CPD - Práctica 6

Elena Cantero Molina

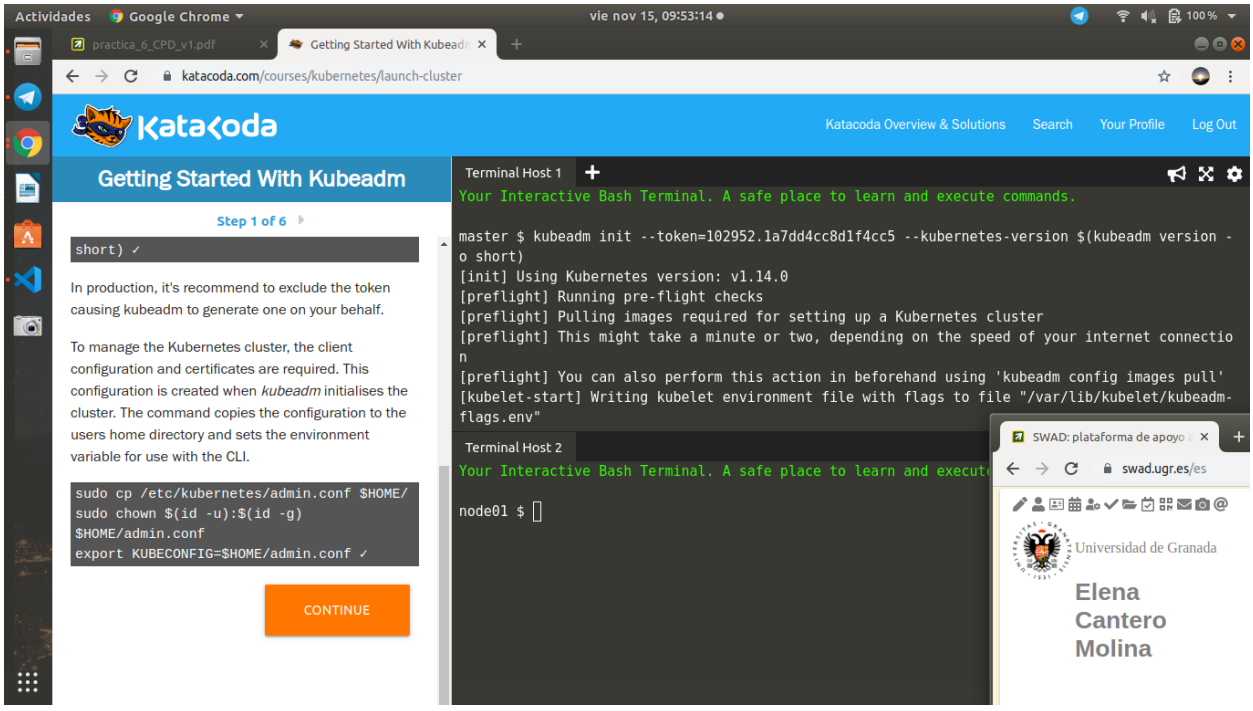
Kubernetes 101

I) Kubernetes

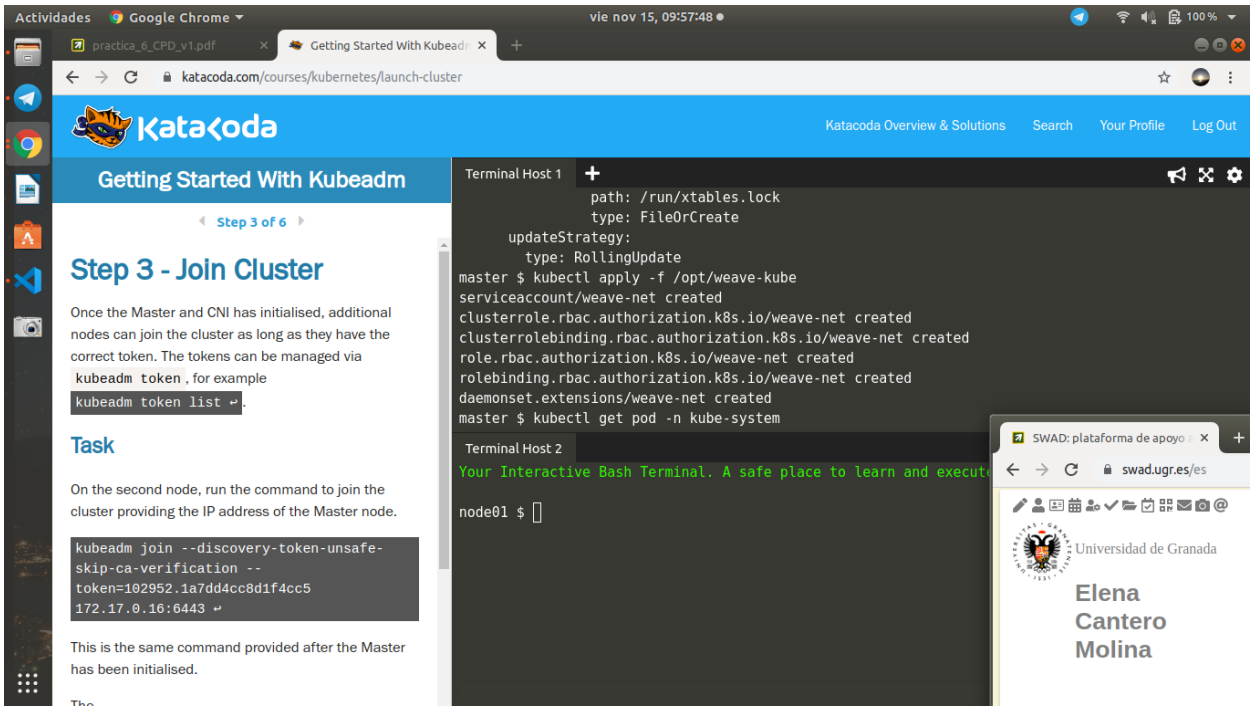
II) Primeros pasos con Kubernetes

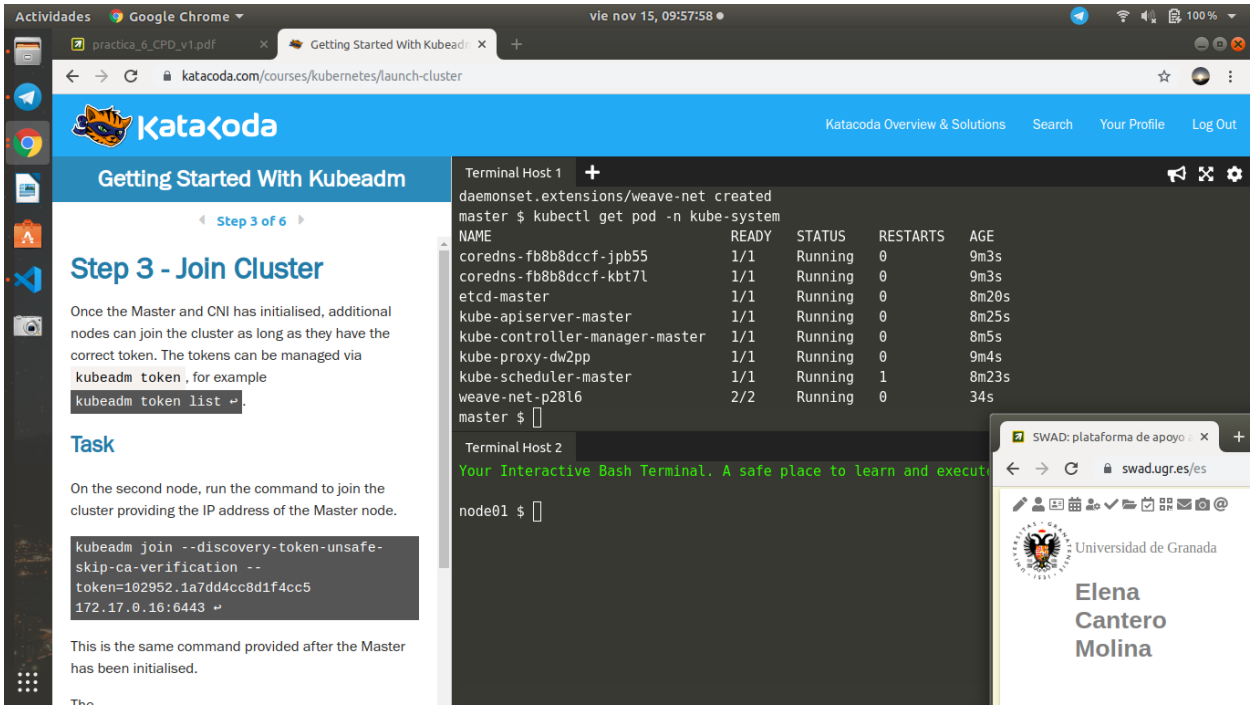
II.a) Lanzamos: “Launch a multi-node cluster using Kubeadm”

Iniciamos los nodos de Kubernetes con Kubeadm

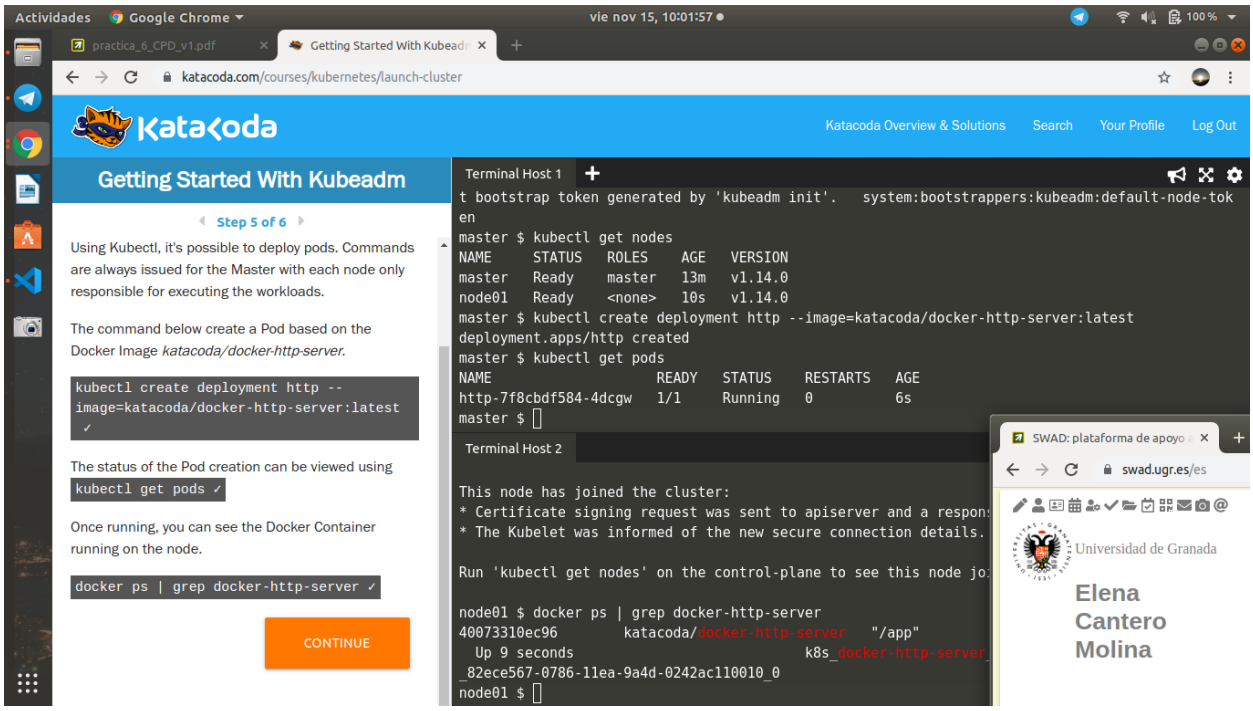


Activamos la capa de red que permite que se vean entre sí de forma transparente todos los contenedores situados en diversas máquinas, Container Network Interface (CNI).



