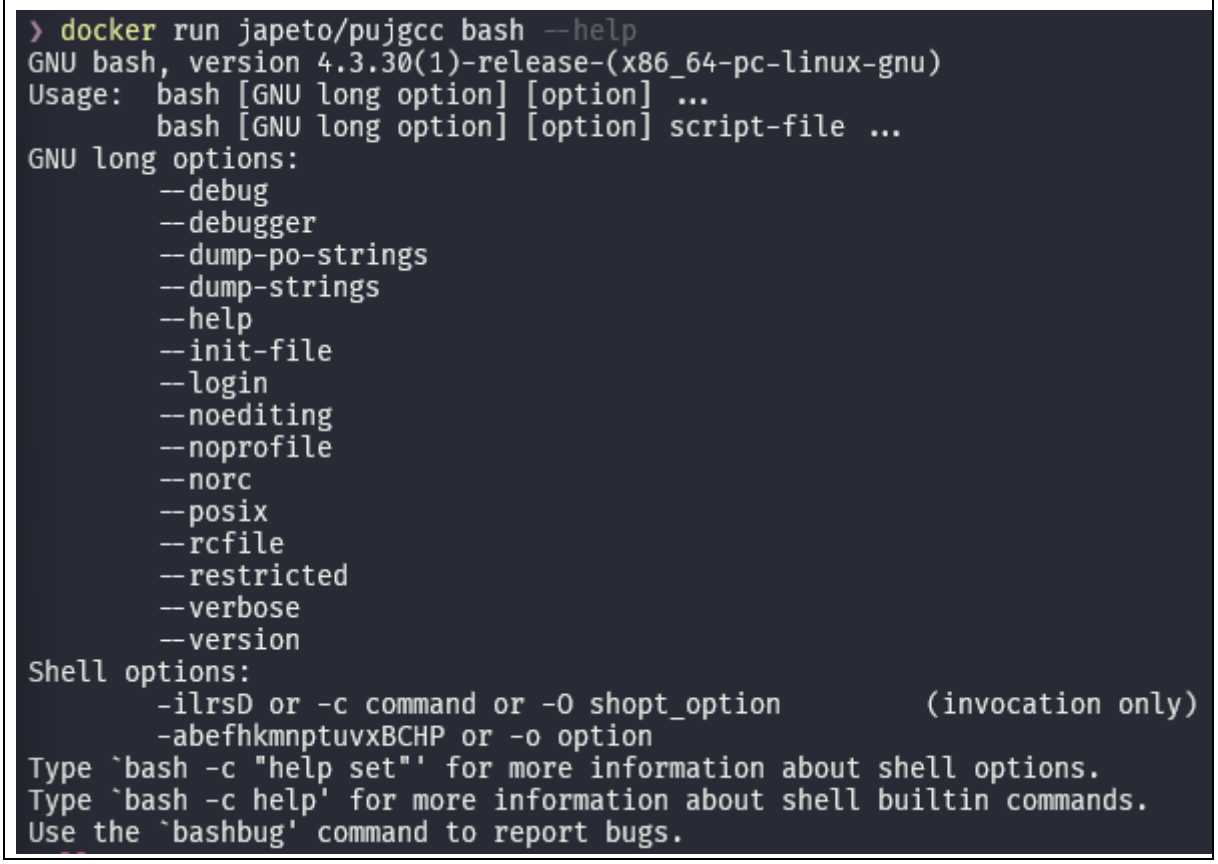
As displayed in the terminal, the description of the command is 'Run a command in a new container'.

Question :

1. What is the difference between an image and a container ?
   R// En Docker, una imagen es una plantilla estática e inmutable que contiene el código, las dependencias y el entorno necesario para ejecutar una aplicación. Se utiliza como un plano para crear contenedores. Un contenedor, por otro lado, es una instancia en ejecución de una imagen que es ligera y mutable. Los contenedores usan la imagen como base para ejecutar la aplicación y pueden ser iniciados, detenidos y modificados durante su ciclo de

vida. Las imágenes definen el entorno y las configuraciones, mientras que los contenedores son las instancias activas que ejecutan el software basado en esas imágenes.

