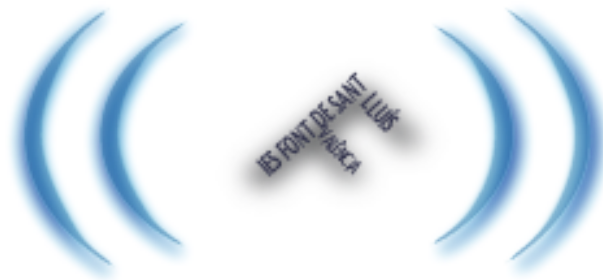


IES FUENTE DE SAN LUIS

Despliegue de aplicaciones web con Kubernetes



- Proyecto de “Administración de sistemas en red”
- Realizado por: Alejandro Forner Artal
- Nombre del tutor del proyecto: Ignacio Navarro Martinez
- Mes y año de presentación: Junio del 2021

índice

1.	Resumen del proyecto	
1.1.	Versión en castellano	2
1.2.	Versión en valenciano	3
1.3.	Versión en inglés	4
2.	Estudio del problema	5
2.1.	Introducción al proyecto	
2.2.	Finalidad del proyecto	
2.3.	Objetivos finales del proyecto	
3.	Recursos necesarios	6
3.1.	Recursos hardware	
3.2.	Recursos software	
4.	Que es Kubernetes y sus ventajas	6
5.	Ejecución del proyecto	8
6.	Fase de pruebas	22
7.	Conclusiones finales	26
8.	Bibliografía	27

1.1 Resumen del proyecto (Castellano)

En este proyecto vamos a ver cómo implantar el sistema Kubernetes que tendrá alojado un servidor web, en concreto bajo Nginx con balanceo de carga.

Lo primero que haremos será la instalación del software necesario para crear nuestra propia imagen de contenedor, necesitaremos tener instalado por un lado Docker y el propio Kubernetes, basado en la nube, para disponer de las herramientas necesarias para poder llevar a cabo este proyecto.

Respecto a la configuración se utilizará una serie de ficheros para poder adaptar el proyecto a las necesidades del mismo, por un lado utilizaremos un script que en él llevará a cabo la instalación de Docker, un fichero de configuración Dockerfile que se encargará de crear una imagen de contenedor Docker para después poderla utilizar con Kubernetes.

Una vez comprobado que los ficheros de configuración hayan terminado de ejecutarse, se procederá a verificar el correcto funcionamiento de la utilidad Kubernetes.

Por último se harán algunos pequeños cambios en los ficheros de configuración para comprobar el correcto funcionamiento de Kubernetes.

Realizados los correspondientes cambios comprobaremos también el mejor método para poderse ejecutar más rápido y con menos carga computacional.



kubernetes

1.2 Resum del projecte (València)

En aquest projecte veurem com implantar el sistema Kubernetes que tindrà allotjat un servidor web, en concret baix Nginx amb balanceig de càrrega.

El primer que farem serà la instal·lació del programari necessari per a crear la nostra pròpia imatge de contenidor, necessitarem tindre instal·lat d'una banda Docker i el propi Kubernetes, basat en el núvol, per a disposar de les eines necessàries per a poder dur a terme aquest projecte.

Respecte a la configuració s'utilitzarà una sèrie de fitxers per a poder adaptar el projecte a les necessitats d'aquest, d'una banda utilitzarem un script que en ell durà a terme la instal·lació de Docker, un fitxer de configuració Dockerfile que s'encarregarà de crear una imatge de contenidor Docker per a després poder-la utilitzar amb Kubernetes.

Una vegada comprovat que els fitxers de configuració hagen acabat d'executar-se, es procedirà a verificar el correcte funcionament de la utilitat Kubernetes.

Finalment es faran alguns xicotets canvis en els fitxers de configuració per a comprovar el correcte funcionament de Kubernetes.

Fets els corresponents canvis comprovarem també el millor mètode per a poder-se executar més ràpid i amb menys càrrega computacional.



kubernetes

1.3 Project summary (English)

In this project we are going to see how to implement the Kubernetes system that will have a web server hosted, specifically under Nginx with load balancing.

The first thing we will do is install the necessary software to create our own container image, we will need to have Docker installed on the one hand and Kubernetes itself, based on the cloud, to have the necessary tools to carry out this project.

Regarding the configuration, a series of files will be used to be able to adapt the project to its needs, on the one hand we will use a script that will carry out the installation of Docker, a configuration file Dockerfile that will be in charge of creating an image container file for later use with Kubernetes.

Once it has been verified that the configuration files have finished executing, we will proceed to verify the correct operation of the Kubernetes utility.

Finally, some small changes will be made in the configuration files to check the correct operation of Kubernetes.

Once the corresponding changes have been made, we will also check the best method to be able to run faster and with less computational load.



kubernetes

2. Estudio del problema

Hola a todos/as, Bienvenidos/as a mi proyecto de fin de curso de administración de sistemas informáticos en red.

El motivo de la realización de este proyecto es porque cada día se intenta buscar soluciones más sencillas, más simples y a menor coste posible a los temas más complejos en relación a la informática y las telecomunicaciones, por eso, mi proyecto busca fomentar esa idea. Además la idea principal de este proyecto es por un lado ayudar a las personas particulares y a las empresas, sea cual sea su situación económica a implantar soluciones que ayuden a abaratar costes, aprovechar al máximo el espacio de trabajo y dedicarle el menor tiempo posible.

2.1 Introducción al proyecto

En este proyecto vamos a ver como se puede implementar el sistema Kubernetes que aloja una página web bajo Nginx con balanceo de carga, para ello se utilizarán una serie de ficheros necesarios para poder personalizar y automatizar el desarrollo de este proyecto.

2.2 Finalidad del proyecto

La finalidad de este proyecto radica en poder simplificar y administrar de forma más eficiente los recursos tanto hardware como software, ahorrando costes de mantenimiento, ahorro de tiempo al estar mucho más automatizado y ahorro también en el espacio de almacenamiento ya que cada día más se busca este tipo de soluciones.

2.3 Objetivos finales del proyecto

Entre los objetivos de este proyecto está la idea de poder demostrar la fiabilidad de cambiar sin miedo del entorno de trabajo adaptándolo a las necesidades y a las exigencias del mercado tecnológico.

