

Reto 1

```
docker run -it --name reto-ubuntu ubuntu
```

En la terminal:

```
apt update  
apt install python3
```

Para instalar librerías:

```
apt install python3-pip -y  
apt install python3-requests -y
```

Para Mysql

```
apt install mysql-client -y  
apt install python3-mysql.connector -y  
apt install python3-mysqldb -y
```

Para luego asociar una carpeta del contenedor al hacer un volumen Bind, lo mejor es ya definirla en el contenedor.

```
root@f81b1e2681be: /home/ubuntu/python#
```

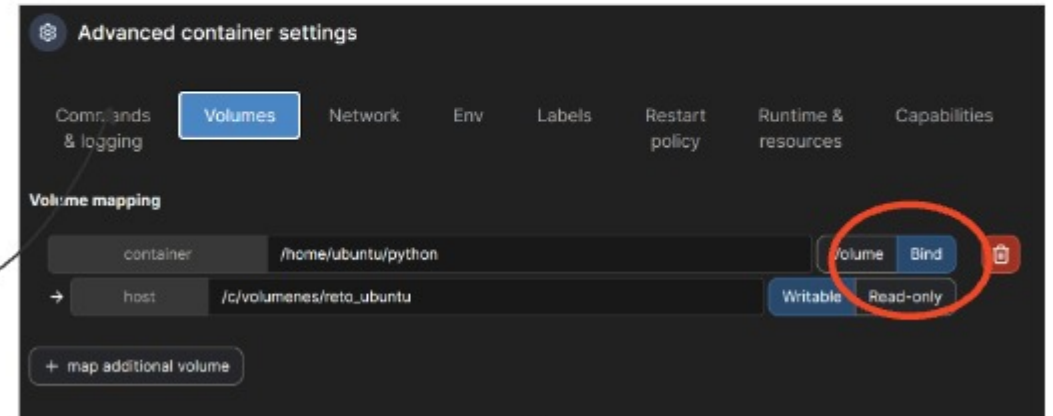
Creo una imagen personalizada, para poder hacer contenedores con esta configuración en el futuro.

```
docker commit mi-ubuntu mi-imagen-personalizada
```

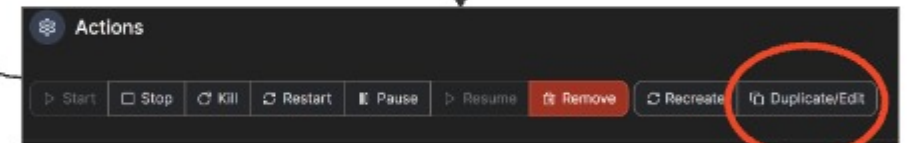
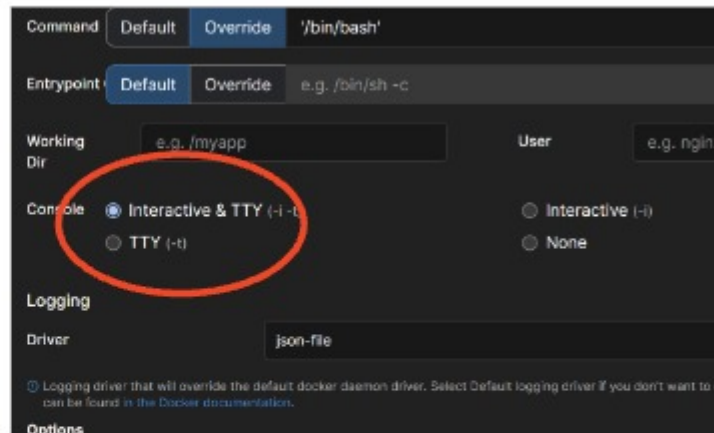
Reto 2

Creamos un nuevo contenedore con la imagen creada,

Con portainer



Para hacer el contenedor interactivo (el -it que ponemos en terminal)
Desde portainer, nos vamos a duplicate edit y marcamos el valor interactive TTY



