



Centro Universitario de Ciencias

Exactas e Ingenierías.



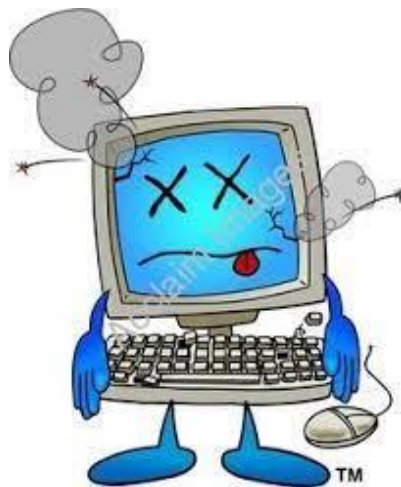
Ingeniería en computación.

Alumno: Vargas López David Guadalupe.

Computación tolerante a fallos.

Profesor: López Franco Michel Emanuel.

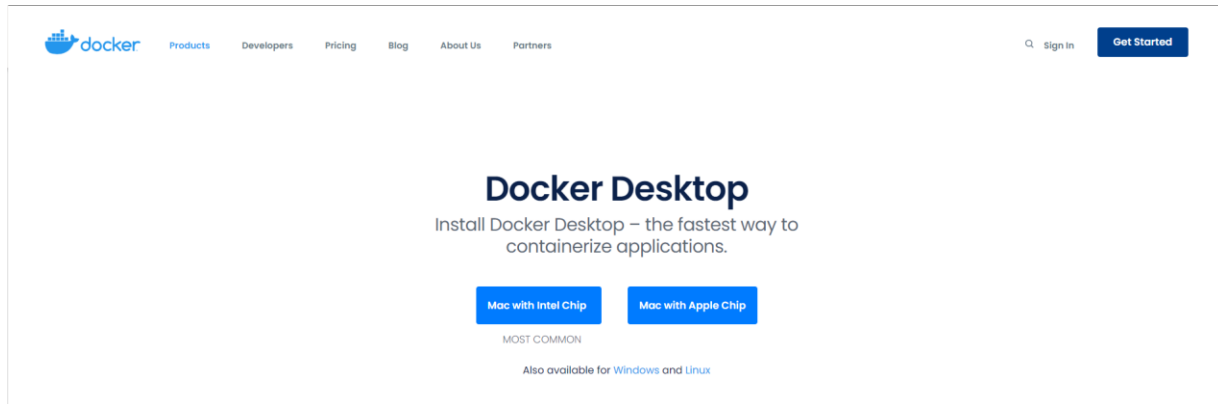
Sección: D06.



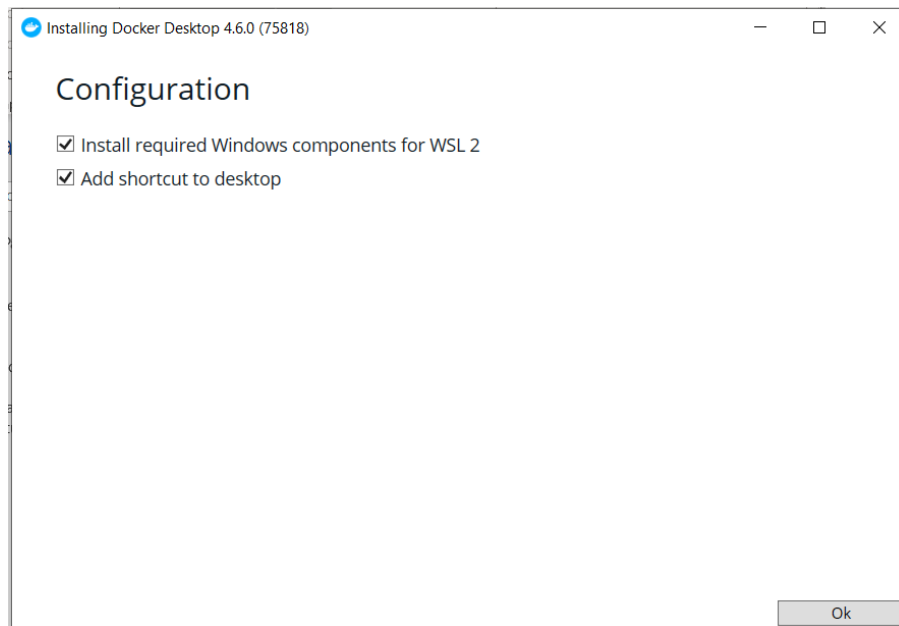
Guadalajara Jal. marzo del 2022.

# Docker:

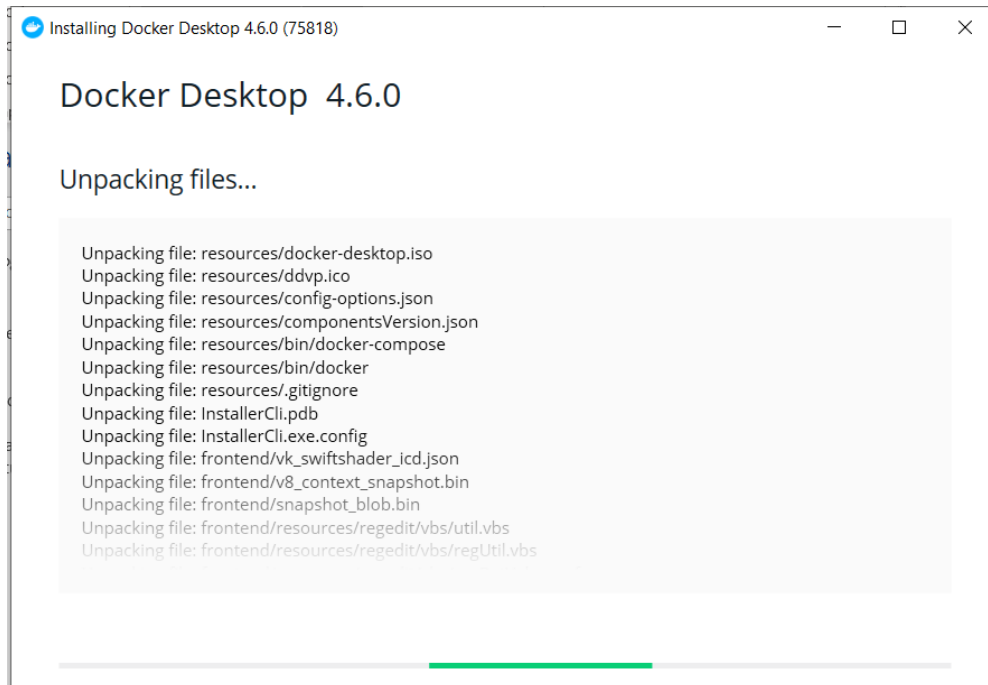
Se instala Docker de la página oficial.



