

- Arreglo de documentación en DEV para el despliegue
- prueba de primer despliegue con Docker y AWS
- Empezar a desarrollar con SOLID
- BD MongoDB. Diseñar una de prueba y su modelo

SOLID son los principios para mantener los POD

API REST = intermediario entre usuarios/clientes y recursos/servicios web a obtener
no necesitamos saber cómo se recibe el recurso ni de donde viene

PRUEBA PRIMER DESPLIEGUE CON DOCKER

Desplegar una máquina virtual usando SSH. Funciona con cualquier servicio en la nube, o nuestro propio servidor



Crear una MV en un servicio en la nube,
instalar docker y docker-compose,
y desplegar nuestros contenedores ahí usando Github Actions y SSH

OPCION 1) Microsoft Azure

OPCION 2) AWS

OPCION 3) Google Cloud

Microsoft Azure

Dentro de la MV, en Networking (Redes):

- IP pública: para acceder a la máquina 20.168.215.236
- configurar puertos accesibles (3000, 5000)

Ahora accedemos a la MV usando SSH e instalaremos Docker.

- Importante que a la clave privada solo tenga acceso de lectura, y solo yo.
- ```
ssh -i <rutaaccesoclaveprivada> azureuser@<IP-PUBLICA>
```

### GitHub Actions

Nuestra MV ya puede ejecutar contenedores de Docker

Vamos a crear 3 GitHub Secrets:

- DEPLOY\_HOST la IP de la MV
- DEPLOY\_USER usuario con permisos para acceder a la MV (azureuser)
- DEPLOY\_KEY contenido del fichero con la clave privada

Necesitamos crear un nuevo token de acceso en GitHub con permisos de escritura de paquetes y establecerlo en el "secreto"

- DOCKER\_PUSH\_TOKEN (ya venía hecho en el repo)

docker-compose-deploy.yaml

copiamos el código, ya hecho de base

Contiene las instrucciones para desplegar la aplicación

```

azureuser@deploymentASW223:~$ sudo apt update
[sudo] password for azureuser:
sudo apt install docker-ce
sudo usermod -s6 docker $(USER)
sudo curl -L "https://github.com/docker/compose/releases/download/1.28.5/docker-compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose
Hit:1 http://azure.archive.ubuntu.com/ubuntu focal InRelease
Hit:2 http://azure.archive.ubuntu.com/ubuntu focal-updates InRelease
Hit:3 http://azure.archive.ubuntu.com/ubuntu focal-backports InRelease
Hit:4 http://azure.archive.ubuntu.com/ubuntu focal-security InRelease
Reading package lists... Done
Building dependency tree
Reading state information... Done
16 packages can be upgraded. Run 'apt list --upgradable' to see them.
azureuser@deploymentASW223:~$ sudo apt install apt-transport-https ca-certificates curl software-properties-common
Reading package lists... Done
Building dependency tree
Reading state information... Done
ca-certificates is already the newest version (20211016ubuntu0.20.04.1).
ca-certificates set to manually installed.
The following additional packages will be installed:
 libcurl4 python3-software-properties
The following NEW packages will be installed:
 apt-transport-https
The following packages will be upgraded:
 curl libcurl4 python3-software-properties software-properties-common
upgraded, 1 newly installed, 0 to remove and 12 not upgraded.
Need to get 439 kB of archives.
After this operation, 162 kB of additional disk space will be used.
Do you want to continue? [Y/n] Abort.
azureuser@deploymentASW223:~$

```

Falta 162 KB de espacio

lomap\_esGb.yml

descomentamos la sección para desplegar en SSH

Desde mi rama local lo modifiqué

## Crear una nueva versión

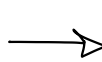
Ya está todo listo para desplegar.

Crearemos una versión 0.00 que desplegará el proceso que acabamos de configurar

## PREGUNTAR PORQUÉ FALLÓ AL DESPLEGAR

¿No se pueden hacer pruebas desde ramas que no sean Master?

- OPCIÓN 1: no se instaló bien Docker en la MV  
↳ repito el proceso de instalaci, pero esta vez línea a línea



**SOLUCIONADO** ✓

Ya **pasa todos los test** de la versión

Aunque aun **no despliega la página**  
desde localhost:3000

---

## EMPEZAR A DESARROLLAR CON SOLID

Para empezar necesitamos un POD de Solid, y un WebID

### SERVIDOR DE PODS EXISTENTE + APLICACIÓN DEMO:

Empecé a probar usando un servidor de PODs existente: **INRUPT POD SPACES** y obtuve mi propio POD y WebID.

- Cree una carpeta nueva en mi escritorio para almacenar esta demo
  - Dentro de ella, `npm init --yes`, para crear la app.
  - Instalamos las librería `Inrupt Client Libraries`
  - instalamos `Parcel`
  - Edita `package.json` para listar los navegadores soportados por la aplicación.
  - Crea los ficheros de la app dentro del directorio para la aplicación.
  - Ejecuta la app con `npx parcel index.html`
  - Abre `localhost:1234` en el navegador
  - Para cerrar, `Ctrl + C`
-

# MODELO BASE DE DATOS

Me creé una cuenta personal en MongoDB @aliciatp15 y añadí la IP de mi ordenador.

## INFORMACIÓN CENTRALIZADA (Por rendimiento)

| Placemark<br>Chincheta |
|------------------------|
| - ID: Integer          |
| - longitud: Double     |
| - latitud: Double      |
| - ID_POD: String       |

Las chinchetas nos sirven para colocarlas en el mapa sin mostrar info privada.

Relaciona el punto con el POD. Así ya se relaciona con el usuario sin interferir en su intimidad

- categoría: String  
Ej: Bar, tienda, monumento, paisaje, restaurante, ...  
→ no creo que deba ir aquí

| Pod          |
|--------------|
| - ID: String |

Acceso a la información privada.  
El contenido relevante estará descentralizado

Es el usuario de cada POD (el contenido de Web-ID)

---

## INFORMACIÓN DESCENTRALIZADA (pod por cada usuario)

| USUARIO                |
|------------------------|
| - ID: Integer          |
| - ID_POD: String       |
| - amigos: List<String> |
| - lugares:             |

Almacena su POD. Para cuando necesitemos p.e "mostrar todos los puntos del usuario 2."

| Place          |
|----------------|
| - ID: Integer  |
| - categoría    |
| - puntuaciones |
| - comentarios  |
| - fotos        |

esto es personal, no tengo claro donde va