Now, to run the application, execute the following command:
Paste a screenshot of result

docker run japeto/pujgcc bash --help


```
❯ docker run japeto/pujgcc bash --help
GNU bash, version 4.3.30(1)-release-(x86_64-pc-linux-gnu)
Usage:  bash [GNU long option] [option] ...
        bash [GNU long option] [option] script-file ...
GNU long options:
        --debug
        --debugger
        --dump-po-strings
        --dump-strings
        --help
        --init-file
        --login
        --noediting
        --noprofile
        --norc
        --posix
        --rcfile
        --restricted
        --verbose
        --version
Shell options:
        -ilrsD or -c command or -O shopt_option         (invocation only)
        -abefhkmnptuvxBCHP or -o option
Type `bash -c "help set"' for more information about shell options.
Type `bash -c help' for more information about shell builtin commands.
Use the `bashbug' command to report bugs.
```

Congratulations!
You just successfully downloaded and used your first Docker image!

Running *PUJGCC* without parameters was interesting as a demonstration of Docker's features. But if we want to really run *PUJGCC*, we also need to provide parameters and, most importantly, input files.

Find the paths to bind
To bind our current folder to the /data/ folder located inside a container, we first need the absolute path of the current folder, obtained through the unix pwd command. ***Paste a screenshot of result***

pwd

```
❯ pwd

Path
----
C:\Users\PC
```

This path will be used in further commands through ${PWD}.

Instead of running ls command to /home/ files, we will now just list the content of the /data/ folder inside the container but bind with the host.
Paste a screenshot of result

docker run japeto/pujgcc:latest ls /data
❯ docker run japeto/pujgcc:latest ls /data ls: cannot access /data: No such file or directory

If nothing appears, it is normal: the folder is empty and only serves as a "*branching point*".

We now have the paths of the two folders we want to bind together.

Bind a local folder into a container

To perform the folder mapping between the current folder and /data inside the image, the syntax is simple. ***Paste a screenshot of result***

docker run -v ${PWD}:/data/ japeto/pujgcc:latest ls /data/
❯ docker run -v ${PWD}:/data/ japeto/pujgcc:latest ls /data/ helloworld.c

Question:

1. Is the displayed list the same as what is in your current folder?
   R// Si, las carpetas tiene los mismos archivos.

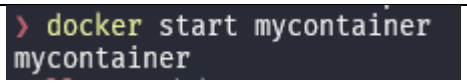
Finally, we can run C on a C or C++ file located in the Data folder. Change the name of the file to any of the provided files.
Paste a screenshot of result

docker run -v ${PWD}:/data/ japeto/pujgcc:latest gcc /data/helloworld.c
❯ docker run -v ${PWD}:/data/ japeto/pujgcc:latest gcc /data/helloworld.c Collazos ~\data 1.048s

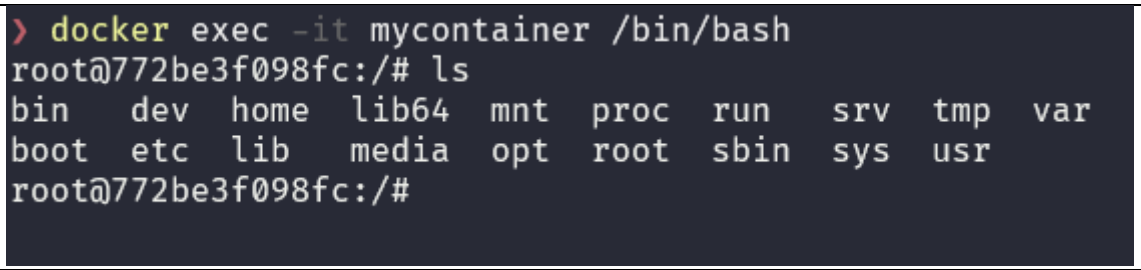
Restart and detach a container

Learn how to re-use a container where you installed something

Use the start command to restart the container created in the last exercise ***Paste a screenshot of result***

docker start mycontainer


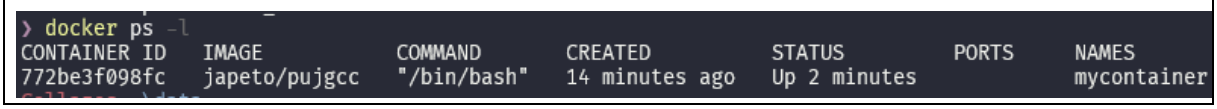
Go back to the container using the exec command instead of the run command. ***Paste a screenshot of result***

docker exec -ti mycontainer /bin/bash


Question :

1. *What happens now when you exit the container? Is it stopped?*
   R// No se detiene, sigue ejecutándose el contenedor.

Check with and ***Paste a screenshot of result***

docker ps -l


In fact, the container keeps on running. This is because re-starting a container turns it into a “detached process” running in the background. Alternatively, we could have added the -d option to the first docker run command, creating directly a detached container.

Finally, you can stop the container.

docker stop mycontainer