

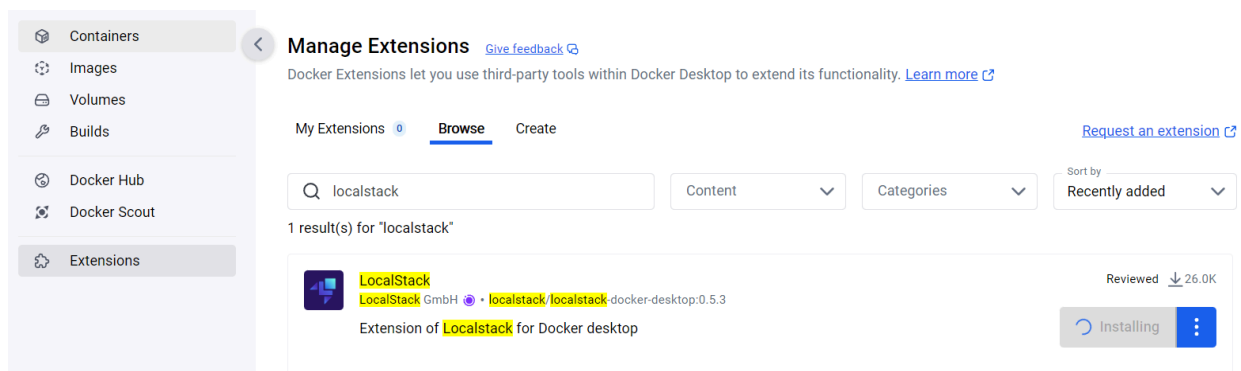
# LocalStack

## Parte 1 - Preparación del entorno

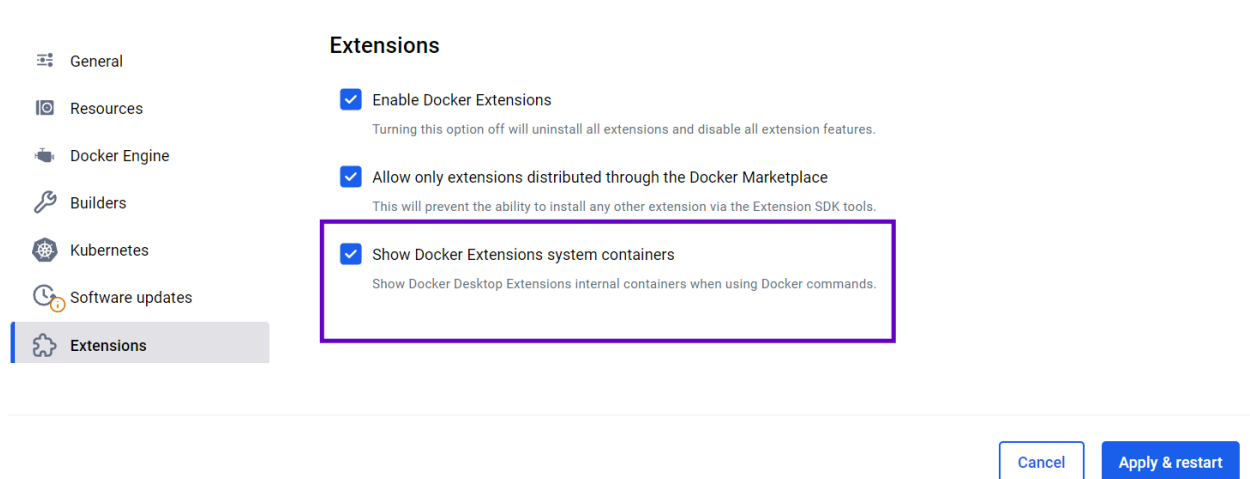
### 1. Instalar Ubuntu desde la terminal:

```
PS C:\Users\sofia> wsl -l -v
  NAME                STATE      VERSION
* docker-desktop      Running    2
PS C:\Users\sofia> wsl --install -d Ubuntu
Descargando: Ubuntu
Instalando: Ubuntu
Distribución instalada correctamente. Se puede iniciar a través de "wsl.exe -d Ubuntu"
PS C:\Users\sofia> wsl -l -v
  NAME                STATE      VERSION
* docker-desktop      Running    2
  Ubuntu              Stopped    2
PS C:\Users\sofia>
```

### 2. Instalar LocalStack desde la interfaz gráfica de Docker Desktop:



### 3. Ajustes en Docker Desktop:



General

Resources

Advanced

Proxies

Network

WSL integration

Docker Engine

Builders

Resources

WSL integration

Configure which WSL 2 distros you want to access Docker from.