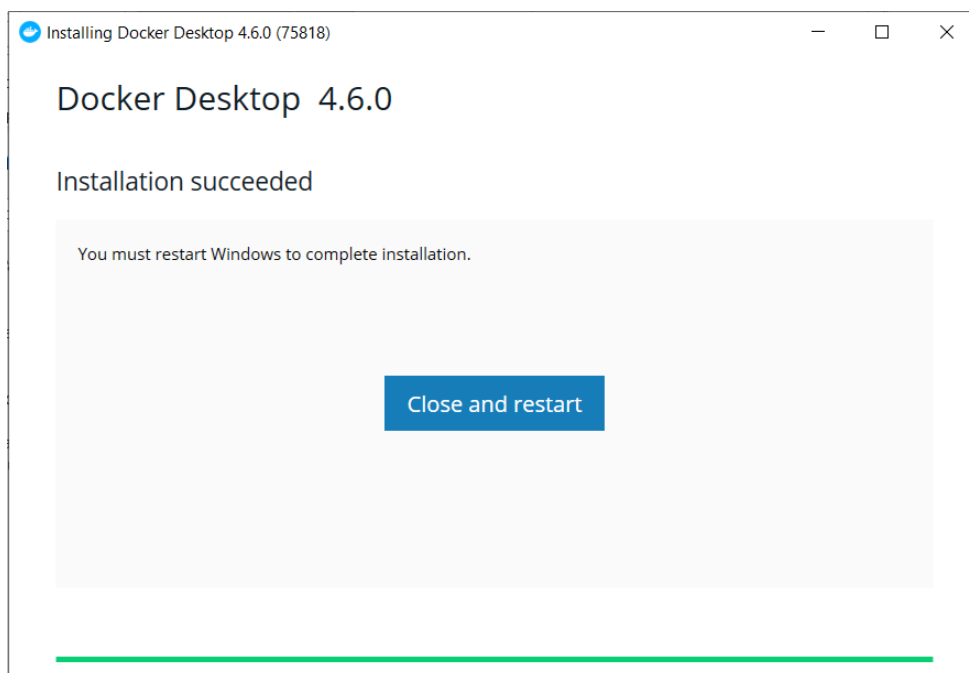
Una vez descargado el ejecutable se procede con la instalación de Docker como cualquier aplicación.



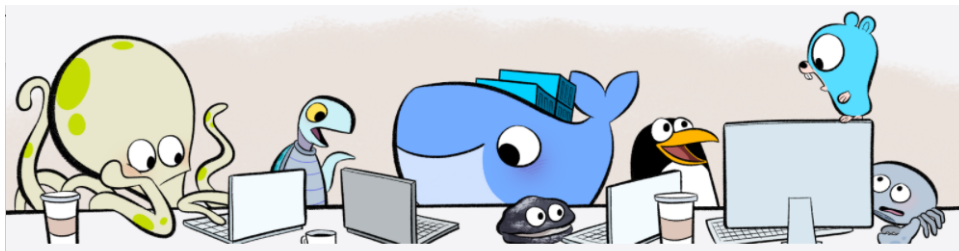
Una vez descargado, se comienzan a descargar los paquetes necesarios para su correcto uso.



Posterior a esto, pide un reinicio de la maquina con el fin de aplicar los cambios necesarios para su utilización correcta.



Se aceptan los términos y condiciones para comenzar a utilizarlo.



### Our Service Agreement has Changed

We've updated the [Docker Subscription Service Agreement](#). Please read the [Blog](#) and [FAQs](#) to learn how companies using Docker Desktop may be affected. By checking "I accept the terms" you agree to the [Subscription Service Agreement](#), the [Data Processing Agreement](#), and the [Data Privacy Policy](#).

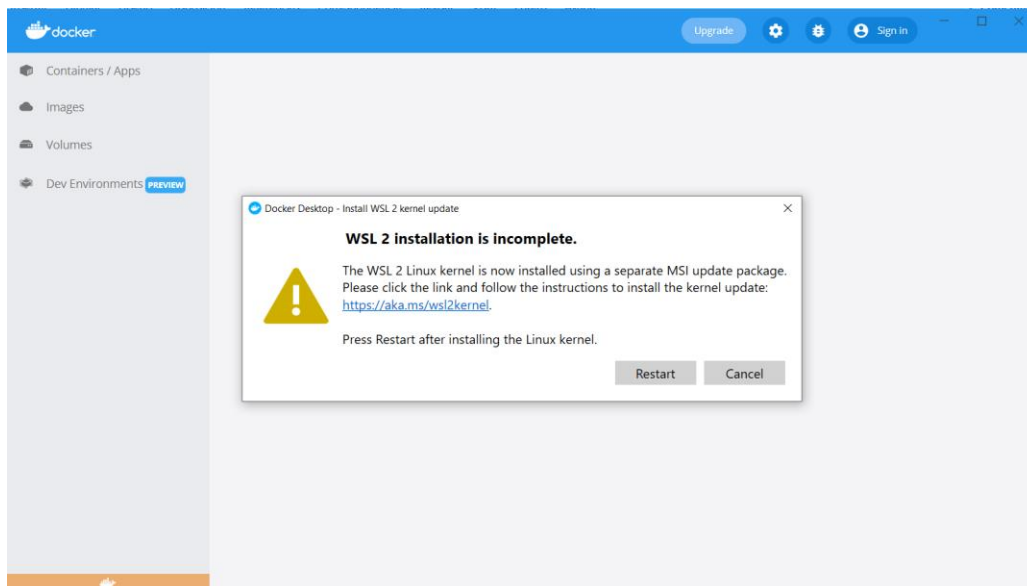
Here's a summary of key changes:

- Our Docker Subscription Service Agreement include a change to the terms of use for Docker Desktop.
  - It **remains free** for small businesses (fewer than 250 employees AND less than \$10 million in annual revenue), personal use, education, and non-commercial open source projects.
  - It requires a paid subscription for professional use in larger enterprises.
- The effective date of these terms is August 31, 2021. There was a **grace period** until January 31, 2022 for those that require a paid subscription to use Docker Desktop. Docker trusts our customers to be in compliance and Docker Desktop will continue to function normally after January 31st, but this is a

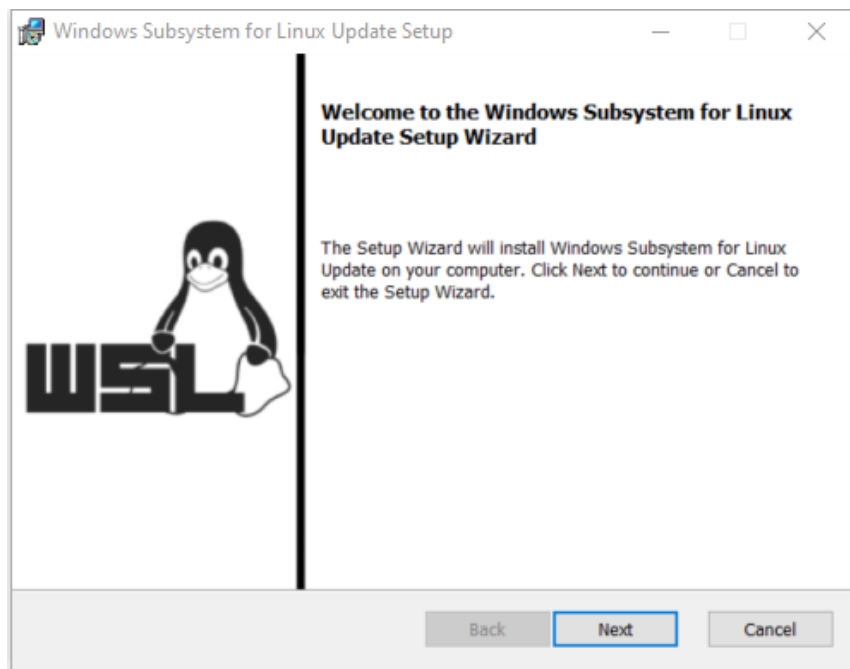
I accept the terms ☐

[View Full Terms](#) [Decline and Close Application](#) [Accept](#)