3. Recursos necesarios

3.1 Recursos hardware

Para la realización de este proyecto no se necesita tener gran cantidad de hardware, los requisitos disponibles para ejecutar este proyecto son los siguientes:

- Procesador: Intel core I3 4150 a 3,5 GHz
- Memoria RAM: 12 GB DDR3 marca Kingston
- Almacenamiento: 120 GB SSD y 1TB HDD

3.2 Recursos software

- Docker versión: 19.03.6
- Kubernetes versión: v1.20.1
- Sistema operativo: Gnu/Linux kernel 5.4.0 Generic (Zorin OS)

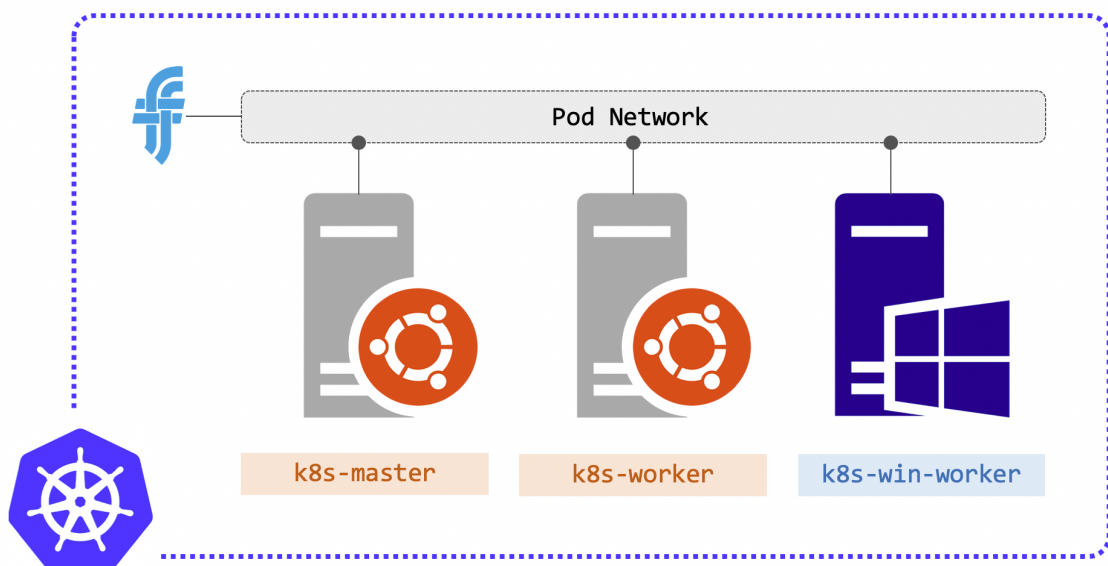
4. ¿Qué es Kubernetes?

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

Google liberó el proyecto Kubernetes en el año 2014. Kubernetes se basa en la experiencia de Google corriendo aplicaciones en producción a gran escala por década y media, junto a las mejores ideas y prácticas de la comunidad.

Ventajas que ofrece Kubernetes

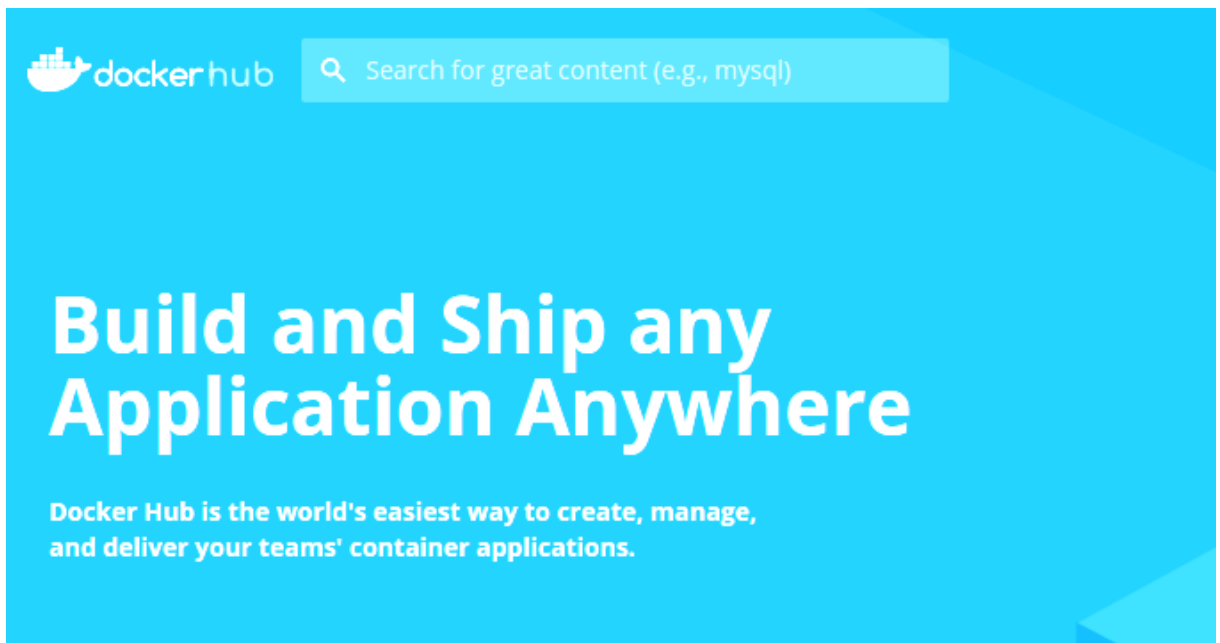
- **Ágil creación y despliegue de aplicaciones:** Mayor facilidad y eficiencia al crear imágenes de contenedor en vez de máquinas virtuales.
- **Desarrollo, integración y despliegue continuo:** Permite que la imagen del contenedor se construya y despliegue de forma frecuente y confiable.
- **Consistencia entre los entornos de desarrollo, pruebas y producción:** La aplicación funciona igual en un laptop y en la nube.
- **Portabilidad entre nubes y distribuciones:** Funciona en Ubuntu, RHEL, CoreOS, tu datacenter físico, Google Kubernetes Engine.
- **Utilización de recursos:** Permite mayor eficiencia y densidad.