Enable integration with my default WSL distro

Enable integration with additional distros:

Ubuntu

Refetch distros

4. Arrancar LocalStack

Containers

Images

Volumes

Builds

Docker Hub

Docker Scout

Extensions

Manage

LocalStack

LocalStack

LocalStack Web Application

Stop

RUNNING

System Status

Configurations

Logs

Refresh

Available

acm

available

config

available

es

available

kinesis

available

opensearch

available

route53

available

scheduler

available

sqs

apigateway

available

dynamodb

available

events

available

kms

available

redshift

available

route53resolver

available

secretsmanager

available

ssm

cloudformation

available

dynamodbstreams

available

firehose

available

lambda

available

resource-groups

available

s3

available

ses

available

stepfunctions

cloudwatch

available

ec2

available

iam

available

logs

available

resourcegroupstaggingapi

available

s3control

available

sns

available

sts

5. Configurar AWS CLI en el contenedor principal

Containers / localstack-main

localstack-main

3ff7acbd582f

localstack/localstack:latest

4510:4510

4511:4511

Show all ports (54)

STATUS

Running (6 seconds ago)

Logs

Inspect

Bind mounts

Exec

Files

Stats

Debug mode

Open in external terminal

# aws configure

#

#

# aws configure

AWS Access Key ID [\*\*\*\*\*mbre]: sofia

AWS Secret Access Key [\*\*\*\*\*1234]: 1234

Default region name [Default region name: us-east-1]: us-east-1

Default output format [Default output format: json]: json

#

© 2023 Docker

## Parte 2 - S3

**1. Amazon S3 (Simple Storage Service):** es un servicio de almacenamiento de objetos en la nube que permite guardar y recuperar cualquier tipo de archivo a través de Internet. Es utilizado para almacenar datos como imágenes, documentos, copias de seguridad... Los archivos se guardan en unidades llamadas buckets, y cada bucket tiene un nombre único.

### 2. Crear dos buckets de S3:

```
# aws s3 mb s3://sofia-bucket-provisional --endpoint-url=http://localhost:4566
aws s3 mb s3://sofia-bucket-definitivo --endpoint-url=http://localhost:4566
make_bucket: sofia-bucket-provisional
# make_bucket: sofia-bucket-definitivo
#
```

### 3. Listar los buckets y comprobar que S3 está activo

```
# aws s3 ls --endpoint-url=http://localhost:4566      Running
2025-04-23 14:33:16 sofia-bucket-provisional
2025-04-23 14:33:17 sofia-bucket-definitivo           s3
#                                                       ● running
```

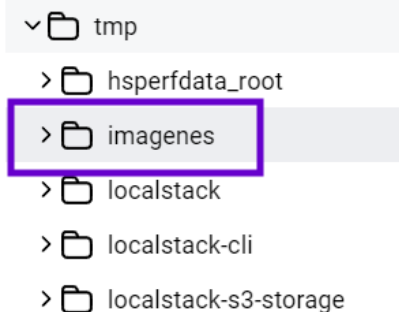
### 4. Crear y subir un archivo archivo.txt

```
# echo "Estoy aprendiendo a utilizar aws" > archivo.txt
# aws s3 cp archivo.txt s3://sofia-bucket-provisional --endpoint-url=http://localhost:4566
upload: ./archivo.txt to s3://sofia-bucket-provisional/archivo.txt
# aws s3 ls s3://sofia-bucket-provisional --endpoint-url=http://localhost:4566
2025-04-23 14:37:49          33 archivo.txt
#
```

---

## 5. Subir una imagen al bucket

### 5.1. Crear una carpeta dentro de la carpeta tmp del contenedor:



```
▼ tmp
  > hsperrdata_root
  > imagenes
  > localstack
  > localstack-cli
  > localstack-s3-storage
```

### 5.2. Desde la terminal, subir la imagen a la carpeta:

```
# cd /tmp
# aws s3 cp imagen.jpg s3://sofia-bucket-provisional --endpoint-url=http://localhost:4566
upload: ./imagen.jpg to s3://sofia-bucket-provisional/imagen.jpg
```

### 5.3. Verificar si se ha subido:

```
# aws s3 ls s3://sofia-bucket-provisional --endpoint-url=http://localhost:4566
2025-04-23 14:37:49          33 archivo.txt
2025-04-23 14:51:15       102 imagen.jpg
# █
```

## 6. Copiar imagen y archivo al bucket definitivo

```
# aws s3 cp s3://sofia-bucket-provisional/archivo.txt s3://sofia-bucket-definitivo/ --endpoint-url=http://localhost:4566
copy: s3://sofia-bucket-provisional/archivo.txt to s3://sofia-bucket-definitivo/archivo.txt
# aws s3 cp s3://sofia-bucket-provisional/imagen.jpg s3://sofia-bucket-definitivo/ --endpoint-url=http://localhost:4566
copy: s3://sofia-bucket-provisional/imagen.jpg to s3://sofia-bucket-definitivo/imagen.jpg
# █
```

