KubeVela: Plataforma de entrega de aplicaciones web

Proyecto Integrado

2º Grado Superior en Administración de Sistemas Informáticos en Red

Mayo 2024

IES Gonzalo Nazareno

By: Mario Zayas García

1. Descripción del proyecto

Se quiere realizar la introducción a una nueva tecnología la cual es KubeVela, el objetivo a conseguir es tener una plataforma de entrega de software interactiva que hace la implementación y operación de aplicaciones. En este proyecto se realizará una introducción básica a dicha tecnología.

Teniendo en cuenta las ventajas/desventajas y funcionalidades de está nueva tecnología, se ha elegido este proyecto ya que parece una buena forma de desarrollar e implementar aplicaciones desde un entorno gráfico de manera sencilla.

Con Kubernetes se pueden crear clústers para el uso de KubeVela. Y gracias a KubeVela puedes desarrollar e implementar la aplicación de manera gráfica y simple.

1.1 Resultados que se esperan obtener

Al realizar este proyecto se espera que tengamos una pequeña base sobre KubeVela.

El objetivo principal sería tener instalado KubeVela y desplegar nuestra primera aplicación, esta instalación se realizará usando VelaD que es el instalador de KubeVela sin uso de un cluster. También aprenderemos cómo instalar KubeVela en un cluster de Kubernetes.

1. Desplegar la primera aplicación (My First App). Se espera tener una página web funcionando en la que salga una imagen ASCII de KubeVela y en ella ponga “Hello KubeVela!”
2. Desplegar la segunda aplicación. Se espera ver el funcionamiento de despliegue mediante la UI de KubeVela y observar como funciona.

2. Fundamentos

2.1 Introducción a Kubernetes

Kubernetes es una plataforma portable y extensible de código abierto para administrar cargas de trabajo y servicios. Kubernetes facilita la automatización y la configuración declarativa. Tiene un ecosistema grande y en rápido crecimiento. El soporte, las herramientas y los servicios para Kubernetes están ampliamente disponibles.

Kubernetes tiene varias características.

- Plataforma de contenedores
- Plataforma de microservicios
- Plataforma portable de nube

¿Por qué Kubernetes?

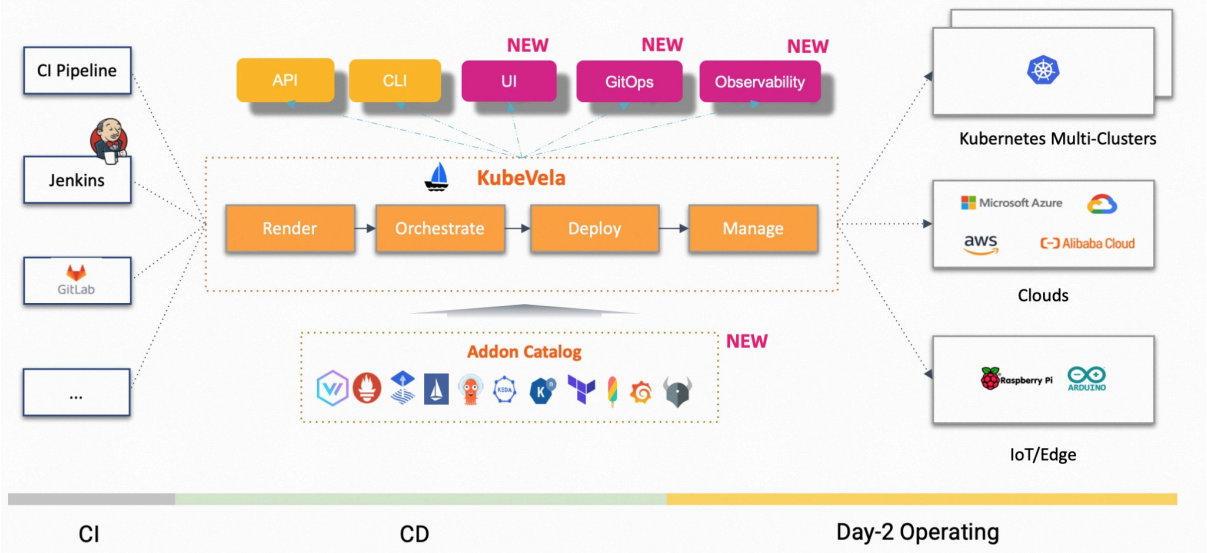
- Ágil creación y despliegue de aplicaciones.

- Desarrollo, integración y despliegue continuo.

- Observabilidad

2.2 Introducción a KubeVela

KubeVela es un moderno plano de control de gestión y entrega de software. El objetivo es hacer que la implementación y operación de aplicaciones en entornos híbridos y multinubes actuales sean más fáciles, rápidas y confiables.



- La aplicación moderna debería poder implementarse en entornos híbridos, incluyendo Kubernetes o la nube.

- Tiene un plano de control de entrega de la aplicación que se adapta a cualquier infraestructura.

- Implementación como código, se puede ejecutar en cualquier sistema CI/CD o GitOps. El flujo de trabajo de KubeVela funciona con Open Application Model (OAM).

- Arquitectura ligera pero altamente extensible, la implementación del plano de control puede estar en un solo pod .

2.2.1 KubeVela frente a otros

2.2.2 KubeVela frente a CI/CD (GitHub Actions, GitLab, CircleCI, Jenkins, etc.)

KubeVela es una plataforma de entrega continua que funciona en el sentido descendente de su proceso de CI. Reutilizando el proceso de CI que ya adoptó y KubeVela se hará cargo del proceso de CD dotándolo de las mejores prácticas modernas de entrega de aplicaciones. Puede ser compatible de forma nativa con GitOps si lo desea.

2.2.3 KubeVela frente a GitOps (ArgoCD, FluxCD, etc.)

KubeVela adopta su proceso de GitOps y lo mejora agregando capacidades de nube híbrida/clúster múltiple:

- KubeVela tiene un flujo trabajo fácil de usar que le permite ampliar, reprogramar o compartir cualquiera de sus procesos de entrega, incluidos los flujos de seguridad y cumplimiento.

- KubeVela considera la entrega de aplicaciones en entornos híbridos/nube múltiple como un ciudadano de primera clase, proporciona estrategias de implementación enriquecidas en clústers y nubes con provisión de entornos de nube totalmente administrados.

2.2.4 KubeVela frente a PaaS (Heroku, Cloud Foundry, etc.)

KubeVela comparte objetivos con la PaaS tradicional y apunta a mejorar la experiencia y eficiencia de los desarrolladores.

La mayor diferencia está en la flexibilidad.
KubeVela es totalmente programable y se pueden ampliar o eliminar in situ cuando cambien sus necesidades. Como plano de control CD, KubeVela le permite tomar el control total de su infraestructura y herramientas.

2.2.5 KubeVela frente a Helm

Para empezar ¿Qué es Helm?, bueno, Helm es un administrador de paquetes para Kubernetes que proporciona paquete, instalación y actualización de un conjunto de archivos YAML para Kubernetes como una unidad.

KubeVela, como plano de control de entrega de software moderno, puede implementar de forma natural gráficos Helm. Pudiendo utilizar KubeVela para definir una aplicación compuesta por un gráfico de WordPress y un módulo AWS, orquestar la topología de los componentes e implementarlos en distintos entornos.

También admite otros formatos como por ejemplo Kustomize.

2.2.6 Open Application Model [OAM]

Ofrece un diseño modular, extensible y portátil para modelar la implementación de aplicaciones con una API de mayor nivel pero consistente.

Esta es la clave para permitir una entrega de aplicaciones simples pero sólidas en entornos híbridos, incluyendo Kubernetes, nube o IoT.

2.3 VelaD

VelaD es una herramienta de **implementación liviana** para configurar **KubeVela**. Esta facilita la configuración del entorno KubeVela, incluido un clúster con **KubeVela** instalado y **VelaUX/Vela CLI** preparado. **VelaD** es la forma más rápida de comenzar con **KubeVela**, por ello será la primera en verse cómo se realiza su instalación.

2.4 VelaUX

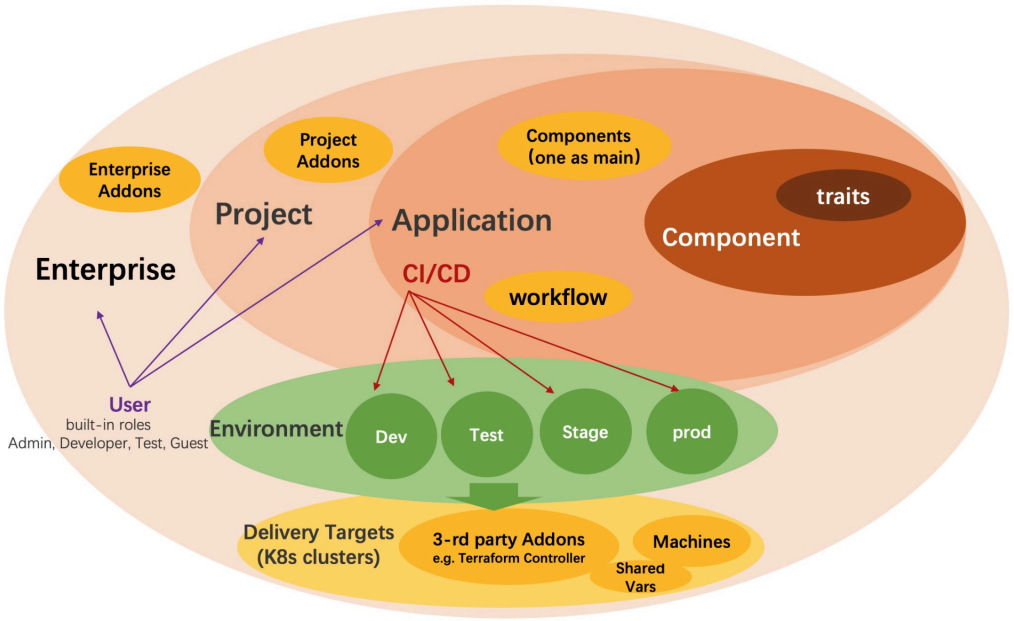
VelaUX es la herramienta que proporciona la **consola UI** de **KubeVela**, desde ella puedes ver todas las aplicaciones que tengas, también puedes configurarlas hacerles deploy y muchas más cosas.

Para entrar a la UI primero debes activar el addon que viene con VelaD, una vez activado deberás registrarte y configurar la contraseña y usuario.

Para poder visitar el panel de VelaUX por puerto local, debes poner el siguiente comando.

```
vela port-forward addon-velaux -n vela-system
```

El concepto de VelaUX es ser un complemento más de KubeVela pero que funcione como interfaz gráfica para el usuario que tiene KubeVela.