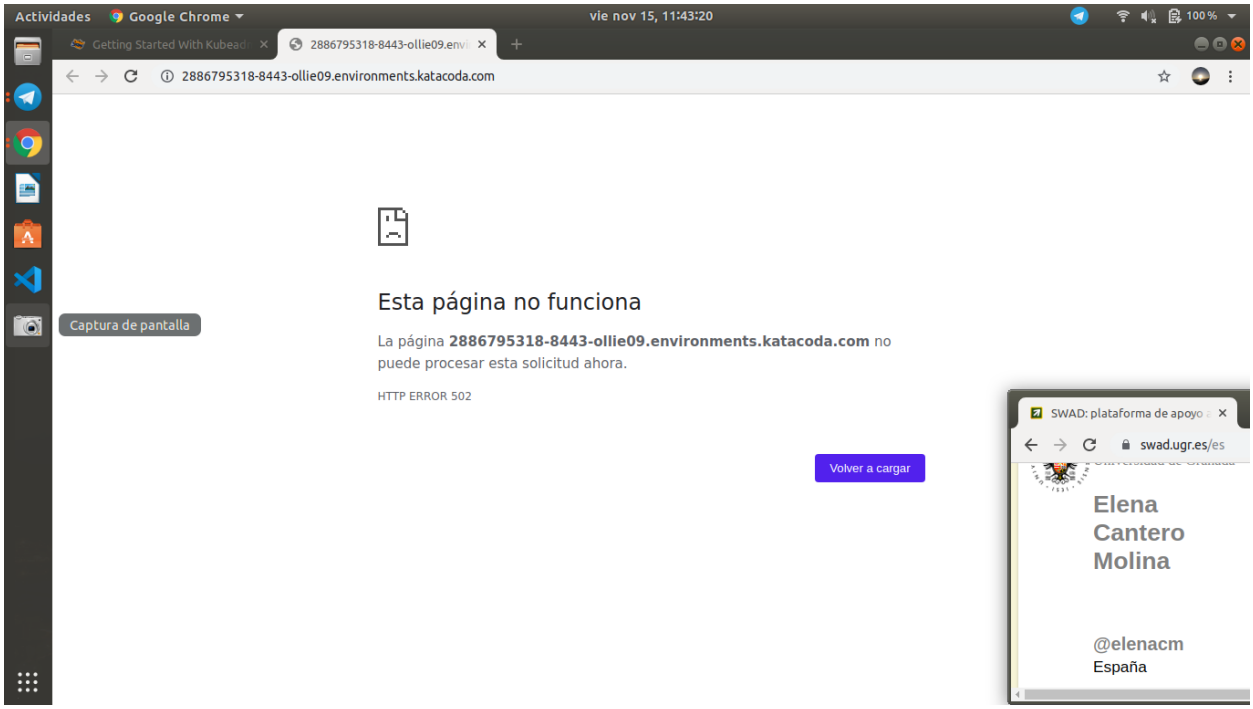


Iniciamos un Pod

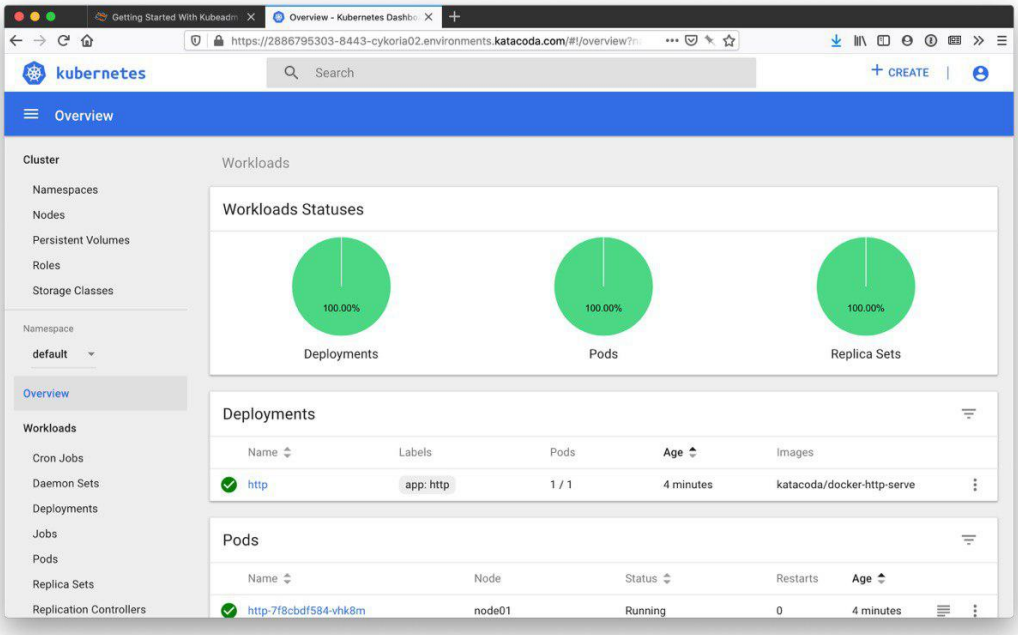


Activamos el Dashboard

A la hora de realizar esta parte me da este error

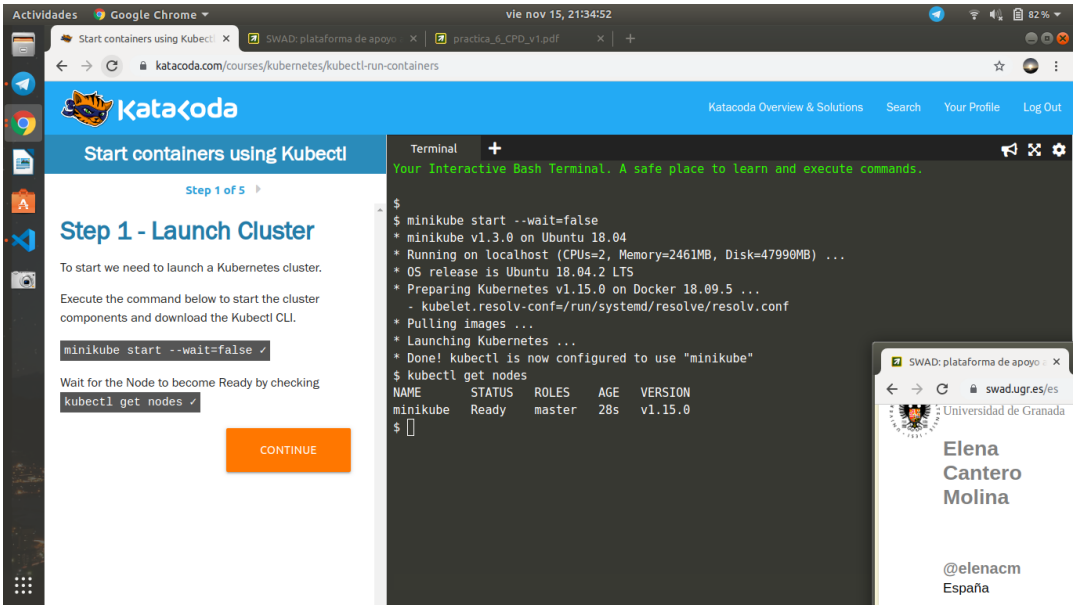


Como no he logrado arreglarlo, en la siguiente captura de pantalla le muestro lo que deberia salir con la captura de pantalla de un compañero de clase

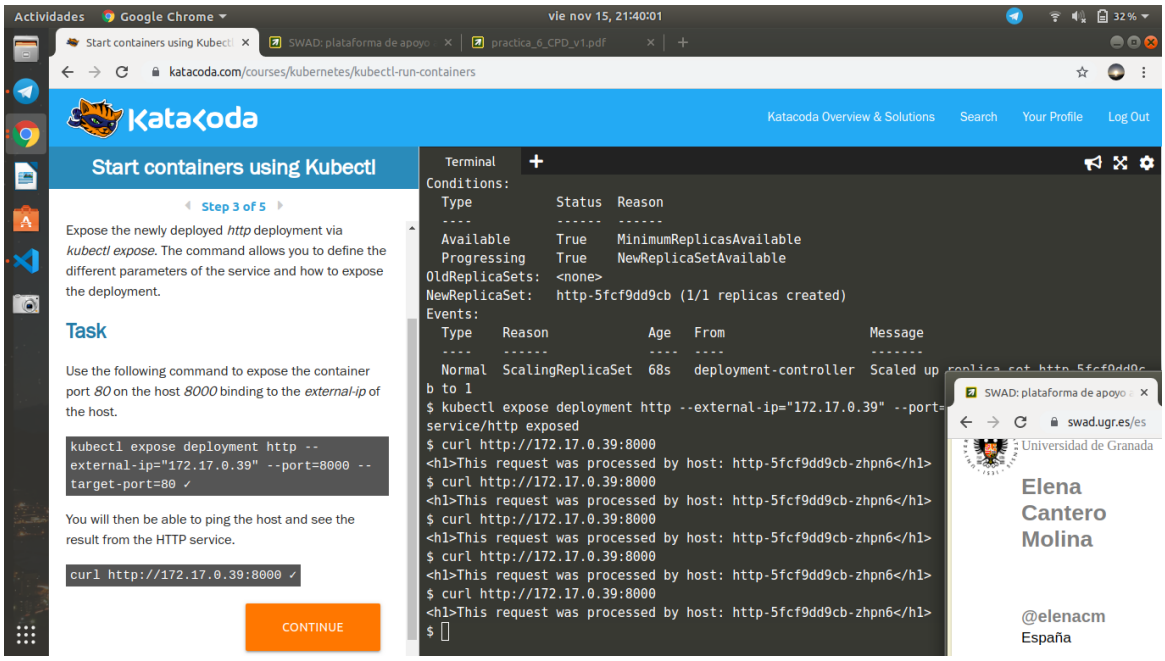


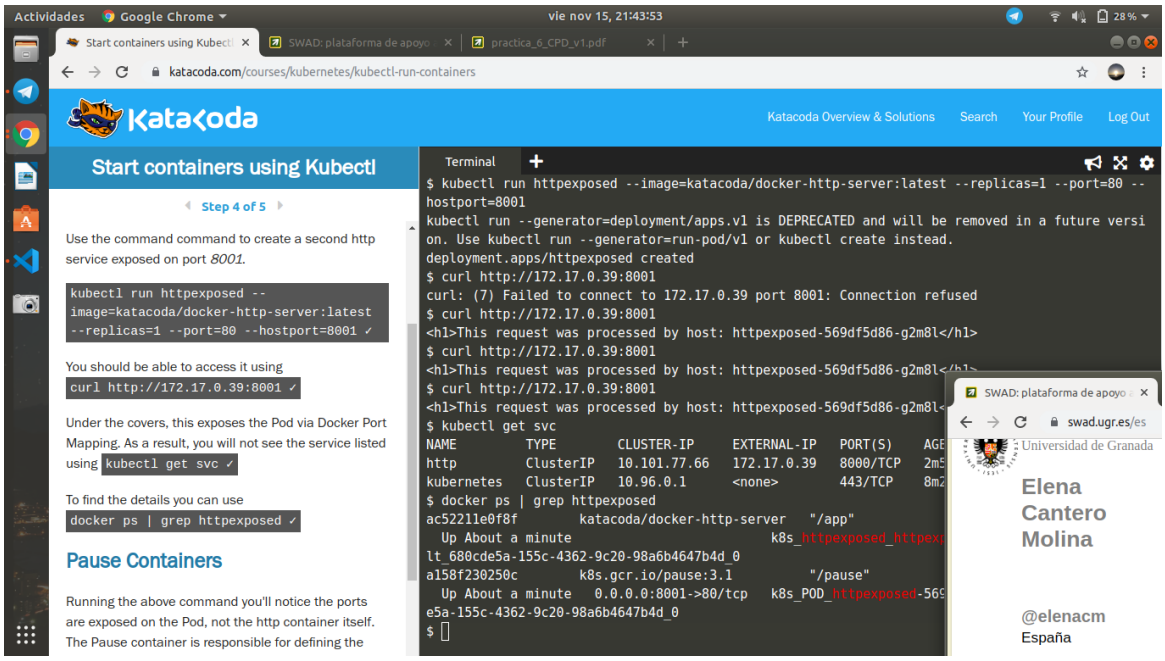
II.b) Lanzamos “Deploy Containers Using Kubectl”

Lanzamos minikube

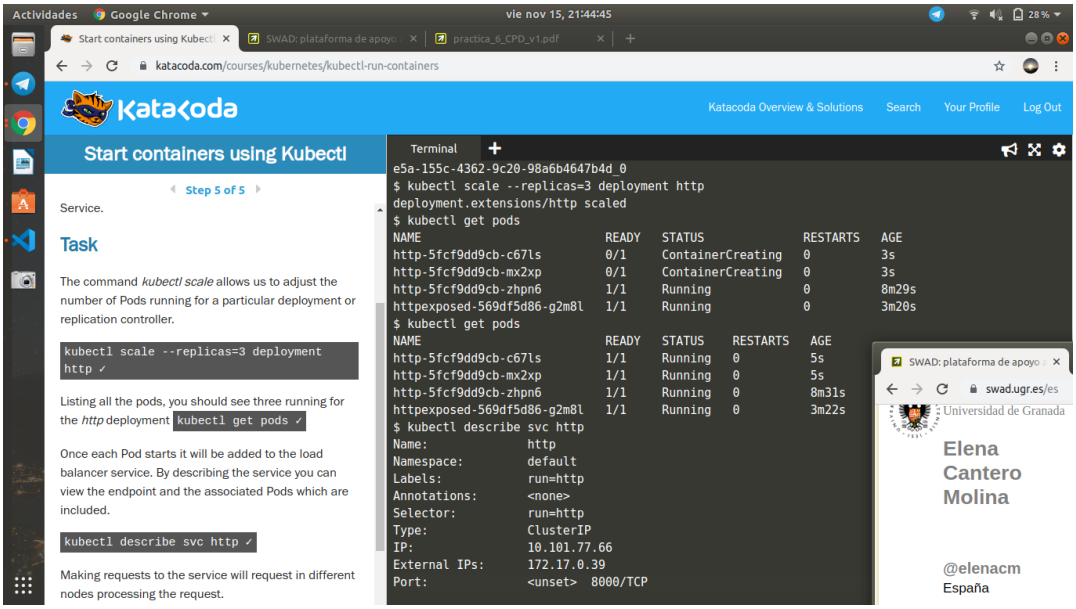


Ejecutamos un despliegue y exportamos el puerto para acceder


Actividades
Google Chrome
vie nov 15, 21:40:01
katacoda.com/courses/kubernetes/kubectl-run-containers
Katacoda
Katacoda Overview & Solutions
Search
Your Profile
Log Out
Start containers using Kubectl
Step 3 of 5
Expose the newly deployed http deployment via kubectl expose. The command allows you to define the different parameters of the service and how to expose the deployment.
Task
Use the following command to expose the container port 80 on the host 8000 binding to the external-ip of the host.
kubectl expose deployment http --external-ip="172.17.0.39" --port=8000 --target-port=80 ✓
You will then be able to ping the host and see the result from the HTTP service.
curl http://172.17.0.39:8000 ✓
CONTINUE
Terminal
Conditions:
Type Status Reason
Available True MinimumReplicasAvailable
Progressing True NewReplicaSetAvailable
OldReplicaSets: <none>
NewReplicaSet: http-5fcf9dd9cb (1/1 replicas created)
Events:
Type Reason Age From Message
Normal ScalingReplicaSet 68s deployment-controller Scaled up
$ kubectl expose deployment http --external-ip="172.17.0.39" --port=
service/http exposed
$ curl http://172.17.0.39:8000
<h1>This request was processed by host: http-5fcf9dd9cb-zhpn6</h1>
$ curl http://172.17.0.39:8000
<h1>This request was processed by host: http-5fcf9dd9cb-zhpn6</h1>
$ curl http://172.17.0.39:8000
<h1>This request was processed by host: http-5fcf9dd9cb-zhpn6</h1>
$ curl http://172.17.0.39:8000
<h1>This request was processed by host: http-5fcf9dd9cb-zhpn6</h1>
$ curl http://172.17.0.39:8000
<h1>This request was processed by host: http-5fcf9dd9cb-zhpn6</h1>
$
SWAD: plataforma de apoyo
swad.ugr.es/es
Universidad de Granada
Elena
Cantero
Molina
@elenacm
España