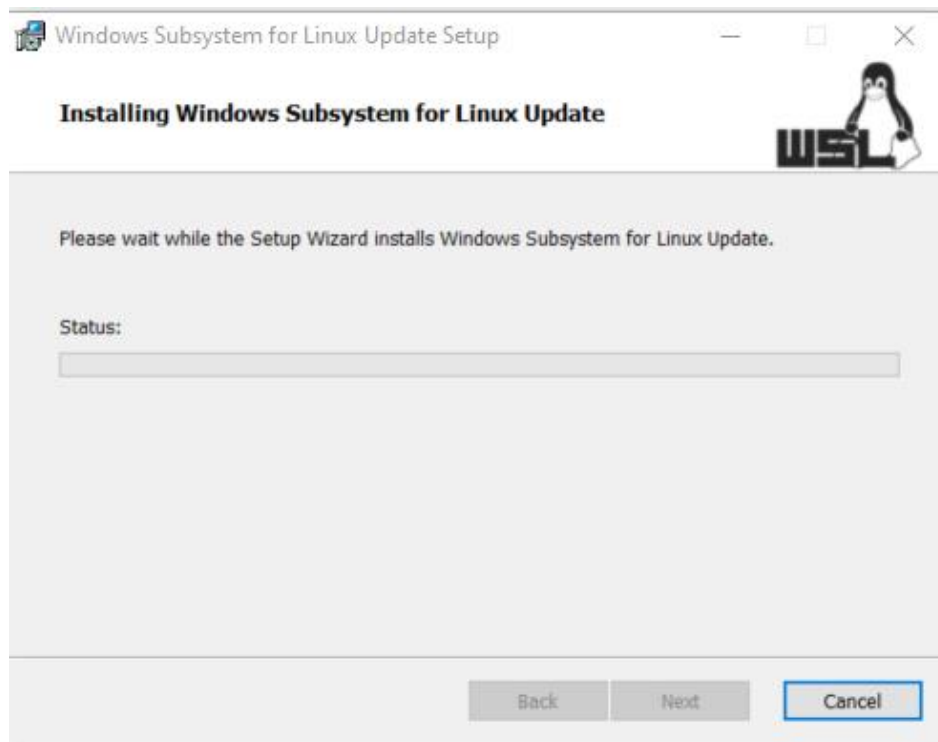
Se pide la instalación de un paquete de actualizaciones de Linux.



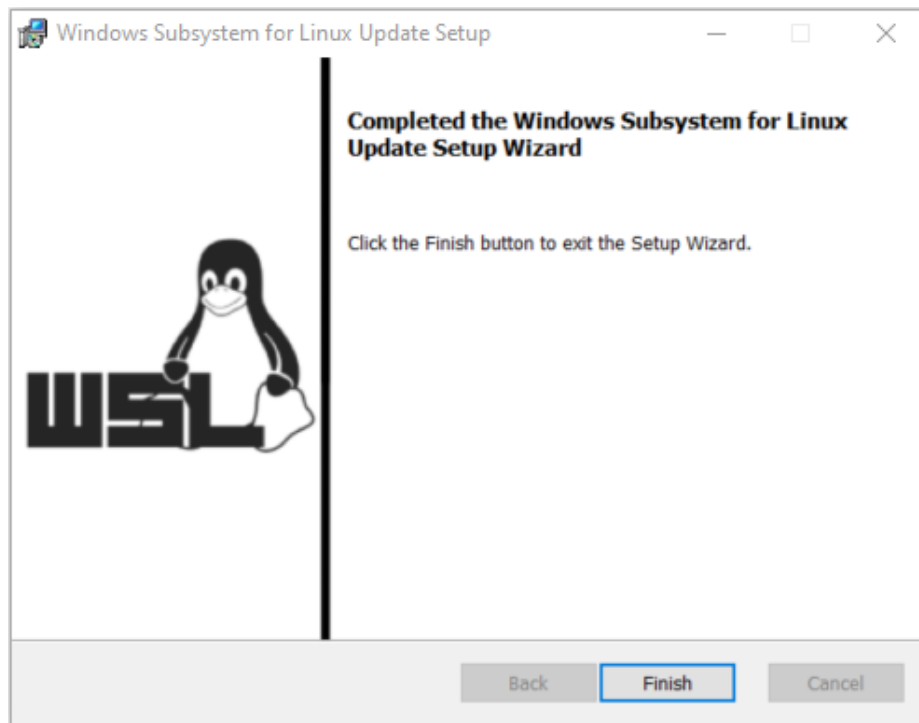
Se acepta que se instalen los paquetes necesarios.



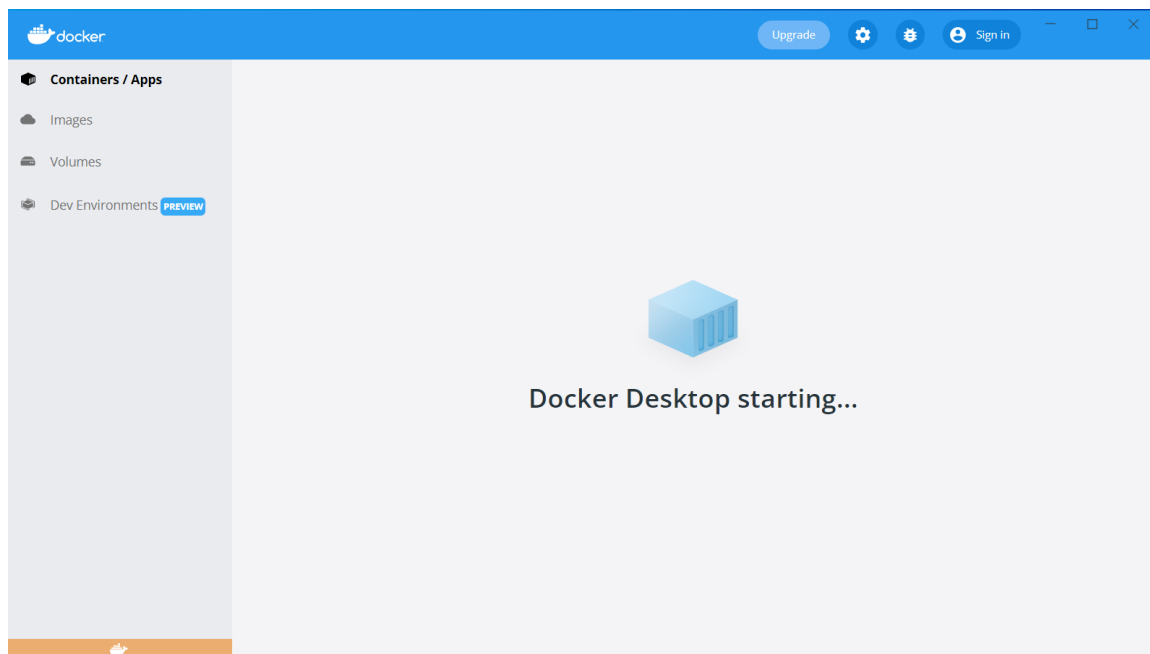
Comienza la instalación de los paquetes necesarios.