El Proyecto es donde se administran las aplicaciones y colaboras con los miembros del equipo.
El entorno es el entorno de desarrollo, prueba y producción y puede incluir múltiples objetivos de entrega.

Una aplicación en VelaUX es un poco diferente con KubeVela, agregamos que el ciclo de vida incluye:

- **Crear** una aplicación es simplemente crear registros de metadatos, no se ejecutará en un clúster real.
- **La implementación** de una aplicación se vinculó con el entorno especificado y creará una instancia del recurso de la aplicación en clústeres de Kubernetes.
- **Reciclar** una aplicación eliminará la instancia de la aplicación y recuperará sus recursos de los clústeres de Kubernetes.
- **Eliminar** una aplicación es en realidad eliminar los metadatos.

El concepto de resto en la aplicación VelaUX está alineado con KubeVela Core.

3. Instalación de KubeVela

3.1 Instalación Independiente de KubeVela Standalone en local

3.1.1 Preparación

Requisitos:

- Si usas Linux o MacOS debes tener instalado `curl`
- Si usas MacOS o Windows debes tener instalado docker
- Comprueba que el daemon de docker esta en funcionamiento

3.1.2 Instalar VelaD

Instalamos VelaD utilizando un `curl`.

```
sudo curl -fsSl https://kubevela.io/script/install-velad.sh | bash
```

```
mario@Mario:~$ sudo curl -fsSl https://kubevela.io/script/install-velad.sh | bash
Your system is linux_amd64
Installing VelaD ...

Getting the latest VelaD...
Downloading https://github.com/kubevela/velad/releases/download/v1.9.5/velad-v1.9.5-linux-amd64.tar.gz .
..
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
  0     0    0     0    0     0      0      0 --:--:-- --:--:-- --:--:--     0
100  414M  100  414M    0     0  32.3M      0  0:00:12  0:00:12 --:--:-- 22.1M
VelaD installed into /usr/local/bin/velad successfully.

Core Version: v1.9.5
VelaD Version: v1.9.5

For more information on how to started, please visit:
  https://kubevela.io
```

Nos aseguramos de que esté instalado, para ello vamos a `/usr/local/bin/`

```
mario@Mario:~$ ls /usr/local/bin/
asar  coraenv  dbhome  firebase  kubectl  oraenv  velad
mario@Mario:~$
```

Configuramos KubeVela, para ello usamos el comando:

`velad install`

```
mario@Mario:~$ sudo velad install
Preparing cluster setup script...
Saving temporary file: k3s-setup-*.sh
Preparing k3s binary...
Saving k3s binary to /usr/local/bin/k3s
Successfully place k3s binary to /usr/local/bin/k3s
Preparing k3s images
Making directory /var/lib/rancher/k3s/agent/images/
Saving K3s air-gap install images to /var/lib/rancher/k3s/agent/images/k3s-airgap-images.tar.gz
Successfully prepare k3s image
Setting up cluster
/bin/bash /root/.vela/tmp/k3s-setup-1546717250.sh --node-name=default
[INFO]  Skipping k3s download and verify
[INFO]  Skipping installation of SELinux RPM
[INFO]  Skipping /usr/local/bin/kubectl symlink to k3s, already exists
[INFO]  Creating /usr/local/bin/crictl symlink to k3s
[INFO]  Skipping /usr/local/bin/ctr symlink to k3s, command exists in PATH at /usr/bin/ctr
[INFO]  Creating killall script /usr/local/bin/k3s-killall.sh
[INFO]  Creating uninstall script /usr/local/bin/k3s-uninstall.sh
[INFO]  env: Creating environment file /etc/systemd/system/k3s.service.env
[INFO]  systemd: Creating service file /etc/systemd/system/k3s.service
[INFO]  systemd: Enabling k3s unit
Created symlink /etc/systemd/system/multi-user.target.wants/k3s.service → /etc/systemd/system/k3s.servic
e.
[INFO]  systemd: Starting k3s
Successfully setup cluster
Checking and installing vela CLI...
vela CLI is not installed, installing...
Installing vela CLI at:  /usr/local/bin/vela
Successfully install vela CLI
Saving and temporary image file: vela-image-cluster-gateway-*.tar
Importing image to cluster using temporary file: vela-image-cluster-gateway-*.tar
unpacking docker.io/oamdev/cluster-gateway:v1.9.0-alpha.2 (sha256:a9baa41c62762dea9cccefeed0ef1a479b695f
7af847ecf7df0ab6b6ecf4e8c5)...done
Successfully import image /root/.vela/tmp/vela-image-cluster-gateway-3850723169.tar
Saving and temporary image file: vela-image-kube-webhook-certgen-*.tar
Importing image to cluster using temporary file: vela-image-kube-webhook-certgen-*.tar
unpacking docker.io/oamdev/kube-webhook-certgen:v2.4.1 (sha256:089374ef23e1d268f138d742b8af056ee1af51bb4
863f6a19cde0634068eaa91)...done
Successfully import image /root/.vela/tmp/vela-image-kube-webhook-certgen-1331836976.tar
Saving and temporary image file: vela-image-vela-core-*.tar
Importing image to cluster using temporary file: vela-image-vela-core-*.tar
unpacking docker.io/oamdev/vela-core:v1.9.5 (sha256:fbff76625847c1183dd371f5fc9acbbd3c33c348faf12984eaae
433b1244f454)...done
Successfully import image /root/.vela/tmp/vela-image-vela-core-319828341.tar
```

Como podemos ver en la siguiente captura, ha sido un éxito.

```
Keep the token below if you want to restart the control plane
K102ee35087d29713d2998ebe70baeb8a5c51a04bb8b274133aaaf2e32655c62c43::server:400e662b7184824e743fed6b2ac7
67f1

🚀 Successfully install KubeVela control plane
💻 When using gateway trait, you can access with 127.0.0.1
🔭 See available commands with `vela help`
💡 To enable dashboard, run `vela addon enable /root/.vela/addons/velaux`
🔑 To access the cluster, set KUBECONFIG:
    export KUBECONFIG=$(velad kubeconfig --name default --host)
mario@Mario:~$
```

Verificamos que la instalación se haya realizado bien, para ello primero deberemos exportar la configuración de **VelaD**, y una vez exportada,

procederemos a usar el siguiente comando que te permite ver los componentes de **VelaD** instalados:

```
export KUBECONFIG=$(velad kubeconfig –name default –host)
vela comp
```

```
mario@Mario:~$ export KUBECONFIG=$(velad kubeconfig --name default --host)
mario@Mario:~$ vela comp
```

Como podemos observar, se ha realizado correctamente.

```
mario@Mario:~$ sudo vela comp
NAME            DEFINITION                  DESCRIPTION

k8s-objects     autodetects.core.oam.dev    K8s-objects allow users to specify raw K8s objects in
                                            properties

raw             autodetects.core.oam.dev    Raw allow users to specify raw K8s object in properties.
                                            This definition is DEPRECATED, please use 'k8s-objects'
                                            instead.

ref-objects     autodetects.core.oam.dev    Ref-objects allow users to specify ref objects to use.
                                            Notice that this component type have special handle logi
c.
task            jobs.batch                  Describes jobs that run code or a script to completion.

cron-task       autodetects.core.oam.dev    Describes cron jobs that run code or a script to complet
ion.
worker          deployments.apps            Describes long-running, scalable, containerized services
                                            that running at backend. They do NOT have network endpoi
nt
                                            to receive external network traffic.

webservice      deployments.apps            Describes long-running, scalable, containerized services
                                            that have a stable network endpoint to receive external
                                            network traffic from customers.

daemon          daemonsets.apps             Describes daemonset services in Kubernetes.

mario@Mario:~$
```

3.1.3 Instalación de VelaUX

Instalamos **VelaUX** con el comando:

```
sudo vela addon enable velaux
```

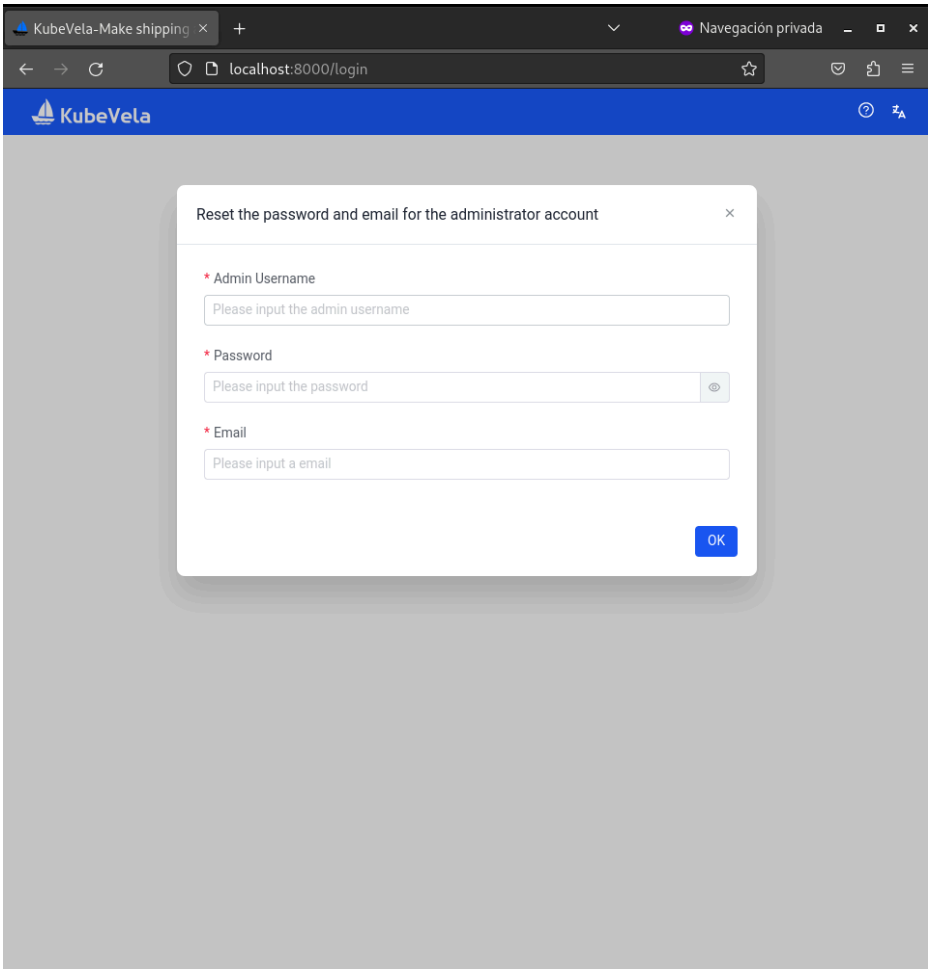
```
mario@Mario:~$ sudo vela addon enable velaux
Addon velaux enabled successfully.
Please access addon-velaux from the following endpoints:
+---------+---------------+-----------------------------------+--------------------------------+-------+
| CLUSTER |   COMPONENT   |     REF(KIND/NAMESPACE/NAME)      |            ENDPOINT            | INNER |
+---------+---------------+-----------------------------------+--------------------------------+-------+
| local   | velaux-server | Service/vela-system/velaux-server | velaux-server.vela-system:8000 | true  |
+---------+---------------+-----------------------------------+--------------------------------+-------+
    To open the dashboard directly by port-forward:

    vela port-forward -n vela-system addon-velaux 8000:8000

    Please refer to https://kubevela.io/docs/reference/addons/velaux for more VelaUX addon installation
and visiting method.
mario@Mario:~$
```