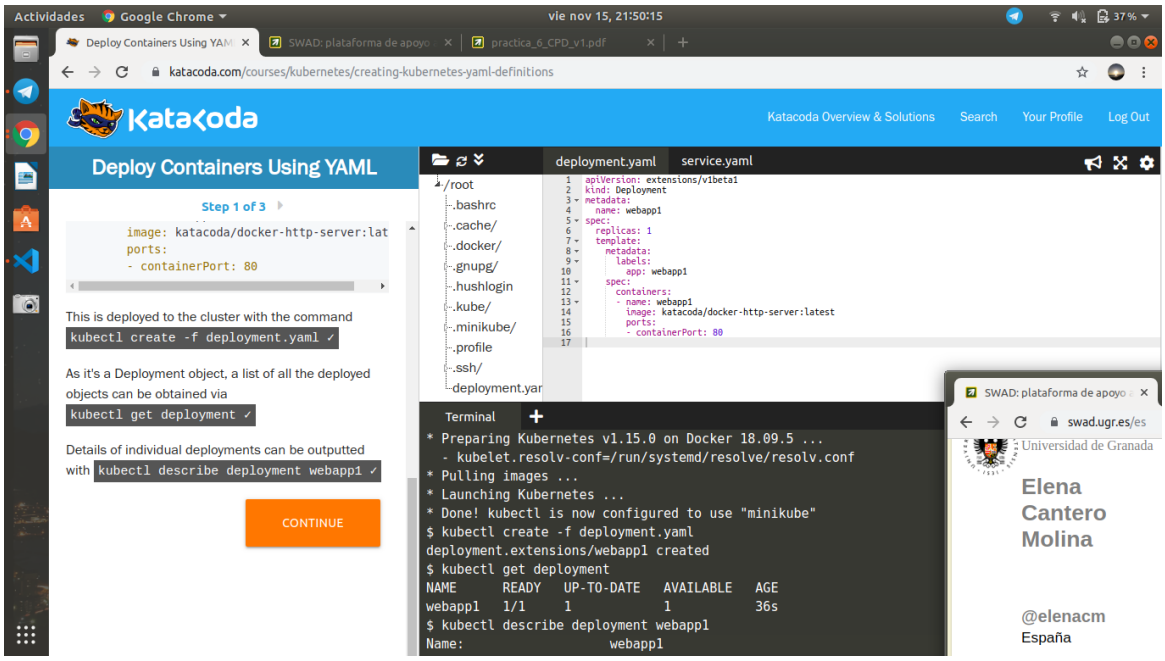

Actividades
Google Chrome
vie nov 15, 21:43:53
katacoda.com/courses/kubernetes/kubectl-run-containers
Katacoda
Katacoda Overview & Solutions
Search
Your Profile
Log Out
Start containers using Kubectl
Step 4 of 5
Use the command command to create a second http service exposed on port 8001.
kubectl run httpexposed --image=katacoda/docker-http-server:latest --replicas=1 --port=80 --hostport=8001 ✓
You should be able to access it using
curl http://172.17.0.39:8001 ✓
Under the covers, this exposes the Pod via Docker Port Mapping. As a result, you will not see the service listed using kubectl get svc ✓
To find the details you can use
docker ps | grep httpexposed ✓
Pause Containers
Running the above command you'll notice the ports are exposed on the Pod, not the http container itself. The Pause container is responsible for defining the
Terminal
$ kubectl run httpexposed --image=katacoda/docker-http-server:latest --replicas=1 --port=80 --hostport=8001
kubectl run --generator=deployment/apps.v1 is DEPRECATED and will be removed in a future version. Use kubectl run --generator=run-pod/v1 or kubectl create instead.
deployment.apps/httpexposed created
$ curl http://172.17.0.39:8001
curl: (7) Failed to connect to 172.17.0.39 port 8001: Connection refused
$ curl http://172.17.0.39:8001
<h1>This request was processed by host: httpexposed-569df5d86-g2m8l</h1>
$ curl http://172.17.0.39:8001
<h1>This request was processed by host: httpexposed-569df5d86-g2m8l</h1>
$ curl http://172.17.0.39:8001
<h1>This request was processed by host: httpexposed-569df5d86-g2m8l
$ kubectl get svc
NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE
http ClusterIP 10.101.77.66 172.17.0.39 8000/TCP 2m5
kubernetes ClusterIP 10.96.0.1 <none> 443/TCP 8m2
$ docker ps | grep httpexposed
ac52211e0f8f katacoda/docker-http-server "/app"
Up About a minute k8s_httpexposed_httpexp
lt_680cde5a-155c-4362-9c20-98a6b4647b4d_0
a158f230250c k8s.gcr.io/pause:3.1 "/pause"
Up About a minute 0.0.0.0:8001->80/tcp k8s_POD_httpexposed-569
e5a-155c-4362-9c20-98a6b4647b4d_0
$
SWAD: plataforma de apoyo
swad.ugr.es/es
Universidad de Granada
Elena
Cantero
Molina
@elenacm
España

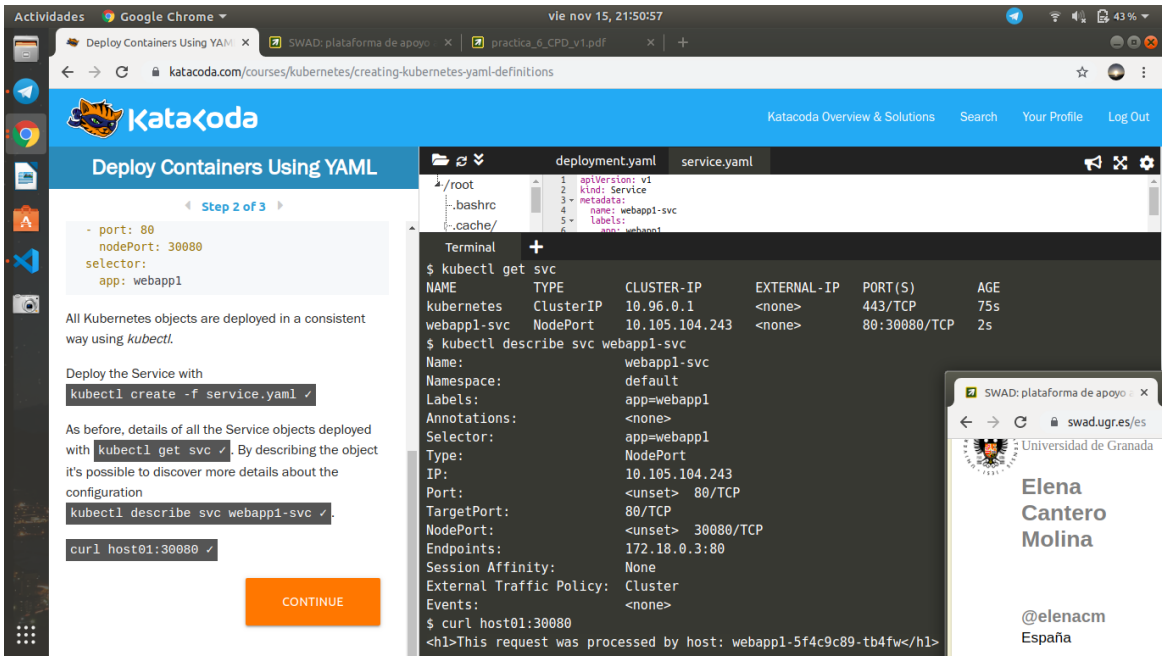
Escalamos

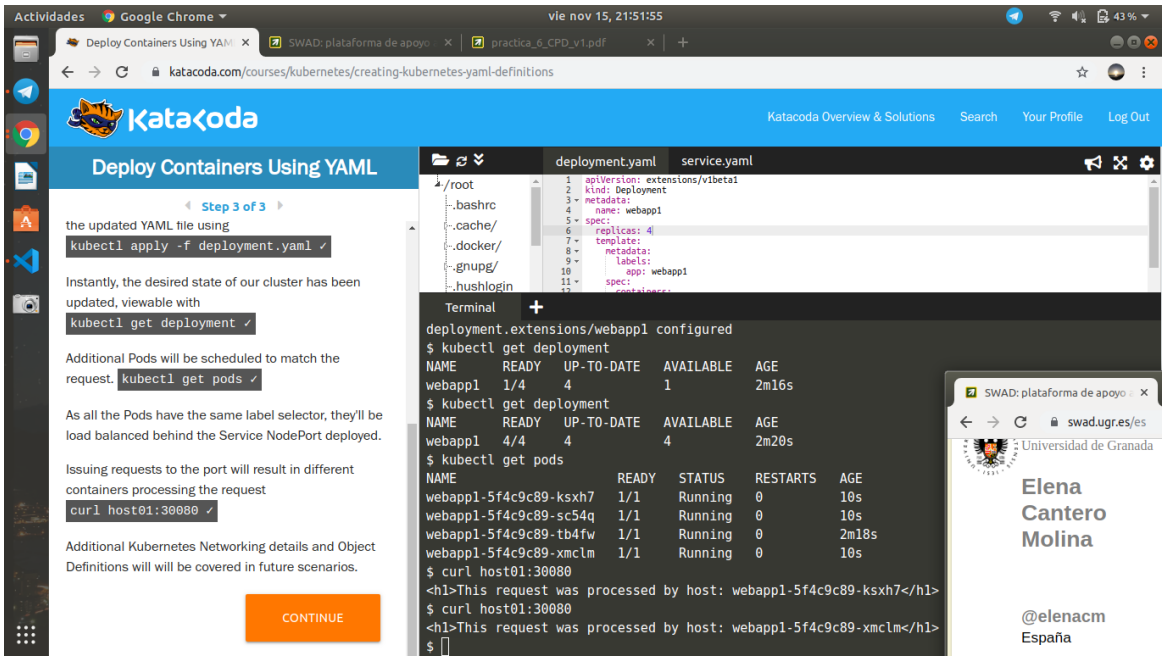


II.c) Lanzamos “Deploy Containers Using YAML”

Creamos un Deployment y un Servicio con ficheros YAML.

