

# AAP\_COP1

## Historial

Owner	Fecha	Detalle
Hernan Conosciuto	Marzo 2021	Creación de la versión 1.0

## Descripción.

El objetivo de esta aplicación es utilizarla como demo de OCP.

La misma crea registros en un lapso de tiempo configurable los cuales se componen de 2 partes, un timestamp y nombres random.

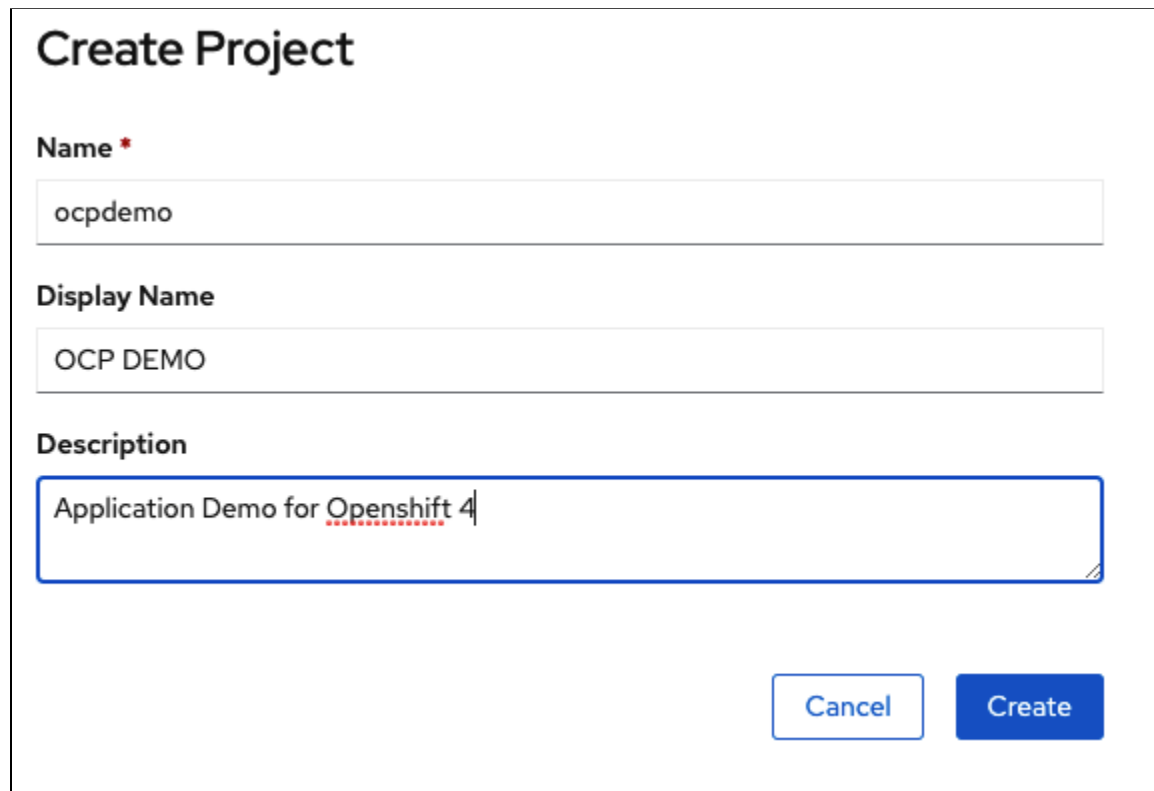
Además de mostrar la **creación de aplicaciones** y cambio de configuración por **environment variables** se puede utilizar para mostrar:

- el uso del pod scaling y generar caga que puede
- visualizarse tanto en el Dashboard de OCP como en la parte de
- ACM Observability.

# Despliegue en Openshift 4.x

Una vez creado el cluster de OCP...