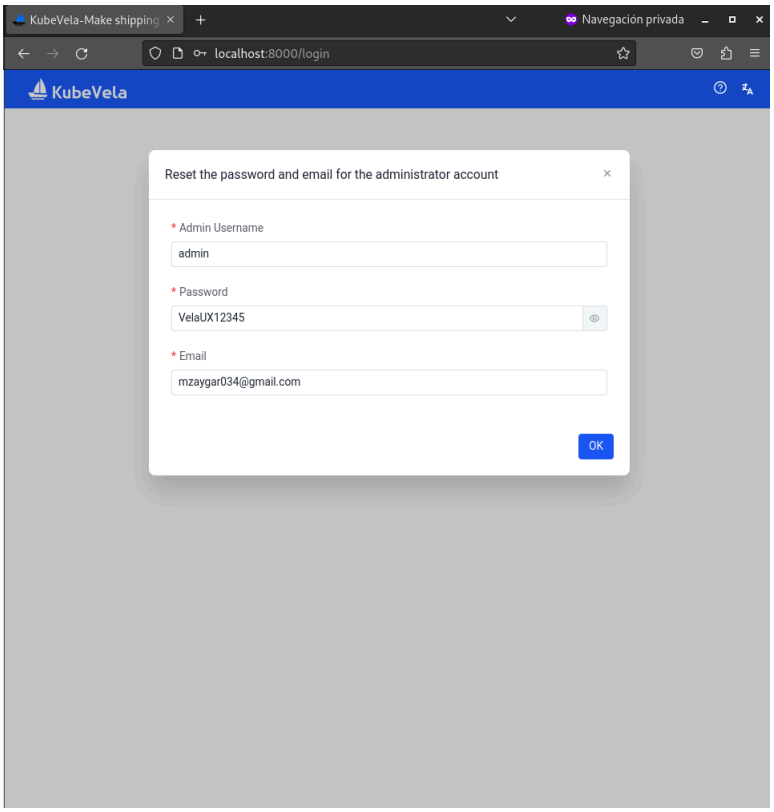
Nos conectamos para ver que realmente funcione.

`sudo vela port-forward -n vela-system addon-velaux 8080:8000`

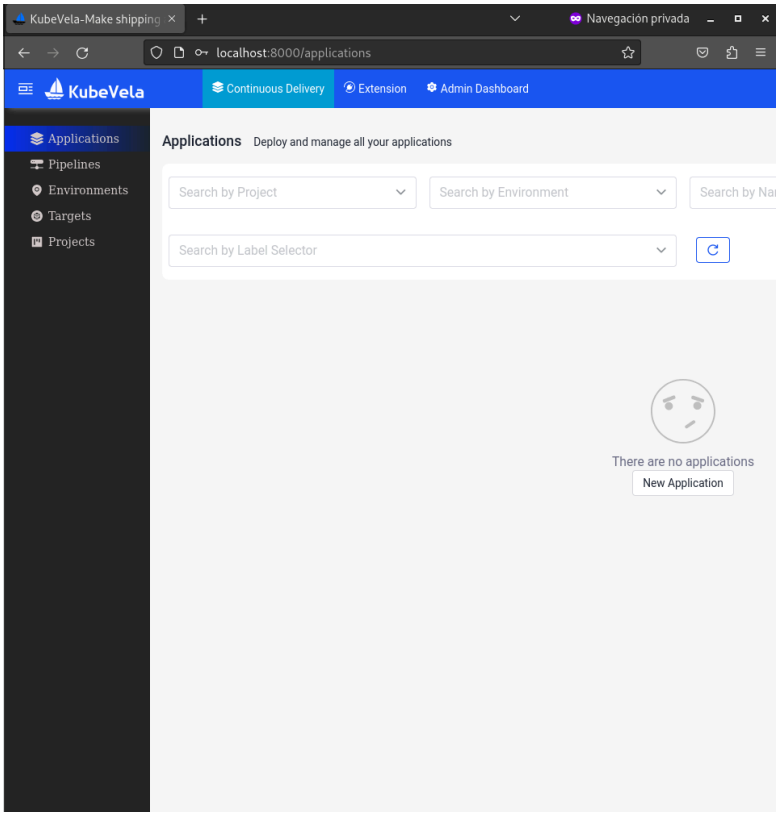
```
mario@Mario:~$ sudo vela port-forward -n vela-system addon-velaux 8000:8000
trying to connect the remote endpoint svc/velaux-server 8000:8000 ..Forwarding from 127.0.0.1:8000 -> 80
00
Forwarding from [::1]:8000 -> 8000

Forward successfully! Opening browser ...
Handling connection for 8000
```

Como podemos observar, hemos entrado en la interfaz gráfica de KubeVela, ahora nos registramos y veremos que hay dentro.


KubeVela-Make shipping
Navegación privada
localhost:8000/login
KubeVela
Reset the password and email for the administrator account
* Admin Username
Please input the admin username
* Password
Please input the password
* Email
Please input a email
OK


KubeVela-Make shipping
Navegación privada
localhost:8000/login
KubeVela
Reset the password and email for the administrator account
* Admin Username
admin
* Password
VelaUX12345
* Email
mzaygar034@gmail.com
OK

Una vez nos hayamos registrado con la contraseña y usuarios anteriores, podremos acceder a la interfaz gráfica.



En ella podemos observar que no hay aplicaciones en este momento.

3.1.4 Desinstalación

Para desinstalar KubeVela solo es necesario usar el siguiente comando:
`velad uninstall`

3.2 Instalación de KubeVela con Kubernetes.

3.2.1 Preparación

- Tener un cluster de Kubernetes versión 1.19 o mayor

- Master = 192.168.121.23
- Nodo1 = 192.168.121.133
- Nodo2 = 192.168.121.197

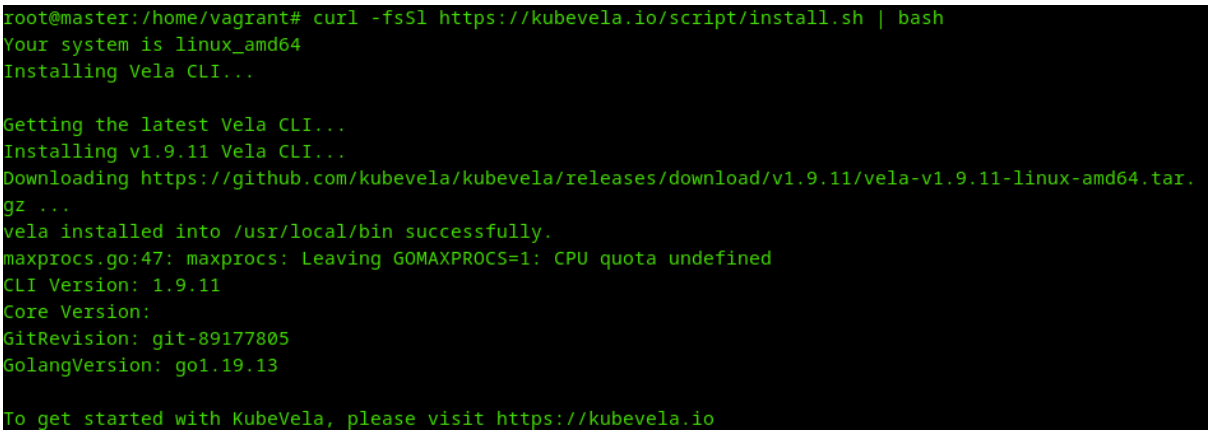
3.2.2 Instalar KubeVela CLI

Para instalarlo hay múltiples opciones.

- Script
- Homebrew
- Descarga directa de los releases
- Asdf-vm
- Docker

En mi caso he utilizado el script:

```
curl -fsSl https://kubevela.io/script/install.sh | bash
```



3.2.3 Instalar KubeVela Core

Hay dos maneras de instalar KubeVela Core

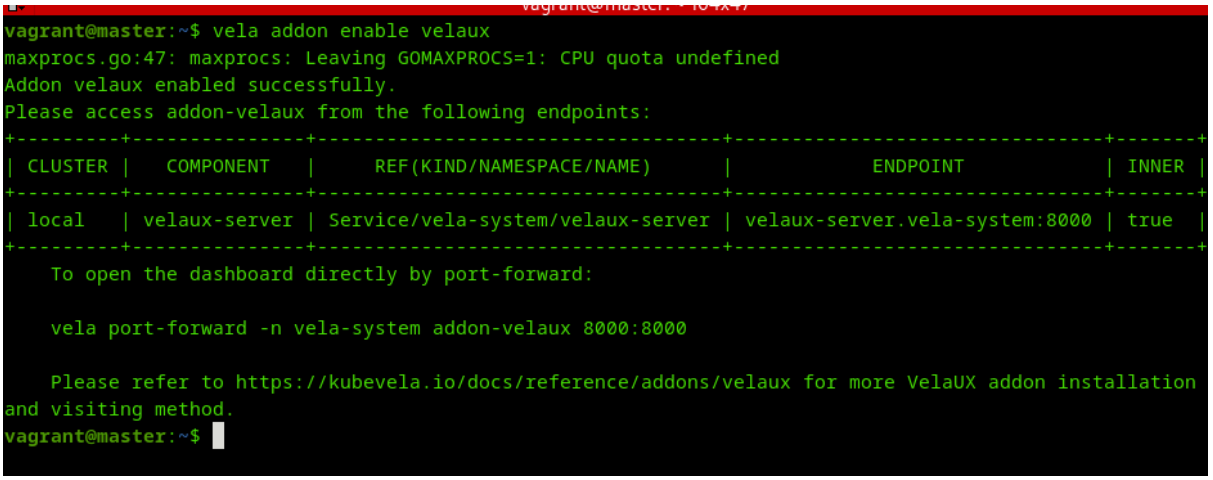
- Default
- Helm

En mi caso he tenido que hacerlo de otra manera debido a fallos en la documentación.

```
mkdir -p ~/.kube
sudo cp /etc/rancher/k3s/k3s.yaml ~/.kube/config
sudo chown $(id -u):$(id -g) ~/.kube/config
sudo kubectl cluster-info
export KUBECONFIG=~/.kube/config
vela install
```