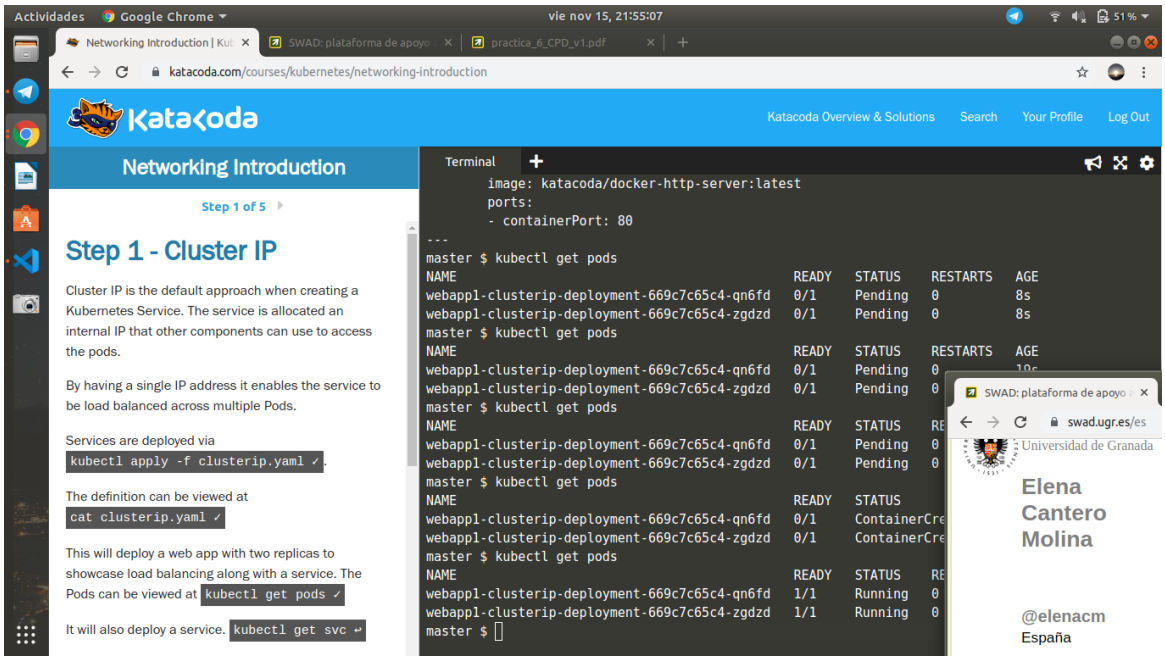


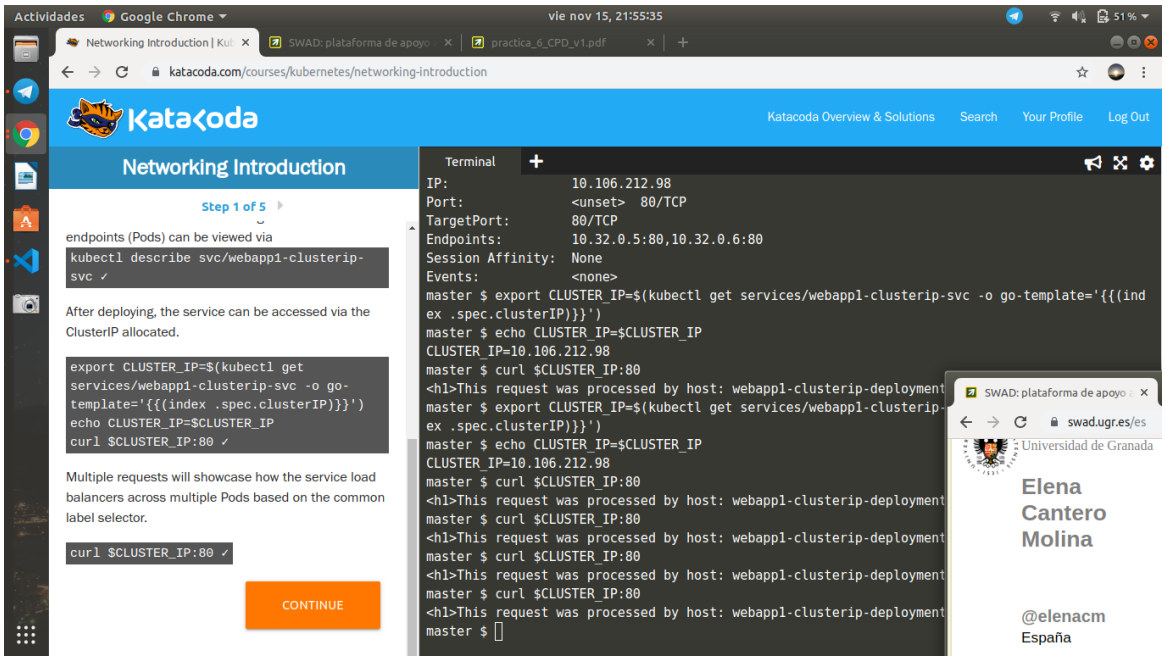




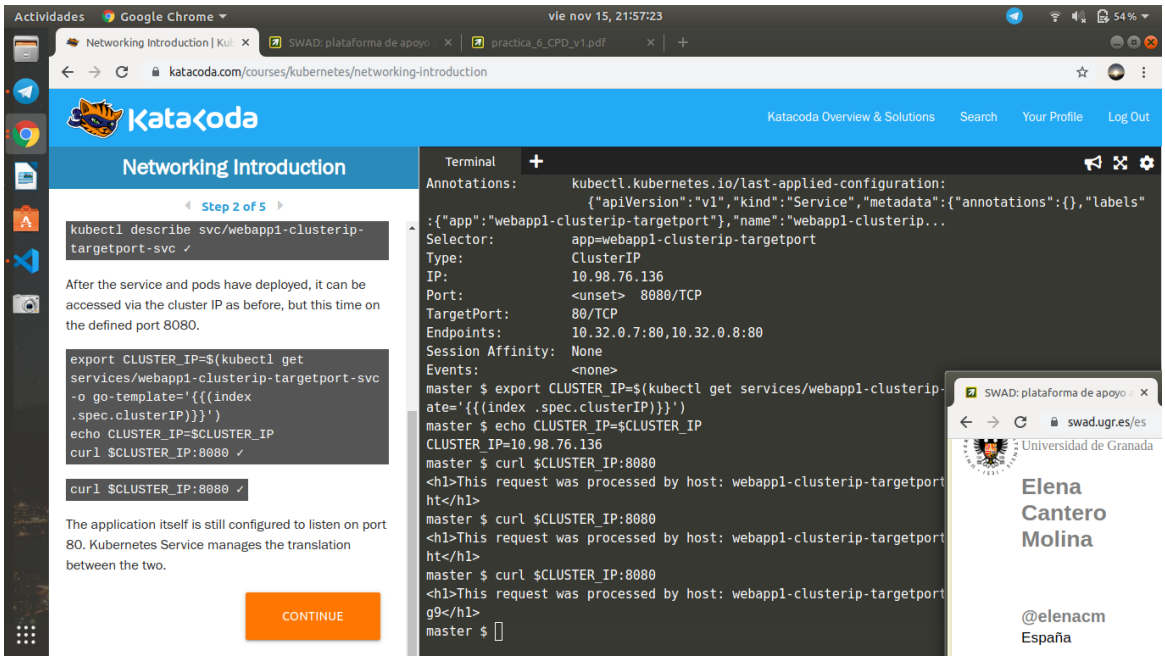
II.d) Lanzamos “Kubernetes - Networking Introduction”