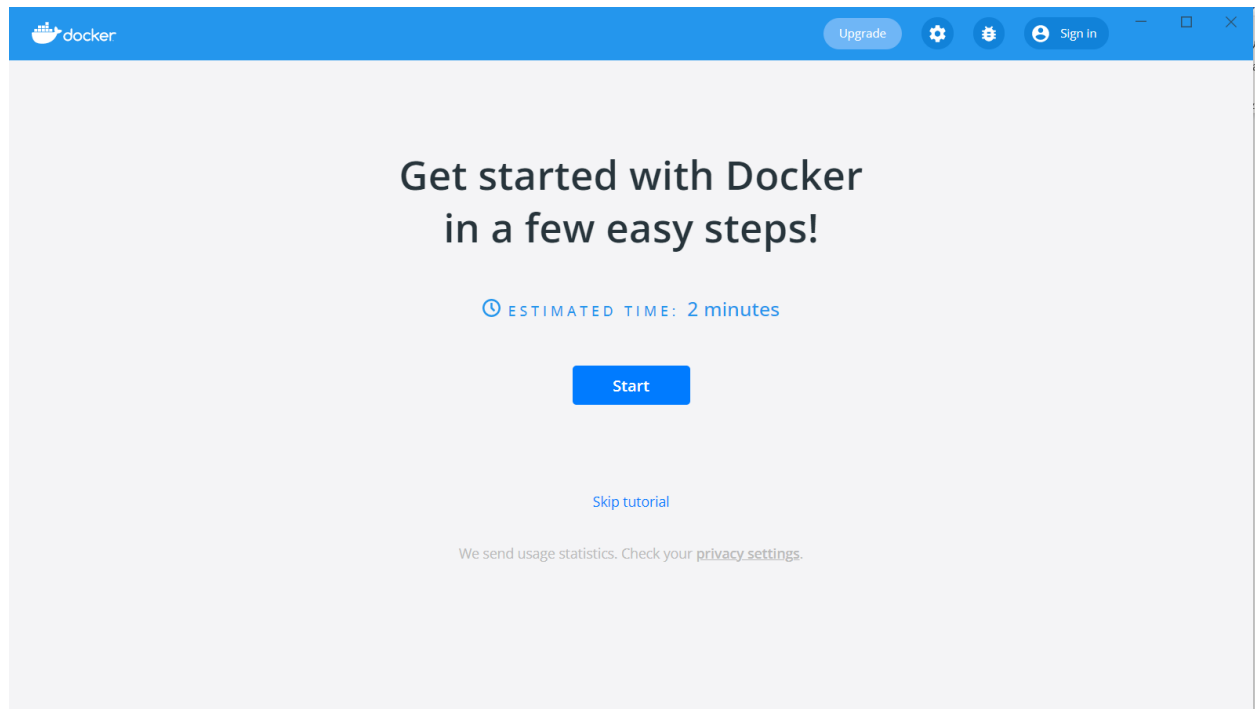
Se finaliza la instalación de los paquetes necesarios.



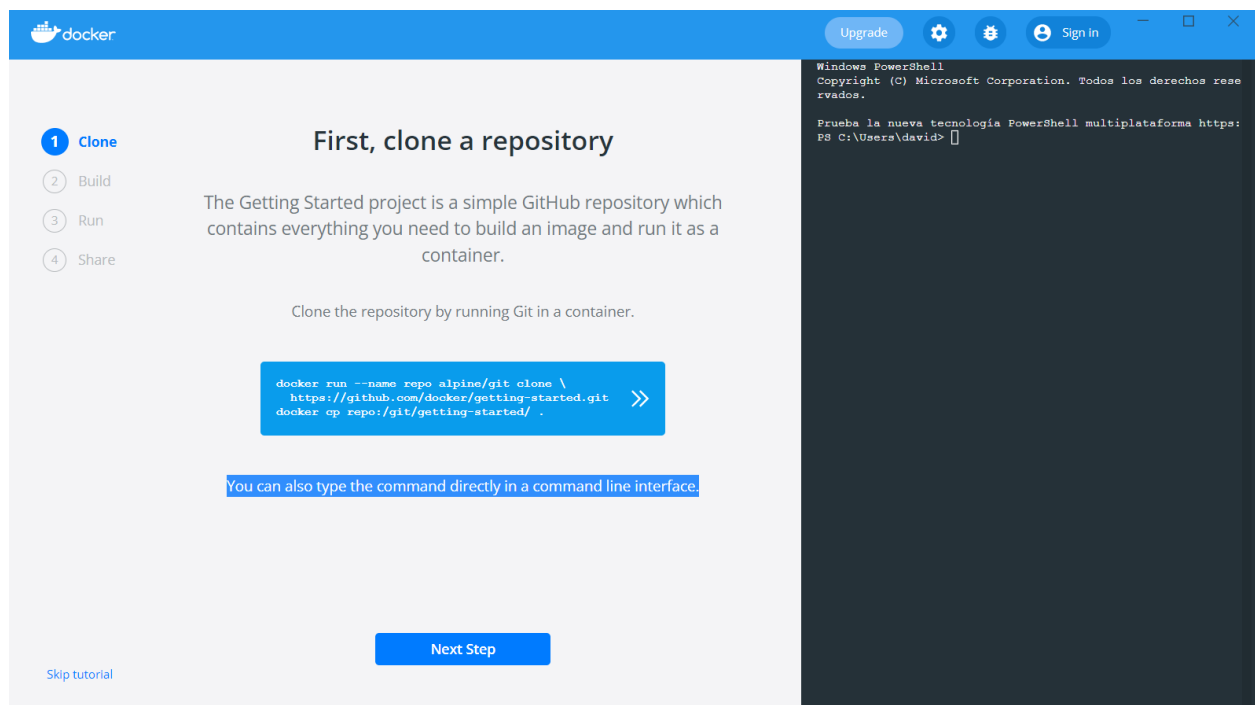
Se inicia nuevamente la aplicación de Docker.



Iniciamos Docker para comenzar a utilizarlo.



Clonamos el primer repositorio de ejemplo, de la pagina de docker que es una especie de tutorial.



Ahora simplemente se carga una imagen de la pagina oficial para seguir el tutorial.

docker

Upgrade

Now, build the image

A Docker image is a private file system just for your container.  
It provides all the files and code your container needs.

od getting-started  
docker build -t docker101tutorial . >>

Next Step

Skip tutorial

Windows PowerShell  
Copyright (C) Microsoft Corporation. Todos los derechos reservados.  
Prueba la nueva tecnología PowerShell multiplataforma https://aka.ms/powershell  
PS C:\Users\david> docker run --name repo alpine/git clone https://github.com/docker/getting-started.git  
Unable to find image 'alpine/git:latest' locally  
97518928ae5f: Pull complete  
b8d269ae55e3: Pull complete  
b863f4504196: Pull complete  
b90e122235c6: Pull complete  
Digest: sha256:d4740deff7f05d2d48771ff66d9d9c26dfb76f0db0aa833b1ea0ee346fale48f  
Status: Downloaded newer image for alpine/git:latest  
Cloning into 'getting-started'...  
PS C:\Users\david> docker cp repo/git/getting-started/.  
PS C:\Users\david>

Se corre el contenedor para crearlos y terminar con el tutorial de inicio.

docker

Upgrade

Run your first container

Start a container based on the image you built in the previous step. Running a container launches your application with private resources, securely isolated from the rest of your machine.