3.2.4 Instalar VelaUX

```
vela addon enable velaux
```

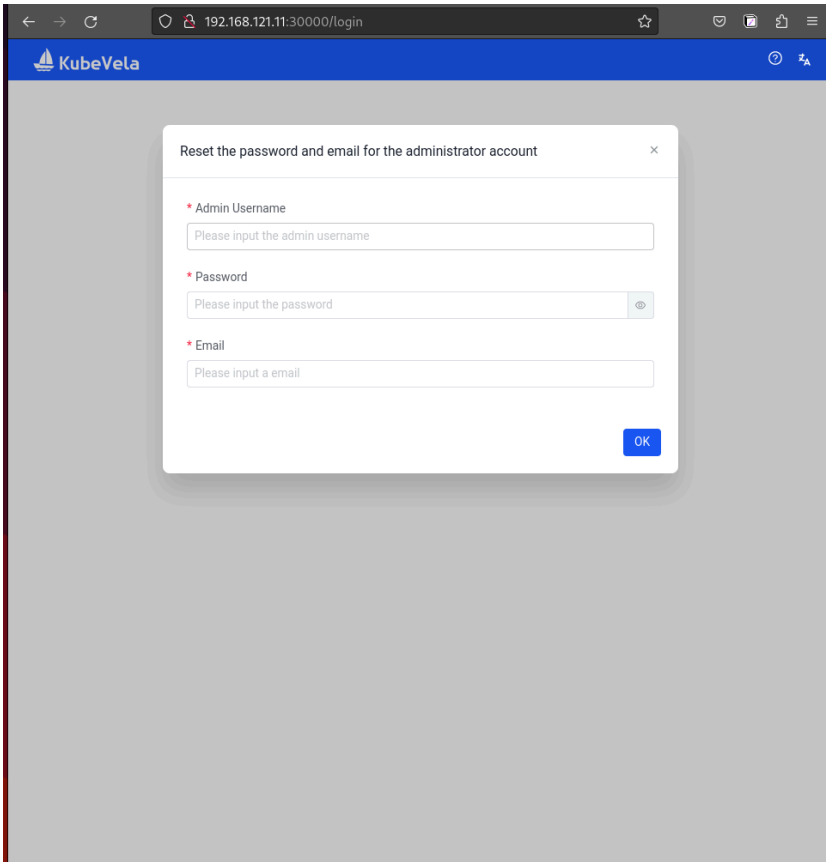


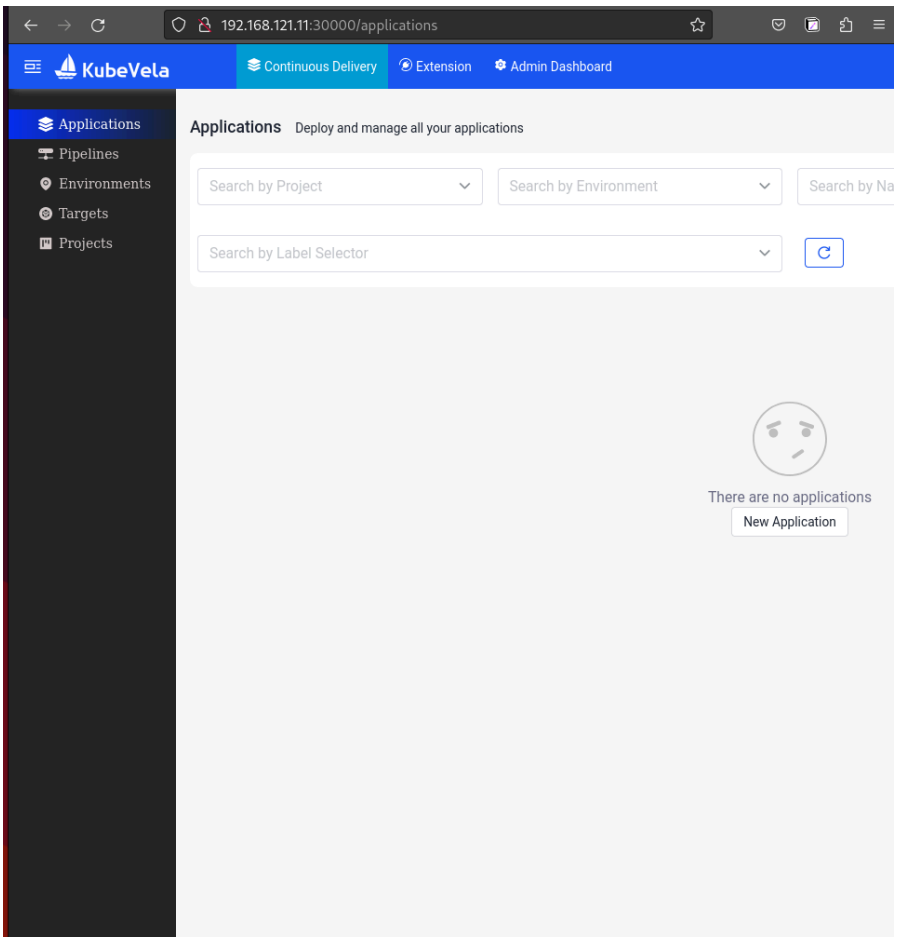
Ahora probaremos a acceder:

```
vela addon enable velaux serviceType=NodePort
```

```
vagrant@master:~$ vela addon enable velaux serviceType=NodePort
maxprocs.go:47: maxprocs: Leaving GOMAXPROCS=1: CPU quota undefined
Addon velaux enabled successfully.
Please access addon-velaux from the following endpoints:
+---------+---------------+-----------------------------------+----------------------+-------+
| CLUSTER |   COMPONENT   |     REF(KIND/NAMESPACE/NAME)      |       ENDPOINT       | INNER |
+---------+---------------+-----------------------------------+----------------------+-------+
| local   | velaux-server | Service/vela-system/velaux-server | 192.168.121.11:30000 | false |
+---------+---------------+-----------------------------------+----------------------+-------+
    To open the dashboard directly by port-forward:

    vela port-forward -n vela-system addon-velaux 8000:8000

    Please refer to https://kubevela.io/docs/reference/addons/velaux for more VelaUX addon installation
and visiting method.
vagrant@master:~$
```

Como podemos ver a continuación hemos podido acceder.

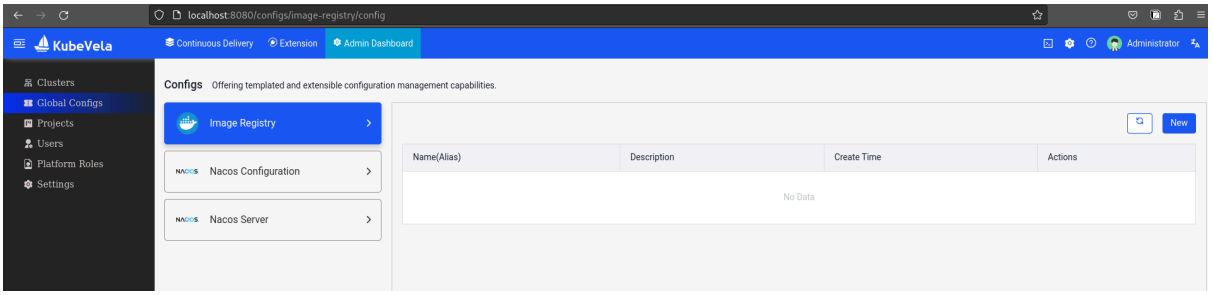




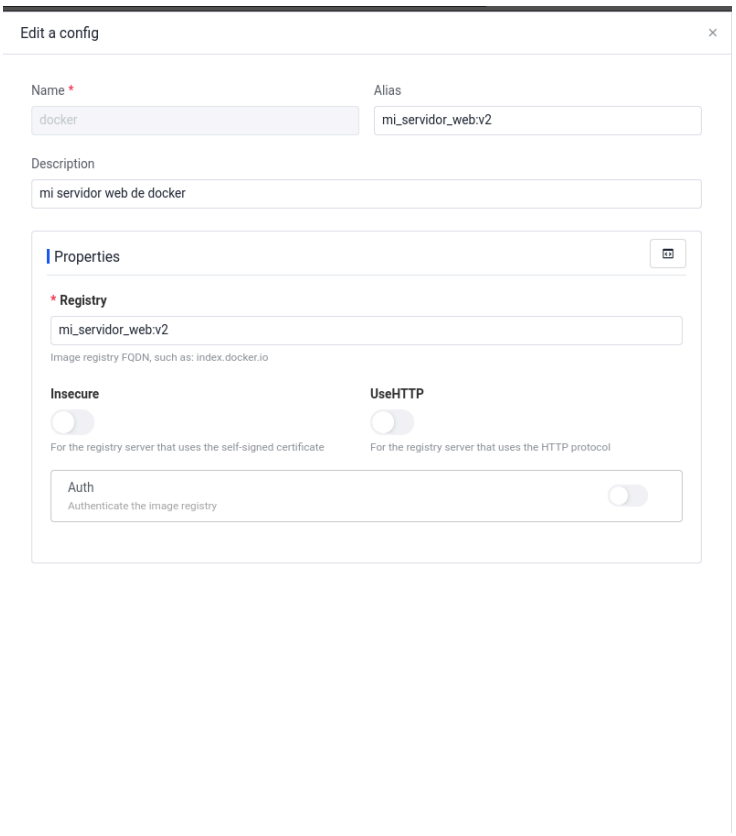
Y Aquí podríamos ver si tenemos alguna aplicación creada o que esté desplegada, así como los fallos que pueda tener.

3.3. Añadir una imagen al registro de imágenes de KubeVela

Para crearlo hay primero que ir a **Configs/ImageRegistry**



Una vez estemos aquí deberemos darle al botón de arriba a la derecha **“New”** y rellenaremos con los datos de nuestra imagen docker.