Cluster IP

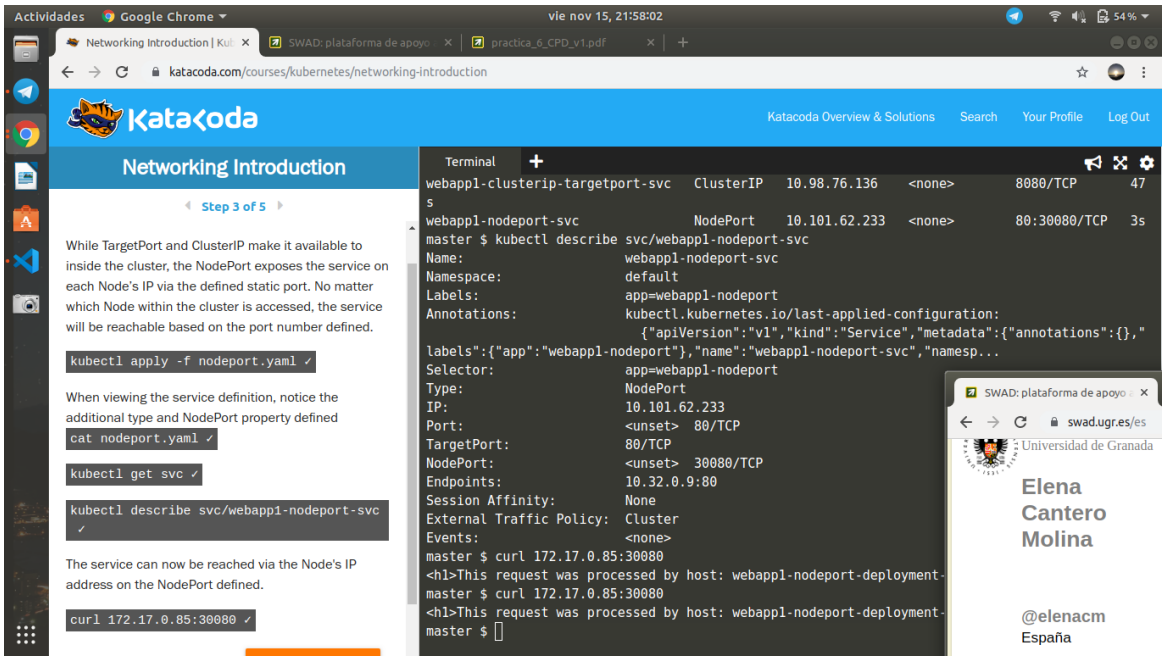




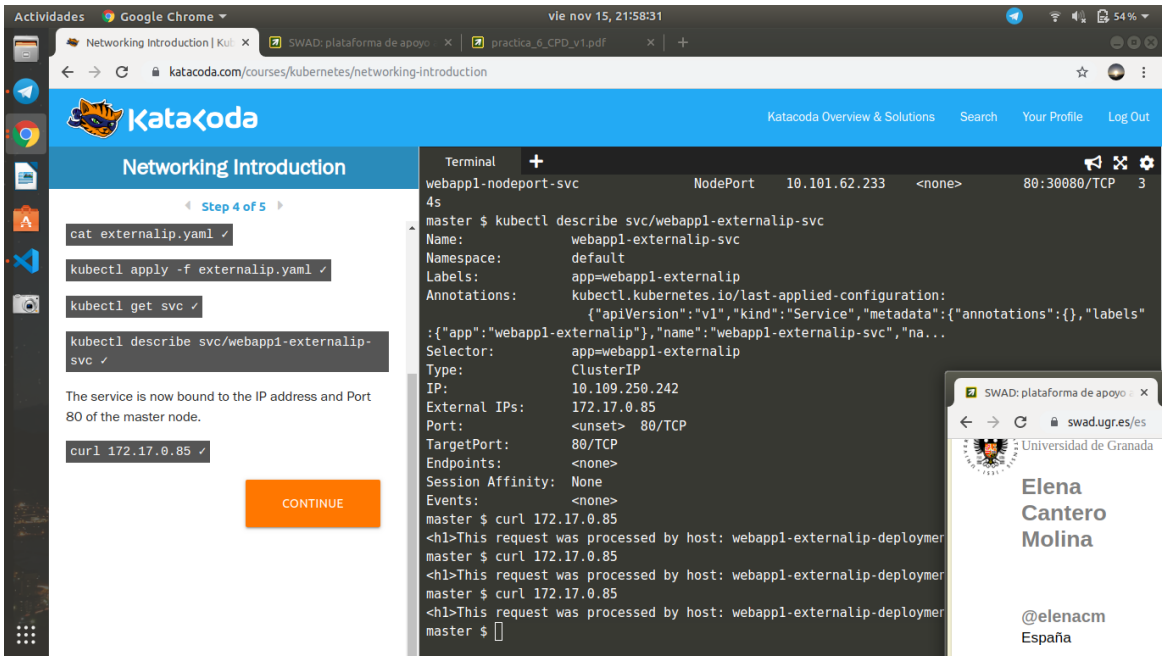
Target Ports



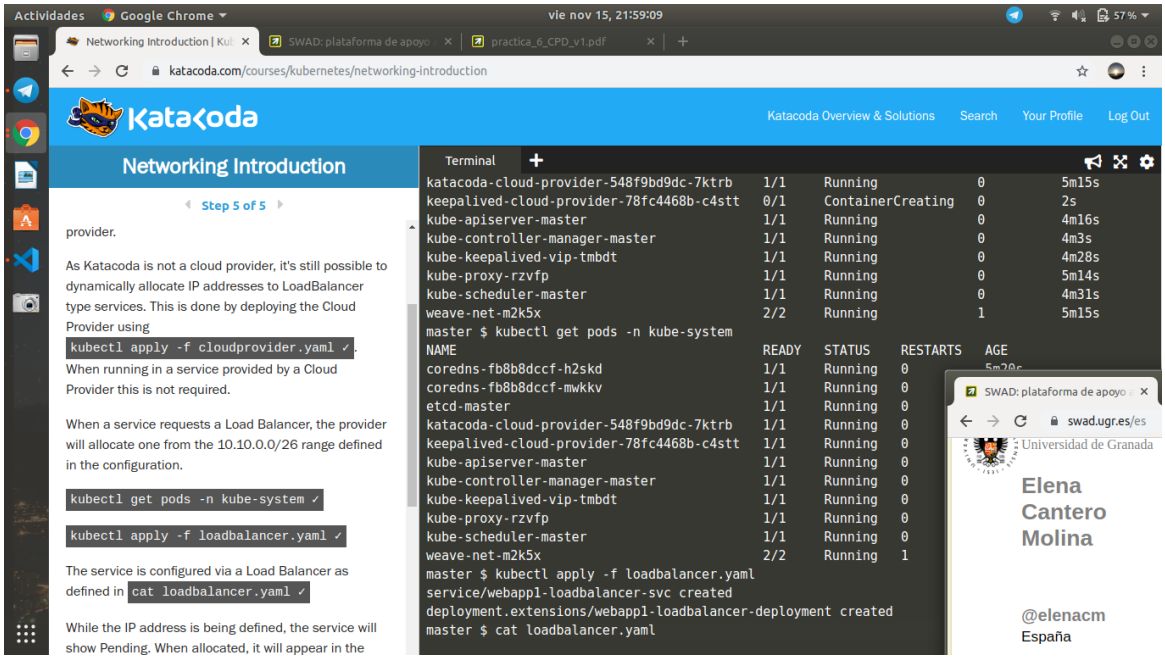
NodePort

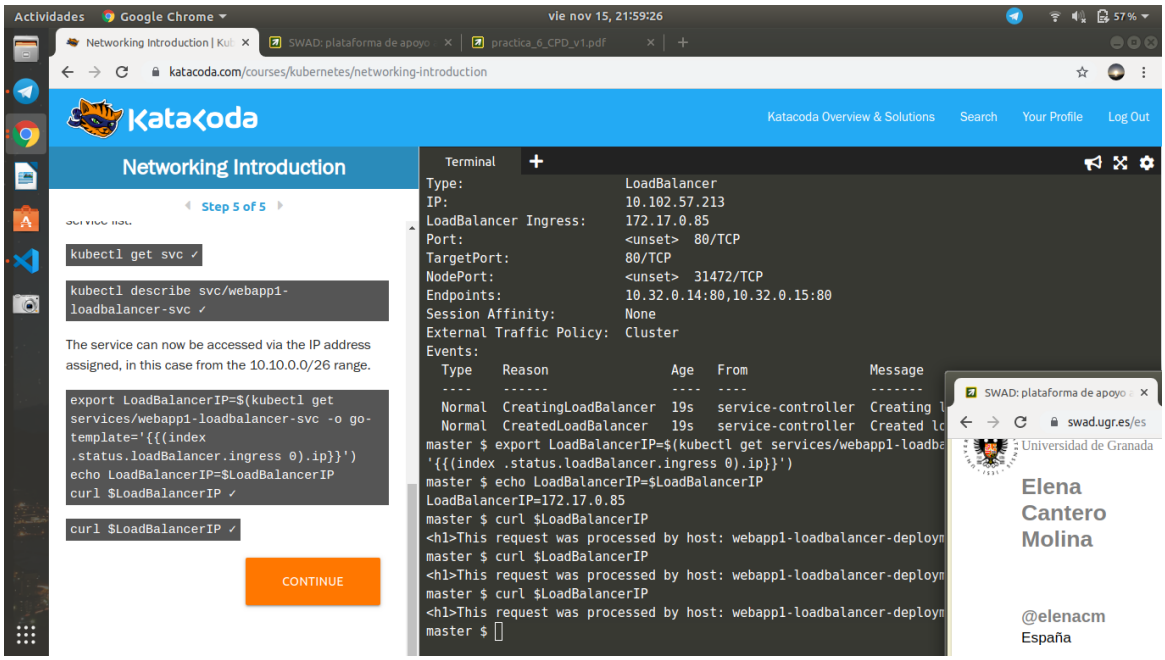


External IPs

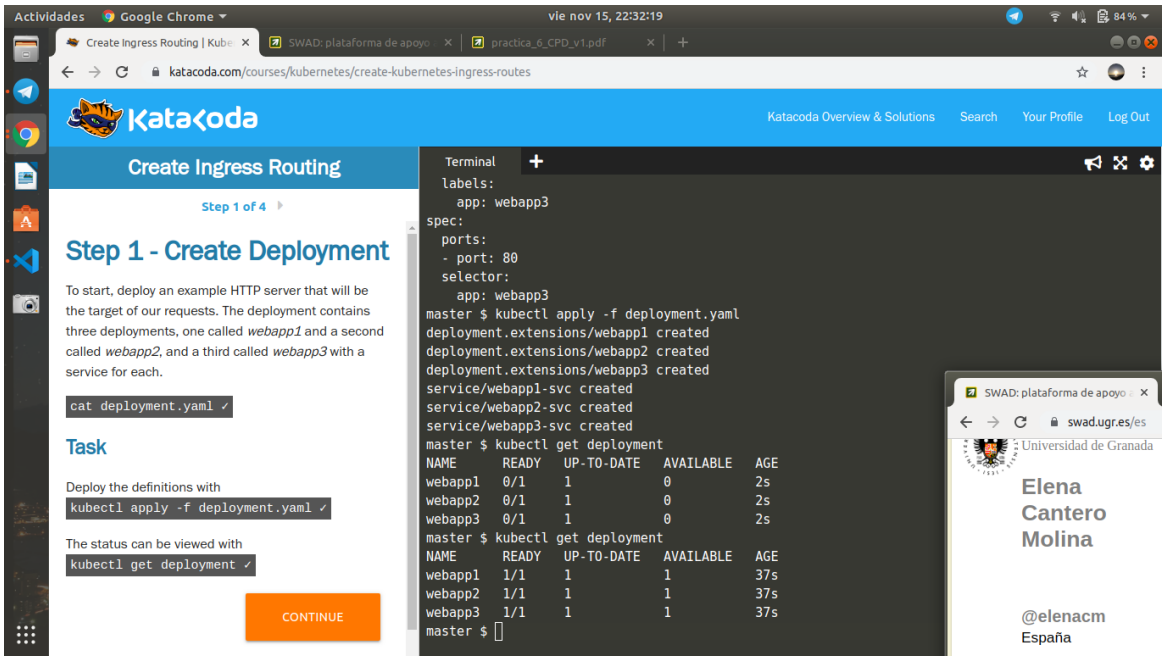


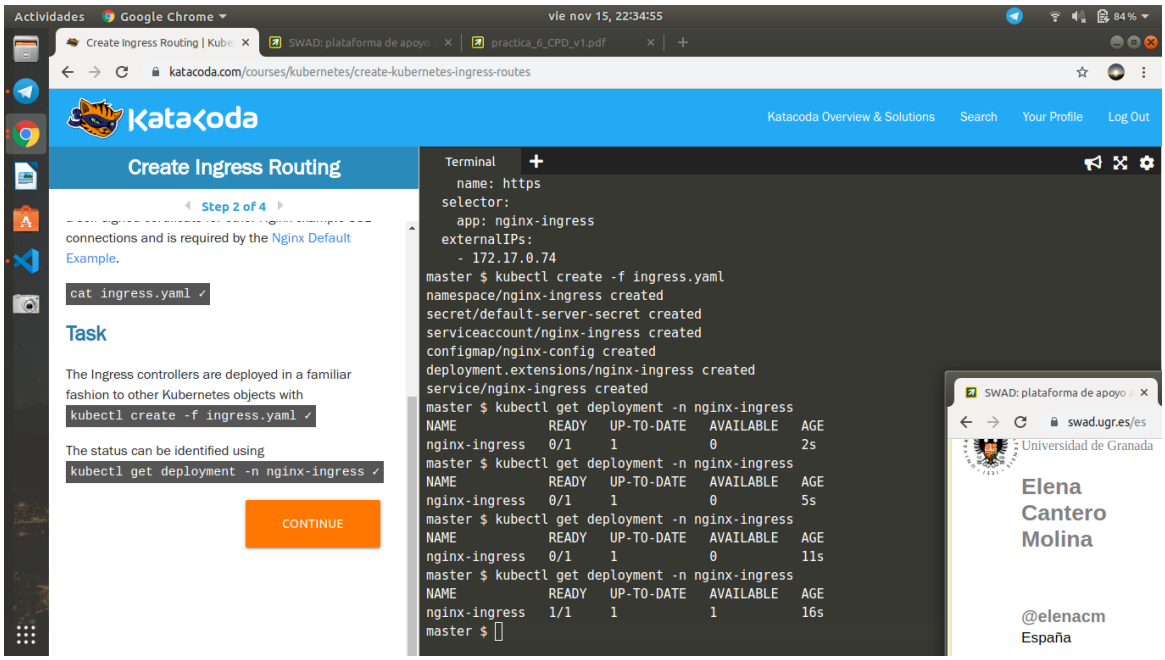
Load Balancer

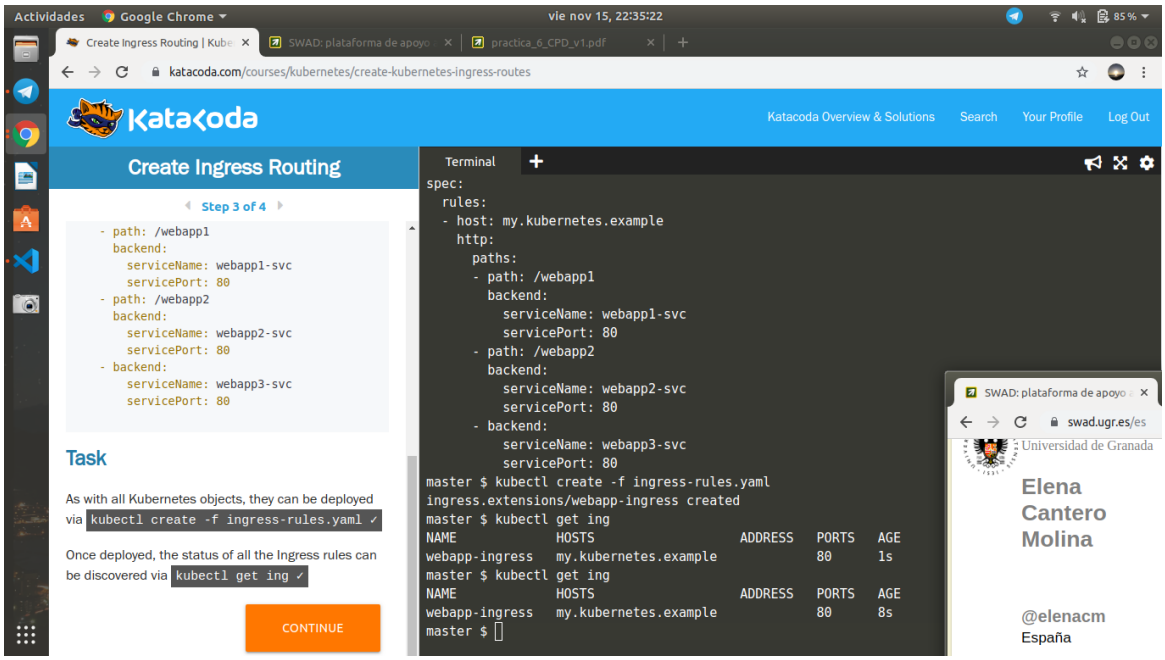


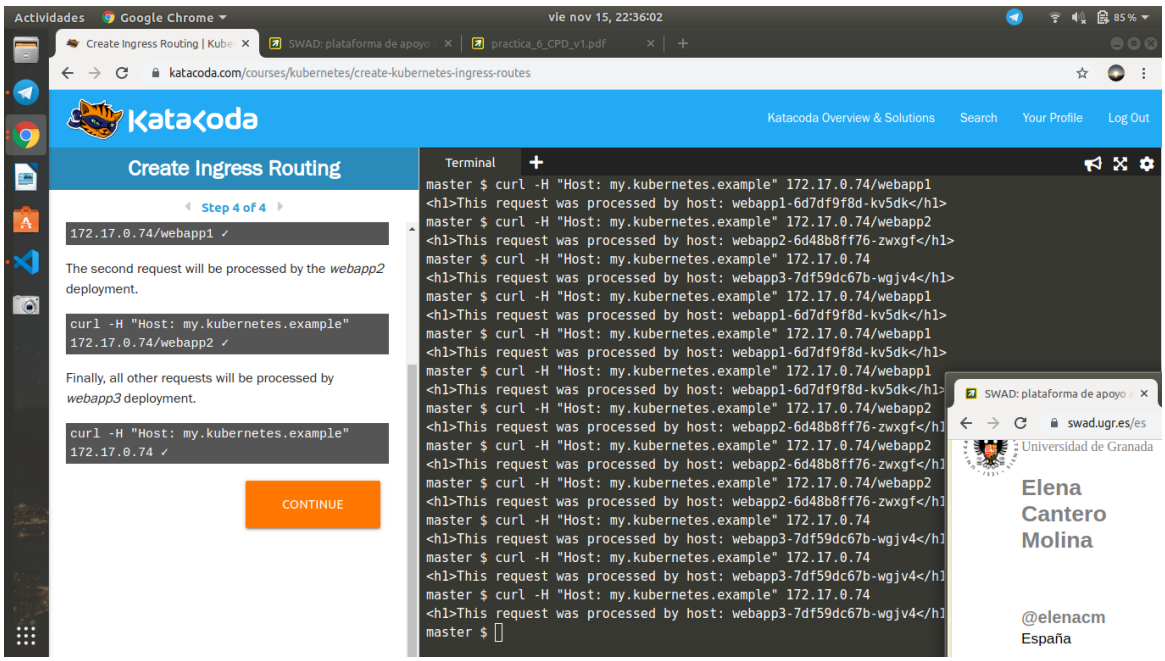


II.e) Lanzamos “Create Ingress Routing”



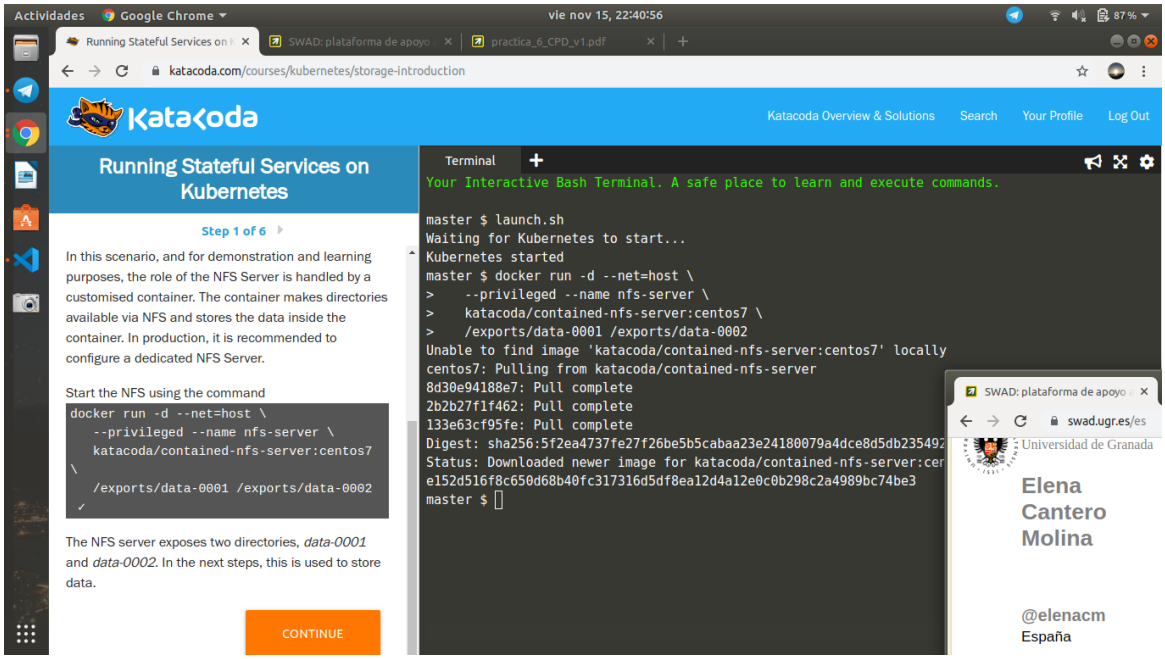

Actividades
Google Chrome
vie nov 15, 22:34:55
Create Ingress Routing | Kube
SWAD: plataforma de apoyo
practica_6_CPD_v1.pdf
katacoda.com/courses/kubernetes/create-kubernetes-ingress-routes
Katacoda
Katacoda Overview & Solutions
Search
Your Profile
Log Out
Create Ingress Routing
Step 2 of 4
connections and is required by the Nginx Default Example.
cat ingress.yaml ✓
Task
The Ingress controllers are deployed in a familiar fashion to other Kubernetes objects with
kubectl create -f ingress.yaml ✓
The status can be identified using
kubectl get deployment -n nginx-ingress ✓
CONTINUE
Terminal
name: https
selector:
app: nginx-ingress
externalIPs:
- 172.17.0.74
master $ kubectl create -f ingress.yaml
namespace/nginx-ingress created
secret/default-server-secret created
serviceaccount/nginx-ingress created
configmap/nginx-config created
deployment.extensions/nginx-ingress created
service/nginx-ingress created
master $ kubectl get deployment -n nginx-ingress
NAME READY UP-TO-DATE AVAILABLE AGE
nginx-ingress 0/1 1 0 2s
master $ kubectl get deployment -n nginx-ingress
NAME READY UP-TO-DATE AVAILABLE AGE
nginx-ingress 0/1 1 0 5s
master $ kubectl get deployment -n nginx-ingress
NAME READY UP-TO-DATE AVAILABLE AGE
nginx-ingress 0/1 1 0 11s
master $ kubectl get deployment -n nginx-ingress
NAME READY UP-TO-DATE AVAILABLE AGE
nginx-ingress 1/1 1 1 16s
master $
SWAD: plataforma de apoyo
swad.ugr.es/es
Universidad de Granada
Elena Cantero Molina
@elenacm
España


