Ejercicio 1

Ejecutamos los comandos en la consola.

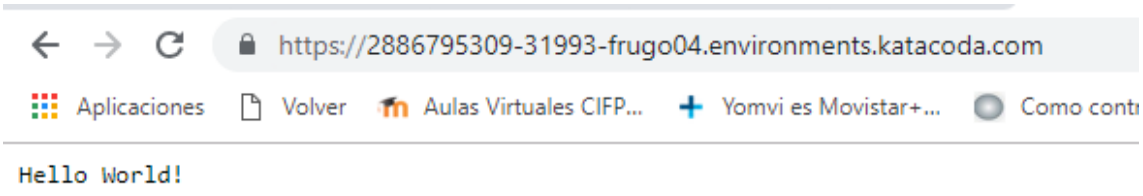
Realizamos el deploy

```
$ kubectl create deployment hello-node --image=gcr.io/hello-minikube-zero-install/hello-node
deployment.apps/hello-node created
$ kubectl get deployments
NAME         READY   UP-TO-DATE   AVAILABLE   AGE
hello-node   1/1     1            1           13s
$
```

Creamos el servicio, Hacemos un expose a la aplicación y obtenemos el puerto de entrada (31993).

```
$ kubectl expose deployment hello-node --type=LoadBalancer --port=8080
service/hello-node exposed
$ kubectl get services
NAME         TYPE           CLUSTER-IP      EXTERNAL-IP   PORT(S)          AGE
hello-node   LoadBalancer   10.110.93.104   <pending>     8080:31993/TCP   10s
kubernetes   ClusterIP      10.96.0.1       <none>        443/TCP          111s
$
```

Accedemos al navegador especificando el puerto.



Eliminamos el servicio y el deploy

```
$ kubectl delete service hello-node
service "hello-node" deleted
$ kubectl delete deployment hello-node
deployment.extensions "hello-node" deleted
$
```

Ejercicio 2

Para realizar la instalación de minikube, hemos utilizado [chocolatey]

```
(c) 2018 Microsoft Corporation. Todos los derechos reservados.

C:\Windows\system32>choco -v
0.10.11
```

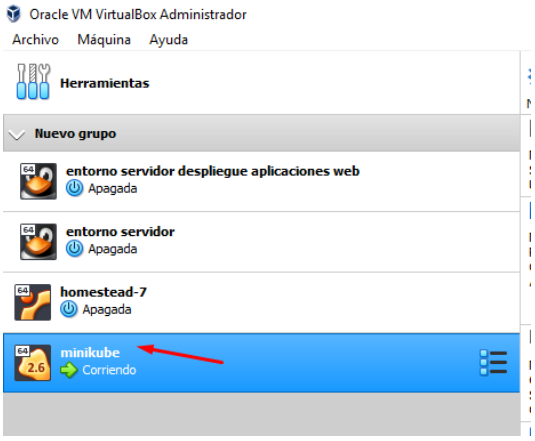
Instalamos minikube

```
C:\Windows\system32>choco install minikube
Chocolatey v0.10.11
Installing the following packages:
minikube
By installing you accept licenses for the packages.
Progress: Downloading kubernetes-cli 1.13.3... 100%
Progress: Downloading Minikube 0.34.1... 100%

kubernetes-cli v1.13.3 [Approved]
kubernetes-cli package files install completed. Performing other installation steps.
The package kubernetes-cli wants to run 'chocolateyInstall.ps1'.
Note: If you don't run this script, the installation will fail.
Note: To confirm automatically next time, use '-y' or consider:
choco feature enable -n allowGlobalConfirmation
Do you want to run the script?([Y]es/[N]o/[P]rint): y

Extracting 64-bit C:\ProgramData\chocolatey\lib\kubernetes-cli\tools\kubernetes-client-windows-amd64.tar.gz to C:\Progra
mData\chocolatey\lib\kubernetes-cli\tools...
C:\ProgramData\chocolatey\lib\kubernetes-cli\tools
Extracting 64-bit C:\ProgramData\chocolatey\lib\kubernetes-cli\tools\kubernetes-client-windows-amd64.tar to C:\ProgramDa
ta\chocolatey\lib\kubernetes-cli\tools...
C:\ProgramData\chocolatey\lib\kubernetes-cli\tools
 ShimGen has successfully created a shim for kubectl.exe
 The install of kubernetes-cli was successful.
```