4. Despliegue de tu primera aplicación usando VelaD

Ahora que tenemos instalado KubeVela es el momento de desplegar nuestra primera aplicación. Lo primero que haremos será crear el .yaml con la aplicación a desplegar.

Una vez creado el .yaml crearemos una variable de entorno llamada prod

```
sudo vela env init prod --namespace prod
```

```
mario@Mario:~$ sudo vela env init prod --namespace prod
environment prod with namespace prod created
mario@Mario:~$
```

Ahora empezamos a desplegar la aplicación.

```
sudo vela up -f
https://kubevela.net/example/applications/first-app.yaml
```

```
mario@Mario:~$ sudo vela up -f https://kubevela.net/example/applications/first-app.yaml
Applying an application in vela K8s object format...
✅ App has been deployed 🚀🚀🚀
    Port forward: vela port-forward first-vela-app -n prod
             SSH: vela exec first-vela-app -n prod
         Logging: vela logs first-vela-app -n prod
      App status: vela status first-vela-app -n prod
        Endpoint: vela status first-vela-app -n prod --endpoint
Application prod/first-vela-app applied.
mario@Mario:~$
```

Comprobamos el estado del despliegue de la aplicación.

`sudo vela status first-vela-app`

```
mario@Mario:~$ sudo vela status first-vela-app
About:

  Name:         first-vela-app
  Namespace:    prod
  Created at:   2024-06-11 18:02:04 +0200 CEST
  Status:       workflowSuspending

Workflow:

  mode: StepByStep-DAG
  finished: false
  Suspend: true
  Terminated: false
  Steps
  - id: t95g7u8dmh
    name: deploy2default
    type: deploy
    phase: succeeded
  - id: i82kwqttkd
    name: manual-approval
    type: suspend
    phase: suspending
    message: Suspended by field suspend

Services:

  - Name: express-server
    Cluster: local  Namespace: default
    Type: webservice
    Healthy Ready:1/1
    Traits:
      ✅ scaler
mario@Mario:~$
```

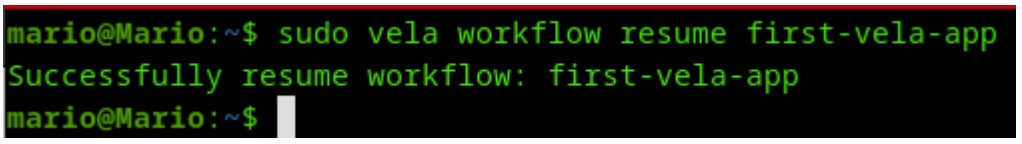
Como podemos ver el estado es **“workflowSuspending”** lo que significa que ha terminado el flujo de trabajo y está esperando una confirmación manual. Observaremos la aplicación y nos aseguraremos de que está funcionando y se ha creado la aplicación.

`sudo vela port-forward first-vela-app 8000:8000`

```
mario@Mario:~$ sudo vela port-forward first-vela-app 8000:8000
trying to connect the remote endpoint svc/express-server 8000:8000 ..Forwarding from 127.0.0.1:8000 -
000
Forwarding from [::1]:8000 -> 8000

Forward successfully! Opening browser ...
Handling connection for 8000
```

Ahora que sabemos que la aplicación está bien podemos confirmar el flujo de trabajo para continuar.

```
sudo vela workflow resume first-vela-app
```



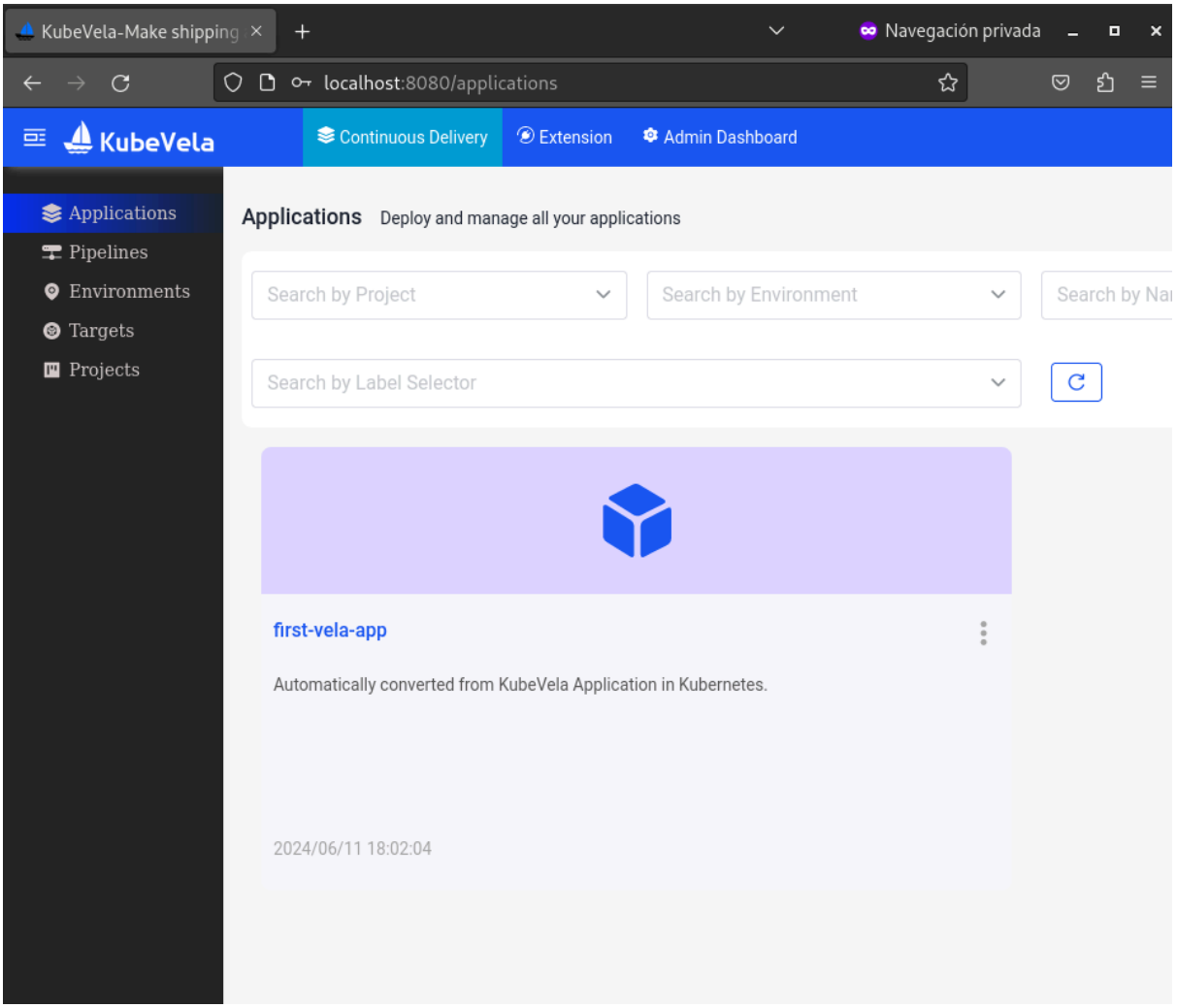
Si vemos el estado de la aplicación podremos observar que está en running

```
mario@Mario:~$ sudo vela status first-vela-app
About:

  Name:         first-vela-app
  Namespace:    prod
  Created at:   2024-06-11 18:02:04 +0200 CEST
  Status:       running

Workflow:

  mode: StepByStep-DAG
  finished: true
  Suspend: false
  Terminated: false
  Steps
  - id: t95g7u8dmh
    name: deploy2default
    type: deploy
    phase: succeeded
  - id: i82kwqttkd
    name: manual-approval
    type: suspend
    phase: succeeded
  - id: hvg772fr6g
    name: deploy2prod
    type: deploy
    phase: succeeded

Services:

  - Name: express-server
    Cluster: local  Namespace: prod
    Type: webservice
    Healthy Ready:2/2
    Traits:
      ✅ scaler
  - Name: express-server
    Cluster: local  Namespace: default
    Type: webservice
    Healthy Ready:1/1
    Traits:
      ✅ scaler
mario@Mario:~$
```

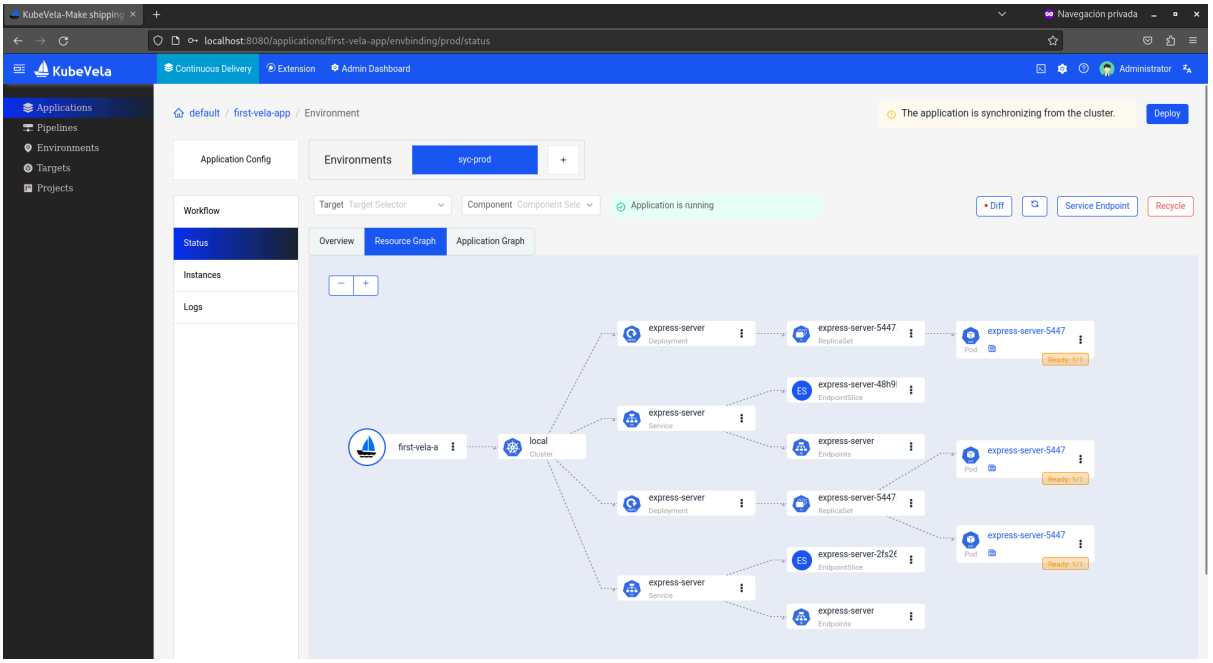
Y con esto hemos terminado de desplegar la primera aplicación, pero, también se puede ver desde la UI de VelaUX, por lo que vamos a ver si eso es posible.

```
sudo vela port-forward addon-velaux -n vela-system 8080:8000
```

```
mario@Mario:~$ sudo vela port-forward addon-velaux -n vela-system 8080:8000
trying to connect the remote endpoint svc/velaux-server 8080:8000 ..Forwarding from 127.0.0.1:8080 -> 80
00
Forwarding from [::1]:8080 -> 8000

Forward successfully! Opening browser ...
Handling connection for 8080
```