Actividades
Google Chrome
vie nov 15, 22:35:22
Create Ingress Routing | Kube
SWAD: plataforma de apoyo
practica_6_CPD_v1.pdf
katacoda.com/courses/kubernetes/create-kubernetes-ingress-routes
Katacoda
Katacoda Overview & Solutions
Search
Your Profile
Log Out
Create Ingress Routing
Step 3 of 4
- path: /webapp1
backend:
serviceName: webapp1-svc
servicePort: 80
- path: /webapp2
backend:
serviceName: webapp2-svc
servicePort: 80
- backend:
serviceName: webapp3-svc
servicePort: 80
Task
As with all Kubernetes objects, they can be deployed via
kubectl create -f ingress-rules.yaml ✓
Once deployed, the status of all the Ingress rules can be discovered via
kubectl get ing ✓
CONTINUE
Terminal
spec:
rules:
- host: my.kubernetes.example
http:
paths:
- path: /webapp1
backend:
serviceName: webapp1-svc
servicePort: 80
- path: /webapp2
backend:
serviceName: webapp2-svc
servicePort: 80
- backend:
serviceName: webapp3-svc
servicePort: 80
master $ kubectl create -f ingress-rules.yaml
ingress.extensions/webapp-ingress created
master $ kubectl get ing
NAME HOSTS ADDRESS PORTS AGE
webapp-ingress my.kubernetes.example 80 1s
master $ kubectl get ing
NAME HOSTS ADDRESS PORTS AGE
webapp-ingress my.kubernetes.example 80 8s
master $
SWAD: plataforma de apoyo
swad.ugr.es/es
Universidad de Granada
Elena Cantero Molina
@elenacm
España

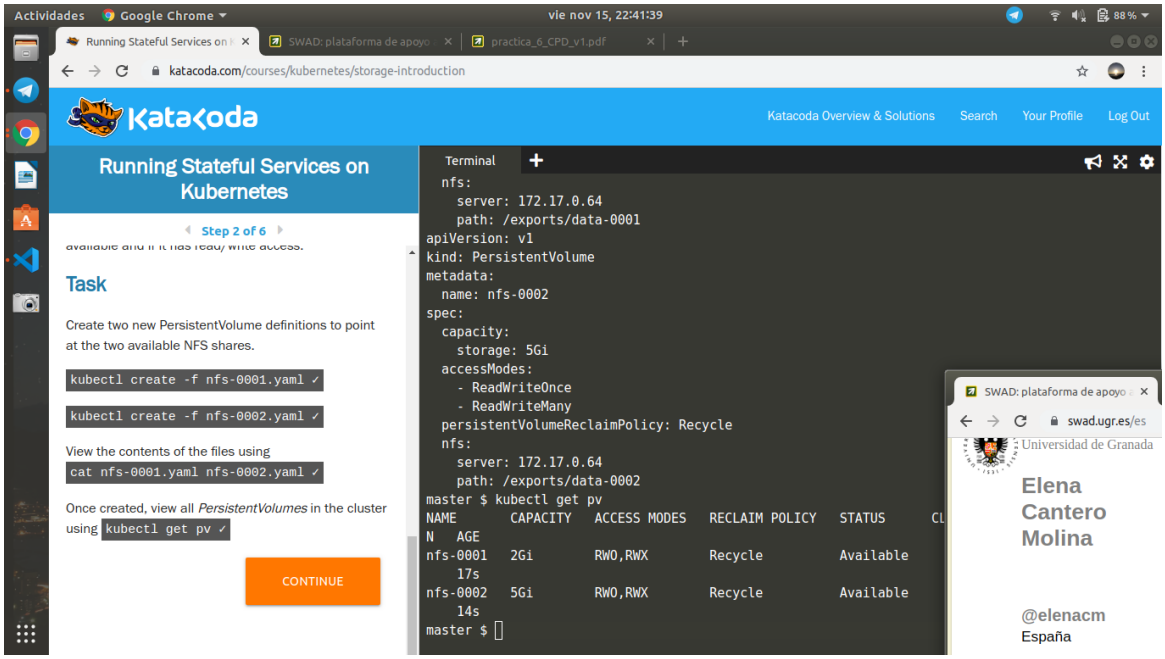

Actividades
Google Chrome
vie nov 15, 22:36:02
Create Ingress Routing | Kube
SWAD: plataforma de apoyo
practica_6_CPD_v1.pdf
katacoda.com/courses/kubernetes/create-kubernetes-ingress-routes
Katacoda
Katacoda Overview & Solutions
Search
Your Profile
Log Out
Create Ingress Routing
Step 4 of 4
172.17.0.74/webapp1 ✓
The second request will be processed by the webapp2 deployment.
curl -H "Host: my.kubernetes.example" 172.17.0.74/webapp2 ✓
Finally, all other requests will be processed by webapp3 deployment.
curl -H "Host: my.kubernetes.example" 172.17.0.74 ✓
CONTINUE
Terminal
master $ curl -H "Host: my.kubernetes.example" 172.17.0.74/webapp1
<h1>This request was processed by host: webapp1-6d7df9f8d-kv5dk</h1>
master $ curl -H "Host: my.kubernetes.example" 172.17.0.74/webapp2
<h1>This request was processed by host: webapp2-6d48b8ff76-zwxgf</h1>
master $ curl -H "Host: my.kubernetes.example" 172.17.0.74
<h1>This request was processed by host: webapp3-7df59dc67b-wgjv4</h1>
master $ curl -H "Host: my.kubernetes.example" 172.17.0.74/webapp1
<h1>This request was processed by host: webapp1-6d7df9f8d-kv5dk</h1>
master $ curl -H "Host: my.kubernetes.example" 172.17.0.74/webapp1
<h1>This request was processed by host: webapp1-6d7df9f8d-kv5dk</h1>
master $ curl -H "Host: my.kubernetes.example" 172.17.0.74/webapp1
<h1>This request was processed by host: webapp1-6d7df9f8d-kv5dk</h1>
master $ curl -H "Host: my.kubernetes.example" 172.17.0.74/webapp2
<h1>This request was processed by host: webapp2-6d48b8ff76-zwxgf</h1>
master $ curl -H "Host: my.kubernetes.example" 172.17.0.74/webapp2
<h1>This request was processed by host: webapp2-6d48b8ff76-zwxgf</h1>
master $ curl -H "Host: my.kubernetes.example" 172.17.0.74/webapp2
<h1>This request was processed by host: webapp2-6d48b8ff76-zwxgf</h1>
master $ curl -H "Host: my.kubernetes.example" 172.17.0.74
<h1>This request was processed by host: webapp3-7df59dc67b-wgjv4</h1>
master $ curl -H "Host: my.kubernetes.example" 172.17.0.74
<h1>This request was processed by host: webapp3-7df59dc67b-wgjv4</h1>
master $ curl -H "Host: my.kubernetes.example" 172.17.0.74
<h1>This request was processed by host: webapp3-7df59dc67b-wgjv4</h1>
master $
SWAD: plataforma de apoyo
swad.ugr.es/es
Universidad de Granada
Elena Cantero Molina
@elenacm
España

II.f) Lanzamos “Running Stateful Services on Kubernetes”

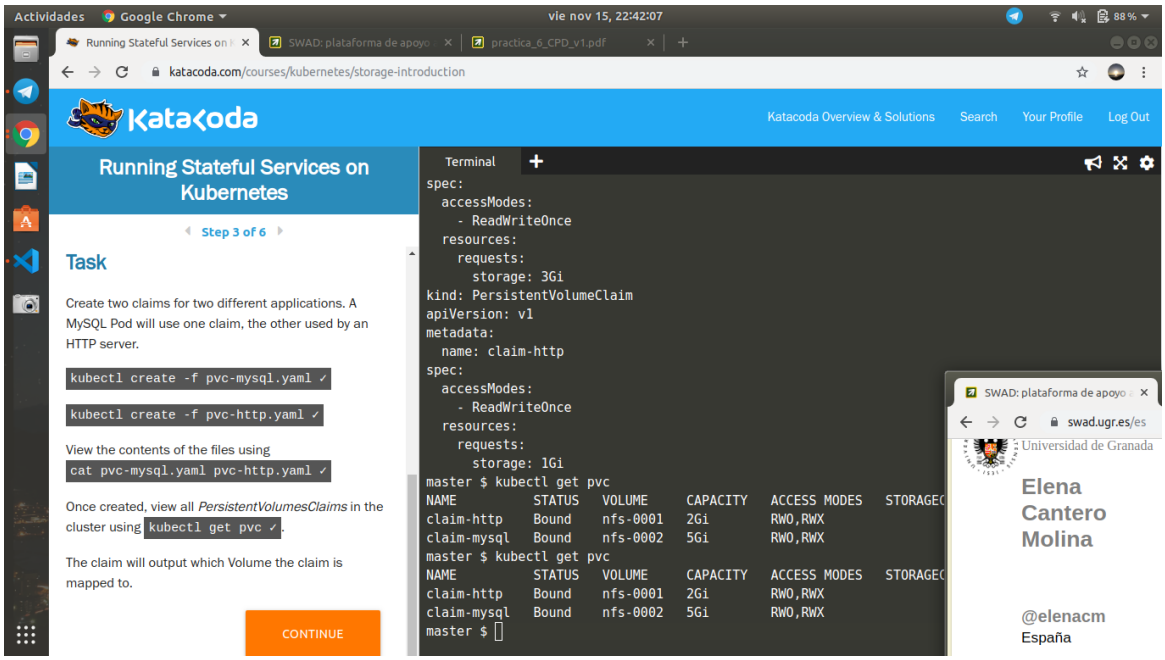
Activamos un servidor NFS.



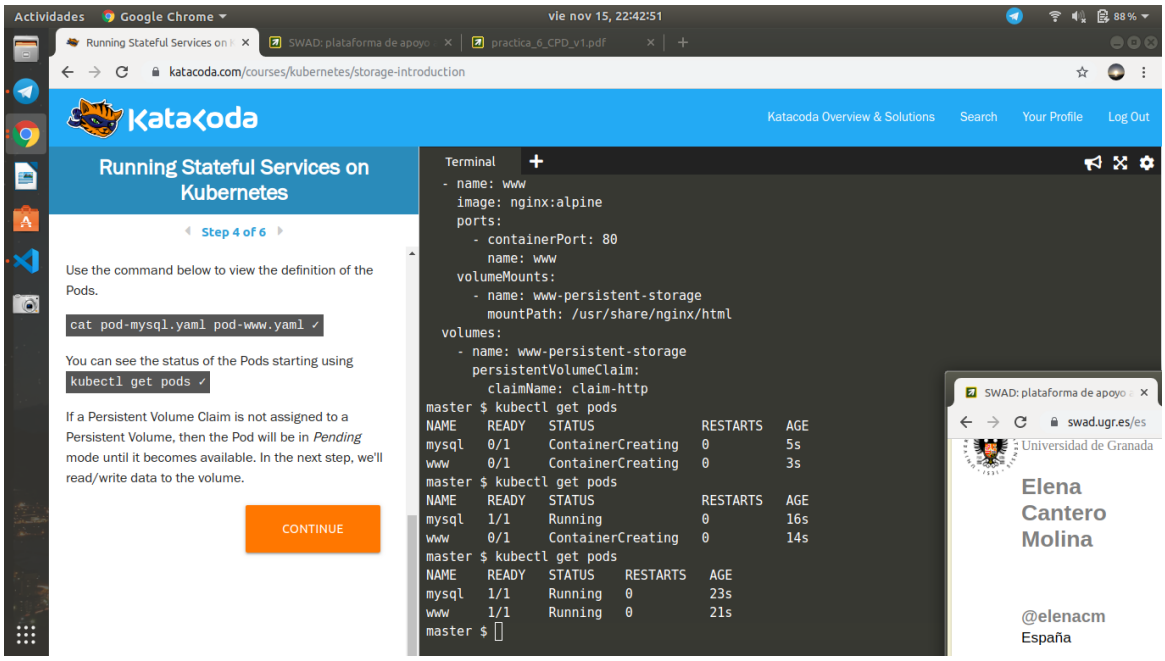
Desplegamos los volúmenes persistentes (PV)