docker run -d -p 80:80 --name docker-tutorial docker101tutorial >>

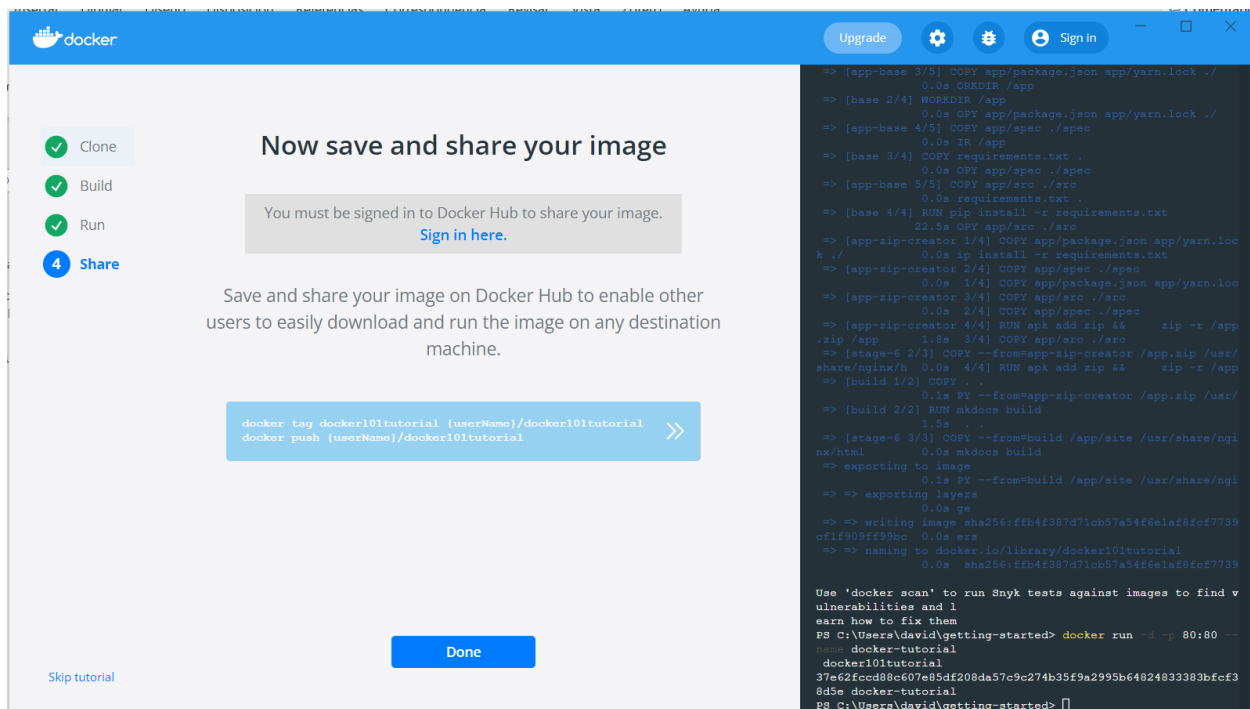
Next Step

Skip tutorial

data444a4a4e38 0.0s  
=> extracting sha256:e27b35b4b1cb7388ae796a8c3006f3e70a  
c0cb0d81943ea 0.3s  
=> [app-base 2/5] WORKDIR /app  
0.4s  
=> [app-base 3/5] COPY app/package.json app/yarn.lock ./  
0.0s  
=> [base 2/4] WORKDIR /app  
0.0s  
=> [app-base 4/5] COPY app/spec ./spec  
0.0s  
=> [base 3/4] COPY requirements.txt .  
0.0s  
=> [app-base 5/5] COPY app/src ./src  
0.0s  
=> [base 4/4] RUN pip install -r requirements.txt  
22.5s  
=> [app-zip-creator 1/4] COPY app/package.json app/yarn.lock ./  
0.0s  
=> [app-zip-creator 2/4] COPY app/spec ./spec  
0.0s  
=> [app-zip-creator 3/4] COPY app/src ./src  
0.0s  
=> [app-zip-creator 4/4] RUN apk add zip && zip -r /app .zip /app  
1.8s  
=> [stage-6 2/3] COPY --from=app-zip-creator /app.zip /usr/share/nginx/html  
0.0s  
=> [build 1/2] COPY . .  
0.1s  
=> [build 2/2] RUN mkdocs build  
1.5s  
=> [stage-6 3/3] COPY --from=build /app/site /usr/share/nginx/html  
0.0s  
=> exporting to image  
0.1s  
=> exporting layers  
0.0s  
=> writing image sha256:ffb4f387d71cb57a54f6e1af8fc7739cf1f909ff99bc  
0.0s  
=> naming to docker.io/library/docker101tutorial  
0.0s  
Use 'docker scan' to run Snyk tests against images to find vulnerabilities and learn how to fix them  
PS C:\Users\david> getting-started>

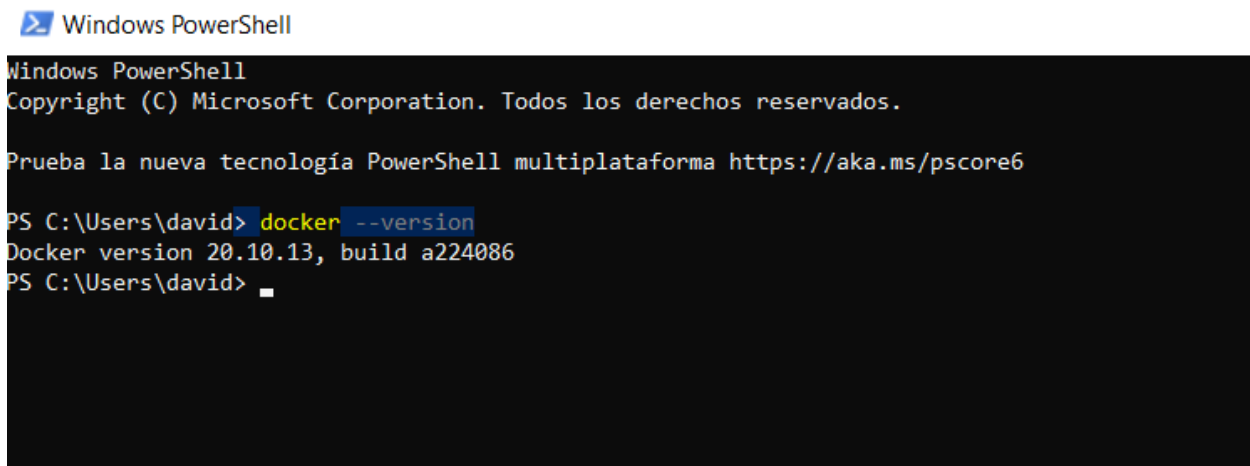


Se crea el contenedor y se guarda en las imágenes de Docker.



The screenshot shows the Docker Desktop interface. On the left, a sidebar lists actions: Clone, Build, Run, and Share (highlighted with a blue circle and the number 4). The main area is titled 'Now save and share your image'. It contains a message: 'You must be signed in to Docker Hub to share your image. Sign in here.' Below this, it says: 'Save and share your image on Docker Hub to enable other users to easily download and run the image on any destination machine.' A blue button with a double arrow icon contains the commands: `docker tag docker101tutorial (userName)/docker101tutorial` and `docker push (userName)/docker101tutorial`. At the bottom left is a 'Skip tutorial' link, and at the bottom center is a 'Done' button. On the right, a terminal window shows the build logs for a Docker image named 'docker101tutorial'. The logs show the build process from base image to final image, including steps like copying files, installing dependencies, and exporting the image.

Por último, comprobamos que Docker esta instalado correctamente observando la versión.



The screenshot shows a Windows PowerShell terminal window. The title bar says 'Windows PowerShell'. The first line of the terminal output is 'Copyright (C) Microsoft Corporation. Todos los derechos reservados.' The second line is 'Prueba la nueva tecnología PowerShell multiplataforma https://aka.ms/pscore6'. The third line shows the command `PS C:\Users\david> docker --version`. The output of the command is 'Docker version 20.10.13, build a224086'. The prompt for the next command is 'PS C:\Users\david> '.

### Ejemplo práctico de uso de Docker:

El código en Python utilizado en este ejemplo es el siguiente:

Es una pequeña aplicación web con la que se muestra un hola mundo dentro de una pagina web que se esta ejecutando bajo el puerto 4000, aunque también se esta utilizando un archivo externo que simplemente se manda llamar para mostrar algunos nombres con el fin de mostrar de una mejor manera el ejemplo.

#### App.py

```
from flask import Flask, jsonify
from users import users

app= Flask(__name__)

@app.route('/', methods=['GET'])
def ping():
    return jsonify({"response": "hello world"})

@app.route('/users')
def userHandler():
    return jsonify({"users": users})

if __name__ == '__main__':
    app.run(host="0.0.0.0", port=4000, debug=True)
```

#### users.py

```
users = [
    {"name": "david"},
    {"name": "ulises"},
    {"name": "gael"}
]
```

Instalamos un entorno virtual para ejecutar código de Python.