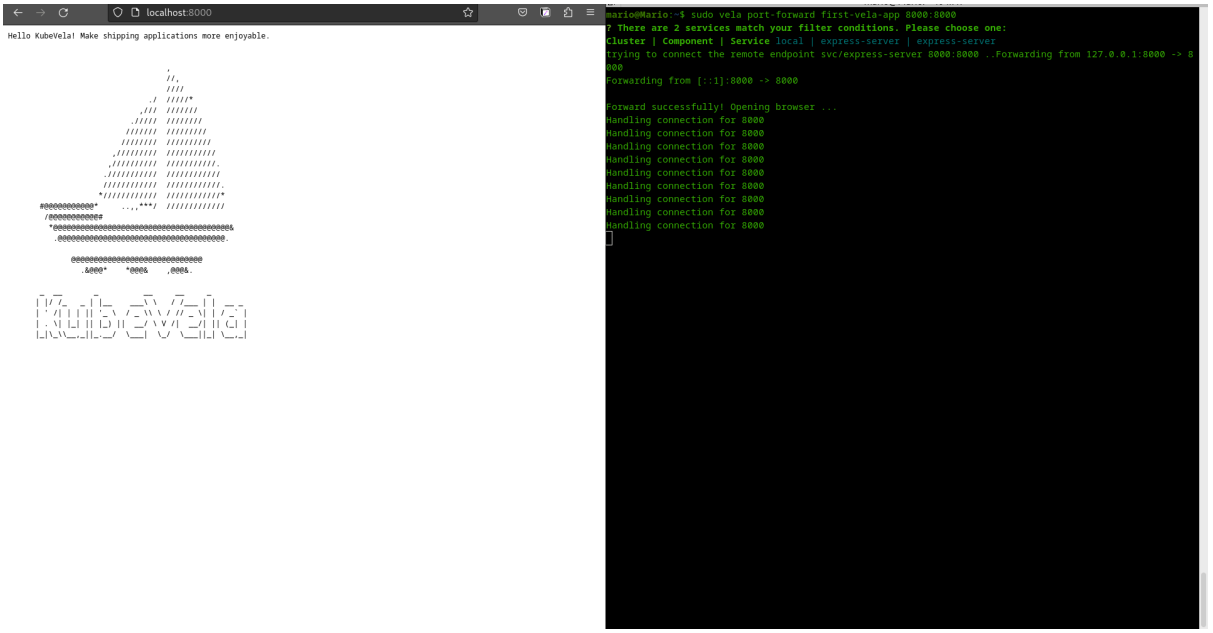


Al entrar veremos que se ha creado en **“Applications”** la aplicación que hemos desplegado, también te dice la hora a la que se ha creado.



Ahora una vez intentemos hacer el **port-forward** para poder acceder a la aplicación web, nos dará a elegir entre 2 opciones que serán los dos servicios que hemos creado, el default y el prod.

```
sudo vela port-forward first-vela-app 8000:8000
```



Para eliminar la aplicación solo hará falta poner el siguiente comando:

```
vela delete first-vela-app
```

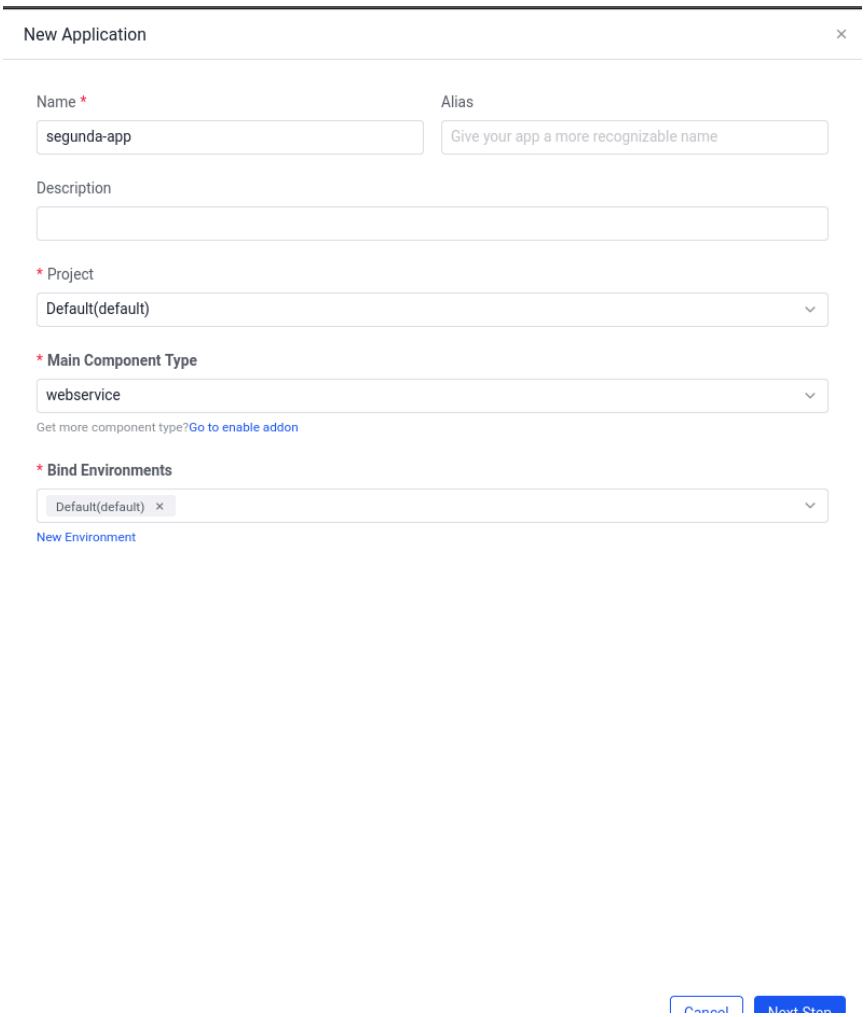
5. Despliegue de tu primera aplicación usando VelaUX

Lo primero que necesitaremos es tener habilitado VelaUX.
Y una imagen docker en nuestro registro local para que KubeVela pueda utilizarla (este paso no es necesario)

Una vez tengamos esto, procederemos a crear la aplicación. Al entrar en la UI veremos la pantalla principal, en ella podremos apreciar una parte llamada **“Applications”** en la cual estaremos por default.

Una vez estemos aquí iremos a la esquina derecha de la pantalla al botón de **“New Application”** y le daremos click.

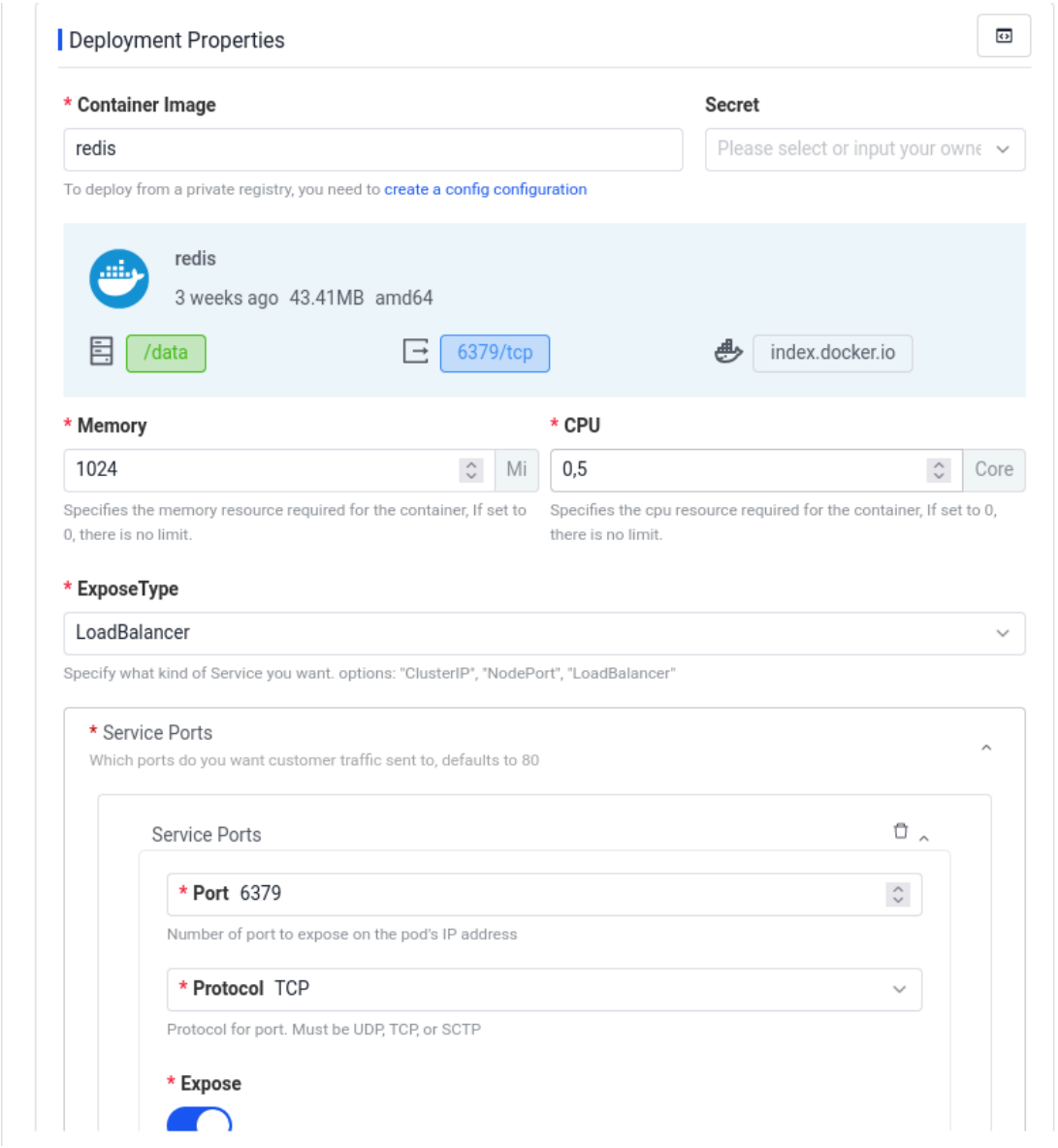
Una vez le demos click saldrá esta pantalla que rellenaremos como queramos, en mi caso he puesto el nombre que he querido y he puesto el Main componente como webservice porque es lo que voy a hacer.