Desde Terminal

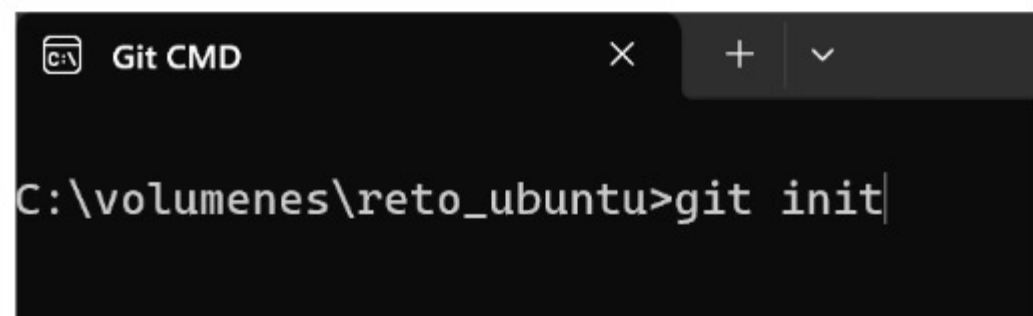
```
docker run --name reto_ubuntu_python -it -v C:\volumenes\reto_ubuntu:/home/ubuntu/python img_reto_ubuntu
```

Reto 3

Instalamos Git en el host (Windows)

<https://git-scm.com/> (Todo Next)

Una vez instalado abrimos el Git CMD desde el boton de inicio (Win)

A screenshot of a Windows terminal window titled "Git CMD". The window shows the command prompt at the path "C:\volumenes\reto_ubuntu" with the command "git init" entered and the cursor at the end of the line.

```
C:\volumenes\reto_ubuntu>git init|
```

En mi cuenta de Github creo un nuevo repositorio y hago:

- commit
- remote
- push

Reto 4

Create container ↻

Name: reto_mysql

Image Configuration

Registry: Docker Hub (anonymous) ▾

Image: docker.io mysql:latest 🔍 Search

Advanced mode

Always pull the image

You are currently using an anonymous account to pull images from DockerHub and will be limited to 100 pulls every 6 hours. You can configure DockerHub authentication in the [Registries](#) view. Remaining pulls: 98/100

Webhook

Create a container webhook Business feature

Network ports configuration

Publish all exposed ports to random host ports

Port mapping

host	3308	→	container	3306	tcp	udp	🗑️
------	------	---	-----------	------	-----	-----	----

Advanced container settings

Commands & logging | **Volumes** | Network | Env | Labels | Restart policy | Runtime & resources | Capabilities

Volume mapping

container	/var/lib/mysql	Volume	Bind	🗑️
→	volume	db_sql_reto - local	Writable	Read-only

+ map additional volume

Reto 5 - A

Conexion a BD desde Python

```
leerBd.py > [e] res
1  import mysql.connector
2
3
4  #Datos de conexion:
5
6  connection = mysql.connector.connect(
7      user='root',
8      password='1234',
9      host='172.17.0.2',
10     database='retobd'
11
12 )
13 #creo el objeto que gestionara la llamada
14 # y el retorno de los datos En python,cursor.
15 cursor = connection.cursor()
16 cursor.execute("select * from coches")
17 #los paso a un diccionario para leerlo con un bucle
18 res = cursor.fetchall()
19 for coche in res:
20     print(coche)
21
22
```

Reto 5 - B

Conexion a BD desde Python - Con funciones

```
import mysql.connector
#Datos de conexion:
config = {
    'user': 'root', 'password': '1234',
    'host': '172.17.0.3', 'database': 'bdcoches'
}
def conectar(config):
    conexion = mysql.connector.connect()
    if conexion.is_connected():
        print("Conexiones establecida a la BD")
        return conexion
    else:
        print("Merda, no he podido conectar. Buscate la vida para ar
```

Reto 5 - C

Recuperacion de datos.

```
def consultaCoches():  
    conexion = conectar()  
    #creo el objeto que gestionara la llamada  
    # y el retorno de los datos En python,cursor.  
    cursor = conexion.cursor()  
    cursor.execute("select * from coches")  
    #los paso a un diccionario para leerlo con un bucle  
    res = cursor.fetchall()  
    for coche in res:  
        print(coche)
```

```
consultaCoches()
```

