UT3 - Práctica 2

Parte 1 - Nextcloud

Buscar y descargar la imagen oficial de Nextcloud en Docker, versión 30.0.8 con Apache.

```
C:\Users\molin>docker pull nextcloud:30.0.8-apache
30.0.8-apache: Pulling from library/nextcloud
2719dd8b3581: Download complete
dacb60b59038: Download complete
64450047668b: Download complete
6a5a37a900f3: Download complete
4d6386e035f7: Download complete
```

docker pull: indica a Docker que descargue una imagen desde el repositorio oficial en Docker Hub.

Nextcloud:30.0.8-apache: especifica la imagen de Nextcloud en su versión **30.0.8** con Apache como servidor web.

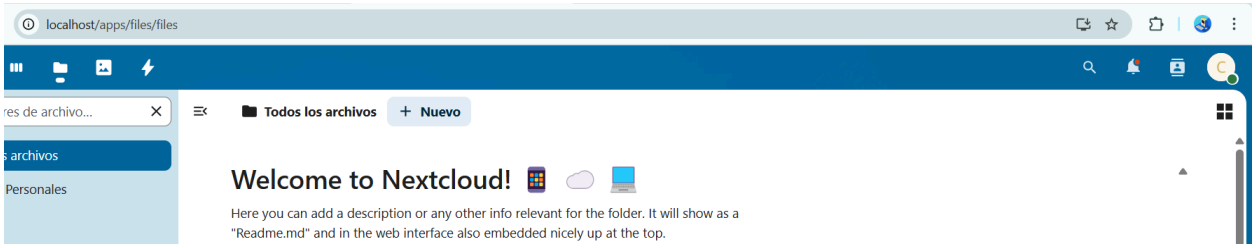
Crear un contenedor con las características dadas

```
PS C:\Users\molin> docker run -d --name ceeciimg-nextcloud -p 80:80 -v ceeciimg-nextvolume:/var/www/html nextcloud:30.0.8-apache
e9444eda8a1d168d4fc15c728f3cf459ff28b30cc86927eb222b7fe7a37c851c
PS C:\Users\molin> docker ps
CONTAINER ID   IMAGE                    COMMAND                  CREATED              STATUS              PORTS                  NAMES
e9444eda8a1d   nextcloud:30.0.8-apache   "/entrypoint.sh apac…"   About a minute ago   Up About a minute   0.0.0.0:80->80/tcp   ceeciimg-nextcloud
PS C:\Users\molin>
```

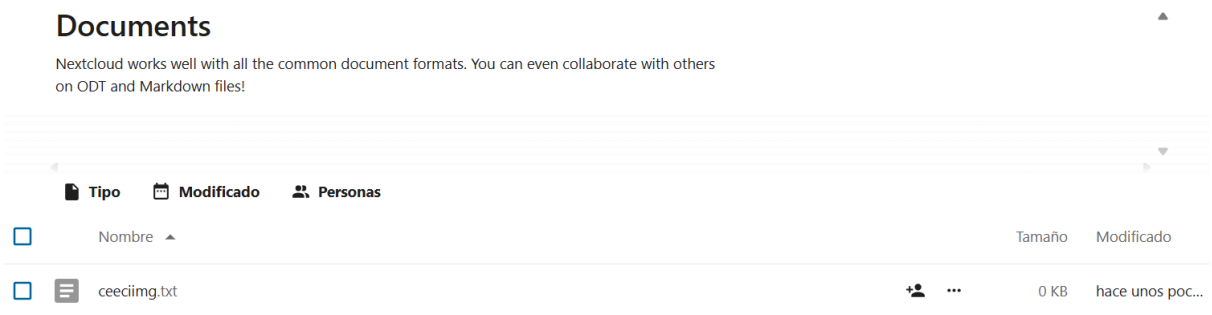
Explicación de los parámetros:

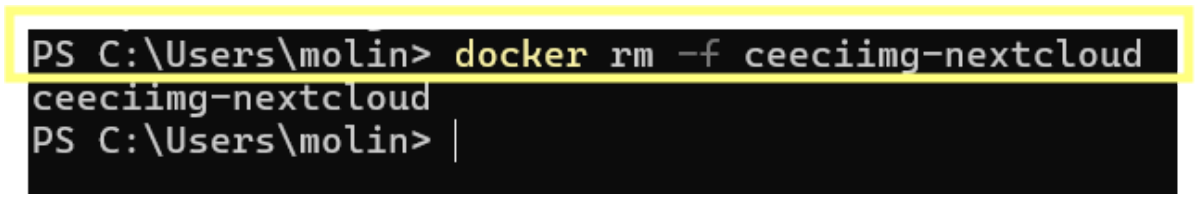
-d: Ejecuta el contenedor en segundo plano (modo demonio).
--name ceeciimg-nextcloud: Asigna el nombre de ceeciimg a mi contenedor
-p 80:80: Vincula el puerto 80 de mi máquina (host) con el puerto 80 del contenedor.
-v tunombre-nextvolume:/var/www/html: Vincula el volumen ceeciimg-nextvolume con el directorio donde Nextcloud guarda los datos (/var/www/html).
nextcloud:30.0.8-apache: Especifica la imagen de Nextcloud versión 30.0.8 con Apache.