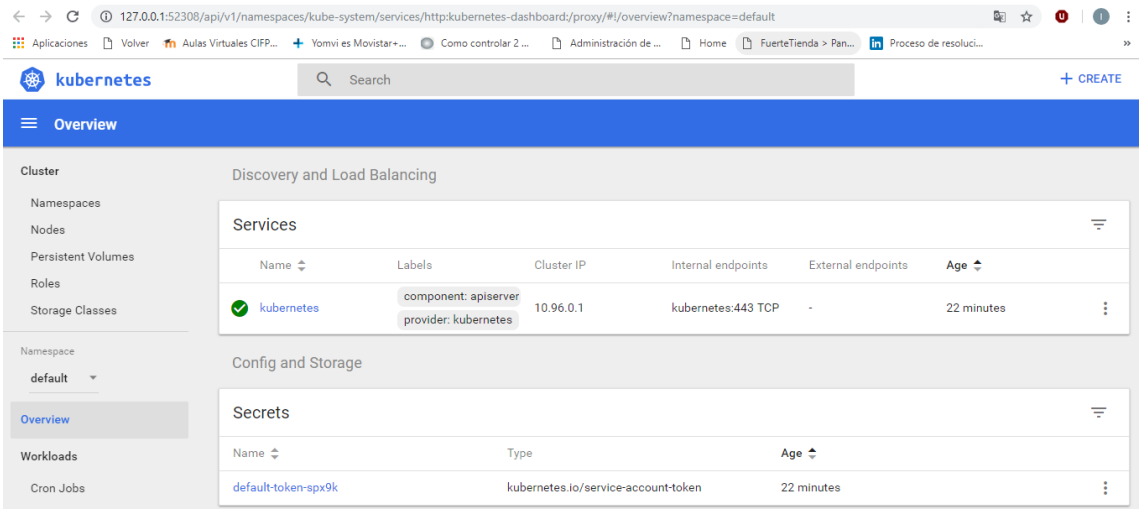
Realizamos el minikube start

```
C:\Windows\system32>minikube start
o   minikube v0.34.1 on windows (amd64)
>   Creating virtualbox VM (CPUs=2, Memory=2048MB, Disk=20000MB) ...
@   Downloading Minikube ISO ...
 184.30 MB / 184.30 MB [============================================] 100.00% 0s
-   "minikube" IP address is 192.168.99.100
-   Configuring Docker as the container runtime ...
-   Preparing Kubernetes environment ...
@   Downloading kubelet v1.13.3
@   Downloading kubeadm v1.13.3
-   Pulling images required by Kubernetes v1.13.3 ...
-   Launching Kubernetes v1.13.3 using kubeadm ...
-   Configuring cluster permissions ...
-   Verifying component health .....
+   kubectl is now configured to use "minikube"
=   Done! Thank you for using minikube!

C:\Windows\system32>
```

Una vez realizado ya tenemos la máquina virtual creada



Arrancamos el dashboard mediante “minikube dashboard”



Ahora siguiendo el tutorial de kubernetes (“Hola Mundo”) creamos el deployment.

```
kubectl create deployment hello-node --image=gcr.io/hello-minikube-zero-install/hello-node
```

```
C:\Windows\system32>kubectl get deployments
NAME         READY   UP-TO-DATE   AVAILABLE   AGE
hello-node   0/1     1            0           61s

C:\Windows\system32>
```

```
C:\Windows\system32>kubectl get pods
NAME                          READY   STATUS    RESTARTS   AGE
hello-node-64c578bdf8-5rmqc   1/1     Running   0          3m57s

C:\Windows\system32>
```

Kubectl get events

```
C:\Windows\system32>kubectl get events
LAST SEEN   TYPE     REASON                   KIND         MESSAGE
4m38s       Normal   Scheduled                Pod          Successfully assigned default/hello-node-64c578bdf8-5rmqc to minikube
4m38s       Normal   Pulling                  Pod          pulling image "gcr.io/hello-minikube-zero-install/hello-node"
2m41s       Normal   Pulled                   Pod          Successfully pulled image "gcr.io/hello-minikube-zero-install/hello-node"
2m41s       Normal   Created                  Pod          Created container
2m41s       Normal   Started                  Pod          Started container
4m38s       Normal   SuccessfulCreate         ReplicaSet   Created pod: hello-node-64c578bdf8-5rmqc
4m38s       Normal   ScalingReplicaSet        Deployment   Scaled up replica set hello-node-64c578bdf8 to 1
33m         Normal   NodeHasSufficientMemory  Node         Node minikube status is now: NodeHasSufficientMemory
33m         Normal   NodeHasNoDiskPressure    Node         Node minikube status is now: NodeHasNoDiskPressure
33m         Normal   NodeHasSufficientPID     Node         Node minikube status is now: NodeHasSufficientPID
32m         Normal   RegisteredNode           Node         Node minikube event: Registered Node minikube in Controller
32m         Normal   Starting                 Node         Starting kube-proxy.

C:\Windows\system32>
```

Realizamos el expose

```
C:\Windows\system32>kubectl expose deployment hello-node --type=LoadBalancer --port=8080
service/hello-node exposed

C:\Windows\system32>minikube service hello-node
-    Opening kubernetes service default/hello-node in default browser...

C:\Windows\system32>
```

Se nos abre una pestaña en el navegador con la aplicación.