5. Ejecución del proyecto

Una vez hayamos explicado un breve resumen de este proyecto y sus características y finalidades llegó la hora de explicar el desarrollo del mismo. Para empezar a realizar este proyecto, tenemos que registrarnos en una web llamada Docker Hub, este registro es necesario para posteriormente poder subir nuestra propia imagen del contenedor que usaremos para realizar este proyecto.

Link de la página web: <https://hub.docker.com/>



Una vez accedido a Docker Hub para registrarnos nos pedirá un identificador de Docker, un correo electrónico y una contraseña.



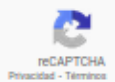
Create a Docker ID.

Already have an account? [Sign In](#)



☐ Send me occasional product updates and announcements.

☐ No soy un robot




Sign Up

By creating an account, you agree to the [Terms of Service](#),
[Privacy Policy](#), and [Data Processing Terms](#).

Una vez registrados nos dará a elegir el tipo de plan que queremos utilizar para esta cuenta, nosotros elegiremos el gratuito que nos ofrece la posibilidad de tener un repositorio privado, para este proyecto es más que suficiente porque el repositorio que vamos a crear es público y aparte son ilimitados.

Free	Pro	Team
FOR INDIVIDUALS	FOR INDIVIDUALS	FOR ORGANIZATIONS
<ul style="list-style-type: none">✓ Unlimited public repositories✓ 1 private repository✓ Community support	<ul style="list-style-type: none">✓ Unlimited public repositories✓ Unlimited image pulls✓ Up to 1 collaborator per private repository✓ 300 vulnerability scans per month✓ 2 parallel builds✓ Email support	<ul style="list-style-type: none">✓ Unlimited public repositories✓ Unlimited private repositories✓ Unlimited image pulls✓ User management with role-based access controls✓ Unlimited teams✓ Unlimited vulnerability scans✓ 3 parallel builds✓ Email support
\$0 /month	\$5 /month With annual plan	\$7 user/month Starts at \$25 for 5 users
Continue with Free	Buy Now	Buy Now

A continuación, nos mandaran un correo electrónico a la dirección del registro para confirmar que la dirección de correo electrónico es válida.



Please verify your email address

Great! You're almost there. Before you can create a repository or configure Docker Hub, you'll need to verify your email address.

We've sent a verification email to **a.forner1998@gmail.com.**

Tras haber verificado el correo electrónico ya estaremos correctamente registrados en la página de Docker Hub.



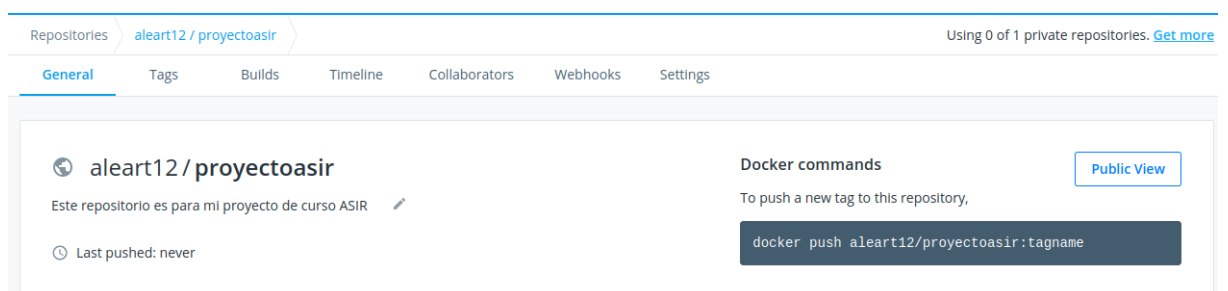
A continuación crearemos nuestro repositorio de forma pública que en él subiremos la imagen de contenedor creada por nosotros mismos.



Nota: La dirección de un repositorio se crea utilizando el Docker ID que en este caso es aleart12 y el nombre del repositorio que en este caso lo he llamado proyectoasir.

The screenshot shows the 'Create Repository' form on Docker Hub. At the top, there's a section for repository details with a dropdown menu set to 'aleart12' and a text input field containing 'proyectoasir'. Below this, a description reads 'Este repositorio es para mi proyecto de curso ASIR'. The 'Visibility' section shows two options: 'Public' (selected with a radio button) and 'Private'. The 'Public' option is described as 'Public repositories appear in Docker Hub search results', while the 'Private' option is 'Only you can view private repositories'. At the bottom, there's a section for 'Build Settings (optional)'.

En definitiva, se vería así el repositorio: **aleart12/proyectoasir**



Perfecto, con todos estos pasos ya tendremos nuestro repositorio creado en docker Hub, ahora crearemos un fichero Dockerfile que lo editaremos con una serie de parámetros de configuración para poder construir una imagen de contenedor que procederemos a subir.

Los parámetros son los siguientes:

- FROM nginx:latest → Este parámetro sirve para empezar a construir nuestra imagen personalizada a través de esta imagen de Docker Hub.
- COPY index.html → Este parámetro sirve para copiar un archivo a el directorio especificado.

Nota: Con este proceso conseguimos que nuestra imagen personalizada contenga como página por defecto nuestra propia web.

- EXPOSE 80 Y 8080: Este parámetro se utiliza para que nuestra imagen tenga abiertos estos puertos para permitir peticiones web.

```
FROM nginx:latest
COPY index.html /usr/share/nginx/html
EXPOSE 80
EXPOSE 8080
```

Os muestro el contenido del fichero index.html:

Bienvenidos/as al proyecto de Alejandro Forner Artal

Nombre del proyecto en distintos idiomas

Español	Implantación de un sistema en la nube con Kubernetes con balanceo de carga
Inglés	Implementing a cloud system with load balanced Kubernetes
Francés	Implémentation d'un système cloud avec Kubernetes à charge équilibrée
Alemán	Implementierung eines Cloud-Systems mit Kubernetes mit Lastenausgleich
Portugués	Implementar um sistema de nuvem com Kubernetes com carga balanceada
Ruso	Внедрение облачной системы с Kubernetes с балансировкой нагрузки

A continuación nos toca compilar el fichero Dockerfile para que pase a ser una imagen de contenedor completamente operativa.