Reto 5 - D



```
root@efb1997f2808:/home/ubuntu/python# python3 LeerBd.py
(1, 'Toyota', 'Corolla', 'Blanco', 45000, 15000.0)
(2, 'Honda', 'Civic', 'Negro', 60000, 17000.0)
(3, 'Ford', 'Focus', 'Azul', 50000, 14000.0)
(4, 'Chevrolet', 'Malibu', 'Rojo', 70000, 16000.0)
(5, 'Nissan', 'Sentra', 'Gris', 55000, 15500.0)
(6, 'Volkswagen', 'Jetta', 'Blanco', 40000, 16500.0)
(7, 'Hyundai', 'Elantra', 'Negro', 45000, 14500.0)
(8, 'Kia', 'Rio', 'Azul', 30000, 13500.0)
(9, 'Mazda', 'Mazda3', 'Rojo', 48000, 15800.0)
(10, 'Subaru', 'Impreza', 'Gris', 53000, 16800.0)
(11, 'BMW', 'Serie 3', 'Blanco', 75000, 28000.0)
(12, 'Mercedes-Benz', 'Clase C', 'Negro', 82000, 32000.0)
(13, 'Audi', 'A4', 'Azul', 69000, 29000.0)
(14, 'Lexus', 'IS', 'Rojo', 77000, 31000.0)
(15, 'Tesla', 'Model 3', 'Blanco', 25000, 45000.0)
(16, 'Jeep', 'Cherokee', 'Verde', 85000, 20000.0)
(17, 'Dodge', 'Charger', 'Negro', 62000, 26000.0)
(18, 'Ford', 'Mustang', 'Rojo', 54000, 27000.0)
(19, 'Chevrolet', 'Camaro', 'Amarillo', 50000, 26500.0)
(20, 'Nissan', 'Altima', 'Gris', 72000, 17500.0)
(21, 'Hyundai', 'Tucson', 'Azul', 41000, 19000.0)
(22, 'Kia', 'Sportage', 'Blanco', 38000, 18500.0)
(23, 'Toyota', 'RAV4', 'Negro', 60000, 22500.0)
```

Reto 5 - Problemas



Existe acutalmente un problema con la version de ubuntu y el connector de mysql. Para solucionarlo vamos a remover el modulo actual e intalar una version anterior.

1 - Purgamos la isntalcion del modulo del conector.

apt remove --purge python3-mysql.connector

2- descargamos la version anterior, en este caso la 8,4

```
root@efb1997f2808:/home/ubuntu/python# wget https://downloads.mysql.com/archives/get/p/29/file/mysql-connector-python-py3_8.4.0-1ubuntu24.04_amd64.deb
```

- Para ejecutar el wget tendremos que instalarlo con **apt install wget**

3 - Desempaquetamos el paquete .deb descargado

dpkg -i mysql-connector-python-py3_8.4.0-1ubuntu24.04_amd64.deb

Y ya deberia funcionar.

Otra forma ser'ia usando el PIP con

pip install mysql-connector-python --break-system-packages

<https://downloads.mysql.com/archives/c-python/>

A screenshot of a web browser displaying the MySQL Product Archives page. The page title is "MySQL Product Archives" and the sub-page is "MySQL Connector/Python (Archived Versions)". A yellow warning banner at the top states: "Please note that these are old versions. New releases will have recent bug fixes and features! To download the latest release of MySQL Connector/Python, please visit MySQL Downloads." Below the banner are three dropdown menus for "Product Version" (set to 8.4.0), "Operating System" (set to Ubuntu Linux), and "OS Version" (set to Ubuntu Linux 24.04 (Bionic, C4.04)). A table below lists several "DEB Package, Python" entries with their respective dates (Apr 25, 2024), sizes (1.0M, 0.7M, 142.1K, 8.4M), and "Download" buttons. The first entry is for "mysql-connector-python-py3_8.4.0-1ubuntu24.04_amd64.deb".

Reto - Mostrat los datos en tabla

Si queremos podemos hacer que los datos los muestre de una forma mas bonita, como en una tabla. Tenemos muchas opciones, vamos a probar prettytable.