Cuando le damos a **"Next Step"** nos saldrá la siguiente interfaz, donde podremos poner el nombre de la imagen del contenedor que usaremos, en mi caso he utilizado la redis.

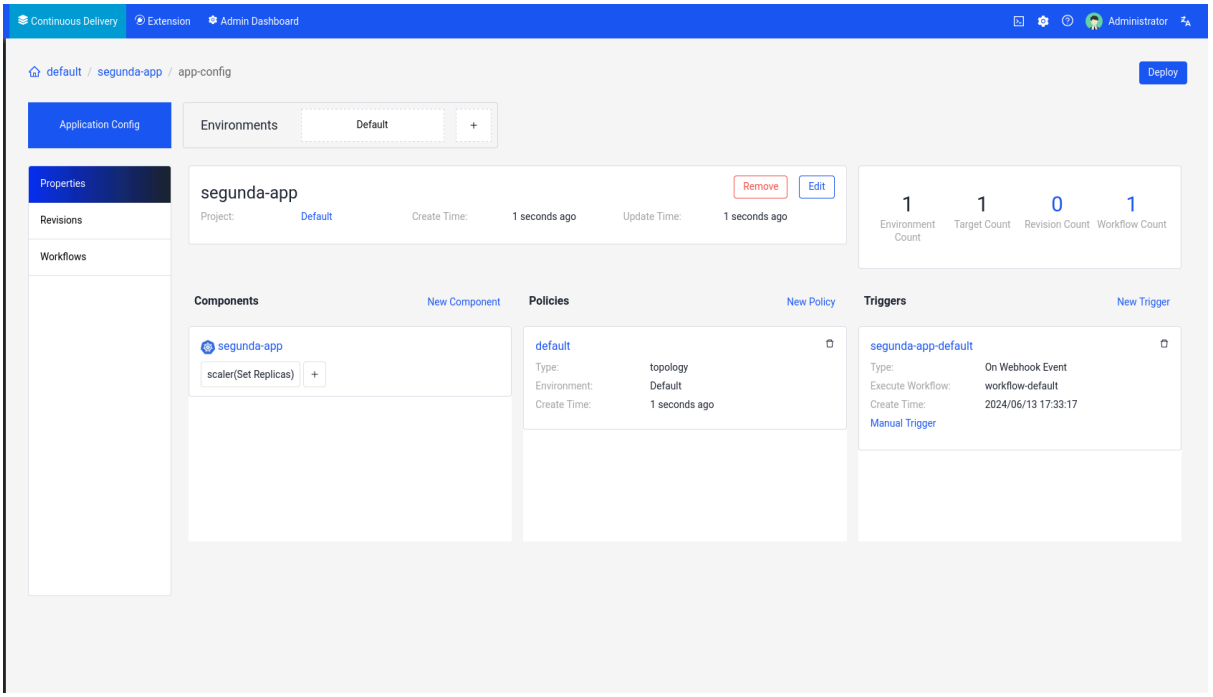
Como se puede apreciar también puedes escoger la cantidad de CPU y de memoria que tendrá el contenedor así como el puerto el protocolo y si quieres que esté expuesto.

Además hay una pestaña llamada **"Advanced settings"** que te permite usar más contenido.

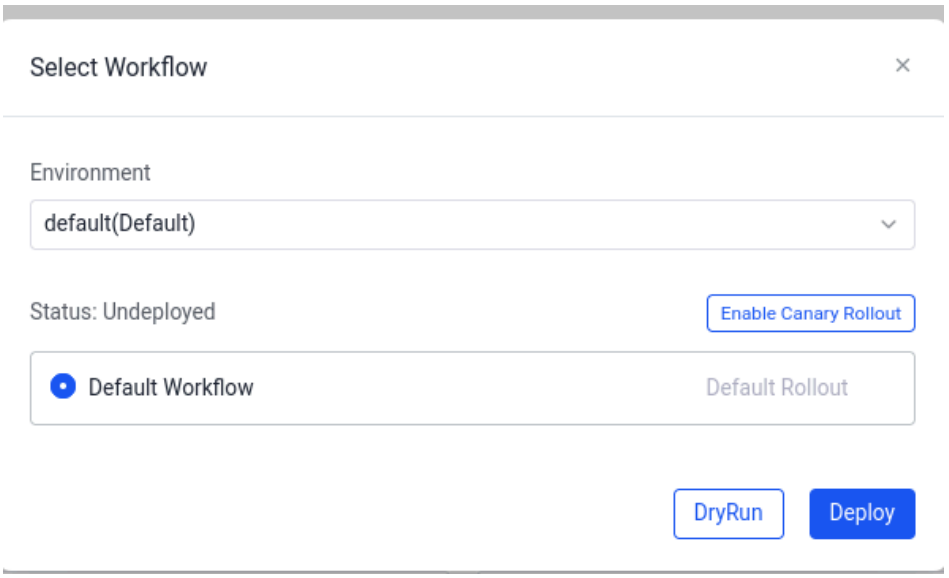


Una vez terminado el paso de arriba, tendremos la siguiente interfaz, la cual te indica lo que se va a crear, los triggers que tiene, sus políticas, el componente...

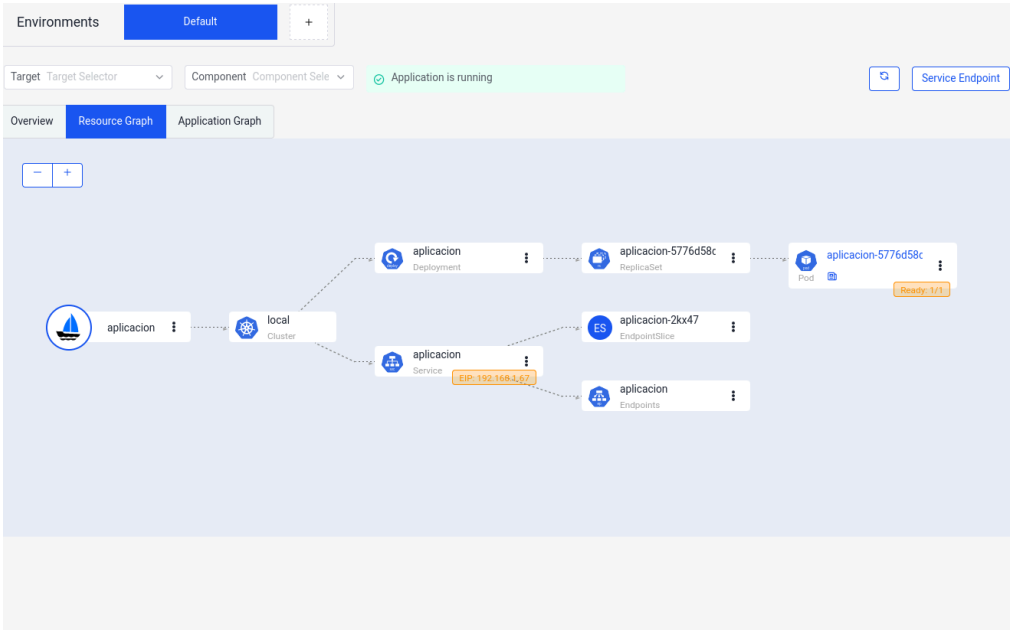
Puedes añadir más si quieres, en mi caso, he utilizado los que vienen al crearse. Ahora procederemos a desplegar la aplicación. Para ello habrá que pulsar el botón de arriba a la derecha que pone **“Deploy”**.



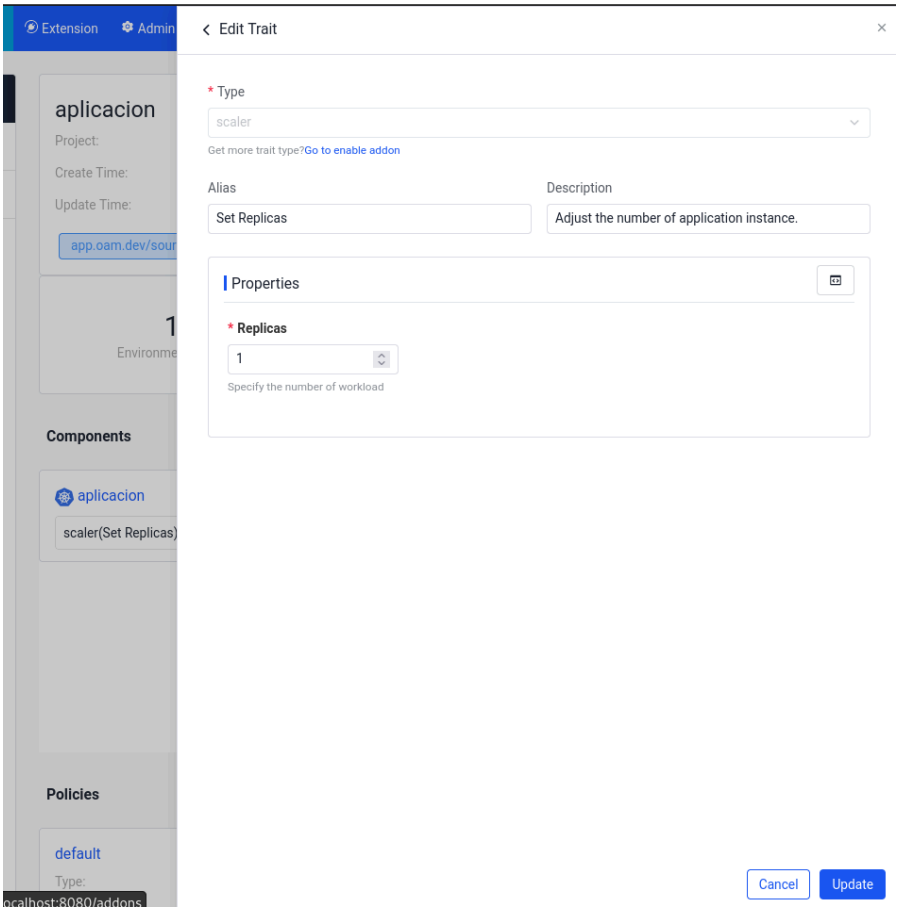
Una vez pulsado dicho botón nos saldrá esta interfaz, la cual podemos dejar como está y darle a **"Deploy"**.