Comando: `docker build -t pasir .`

```

alejandro@alejandro-PC:~/Escritorio/Curso/Proyecto$ docker build -t pasir .
Sending build context to Docker daemon 95.66MB
Step 1/4 : FROM nginx:latest
latest: Pulling from library/nginx
a076a628af6f: Pull complete
0732ab25fa22: Pull complete
d7f36f6fe38f: Pull complete
f72584a26f32: Pull complete
7125e4df9063: Pull complete
Digest: sha256:10b8cc432d56da8b61b070f4c7d2543a9ed17c2b23010b43af434fd40e2ca4aa
Status: Downloaded newer image for nginx:latest
--> f6d0b4767a6c
Step 2/4 : COPY index.html /usr/share/nginx/html
--> bbd2459bac5e
Step 3/4 : EXPOSE 80
--> Running in 8fee2b3547ed
Removing intermediate container 8fee2b3547ed
--> e4d1b966cf70
Step 4/4 : EXPOSE 8080
--> Running in 91843b5903c6
Removing intermediate container 91843b5903c6
--> 8298dd1e3d6d
Successfully built 8298dd1e3d6d
Successfully tagged pasir:latest

```

Nota 1: El comando se tiene que realizar en el mismo directorio de donde se encuentra el fichero Dockerfile.

Nota 2: El nombre pasir es el nombre que se le asigna como tag a la imagen.

Como vemos aquí en todas las imágenes que tenemos disponibles, veremos una que se llama pasir, esa imagen corresponde a la creada recientemente por el Dockerfile.

```

alejandro@alejandro-PC:~/Escritorio/Curso/Proyecto$ docker images
REPOSITORY          TAG         IMAGE ID      CREATED        SIZE
pasir                latest     8298dd1e3d6d  4 minutes ago  133MB
gcr.io/k8s-minikube/kicbase  v0.0.15-snapshot4  06db6ca72446  4 weeks ago   941MB
alejandro@alejandro-PC:~/Escritorio/Curso/Proyecto$

```

Perfecto, ahora tenemos que iniciar sesión en la terminal con docker porque si no no podrá subir la imagen del contenedor.

```

alejandro@alejandro-PC:~/Escritorio/Curso/Proyecto$ docker login
Login with your Docker ID to push and pull images from Docker Hub. If you don't have a Docker ID, head over to https://hub.docker.com to create one.
Username: aleart12
Password:
WARNING! Your password will be stored unencrypted in /home/alejandro/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded
alejandro@alejandro-PC:~/Escritorio/Curso/Proyecto$

```

El siguiente paso es agregarle un tag a nuestra imagen.

Comando: `docker push aleart12/proyectoasir:pasir`

```

alejandro@alejandro-PC:~/Escritorio/Curso/Proyecto$ docker images

```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
aleart12/pasir	v1	8298dd1e3d6d	17 minutes ago	133MB
aleart12/proyectoasir	pasir	8298dd1e3d6d	17 minutes ago	133MB

Hecho ese paso lo único que nos queda es subir nuestra imagen a nuestro repositorio de Docker Hub.

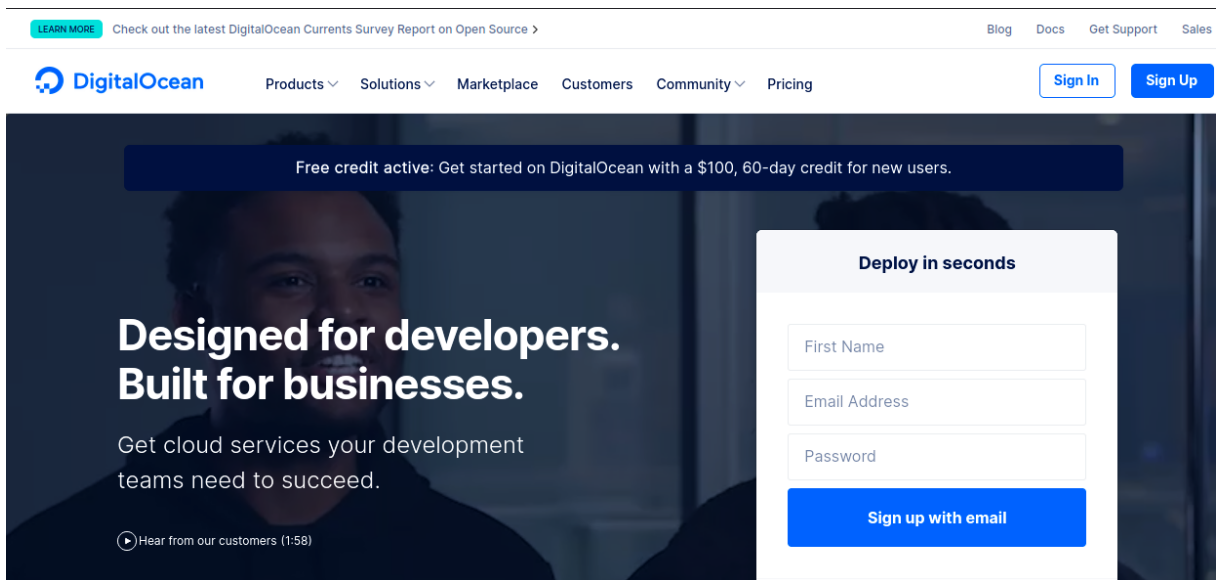
```

alejandro@alejandro-PC:~/Escritorio/Curso/Proyecto$ docker push aleart12/proyectoasir:pasir
The push refers to repository [docker.io/aleart12/proyectoasir]
4b892ef17462: Mounted from aleart12/pasir
85fcec7ef3ef: Mounted from aleart12/pasir
3e5288f7a70f: Mounted from aleart12/pasir
56bc37de0858: Mounted from aleart12/pasir
1c91bf69a08b: Mounted from aleart12/pasir
cb42413394c4: Mounted from aleart12/pasir
pasir: digest: sha256:0c1870d094a1c4de850261d365f58a67cd592ebc17ced35d87dcc1a3556c530d size: 1570

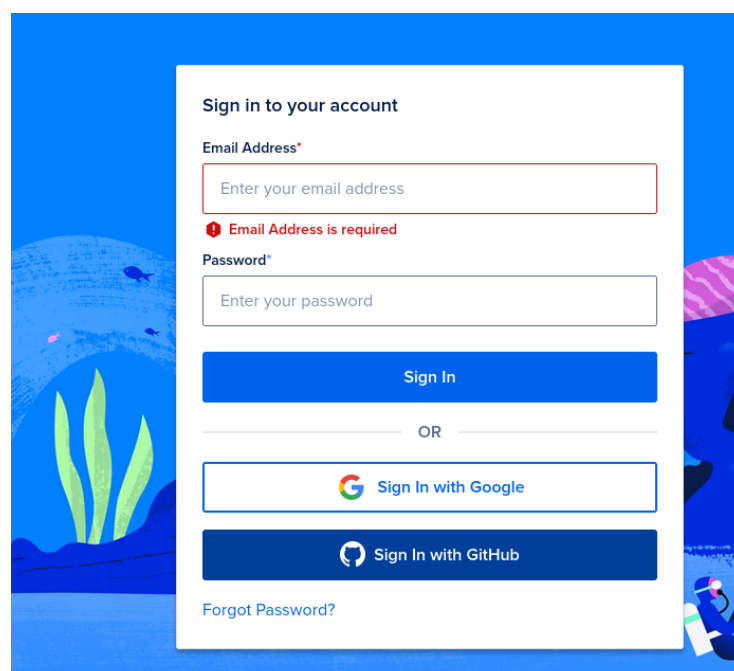
```