1- Instalamos la libreria

apt install python3-prettytable

2 - Modificamos el codigo.

```
import mysql.connector  
from prettytable import PrettyTable;
```

```
res = cursor.fetchall()  
tabla = PrettyTable(["ID", "Marca", "Modelo", "Color", "Kilometraje", "Precio"])  
for fila in res:  
    tabla.add_row(fila)  
print(tabla)
```



ID	Marca	Modelo	Color	Kilometraje	Precio
1	Toyota	Corolla	Blanco	45000	15000.0
2	Honda	Civic	Negro	60000	17000.0
3	Ford	Focus	Azul	50000	14000.0
4	Chevrolet	Malibu	Rojo	70000	16000.0
5	Nissan	Sentra	Gris	55000	15500.0
6	Volkswagen	Jetta	Blanco	40000	16500.0
7	Hyundai	Elantra	Negro	45000	14500.0
8	Kia	Rio	Azul	30000	13500.0
9	Mazda	Mazda3	Rojo	48000	15800.0
10	Subaru	Impreza	Gris	53000	16800.0
11	BMW	Serie 3	Blanco	75000	28000.0
12	Mercedes-Benz	Clase C	Negro	82000	32000.0
13	Audi	A4	Azul	69000	29000.0
14	Lexus	IS	Rojo	77000	31000.0
15	Tesla	Model 3	Blanco	25000	45000.0
16	Jeep	Cherokee	Verde	85000	20000.0
17	Dodge	Charger	Negro	62000	26000.0
18	Ford	Mustang	Rojo	54000	27000.0
19	Chevrolet	Camaro	Amarillo	50000	26500.0
20	Nissan	Altima	Gris	72000	17500.0
21	Hyundai	Tucson	Azul	41000	19000.0
22	Kia	Sportage	Blanco	38000	18500.0
23	Toyota	RAV4	Negro	60000	22500.0
24	Mazda	CX-5	Rojo	47000	23000.0
25	Volkswagen	Tiguan	Gris	52000	24000.0

Reto MongoDB - a

Ahora vamos a crear un contenedor para utilizar MongoDB.

Los datos importantes del contenedor son:

Imagen: mongo

Puerto:27017

Variables de Entorno:

MONGO_INITDB_ROOT_USERNAME,

MONGO_INITDB_ROOT_PASSWORD

Puedes crear un volumen asociando la ruta al contenedor en **/data/db**

Docker ademas creara otro volumen asociado a **/data/configdb** automaticamente.

Reto MongoDB - Acceder a mongo desde la terminal

Como estamos utilizando un conetenedor, accedemos mediante **docker exec -it nombreContenedor bash**

Una vez en la terminal del contenedor, accedemos a Mongo con usuario y contraseña:

mongosh -u usuario -p pass --authenticationDatabase admin

```
root@8a6d4743ba5a:/# mongosh -u admin -p 1234 --authenticationDatabase admin
Current Mongosh Log ID: 679c8606ee0dc94ea7e94969
Connecting to:      mongodb://<credentials>@127.0.0.1:27017/?directConnection=true&source=admin&appName=mongosh+2.3.4
Using MongoDB:     8.0.4
Using Mongosh:     2.3.4

For mongosh info see: https://www.mongodb.com/docs/mongodb-shell/

-----
The server generated these startup warnings when booting
2025-01-30T12:03:24.944+00:00: Using the XFS filesystem is strongly recommended with the
XFS file system. See http://dochub.mongodb.org/core/prodnotes-filesystem
2025-01-30T12:03:25.798+00:00: For customers running the current memory allocator with
the following sysfsFile
2025-01-30T12:03:25.798+00:00: We suggest setting the contents of sysfsFile to 0.
2025-01-30T12:03:25.798+00:00: Your system has glibc support for rseq built in, with
c-google and has critical performance implications. Please set the environment variable
rseq=0
2025-01-30T12:03:25.798+00:00: vm.max_map_count is too low
```

Reto MongoDB - Mongo Compass

Tambien podemos acceder a mongo utilizando IDEs como Mongo Compass.

Mongo nos da la opci'on de utilizar Mongo Atlas (de pago). En nuestro caso usaremos el Compass que es mas que suficiente.

Descargamos la ultima versi'on de MongoDB Compass Download (GUI) en:

<https://www.mongodb.com/try/download/compass>

Lo instalamos. (Windows da un aviso de seguridad, lo ingnoramos pues es una utilidad extendida y de confianza.)

Reto MongoDB - Mongo Compass (Conexion)

Para conectar solo tenemos que añadir una conexión poniendo el puerto, usuario y contraseña.