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL

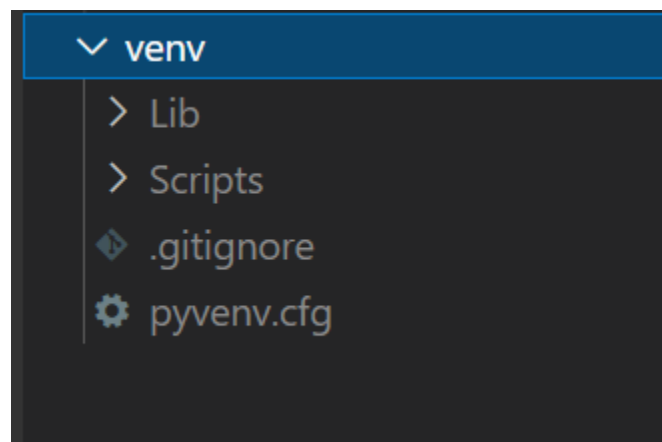
Prueba la nueva tecnología PowerShell multiplataforma https://aka.ms/pscore6

PS C:\Users\david\OneDrive\Escritorio\docker> pip install virtualenv
Collecting virtualenv
  Downloading virtualenv-20.13.4-py2.py3-none-any.whl (8.7 MB)
    | 8.7 MB 6.4 MB/s
Collecting platformdirs<3,>=2
  Downloading platformdirs-2.5.1-py3-none-any.whl (14 kB)
Requirement already satisfied: six<2,>=1.9.0 in c:\users\david\appdata\local\programs\python\python310\lib\site-packages (from virtualenv) (1.16.0)
Collecting distlib<1,>=0.3.1
  Downloading distlib-0.3.4-py2.py3-none-any.whl (461 kB)
    | 461 kB 3.2 MB/s
Collecting filelock<4,>=3.2
  Downloading filelock-3.6.0-py3-none-any.whl (10.0 kB)
Installing collected packages: platformdirs, filelock, distlib, virtualenv

```

Se crea una carpeta llamada venv.

```
PS C:\Users\david\OneDrive\Escritorio\docker> virtualenv venv
created virtual environment CPython3.10.2.final.0-64 in 12970ms
creator CPython3Windows(dest=C:\Users\david\OneDrive\Escritorio\docker\venv, clear=False, no_vcs_ignore=False, global=False)
seeder FromAppData(download=False, pip=bundle, setuptools=bundle, wheel=bundle, via=copy, app_data_dir=C:\Users\david\AppData\Local\pypa\virtualenv)
added seed packages: pip==22.0.4, setuptools==60.10.0, wheel==0.37.1
activators BashActivator,BatchActivator,FishActivator,NushellActivator,PowerShellActivator,PythonActivator
PS C:\Users\david\OneDrive\Escritorio\docker>
```



Instalamos flask para que funcione el entorno virtual.

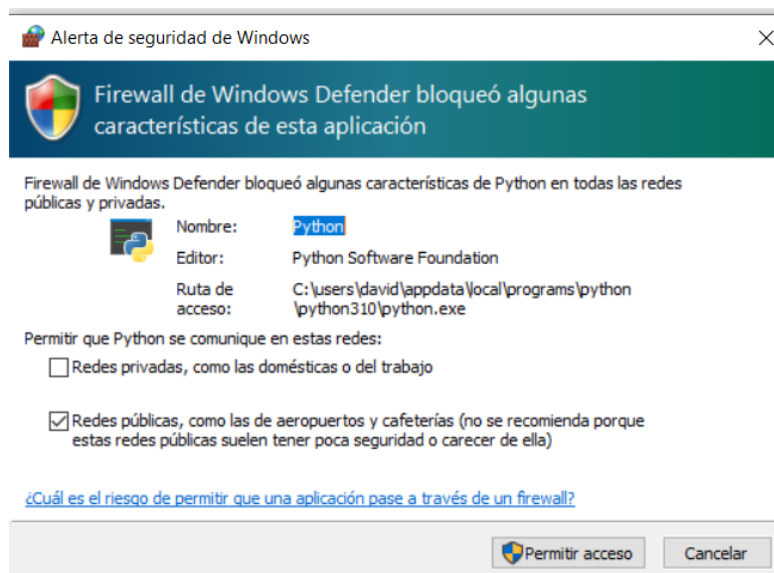
```
Microsoft Windows [Versión 10.0.19044.1586]
(c) Microsoft Corporation. Todos los derechos reservados.

(venv) C:\Users\david\OneDrive\Escritorio\Nueva carpeta\venv\Scripts>pip install flask
Collecting flask
  Downloading Flask-2.0.3-py3-none-any.whl (95 kB)
    ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 95.6/95.6 KB 1.8 MB/s eta 0:00:00
Collecting Werkzeug>=2.0
  Downloading Werkzeug-2.0.3-py3-none-any.whl (289 kB)
    ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 289.2/289.2 KB 1.4 MB/s eta 0:00:00
Collecting click>=7.1.2
  Using cached click-8.0.4-py3-none-any.whl (97 kB)
Collecting itsdangerous>=2.0
  Downloading itsdangerous-2.1.1-py3-none-any.whl (15 kB)
Collecting Jinja2>=3.0
  Using cached Jinja2-3.0.3-py3-none-any.whl (133 kB)
Collecting colorama
  Using cached colorama-0.4.4-py2.py3-none-any.whl (16 kB)
```

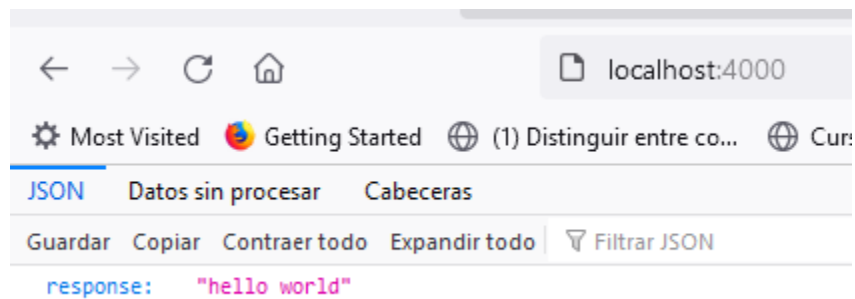
Se corre el servidor virtual en el puerto 4000