Perfecto, ahora que tenemos nuestra imagen de contenedor personalizada subida a Docker Hub procederemos a crearnos una cuenta en el sitio web de DigitalOcean que servirá para posteriormente crear un cluster de Kubernetes.

Página web de DigitalOcean: <https://www.digitalocean.com/>



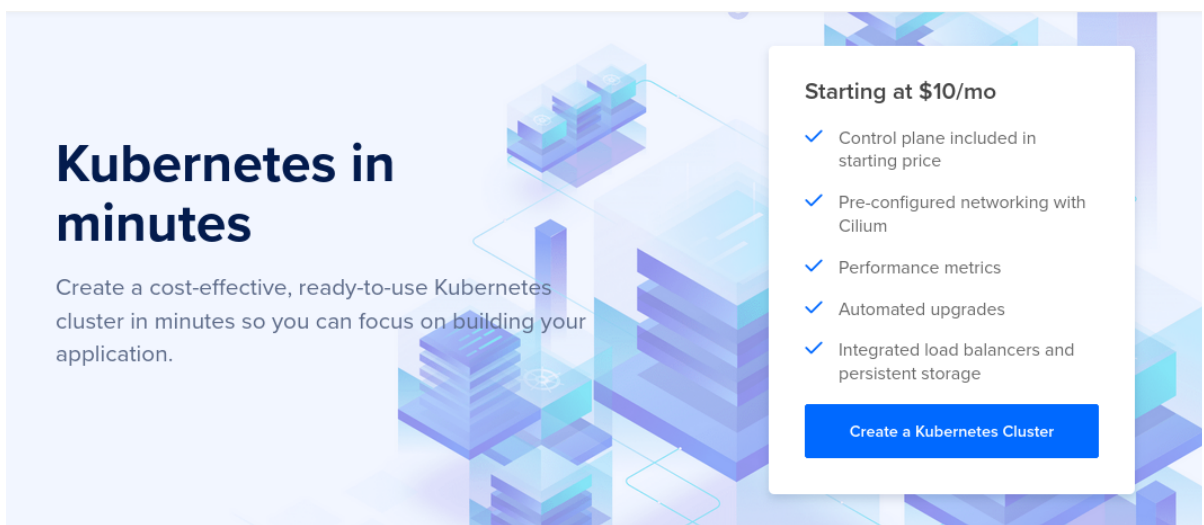
A continuación crearemos una cuenta, en mi caso me he registrado a través de mi cuenta de Google ya que nos ofrece esa opción para el registro y aparte es mucho más rápido porque toda la información requerida ya está almacenada en la cuenta de google.



Una vez realizado correctamente el registro se nos pedirá para verificar la cuenta que hagamos un ingreso, ya sea por medio de cuenta bancaria o a través de PayPal.

A continuación cuando se haya verificado la cuenta procederemos a crear un cluster de Kubernetes.

Al principio nos dice que el mantenimiento del cluster asciende a 10 dólares al mes pero ese precio es relativo, es decir, dependiendo del número de GB tengamos, del uso de la cpu y de la memoria, de todas formas como he invertido 25 dólares de momento no nos preocupamos.



Durante el proceso de creación del cluster nos pedirá una serie de opciones a elegir para poder perfilar y crear el mismo.

Las primeras opciones nos indican la versión de Kubernetes a elegir, en mi caso es la que viene por defecto.

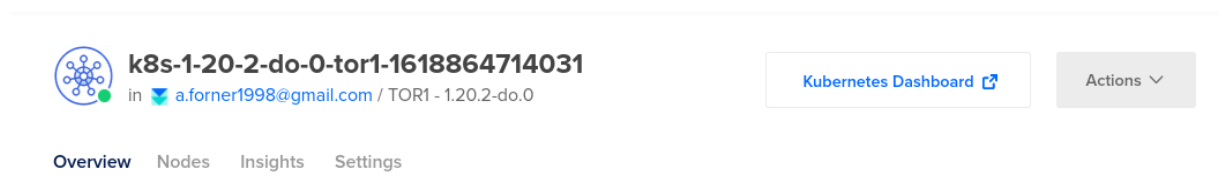
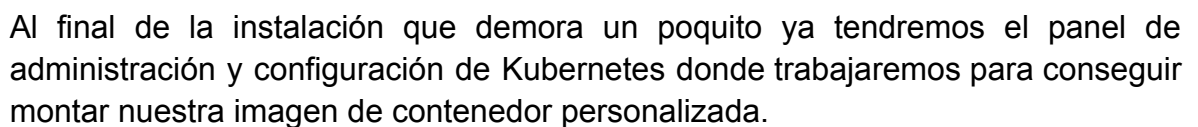
Luego nos indica que centro de datos queremos para alojar nuestro cluster, en mi caso he escogido Toronto, Canadá.

Las siguientes opciones son las de elegir la capacidad del cluster, en mi caso he escogido el paquete más básico y por ende el más económico, como dije anteriormente, 10 dólares al mes pero puede variar.