New Connection

Manage your connection settings

URI ⓘ Edit Connection String

mongodb://admin:*****@localhost:27017/

Name

Color

Favorite this connection
Favoriting a connection will pin it to the top of your list of connections

▼ Advanced Connection Options

General **Authentication** TLS/SSL Proxy/SSH In-Use Encryption Advanced

Authentication Method

Username/Password OIDC X.509 Kerberos LDAP AWS IAM

Username Optional

Password Optional

How do I find my connection string in Atlas?

If you have an Atlas cluster, go to the Cluster view. Click the 'Connect' button for the cluster to which you wish to connect. [See example](#)

How do I format my connection string?

[See example](#)

Reto MongoDB - Comandos basicos

Crear una base de datos use concesionario

Crear una colección e insertar un coche

```
db.coches.insertOne({  
  marca: "Toyota",  
  modelo: "Corolla",  
  año: 2022,  
  color: "Rojo"  
})
```

Ver los datos insertados

```
db.coches.find().pretty()
```

RECORDANDO

Descargar imagenes:

```
docker pull ubuntu
```

```
docker pull ubuntu:20.04
```

Crear y ejecutar contenedores

```
docker run -it ubuntu
```

```
docker run -it --name mi-ubuntu ubuntu
```

```
docker run --name mi-ubuntu -it -v /datos-  
persistentes ubuntu
```

```
docker run --name mi-ubuntu -it -v mi-  
volumen:/datos ubuntu
```

```
docker run -it -v C:\ruta\local:/ruta/en/contenedor  
ubuntu
```

Crear Volumenes

```
docker volume create mi-volumen
```

Consultar informacion

```
docker ps
```

```
docker ps -a
```

```
docker images
```

Imagen personalizadas

```
docker commit mi-ubuntu mi-imagen-personalizada
```

Operaciones con contenedores

```
docker start mi-ubuntu
```

```
docker exec -it mi-ubuntu bash
```

```
docker start -ai mi-ubuntu
```

```
docker rm mi-ubuntu
```

```
docker container prune
```

Reto MongoDB - Con Python

Lo primero es instalar el modulo de pymongo en el contenedor

```
apt install python3-pymongo
```

Reto MongoDB - Con Python

Datos de usuario

```
from pymongo import MongoClient

# Datos de conexión
usuario = "admin" # Reemplaza con tu usuario
contrasena = "1234" # Reemplaza con tu contraseña
base_datos = "coches" # Reemplaza con el nombre de tu base de datos

# Construcción de la URI para conexión
uri = f"mongodb://{usuario}:{contrasena}@172.17.0.4:27017/{base_datos}?authSource=admin"

# Conexión a MongoDB
cliente = MongoClient(uri)

# Verificar que la base de datos existe
print("Bases de datos disponibles:", cliente.list_database_names())

db = cliente["parking"]

# Obtener un documento de una colección
coleccion = db["autos"]

coches = coleccion.find() # Retorna un cursor con todos los documentos

for coche in coches:
    print(coche)
```

```
uri = f"mongodb://{usuario}:{contrasena}@172.17.0.4:27017/{base_datos}?authSource=admin"
```

Reto MongoDB - Con Python - En tabla

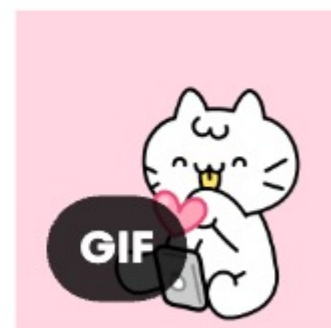
Creamos la tabla PrettyTable con las columnas adecuadas.

Iteramos sobre los coches, asegurándonos de que los valores se obtienen con `.get()` para evitar errores si falta algún campo.

```
tabla = PrettyTable(["_id","ID", "Marca", "Modelo", "Color", "Kilometraje", "Precio"])

for coche in coches:
    tabla.add_row([
        coche.get("_id", ""), # Obtener el ID de MongoDB
        coche.get("id", ""),
        coche.get("marca", ""),
        coche.get("modelo", ""),
        coche.get("color", ""),
        coche.get("km", ""),
        coche.get("precio", "")
    ])

print(tabla)
```



- Acuerdate de poner la importacion del modulo
from prettytable import PrettyTable;