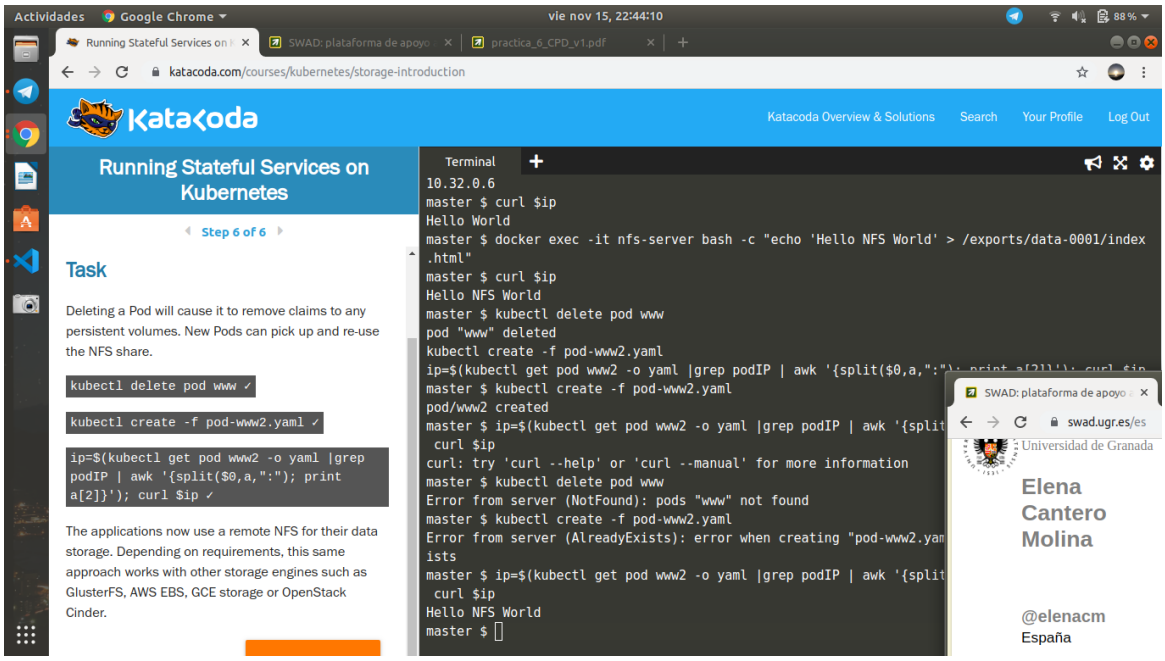
Desplegamos los recursos que demandan volúmenes persistentes (PVC)



Utilizamos los volúmenes

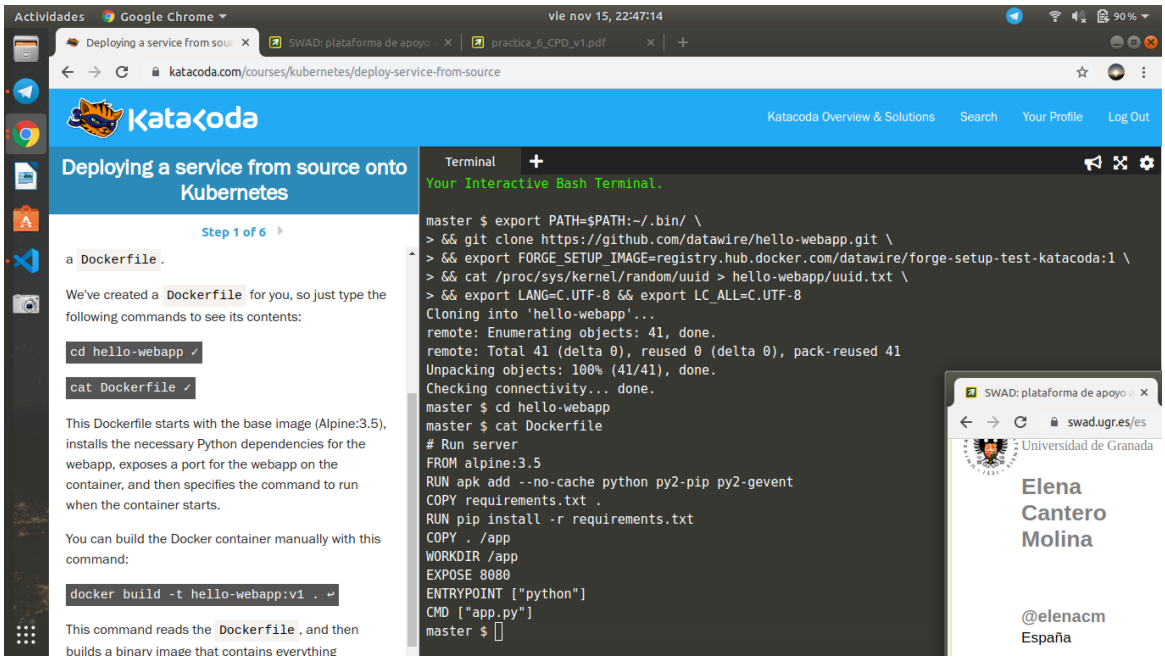


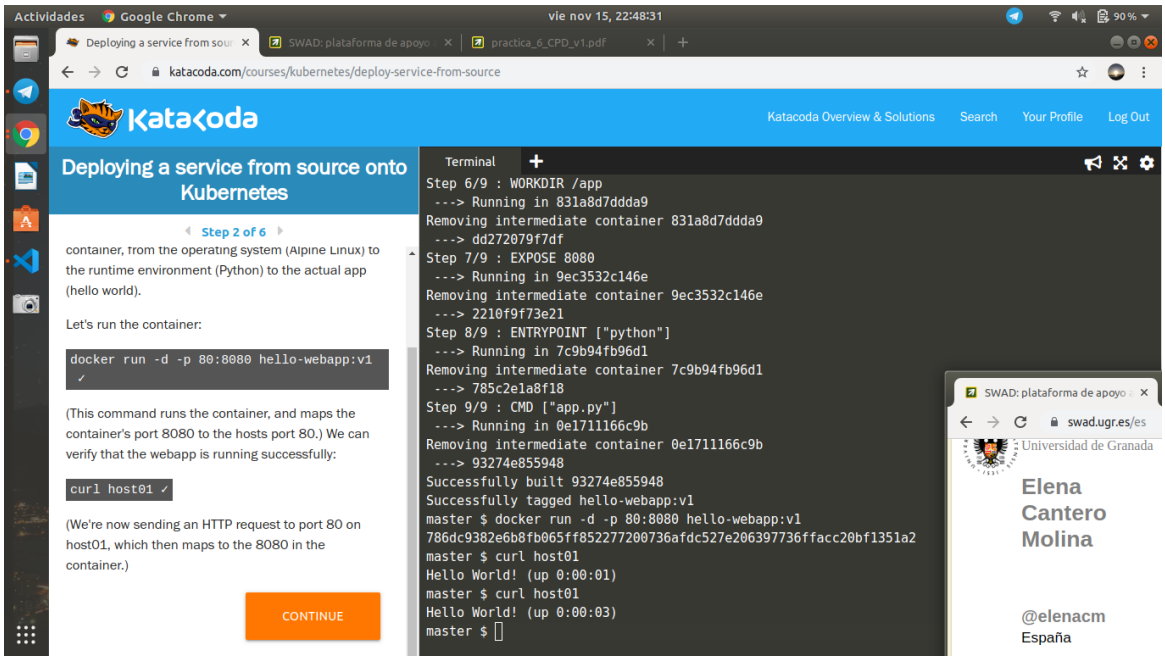
Borramos y reactivamos otro Pod para verificar el almacenamiento persistente.



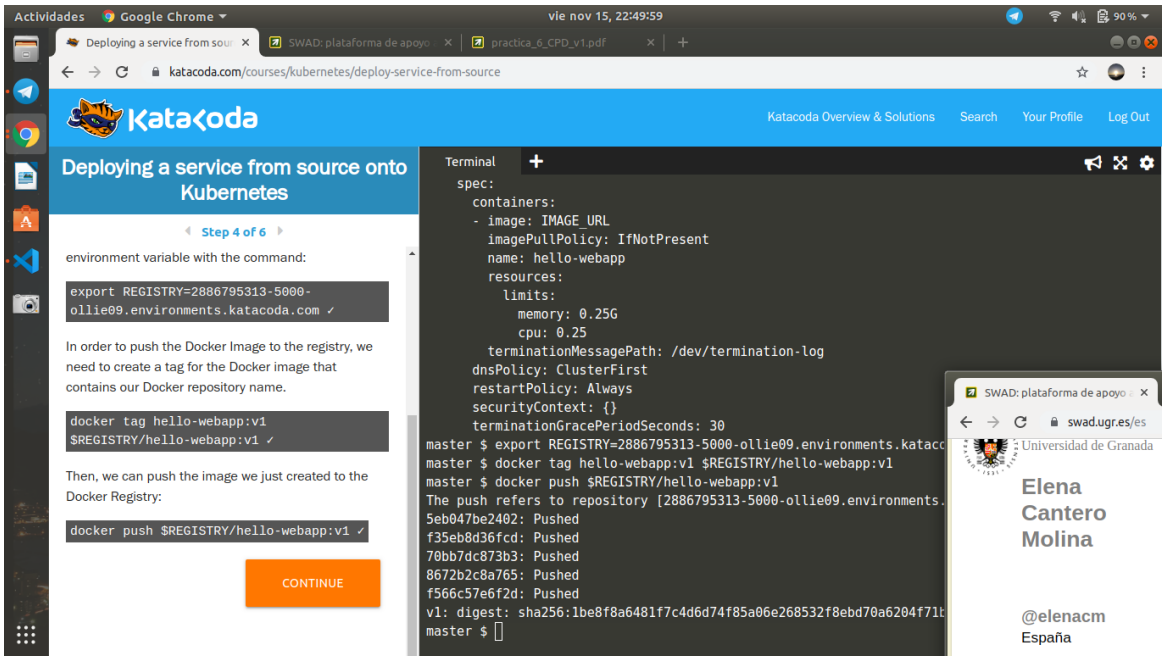
II.g) Lanzamos “Deploying a service from source onto Kubernetes”

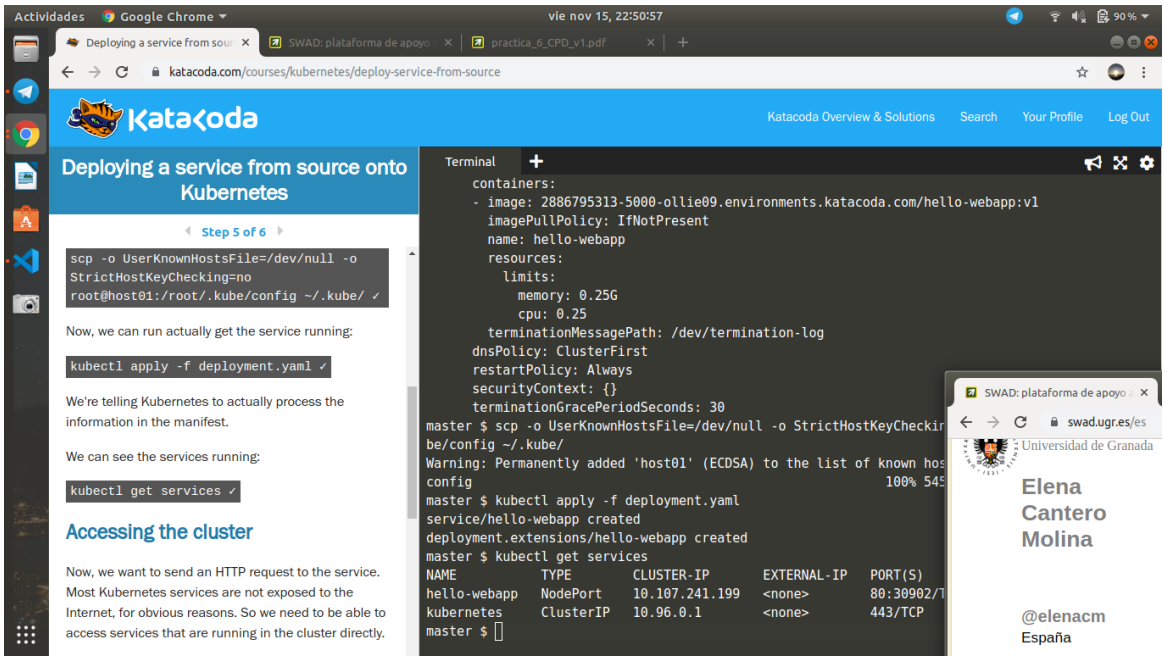
Creamos un contenedor Docker y a lanzarlo dentro de Kubernetes, definiendo límites de uso de CPU y memoria.