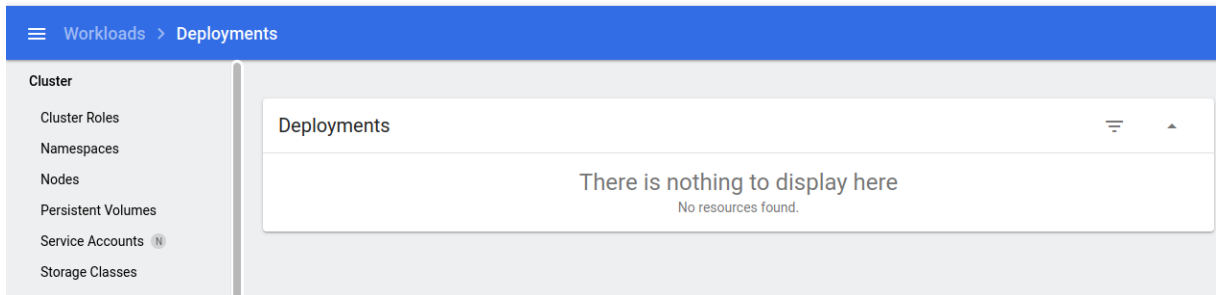
Recommended: A minimum of 2 nodes is required to prevent downtime during upgrades or maintenance.

MONTHLY RATE **\$10.00/month** \$0.01/hour

Con estos sencillos pasos aplicamos los cambios y empezará a crearnos nuestro cluster en base a la información dada anteriormente.



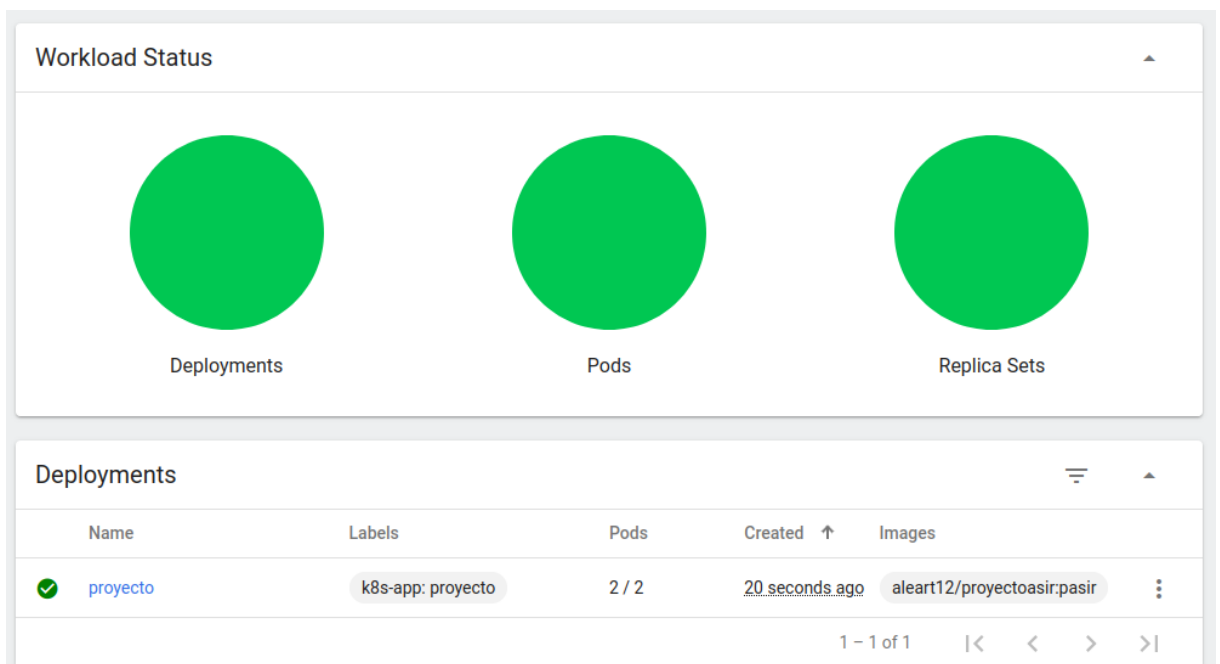
Perfecto, ahora en el panel de control de Kubernetes lo primero que hay que hacer es crear un deployment que almacenará todos los pods y configuraciones respectivas a nuestra imagen.



Mirando un poco la información que nos solicita es la siguiente:

- Nombre de la aplicación: Nombre que le vamos a dar a nuestro deployment.
- Imagen de contenedor: La URL de nuestro contenedor en Docker Hub.
- Número de pods: Cantidad de réplicas a usar con nuestra imagen.
- Servicio External: Servicio externo para que pueda ser accesible desde cualquier red.
- Secreto: un secreto para poder usar la imagen.

Una vez rellenado los campos solicitados, se procederá a crear nuestro deployment. El color verde indica que la imagen ya está corriendo.



Como podemos ver en la sección de pods que todos están funcionando correctamente, más adelante en la batería de pruebas veremos la gran utilidad que tienen estos pods.

Workloads > Pods							
Cluster							
Cluster Roles							
Namespaces							
Nodes							
Persistent Volumes							
Service Accounts							
Storage Classes							
Workloads							
Cron Jobs							
Daemon Sets							
Deployments							

Pods							
Name	Labels	Node	Status	Restarts	CPU Usage (cores)	Memory Usage (bytes)	Created ↑
✓ proyecto-6bb6f7487d-bb7ln	k8s-app: proyecto pod-template-hash: 6bb6f7487d	pool-eihwumg1i-t	Running	0	-	-	a minute ago
✓ proyecto-6bb6f7487d-pfdtz	k8s-app: proyecto pod-template-hash: 6bb6f7487d	pool-eihwumg1i-t	Running	0	-	-	a minute ago

Para comprobar que podemos acceder a nuestra página web nos iremos a la sección de servicios, que si todo ha ido bien nos dará una IP pública para poder visualizar la página desde cualquier lugar y a través de cualquier dispositivo.