```
C:\Users\david\OneDrive\Escritorio\docker>python src/app.py
* Serving Flask app 'app' (lazy loading)
* Environment: production
  WARNING: This is a development server. Do not use it in a production deployment.
  Use a production WSGI server instead.
* Debug mode: on
* Restarting with stat
* Debugger is active!
* Debugger PIN: 102-024-315
* Running on all addresses.
  WARNING: This is a development server. Do not use it in a production deployment.
* Running on http://192.168.100.17:4000/ (Press CTRL+C to quit)
```

Se permite el acceso a la aplicación.



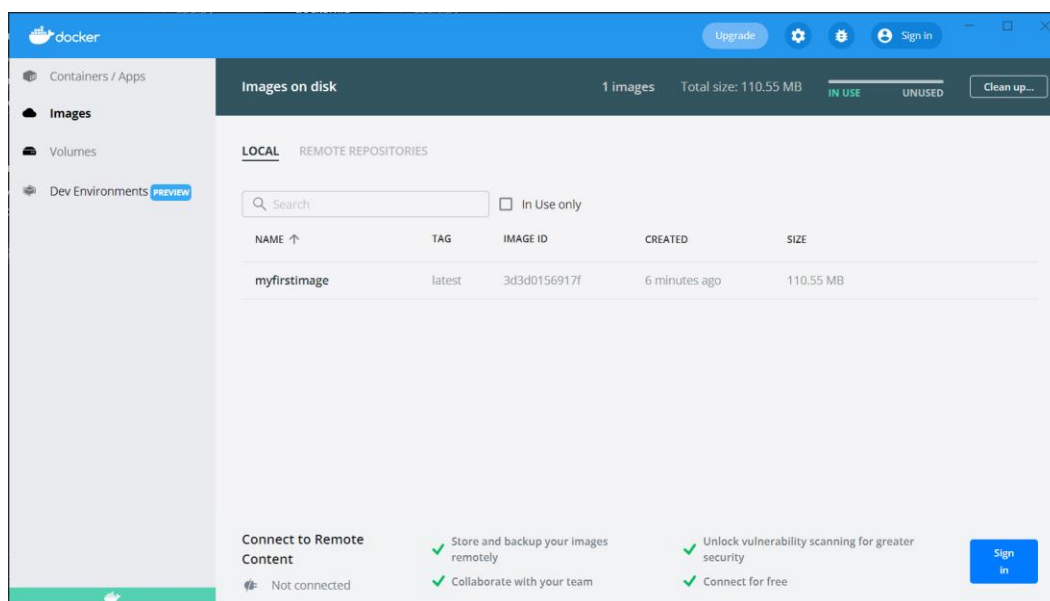
Y observamos en el explorador de Google que efectivamente está en uso.



Se crea la imagen para pasar el código con sus componentes.

```
C:\Users\david\OneDrive\Escritorio\docker>c:/Users/david/OneDrive/Escritorio/docker/venv/Scripts/activate.bat
(venv) C:\Users\david\OneDrive\Escritorio\docker>docker build -t myfirstimage:dev -f Dockerfile .
[+] Building 20.6s (6/6) FINISHED
=> [internal] load build definition from Dockerfile                                0.1s
=> => transferring dockerfile: 121B                                              0.0s
=> [internal] load .dockerignore                                                 0.0s
=> => transferring context: 2B                                                  0.0s
=> [internal] load metadata for docker.io/library/alpine:3.10                  4.5s
=> [1/2] FROM docker.io/library/alpine:3.10@sha256:451eee8bedcb2f029756dc3e9d73bab0e7943c1ac55cff3a4861 0.7s
=> => resolve docker.io/library/alpine:3.10@sha256:451eee8bedcb2f029756dc3e9d73bab0e7943c1ac55cff3a4861 0.0s
=> => sha256:e515aad2ed234a5072c4d2ef86a1cb77d5bfe4b11aa865d9214875734c4eeb3c 528B / 528B          0.0s
=> => sha256:e7b300aee9f9bf3433d32bc9305bfdd22183beb59d933b48d77ab56ba53a197a 1.47kB / 1.47kB      0.0s
=> => sha256:396c31837116ac290458afcb928f68b6cc1c7bdd6963fc72f52f365a2a89c1b5 2.80MB / 2.80MB      0.5s
=> => sha256:451eee8bedcb2f029756dc3e9d73bab0e7943c1ac55cff3a4861c52a0fdd3e98 1.64kB / 1.64kB      0.0s
```

Observamos la imagen dentro de la interfaz.



Para activar el modo interactivo de Docker.

```
(venv) C:\Users\david\OneDrive\Escritorio\docker>docker images
REPOSITORY      TAG         IMAGE ID      CREATED        SIZE
myfirstimage     latest     3d3d0156917f  8 minutes ago  111MB

(venv) C:\Users\david\OneDrive\Escritorio\docker>docker run -it myfirstimage /bin/sh
/ #
```

Observamos que se encuentra dentro de la carpeta.

```
(venv) C:\Users\david\OneDrive\Escritorio\docker>docker run -it myfirstimage /bin/sh
/ # ls
bin      etc      lib      mnt      proc     run      srv      tmp      var
dev      home    media    opt      root     sbin     sys      usr
/ #
```

Creamos una carpeta testing dentro de ese directorio.

```
/ # mkdir testing
/ # ls
bin      etc      lib      mnt      proc     run      srv      testing  usr
dev      home    media    opt      root     sbin     sys      tmp      var
/ #
```

Observamos las versiones de Python y del pip.

```
/ # python --version
/bin/sh: python: not found
/ # python3 --version
Python 3.7.10
/ # pip --version
pip 22.0.4 from /usr/lib/python3.7/site-packages/pip (python 3.7)
/ #
```

Observamos los requerimientos de la aplicación con pip freeze

```
(venv) C:\Users\david\OneDrive\Escritorio\docker>pip freeze
certifi==2021.10.8
charset-normalizer==2.0.12
click==8.0.4
cloudpickle==2.0.0
colorama==0.4.4
commonmark==0.9.1
croniter==1.3.4
```

Creamos un archivo requirements.txt dentro de la aplicación.

```
(venv) C:\Users\david\OneDrive\Escritorio\docker>pip freeze > requirements.txt
```

```
(venv) C:\Users\david\OneDrive\Escritorio\docker>
```

≡ requirements.txt

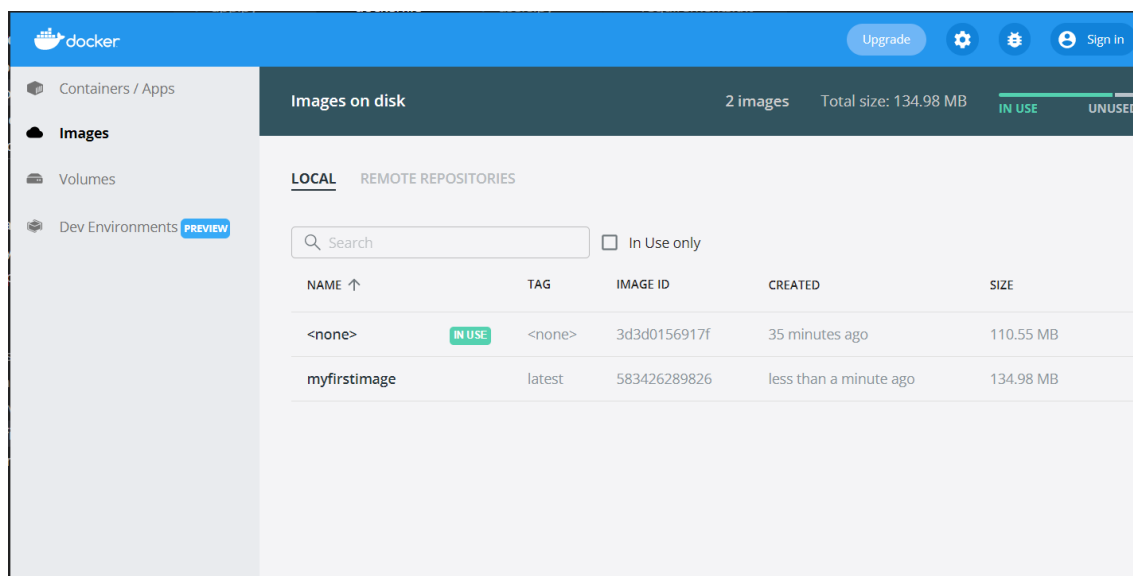
```
1  certifi==2021.10.8
2  charset-normalizer==2.0.12
3  click==8.0.4
4  cloudpickle==2.0.0
5  colorama==0.4.4
6  commonmark==0.9.1
7  croniter==1.3.4
8  cycler==0.11.0
9  dask==2022.2.1
10 distlib==0.3.4
11 distributed==2022.2.1
12 docker==4.2.2
13 filelock==3.6.0
14 Flask==2.0.3
15 fonttools==4.29.1
16 fsspec==2022.2.0
17 graphviz==0.19.1
18 HeapDict==1.0.1
19 idna==3.3
20 importlib-resources==5.4.0
21 itsdangerous==2.1.1
22 Jinja2==3.0.3
23 kiwisolver==1.3.2
24 locket==0.2.1
25 MarkupSafe==2.1.0
26 marshmallow==3.14.1
```

Y se vuelve a crear la imagen

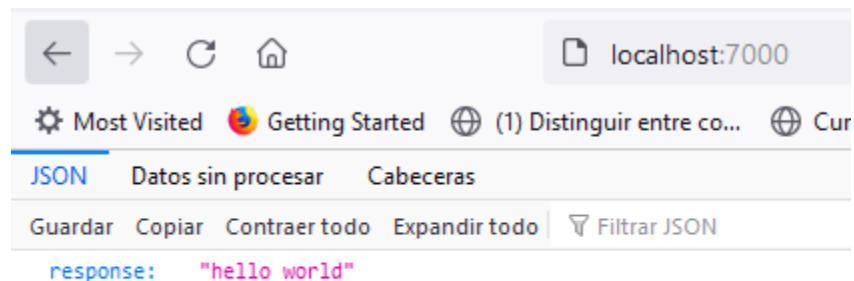
```
(venv) C:\Users\david\OneDrive\Escritorio\docker>docker build -t myfirstimage -f Dockerfile .
[+] Building 0.1s (2/2) FINISHED
=> [internal] load build definition from Dockerfile                                0.0s
=> => transferring dockerfile: 239B                                              0.0s
=> [internal] load .dockerignore                                                 0.0s
=> => transferring context: 2B                                                  0.0s
failed to solve with frontend dockerfile.v0: failed to create LLB definition: dockerfile parse error line 6: COPY requires at least two arguments, but only one was provided. Destination could not be determined.