Verificar:

```
# aws s3 ls s3://sofia-bucket-definitivo --endpoint-url=http://localhost:4566
2025-04-23 14:53:22          33 archivo.txt
2025-04-23 14:53:32       102 imagen.jpg
# █
```

## 7. Descargar y mostrar el contenido del archivo

```
# aws s3 cp s3://sofia-bucket-definitivo/archivo.txt archivo_descargado.txt --endpoint-url=http://localhost:4566
download: s3://sofia-bucket-definitivo/archivo.txt to ./archivo_descargado.txt
# cat archivo_descargado.txt
Estoy aprendiendo a utilizar aws
# █
```

## 8. Eliminar el bucket provisional

### 8.1. Primero eliminar el contenedor

```
# aws s3 rm s3://sofia-bucket-provisional/archivo.txt --endpoint-url=http://localhost:4566
delete: s3://sofia-bucket-provisional/archivo.txt
# aws s3 rm s3://sofia-bucket-provisional/imagen.jpg --endpoint-url=http://localhost:4566
delete: s3://sofia-bucket-provisional/imagen.jpg
# █
```

### 8.2. Eliminar el bucket

```
# aws s3 rb s3://sofia-bucket-provisional --endpoint-url=http://localhost:4566
remove_bucket: sofia-bucket-provisional
# █
```

## Parte 3 - Lambda

**1. AWS Lambda:** es un servicio que permite ejecutar funciones (código) sin necesidad de gestionar servidores. Solo se sube una función, se define un evento que la dispare, y AWS se encarga de ejecutarla automáticamente cuando se necesita.

## 2. Crear el archivo handler.py

```
handler.py ●
C: > Users > sofia > Desktop > handler.py
1 def handler(event, context):
2     nombre = event.get("nombre", "desconocido")
3     return f"Hola, {nombre}. Esta es tu función Lambda funcionando."
4
```

## 3. Comprimir el archivo en .zip y moverlo al contenedor de LocalStack

3.1. Copiar el archivo handler.zip al contenedor localstack-main, dentro de la ruta /tmp/lambda/

```
PS C:\Users\sofia> docker cp "C:\Users\sofia\Desktop\handler.zip" localstack-main:/tmp/lambda/
Successfully copied 2.05kB to localstack-main:/tmp/lambda/
PS C:\Users\sofia>
```

### 3.2.

```
# ls /tmp/lambda
handler.zip
#
```

## 4. Crear la función Lambda en LocalStack

```
# aws lambda create-function \
--function-name sofia-lambda \
--runtime python3.9 \
--handler handler.handler \
--zip-file fileb:///tmp/lambda/handler.zip \
--role arn:aws:iam::000000000000:role/lambda-role \
--endpoint-url=http://localhost:456> 6
> > > > {
  "FunctionName": "sofia-lambda",
  "FunctionArn": "arn:aws:lambda:us-east-1:000000000000:function:sofia-lambda",
  "Runtime": "python3.9",
  "Role": "arn:aws:iam::000000000000:role/lambda-role",
  "Handler": "handler.handler",
  "CodeSize": 192,
  "Description": "",
  "Timeout": 3,
  "MemorySize": 128,
  "LastModified": "2025-04-23T15:43:03.679920+0000",
  "CodeSha256": "YjKhNCQKiuFa4jw7PcZCK5gJKT2GXn5QK8NqO3Am6TQ=",
  "Version": "$LATEST",
```

---

## 5. Ejecutar la función Lambda

```
# aws lambda invoke \  
--function-name sofia-lambda \  
--payload '{"nombre":"Sofia"}' \  
--endpoint-url=http://localhost:4566 \  
respuesta.json  
> > > >  
{  
  "StatusCode": 200,  
  "FunctionError": "Unhandled",  
  "ExecutedVersion": "$LATEST"  
}  
# #
```

## 6. Abrir el archivo respuesta.json

```
# cat respuesta.json  
{"errorMessage": "Handler 'handler' missing on module 'handler'", "errorType": "Runtime.HandlerNotFound", "requestId": "", "stackTrace": []}#
```

Este error aparece porque LocalStack (en su versión gratuita) no puede ejecutar funciones Lambda como lo haría AWS. Aun así, el hecho de que la función se haya creado y devuelto una respuesta ya se considera correcto para esta práctica. El error no es por algo que haya hecho mal, sino por las limitaciones del entorno, así que con esto ya tendría la parte de Lambda completada.