La URL es : <http://159.203.52.118/>

Services						
Name	Namespace	Labels	Cluster IP	Internal Endpoints	External Endpoints	Created ↑
✓ proyecto	default	k8s-app: proyecto	10.245.100.164	proyecto:80 TCP proyecto:30069 TCP	159.203.52.118	22 days ago

Bienvenidos/as al proyecto de Alejandro Forner Artal

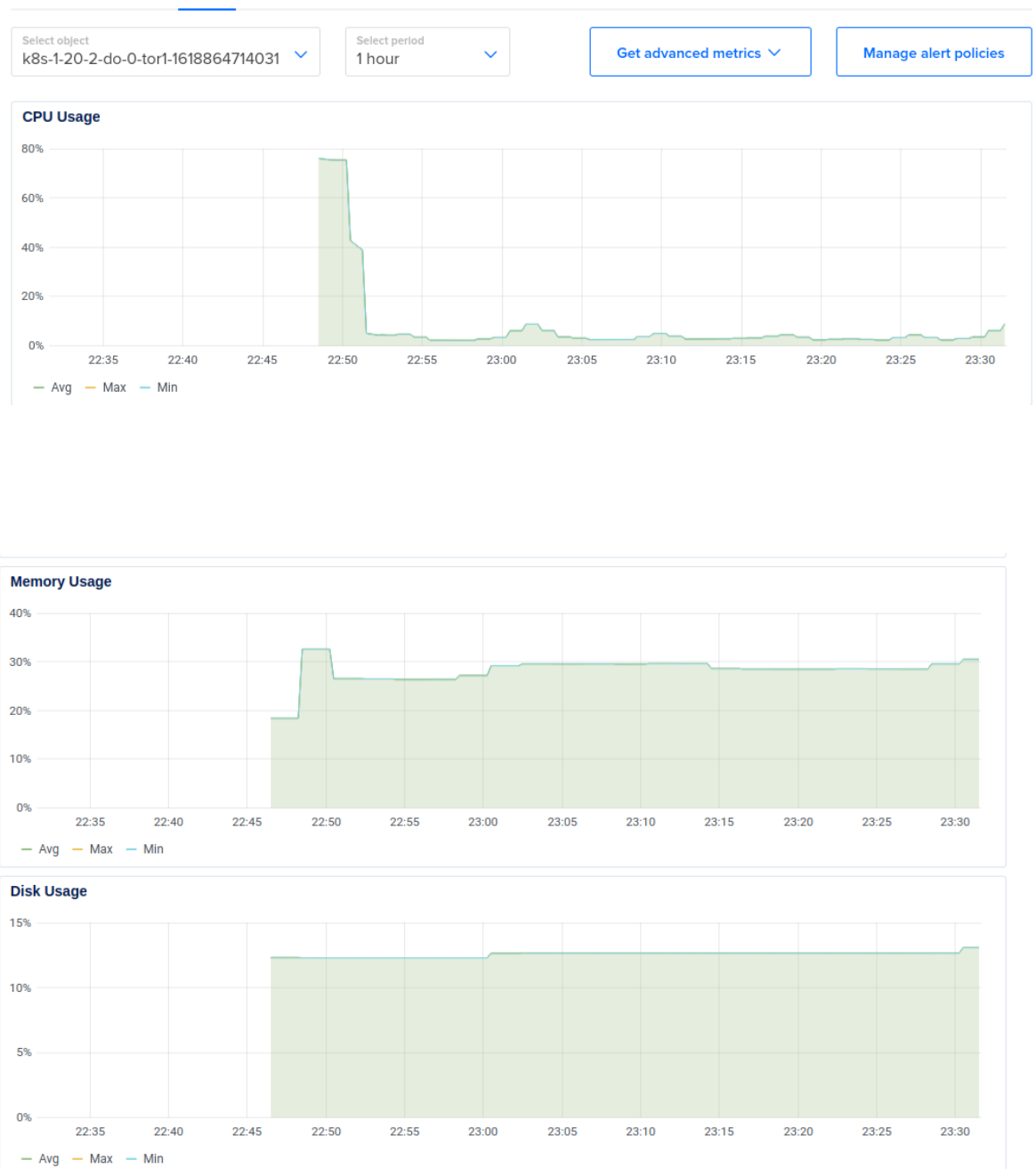
Nombre del proyecto en distintos idiomas

Español	Despliegue de aplicaciones web con kubernetes
Inglés	Deploying web applications with kubernetes
Francés	Déployer des applications Web avec Kubernetes
Alemán	Bereitstellen von Webanwendungen mit Kubernetes
Portugués	Implantar aplicativos da web com kubernetes
Ruso	Развертывание веб-приложений с помощью кubernetes

Con esto ya tendremos activo nuestro cluster Kubernetes, podemos acceder a nuestra página web mostrada anteriormente, eso significa que está respondiendo.

Con el uso, especialmente en otros proyectos o servicios donde exija más movimiento de datos o más tráfico de datos y más rendimiento se podrá observar que la carga de la cpu y de la memoria ram va aumentando o disminuyendo.

En nuestro caso no necesita un alto consumo de cpu ni de memoria.



6. Fase de pruebas

Bien, ahora que tenemos todo instalado y todo configurado correctamente es hora de hacer unas pequeñas pruebas para verificar si soporta cambios y puede funcionar correctamente sin ningún tipo de problema.


Lo primero que haremos será eliminar por nosotros mismos un Pod de nuestro Deployment llamado proyecto. ¿Kubernetes que hará? Lo que Kubernetes hará es relanzar el pod, detectará que un pod a caído por distintas causas, nosotros lo hemos eliminado manualmente y lanzará de nuevo el pod.

Name	Namespac Labels	Node	Status	Restarts	CPU Usage (cores)	Memory Usage (bytes)	Created
✓ proyecto-75d8dff699-655h8	default k8s-app: proyecto pod-template-hash: 75d8dff699	pool-eihwumg1	Running	0	-	-	8 days ago
✓ proyecto-75d8dff699-hsm2b	default k8s-app: proyecto pod-template-hash: 75d8dff699	pool-eihwumg1	Running	0	-	-	8 days ago

Como vemos en la captura de pantalla anterior los dos pods se crearon hace ocho días, lo que vamos a hacer es lo mencionado anteriormente y veremos el resultado.

Delete a resource



Are you sure you want to delete pod *proyecto-75d8dff699-655h8* in namespace *default*?

 This action is equivalent to: `kubectl delete -n default pod proyecto-75d8dff699-655h8`

Delete

Cancel

Efectivamente comprobamos que ha realizado su trabajo correctamente, y es por eso que en la captura siguiente veremos el cambio de nombre del pod y el día de creación.