(venv) C:\Users\david\OneDrive\Escritorio\docker>
```

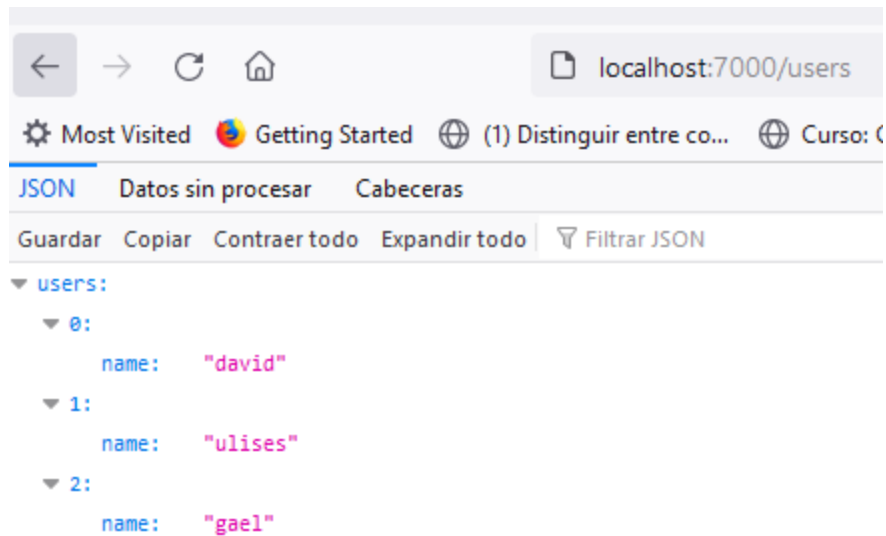
Y crea una imagen llamada none con la cual se cambia el estado a en uso.



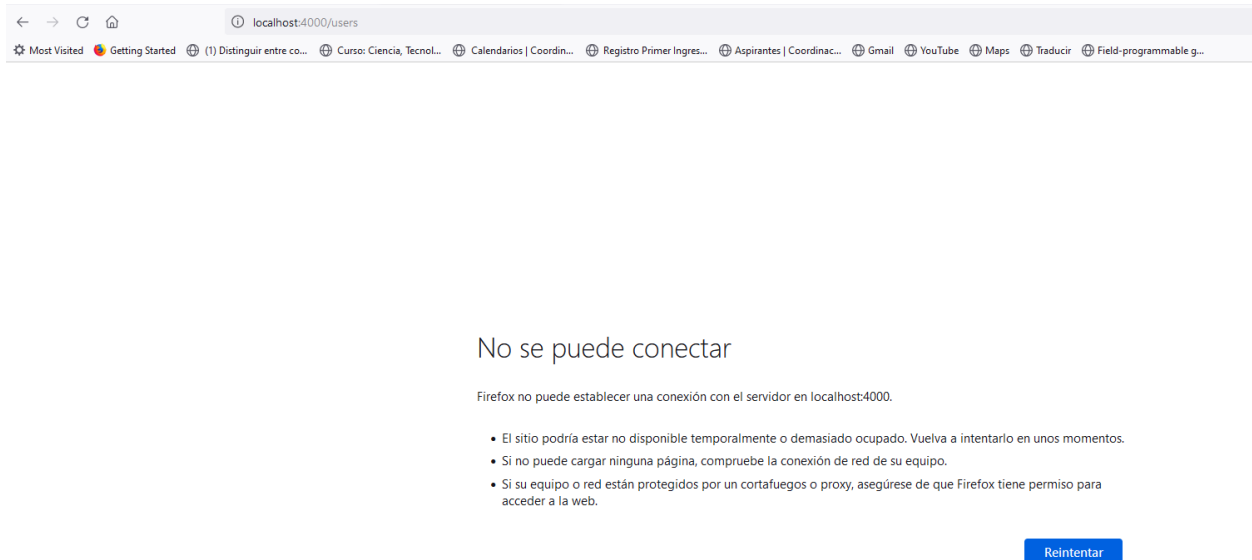
Observamos que siga funcionando tando el hola mundo como la de users para corroborar que siguen activas.







Así como también observamos que en el puerto 4000 ya no están funcionando, debido a que ahí están hosteado la app.py.



Se coloca el comando siguiente para lograr que el proceso se deslinde de la terminal y de esa forma se libere la terminal para otros usos.

```
(venv) C:\Users\david\OneDrive\Escritorio\docker>docker run -it -p 7000:4000 -d myfirstimage  
841fa7fe5e09d78f46783eb78e16832874acd1ccf1ec218c0e5bcabd7fabbaa3
```

Detenemos el contenedor y observamos que en efecto se detuvo la ejecución de este.

```
(venv) C:\Users\david\OneDrive\Escritorio\docker>docker container ls
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS                    NAMES
841fa7fe5e09   myfirstimage   "python3 src/app.py"    9 minutes ago Up 9 minutes   0.0.0.0:7000->4000/tcp   elated_galois

(venv) C:\Users\david\OneDrive\Escritorio\docker>docker stop 841f
841f

(venv) C:\Users\david\OneDrive\Escritorio\docker>docker container ls
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS                    NAMES

(venv) C:\Users\david\OneDrive\Escritorio\docker>
```

### Enlace al repositorio:

<https://github.com/David-1212/docker.py>

### Conclusiones:

Esta practica fue bastante interesante de realizar, ya que no sabía que podía realizarse un contenedor para compartir algún tipo de aplicación, tampoco conocía la aplicación Docker, la cual es bastante fácil de utilizar una vez que te acostumbras.

Tuve algunos problemas al principio de la actividad, debido a que por alguna razón Docker se apago y tuve que desinstalarlo y volver a instalarlo para lograr que funcionara nuevamente, de ahí en fuera, funciono todo de manera fluida y conforme lo planeado, a pesar de que algunos comandos se cambiaron un poco de lo habitual para hacer más fácil el proceso de creación de las imágenes para este ejercicio.