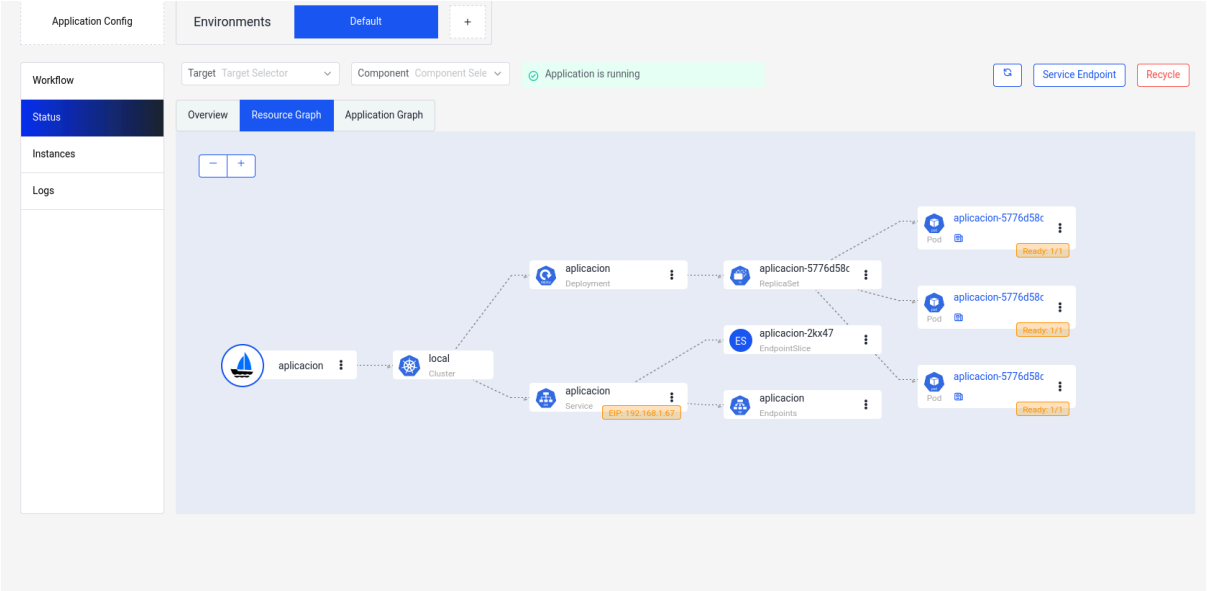
Una vez le hemos dado a deploy esperamos a que esta se despliegue, **habrá un workflow** creado que te indicará si se ha creado con éxito o si por el contrario a fallado, también puedes verlo de otra manera en forma de gráfico para que se entienda mejor como dejo a continuación.



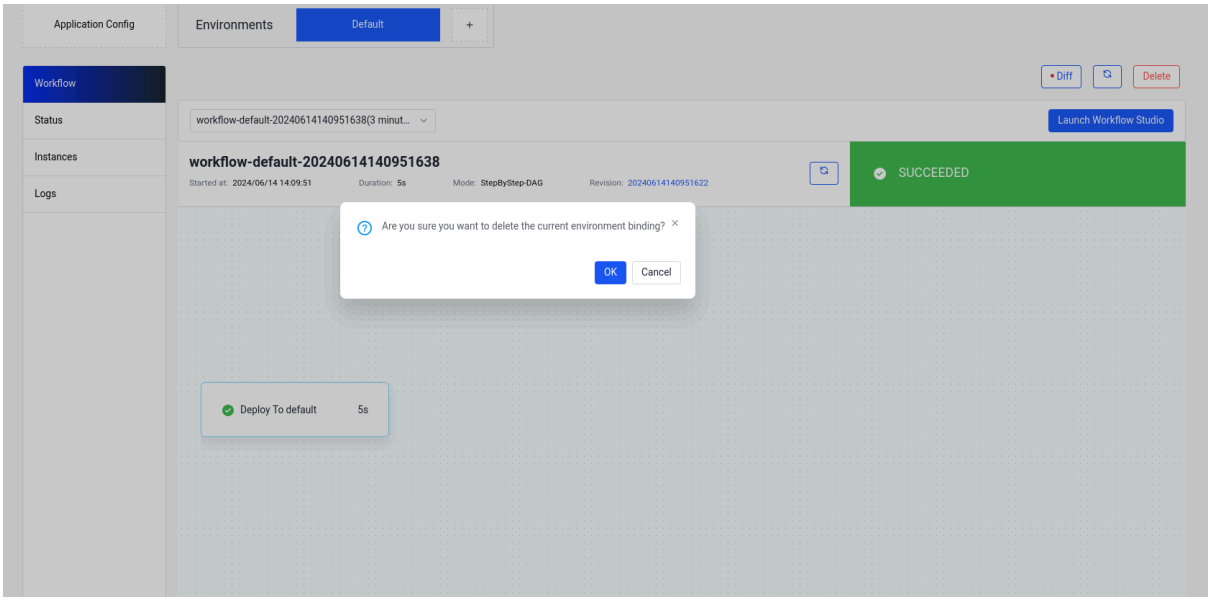
Podemos añadirle mas replicas si queremos



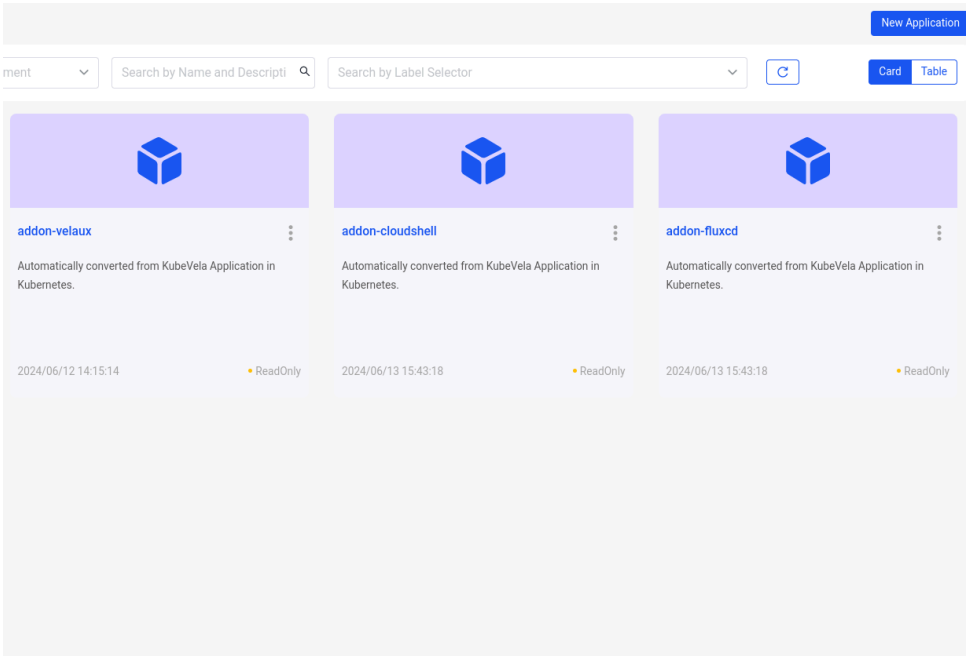
Una vez puestas las réplicas que queremos le damos a **“Update”** y le daremos a **“Deploy”** de nuevo para poder mejorar la aplicación.



Como podemos observar antes había una y ahora tenemos tres. En caso de que necesitemos eliminar la aplicación deberemos darle primero a **“Recycle”** y luego a **“Delete”** y estará completamente eliminado.



Y cuando vayamos al apartado de aplicaciones no estará la que hemos borrado.



6. Conclusiones y propuestas

Con la realización del proyecto hemos podido comprender mejor la infraestructura de **KubeVela,** así como el funcionamiento de está referente a los distintos softwares como por ejemplo **Helm**, **Jenkins**...

Al ser una herramienta en desarrollo hay todavía muchas funciones por pulir y muchas cosas que no se han llegado a realizar en el proyecto. También hemos profundizado en el funcionamiento de **KubeVela** y como utiliza varios de sus componentes.

Para poder utilizar **KubeVela**, dependiendo de lo que vayas a realizar y las opciones que vayas a tomar, debes saber manejarte un poco o al menos conocer varios softwares como puede ser **Helm**, **Kubernetes**, **Docker...** Pero es bastante amigable para el usuario, tanto su interfaz gráfica como el empleo de terminal es relativamente sencillo de comprender y utilizar.

En conclusión, **KubeVela** es una herramienta que personalmente le veo mucho potencial si se pule debida y constantemente, siendo su principal virtud, su **amigable interfaz gráfica y comandos de terminal**. Estoy seguro de que muchas de las cosas que he realizado podrían haberse hecho mejor si comprendiese/funcionasen en su totalidad las herramientas que te da **KubeVela.**

7. Dificultades

La mayor dificultad que he encontrado ha sido sin duda la **falta de información**, es cierto que la información que se da en la página web principal de KubeVela es útil y te permite en la gran mayoría instalar KubeVela.

Sin embargo, he tenido muchos problemas a lo largo de las instalaciones y despliegue de las aplicaciones. De hecho, este proyecto tiene 2 demos pero una de ellas ha tenido que **cambiarse debido a los fallos** que brindaba la aplicación y la herramienta.

Otra dificultad es el hecho de que todo es muy teórico, lo bueno es que es relativamente sencillo, y al entrar en la UI de KubeVela estás bastante vendido, **"si no entiendes lo que ves mejor no lo toques"**, ese es un aviso que da KubeVela ya que la UI está diseñada para gente que sabe mucho del tema, eso no quita que no se pueda usar para ver como va el despliegue. Muchas de las aplicaciones que se muestran te recomiendan tener vacío el entorno en el que ejecutas **KubeVela** porque es posible que lo destruyas.

8. Bibliografía

Documentación de KubeVela: https://kubevela.io/docs/

Ejemplo de aplicaciones de KubeVela: https://github.com/kubevela/samples

Documentación de como desplegar la primera aplicación:
https://kubevela.io/docs/quick-start/

Documentación de VelaD: https://github.com/kubevela/velad

Guía de instalación de VelaUX:
https://kubevela.io/docs/reference/addons/velaux/

https://github.com/kubevela/velaux

¿Cómo encontré KubeVela?: https://landscape.cncf.io/

Uso de la UI:
https://kubevela.io/docs/how-to/dashboard/application/create-application/

Despliegue de aplicación en UI:
https://kubevela.io/docs/tutorials/webservice/