Name	Namespace	Labels	Node	Status	Restarts	CPU Usage (cores)	Memory Usage (bytes)	Created
 proyecto-75d8dff699-29g6j	default	k8s-app: proyect o pod-template-ha sh: 75d8dff699	pool- eihwumg1	Waiting: ContainerC	0	-	-	21 seconds ago
 proyecto-75d8dff699-655h8	default	k8s-app: proyect o pod-template-ha sh: 75d8dff699	pool- eihwumg1	Terminated Completec	0	-	-	8 days ago

Perfecto, ahora hacemos lo mismo con el otro pod y efectivamente funciona a la perfección.

Esperamos unos días hasta que el pod tenga unos cuantos días de vida porque ahora lo vamos a tirar abajo automáticamente mediante peticiones web con el comando curl.

Cuando lanzamos el script que hará cien mil peticiones una detrás de otra lo que hará el pod será sobrecargarse y crear un pod nuevo para que haga el balanceo de carga. Como este proceso lo hace Kubernetes automáticamente y al recargar la página ya se ven creados, es decir no lo podemos ver en directo pero os aseguro que funciona.

```

Abrir ▾ [+]
Ataque.sh
HDD ~/Escritorio/Proyecto/Archivos necesarios

for i in {1..100000}
do
  curl 159.203.52.118
done

```


Comprobamos que cumple con lo dicho anteriormente, como vemos en esta captura al crear los pods, para que no fueran creándose muchos he puesto un límite de cinco pods. Además de poner el tope de cinco también he puesto el umbral de la CPU en 2 % porque si no mi script no lo tira.

Pods			
Name	Labels	Node	Status
✓ proyecto-5c4648f5f7-4z8pz	k8s-app: proyecto pod-template-hash: 5c4648f5f7	pool-ufq4gcjm2-ε	Running
✓ proyecto-5c4648f5f7-6gmwt	k8s-app: proyecto pod-template-hash: 5c4648f5f7	pool-ufq4gcjm2-ε	Running
✓ proyecto-5c4648f5f7-6kqit	k8s-app: proyecto pod-template-hash: 5c4648f5f7	pool-ufq4gcjm2-ε	Running
✓ proyecto-5c4648f5f7-bjzmq	k8s-app: proyecto pod-template-hash: 5c4648f5f7	pool-ufq4gcjm2-ε	Running
✓ proyecto-5c4648f5f7-kh9dk	k8s-app: proyecto pod-template-hash: 5c4648f5f7	pool-ufq4gcjm2-ε	Running

La segunda que vamos a hacer es muy interesante y muy emocionante. Vamos a crear otra página web que llamaremos v2 de versión 2 y con ella la subiremos a nuestro repositorio de Docker Hub y veremos como Kubernetes actualiza los pods automáticamente con esa nueva web v2.

Bienvenidos/as al proyecto de Alejandro Forner Artal

Nombre del proyecto en distintos idiomas

Español	Despliegue de aplicaciones web con kubernetes
Inglés	Deploying web applications with kubernetes
Francés	Déployer des applications Web avec Kubernetes
Alemán	Bereitstellen von Webanwendungen mit Kubernetes
Portugués	Implantar aplicativos da web com kubernetes
Ruso	Развертывание веб-приложений с помощью кubernetes

Creamos una nueva imagen de contenedor con un Dockerfile y la subimos a nuestro repositorio.

```



Abrir  Dockerfilev2
HDD ~/Escritorio/Proyecto

FROM nginx:latest
COPY indexv2.html /usr/share/nginx/html
EXPOSE 80
EXPOSE 8080

```

Tags and Scans VULNERABILITY SCANNING - DISABLED [Enable](#)

This repository contains 1 tag(s).

TAG	OS	PULLED	PUSHED
 pasirv2		---	11 minutes ago

[See all](#)

El proceso de actualización de los pods puede tardar un ratito, cuando se actualicen ya podremos disfrutar de nuestra actualización de página web.

← → ↻ 🏠 174.138.115.165

Bienvenidos/as al proyecto de Alejandro Forner Artal

Nombre del proyecto en distintos idiomas

Español	Despliegue de aplicaciones web con kubernetes
Inglés	Deploying web applications with kubernetes
Francés	Déployer des applications Web avec Kubernetes
Alemán	Bereitstellen von Webanwendungen mit Kubernetes
Portugués	Implantar aplicativos da web com kubernetes
Ruso	Развертывание веб-приложений с помощью кubernetes

7. Conclusiones finales

Como se ha podido comprobar en la ejecución de este proyecto el servicio Kubernetes es una buena herramienta para poder gestionar servicios, tanto web, ficheros, bases de datos e incluso correo electrónico.

Su simplicidad lo hace apto para que cada vez más usuarios o empresas lo implementan gracias a su automatización y gestión de los servicios, es decir, en lugar de tener varias máquinas físicas o virtuales en funcionamiento que además consumen muchos recursos ya que para poder disponer de un servicio implica simular un sistema completo, gracias a esta solución nos ahorramos los costes de mantenimiento.

Cabe mencionar que en este proyecto se ha buscado un proveedor de Kubernetes para trabajar con él directamente desde la nube y así ante un fallo o alguna ampliación futura poderla gestionar desde cualquier dispositivo pero cabe destacar que también es posible realizar la instalación en un ordenador local.

Con respecto a este proyecto cabe destacar que la ejecución del mismo ha sido muy agradable, no he obtenido ningún problema de implementación ni de configuración y la verdad es que mientras iba desarrollando el proyecto cada vez más me gustaba la propuesta de hacerlo y además aparte de haber disfrutado lo que más valoro también es poder aprender a usar y administrar una tecnología nueva.

Entre las modificaciones futuras que añadiría son en relación con el coste del servicio, tal y como os dije en la ejecución del proyecto me costará diez euros mensuales la puesta en marcha del cluster Kubernetes, al final he tenido que agregar más crédito porque el coste va en función del uso que le des al servicio.

8. Bibliografía

- Instalación de Docker:
<https://docs.docker.com/engine/install/ubuntu/>
- Creación de Imagen de contenedor :
<https://docs.docker.com/engine/reference/builder/>
- Subir imagen de contenedor a Docker Hub :
<https://docs.docker.com/engine/reference/commandline/push/>
- Creación de cluster en Kubernetes:
<https://docs.digitalocean.com/products/kubernetes/>
- Instalación de Kubernetes en local para pruebas:
<https://kubernetes.io/es/docs/tasks/tools/install-kubect/>