_id	ID	Marca	Modelo	Color	Kilometraje	Precio
679c9dbda400a51a711370f8	1	Chevrolet	Civic	Rojo	111405	38929
679c9dbda400a51a711370f9	2	Honda	X5	Azul	16368	29465
679c9dbda400a51a711370fa	3	BMW	Corolla	Blanco	41487	14506
679c9dbda400a51a711370fb	4	BMW	Corolla	Blanco	5406	37067
679c9dbda400a51a711370fc	5	Toyota	X5	Rojo	7160	54274
679c9dbda400a51a711370fd	6	Honda	Camaro	Blanco	8248	43683
679c9dbda400a51a711370fe	7	BMW	Camaro	Gris		
679c9dbda400a51a711370ff	8	Toyota	Civic	Azul		
679c9dbda400a51a71137100	9	Ford	X5	Blanco		
679c9dbda400a51a71137101	10	Honda	Camaro	Negro		
679c9dbda400a51a71137102	11	BMW	Corolla	Blanco		
679c9dbda400a51a71137103	12	Chevrolet	Civic	Negro		
679c9dbda400a51a71137104	13	Chevrolet	Corolla	Rojo		
679c9dbda400a51a71137105	14	Toyota	Camaro	Gris		
679c9dbda400a51a71137106	15	Ford	Corolla	Gris		
679c9dbda400a51a71137107	16	Ford	Corolla	Blanco		
679c9dbda400a51a71137108	17	Chevrolet	Corolla	Azul		
679c9dbda400a51a71137109	18	Chevrolet	Mustang	Rojo		
679c9dbda400a51a7113710a	19	Toyota	Civic	Blanco		
679c9dbda400a51a7113710b	20	BMW	Civic	Blanco		
679c9dbda400a51a7113710c	21	Ford	Corolla	Azul		
679c9dbda400a51a7113710d	22	Honda	Mustang	Rojo		
679c9dbda400a51a7113710e	23	BMW	Corolla	Azul		
679c9dbda400a51a7113710f	24	Toyota	Civic	Rojo		
679c9dbda400a51a71137110	25	Chevrolet	Camaro	Gris		
679c9dbda400a51a71137111	26	Toyota	X5	Azul		
679c9dbda400a51a71137112	27	Honda	X5	Rojo		



Subir imagen a Docker-HUB - 1

Recuerda:

Para crear una imagen a partir de un contenedor

```
docker commit 09c2720 reto_python:latest
```

Nos identificamos con:

```
docker login
```

Etiquetar la imagen

Si necesitas subirla a Docker Hub, asegúrate de que esté correctamente etiquetada:

```
docker tag reto_python:latest ciberkaos/reto_python
```

Subimos la imagen

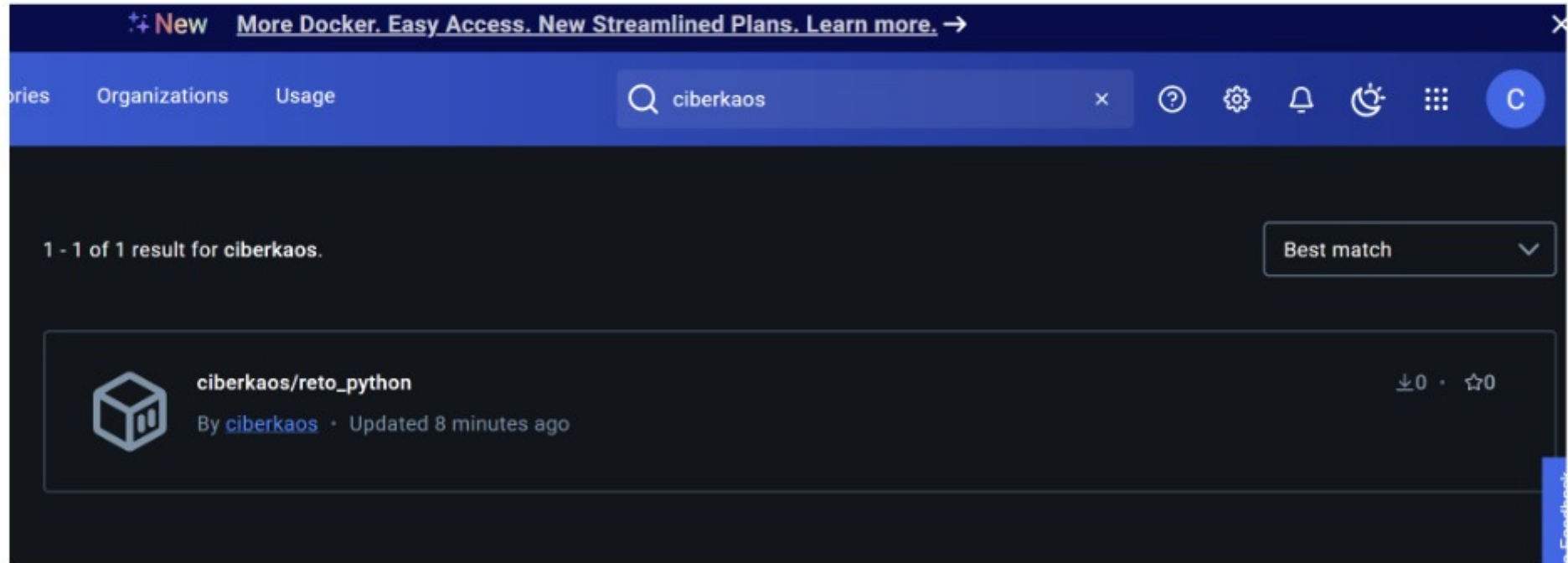
```
docker push ciberkaos/reto_python:latest
```