Crear un proyecto, por ejemplo: **ocpdemo**



**Create Project**

**Name \***

ocpdemo

**Display Name**

OCP DEMO

**Description**

Application Demo for Openshift 4

Cancel Create

Luego dentro del proyecto creado crear un **Postgresql**.

Desde [Topology](#) seleccionar Database y luego [PostgresQL](#)

Completar los siguientes campos:

Database Service Name: **postgresql**

PostgreSQL Connection Username: **postgres**

PostgreSQL Connection Password: **redhat12345**

PostgreSQL Database Nme: **postgres**

Los valores de estos parámetros pueden ser distintos ya que son configurados en la aplicación mediante variables de entorno.

Ahora podemos comenzar a crear los pods de la aplicación.

Primero creamos el pod de la aplicación que genera los registros.

Para esto vamos a [Topology](#) , botón derecho y [Add to Project](#) y luego [From Git](#).

La URL al repo público es: [https://github.com/hconosciuto/app\\_ocp1](https://github.com/hconosciuto/app_ocp1)

Seleccionamos la imagen de [Python](#).

En [Name](#) ponemos: **app-ocp-generaregistros**

Y hacemos click en el botón [Create](#).

Hacemos el mismo paso pero para crear el pod que lee y muestra los registros utilizando la misma URL y en [Name](#): **app-ocp-leeregistros**

Luego debemos setear las variables de entorno.

Desde la vista [Administrator](#),

Vamos a [Workloads](#)

Luego [Deployments](#)

Hacemos click en **app-ocp-generaregistros**

Luego en [Environment](#) y creamos las variables con los valores usados en la creación del postgres, por ejemplo:

```










```

```



(Nombre de la tabla que será creada y almacenará los registros)

```

```




(Tiempo en segundos entre cada creación de registro)

```















**APP\_FILE = Program/genera\_registros.py**

Debajo se observa cómo deberían quedar las variables creadas:

Project: ocpdemo ▾

Container:  app-ocp-generaregistros ▾

Single values (env) ⓘ

NAME	VALUE	
 hac_PDATABASE	postgres	
 hac_PHOST	postgresql	
 hac_PPASSWORD	redhat12345	
 hac_PUSERNAME	postgres	
 hac_TIEMPO	20	
 hac_PTABLA	ocpdemo1	
 APP_FILE	Program/genera_registros.py	

Ambas aplicaciones generan salida por si se genera un error poder ver que está sucediendo.

Si está todo OK la creación de la tabla como la generación de los nombres debería comenzar:

8 lines

```
----> Running application from Python script (Program/genera_registros.py) ...  
Database connected!.  
CREATE TABLE ocpdemo1 (fecha TIMESTAMP without time zone NULL, nombre varchar(30));  
None  
Tabla ocpdemo1 creada!  
Paulo Pereyra  
Martin Pereyra
```

Por último deben configurarse las variables en la aplicación de lectura, en el Deploy:**app-ocp-leeregistros**

```


hac_PDATABASE=postgres
hac_PHOST=postgresql
hac_PPASSWORD=redhat12345
hac_PUSERNAME=postgres
hac_PTABLA=ocpdemo1
APP_FILE = Program/lee_registros.py

```













Debajo se observa cómo deberían quedar las variables creadas:

Project: ocpdemo ▾

---

Container:  app-ocp-leeregistros ▾

Single values (env) ⓘ

NAME	VALUE
 hac_PDATABASE	postgres 
 hac_PHOST	postgresql 
 hac_PPASSWORD	redhat12345 
 hac_PUSERNAME	postgres 
 hac_PTABLA	ocpdemo1 
 APP_FILE	Program/lee_registros.py 

[+ Add More](#) [+ Add from Config Map or Secret](#)

Una vez configurados los pods deberían funcionar sin error.

Desde [Topology](#), se puede ingresar a la opción [Open URL](#) del pod que lee los registros y ver cómo se generan los nombres con el timestamp.

La visualización debería ser similar a la imagen de abajo.