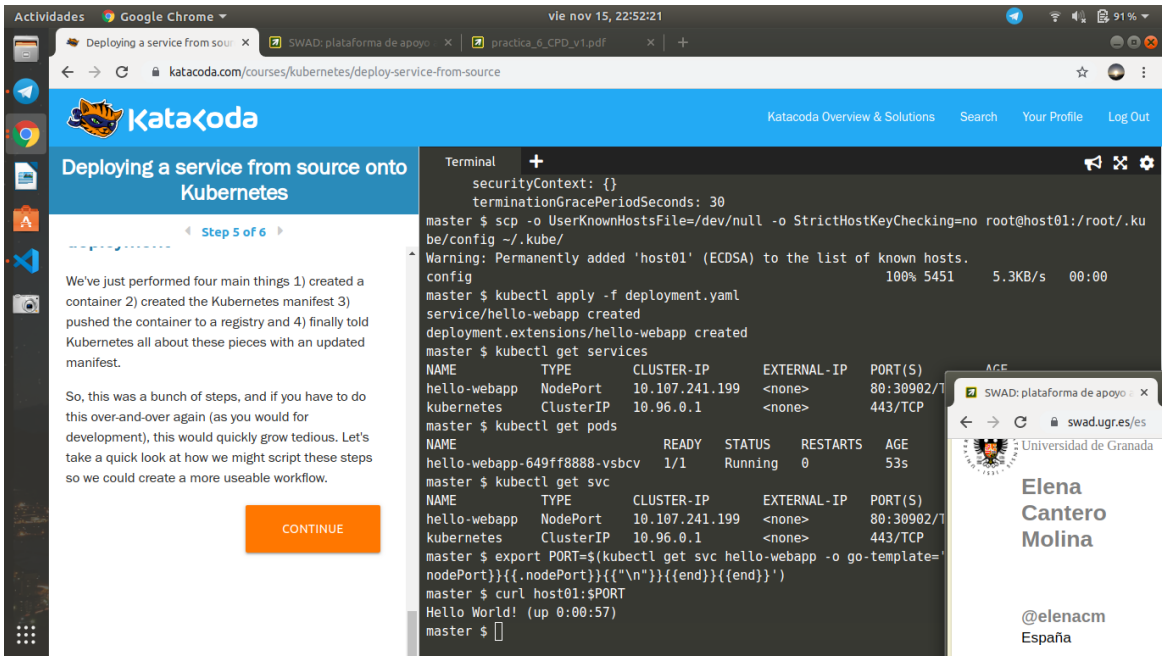




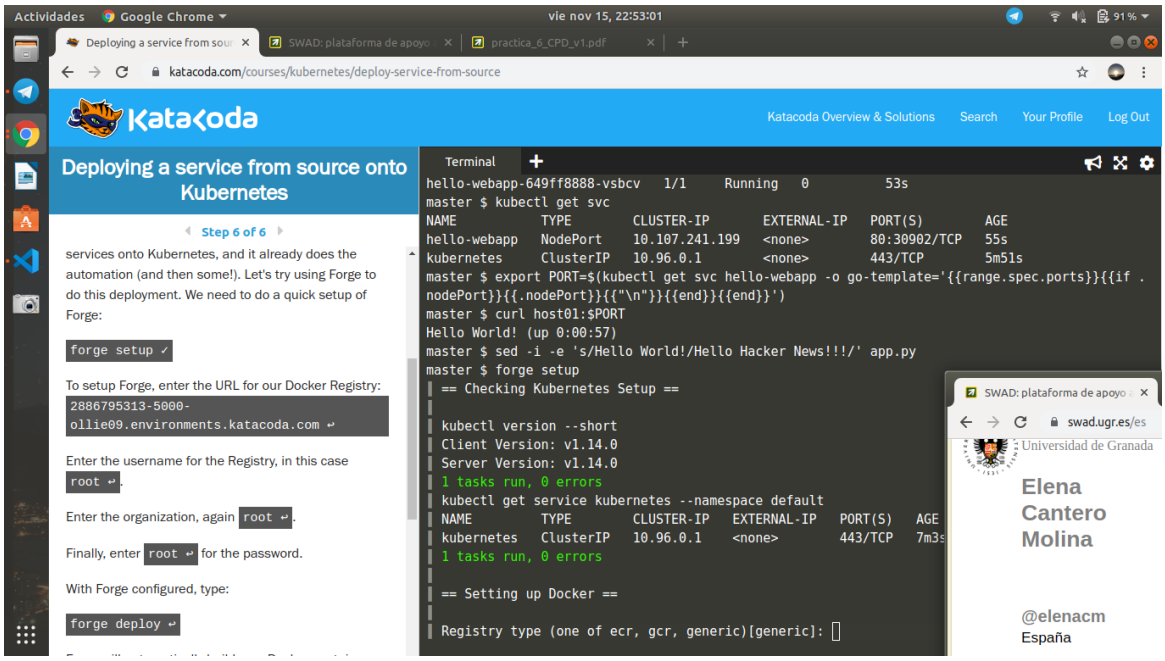
Utilizamos un registro de contenedores.

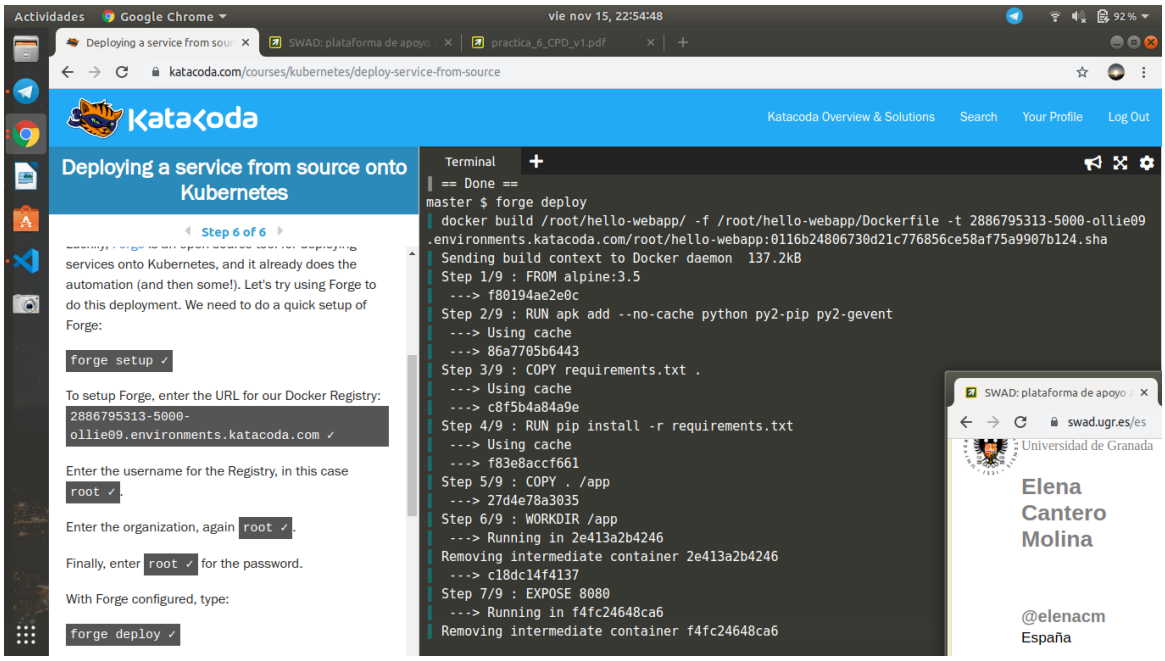


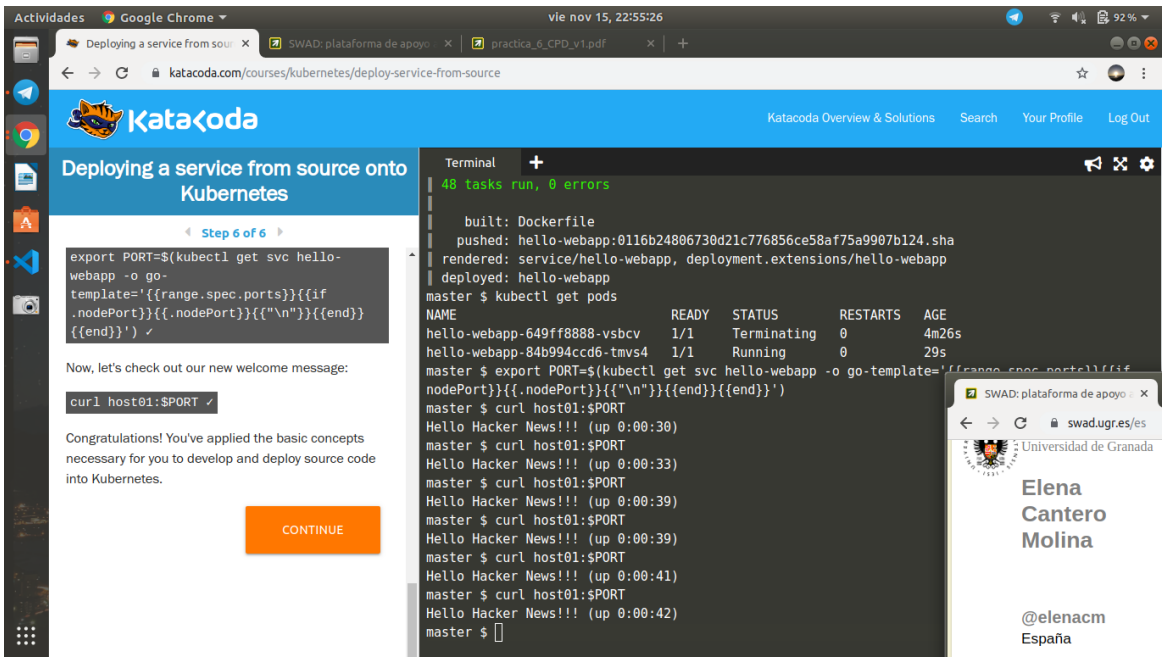




Utilizar forge para automatizar el lanzamiento.


Actividades
Google Chrome
vie nov 15, 22:53:01
katacoda.com/courses/kubernetes/deploy-service-from-source
Katacoda
Katacoda Overview & Solutions
Search
Your Profile
Log Out
Deploying a service from source onto Kubernetes
Step 6 of 6
services onto Kubernetes, and it already does the automation (and then some!). Let's try using Forge to do this deployment. We need to do a quick setup of Forge:
forge setup ↵
To setup Forge, enter the URL for our Docker Registry:
2886795313-5000-ollie09.environments.katacoda.com ↵
Enter the username for the Registry, in this case root ↵.
Enter the organization, again root ↵.
Finally, enter root ↵ for the password.
With Forge configured, type:
forge deploy ↵
Terminal
hello-webapp-649ff8888-vsbcv 1/1 Running 0 53s
master $ kubectl get svc
NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE
hello-webapp NodePort 10.107.241.199 <none> 80:30902/TCP 55s
kubernetes ClusterIP 10.96.0.1 <none> 443/TCP 5m51s
master $ export PORT=$(kubectl get svc hello-webapp -o go-template='{{range.spec.ports}}{{if .nodePort}}{{.nodePort}}{{"\n"}}{{end}}{{end}}')
master $ curl host01:$PORT
Hello World! (up 0:00:57)
master $ sed -i -e 's/Hello World!/Hello Hacker News!!!/' app.py
master $ forge setup
== Checking Kubernetes Setup ==
kubectl version --short
Client Version: v1.14.0
Server Version: v1.14.0
1 tasks run, 0 errors
kubectl get service kubernetes --namespace default
NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE
kubernetes ClusterIP 10.96.0.1 <none> 443/TCP 7m3s
1 tasks run, 0 errors
== Setting up Docker ==
Registry type (one of ecr, gcr, generic)[generic]:
SWAD: plataforma de apoyo
swad.ugr.es/es
Universidad de Granada
Elena Cantero Molina
@elenacm
España


Actividades
Google Chrome
vie nov 15, 22:54:48
katacoda.com/courses/kubernetes/deploy-service-from-source
Katacoda
Katacoda Overview & Solutions
Search
Your Profile
Log Out
Deploying a service from source onto Kubernetes
Step 6 of 6
services onto Kubernetes, and it already does the automation (and then some!). Let's try using Forge to do this deployment. We need to do a quick setup of Forge:
forge setup ✓
To setup Forge, enter the URL for our Docker Registry:
2886795313-5000-ollie09.environments.katacoda.com ✓
Enter the username for the Registry, in this case root ✓.
Enter the organization, again root ✓.
Finally, enter root ✓ for the password.
With Forge configured, type:
forge deploy ✓
Terminal
== Done ==
master $ forge deploy
docker build /root/hello-webapp/ -f /root/hello-webapp/Dockerfile -t 2886795313-5000-ollie09.environments.katacoda.com/root/hello-webapp:0116b24806730d21c776856ce58af75a9907b124.sha
Sending build context to Docker daemon 137.2kB
Step 1/9 : FROM alpine:3.5
---> f80194ae2e0c
Step 2/9 : RUN apk add --no-cache python py2-pip py2-gevent
---> Using cache
---> 86a7705b6443
Step 3/9 : COPY requirements.txt .
---> Using cache
---> c8f5b4a84a9e
Step 4/9 : RUN pip install -r requirements.txt
---> Using cache
---> f83e8accf661
Step 5/9 : COPY . /app
---> 27d4e78a3035
Step 6/9 : WORKDIR /app
---> Running in 2e413a2b4246
Removing intermediate container 2e413a2b4246
---> c18dc14f4137
Step 7/9 : EXPOSE 8080
---> Running in f4fc24648ca6
Removing intermediate container f4fc24648ca6
SWAD: plataforma de apoyo
swad.ugr.es/es
Universidad de Granada
Elena Cantero Molina
@elenacm
España


Actividades
Google Chrome
vie nov 15, 22:55:26
katacoda.com/courses/kubernetes/deploy-service-from-source
Katacoda
Katacoda Overview & Solutions
Search
Your Profile
Log Out
Deploying a service from source onto Kubernetes
Step 6 of 6
export PORT=$(kubectl get svc hello-webapp -o go-template='{{range.spec.ports}}{{if .nodePort}}{{.nodePort}}{{"\n"}}{{end}}{{end}}') ✓
Now, let's check out our new welcome message:
curl host01:$PORT ✓
Congratulations! You've applied the basic concepts necessary for you to develop and deploy source code into Kubernetes.
CONTINUE
Terminal
48 tasks run, 0 errors
built: Dockerfile
pushed: hello-webapp:0116b24806730d21c776856ce58af75a9907b124.sha
rendered: service/hello-webapp, deployment.extensions/hello-webapp
deployed: hello-webapp
master $ kubectl get pods
NAME READY STATUS RESTARTS AGE
hello-webapp-649ff8888-vsbcv 1/1 Terminating 0 4m26s
hello-webapp-84b994ccd6-tmvs4 1/1 Running 0 29s
master $ export PORT=$(kubectl get svc hello-webapp -o go-template='{{range.spec.ports}}{{if .nodePort}}{{.nodePort}}{{"\n"}}{{end}}{{end}}')
master $ curl host01:$PORT
Hello Hacker News!!! (up 0:00:30)
master $ curl host01:$PORT
Hello Hacker News!!! (up 0:00:33)
master $ curl host01:$PORT
Hello Hacker News!!! (up 0:00:39)
master $ curl host01:$PORT
Hello Hacker News!!! (up 0:00:39)
master $ curl host01:$PORT
Hello Hacker News!!! (up 0:00:41)
master $ curl host01:$PORT
Hello Hacker News!!! (up 0:00:42)
master $
SWAD: plataforma de apoyo
swad.ugr.es/es
Universidad de Granada
Elena Cantero Molina
@elenacm
España