Subir imagen a Docker-HUB - 2

```
PS C:\WINDOWS\system32> docker login
Authenticating with existing credentials...
Login Succeeded
PS C:\WINDOWS\system32> docker commit 09c2720 ciberkaos/reto_python:latest
sha256:21d63b2e34344aac279284cb0c3c2408f3fdf453af5fdbd28c3b596b72aa2734
PS C:\WINDOWS\system32> docker tag reto_python:latest ciberkaos:reto_python
Error response from daemon: No such image: reto_python:latest
PS C:\WINDOWS\system32> docker tag reto_python:latest ciberkaos/reto_python:latest
Error response from daemon: No such image: reto_python:latest
PS C:\WINDOWS\system32> docker tag reto_python:latest ciberkaos/reto_python
Error response from daemon: No such image: reto_python:latest
PS C:\WINDOWS\system32> docker commit 09c2720 reto_python:latest
sha256:33a5e0b0c5607f6902d65202a112d6c5b7c034873428c8d0e67a5c4db1c6b1b0
PS C:\WINDOWS\system32> docker tag reto_python:latest ciberkaos/reto_python
PS C:\WINDOWS\system32> docker push ciberkaos/reto_python:latest
The push refers to repository [docker.io/ciberkaos/reto_python]
9c1006369178: Pushed
f000b55df3d7: Pushed
687d50f2f6a6: Mounted from library/mongo
latest: digest: sha256:95cd4dd8032e58cf025f9cc01498792c8eacc0d1601e5a376d26caff05182fd4 size: 954
PS C:\WINDOWS\system32> docker push ciberkaos/reto_python:latest
```

Subir imagen a Docker-HUB - 3

Comprobamos que nuestra imagen esta disponible en el hub de Docker



Utilizar la imagen subida al hub - 1

Para descargar la imagen solo tenemos que descargarla.

Si usamos una distro como parrot, tenemos que configurar el origen del hub, en nuestro caso el de docker.io

```
sudo nano /etc/containers/registries.conf
```

Busca o pega la sección [registries.search] y añade los registros

```
[registries.search]
```

```
registries = ['docker.io', 'quay.io']
```

Reinicia Podman:

```
systemctl --user restart podman
```

Utilizar la imagen subida al hub - 2

Ahora ya puedes descargar la imagen y/o desplegarla

docker pull ciberkaos/reto_ubuntu

```
[ciberkaos@parrot]-[~]  
$ docker run --rm -it ciberkaos/reto_python  
Emulate Docker CLI using podman. Create /etc/containers/nodocker to quiet msg.  
root@2565ac430b26:/# ls  
bin          etc          lib64       proc        sbin.usr-is-merged  usr  
bin.usr-is-merged  home        media       root        srv            var  
boot         lib          mnt         run         sys  
dev          lib.usr-is-merged  opt         sbin       tmp  
root@2565ac430b26:/# cd home  
root@2565ac430b26:/home# ls  
ubuntu  
root@2565ac430b26:/home#
```