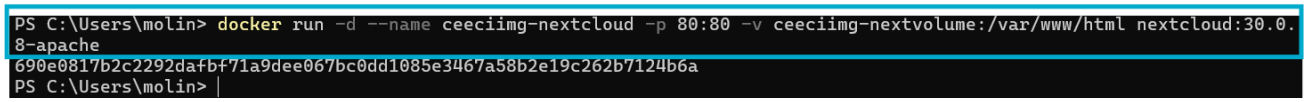
Instalar la VPC

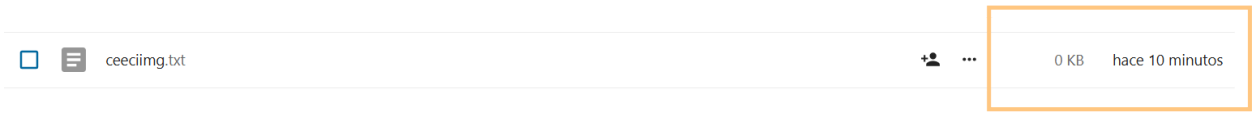


Verificación de persistencia de datos en contenedor Docker con volumen: Subida de archivo y restauración tras eliminar y recrear contenedor









Conclusión: se observa que después de eliminar y volver a crear el contenedor, sigue intacto el archivo que subí llamado “ceeciimgtxt”

Parte 2 - Entorno de desarrollo y uso de bibliotecas Python

Crear una carpeta en el escritorio con nombre “tunombre-python”

Crear el contenedor (sin ejecutarlo)

```
PS C:\Users\molin> docker create -it --name ceeciimg-entornopython `
>>   -v "C:\Users\molin\Desktop\ceeciimg-python:/usr/src/app" `
>>   -w /usr/src/app `
>>   python:3.11-slim
d71c6512e8f3b3f241626acc4aa7b7fd83061b04fbcbb10058c5198973fef991
PS C:\Users\molin>
```

docker create crea un contenedor sin ejecutarlo.
-it permite interactuar con el contenedor desde la terminal.
--name ceeciimg-entornopython asigna el nombre **ceeciimg-entornopython** al contenedor.
-v "C:\Users\molin\Desktop\ceeciimg-python:/usr/src/app" vincula la carpeta local con /usr/src/app dentro del contenedor.
-w /usr/src/app define /usr/src/app como directorio de trabajo dentro del contenedor.
python:3.11-slim usa la imagen oficial de Python 3.1.

Poner en marcha el contenedor

```
PS C:\Users\molin> docker start ceeciimg-entornopython
ceeciimg-entornopython
PS C:\Users\molin>
```

Conectar al terminal del contenedor

```
PS C:\Users\molin> docker exec -it ceeciimg-entornopython bash
root@d71c6512e8f3:/usr/src/app#
```

Descargar la librería de Python ‘NumPy’

```
PS C:\Users\molin> docker exec -it ceeciimg-entornopython bash
root@d71c6512e8f3:/usr/src/app# pip install numpy
Collecting numpy
  Downloading numpy-2.2.4-cp311-cp311-manylinux_2_17_x86_64.manylinux2014_x86_64.whl.metadata (62 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 62.0/62.0 kB 903.2 kB/s eta 0:00:00
Downloading numpy-2.2.4-cp311-cp311-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (16.4 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 6.1/16.4 MB 1.3 MB/s eta 0:00:08
```

Crear en la carpeta vinculada del host un archivo .py

script.py

C: > Users > molin > Desktop > ceeciimg-python > script.py

```python
import numpy as np

# Crear un array de 10 números aleatorios entre 0 y 100
arr = np.random.randint(0, 100, size=10)
print("Array aleatorio:", arr)

# Obtener estadísticas básicas
print("Media:", np.mean(arr))
print("Mediana:", np.median(arr))
print("Desviación estándar:", np.std(arr))
print("Mínimo:", np.min(arr), "Máximo:", np.max(arr))

# Ordenar el array
arr_sorted = np.sort(arr)
print("Array ordenado:", arr_sorted)

# Crear una matriz 3x3 con valores aleatorios entre 0 y 10
matrix = np.random.randint(0, 10, size=(3, 3))
print("\nMatriz aleatoria 3x3:\n", matrix)

# Sumar y multiplicar filas y columnas
print("Suma por filas:", np.sum(matrix, axis=1))
print("Suma por columnas:", np.sum(matrix, axis=0))

# Producto punto de la matriz consigo misma
matrix_product = np.dot(matrix, matrix)
print("Producto punto:\n", matrix_product)

```

Ejecutar el archivo en el contenedor

```
root@d71c6512e8f3:/usr/src/app# python script.py
Array aleatorio: [66 84  4 62 37 94 15 62 71 87]
Media: 58.2
Mediana: 64.0
Desviación estándar: 28.781243892507497
Mínimo: 4 Máximo: 94
Array ordenado: [ 4 15 37 62 62 66 71 84 87 94]

Matriz aleatoria 3x3:
 [[6 3 1]
 [1 7 0]
 [6 3 5]]
Suma por filas: [10  8 14]
Suma por columnas: [13 13  6]
Producto punto:
 [[45 42 11]
 [13 52  1]
 [69 54 31]]
root@d71c6512e8f3:/usr/src/app#
```