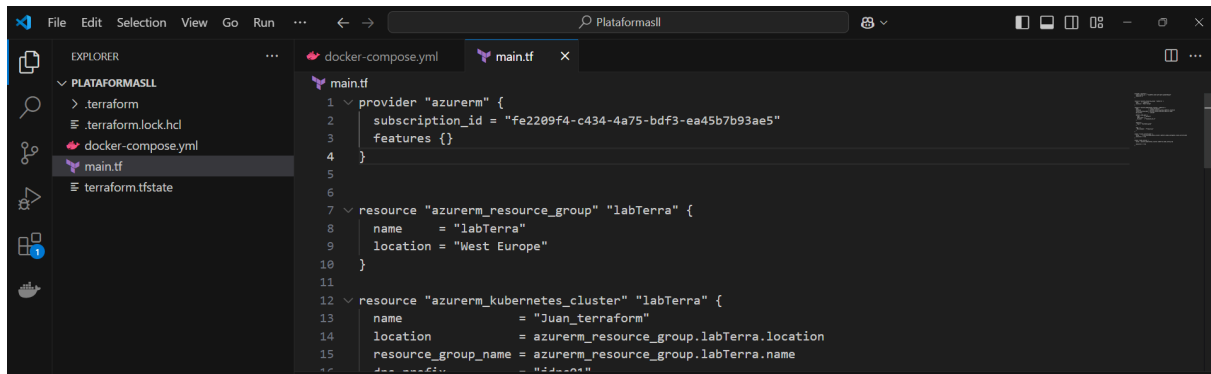


# Terraform Workshop

First of all we need to create an archive to explain everything from our cluster



There are some steps that we need to know before doing the cluster into azure

- We need to have an account with suscription or have credits into azure
- We need to have installed Terraform
- We need to have installed Azure CLI

So whe we want to deploy the cluster we use the next comand lines and specific things like:

In the start of the document put the provider and the suscription id from our account

```
provider "azurerm" {
  subscription_id = "fe2209f4-c434-4a75-bdf3-ea45b7b93ae5"
  features {}
}
```

We use this commands to deploy the cluster

```
terraform init
terraform validate
terraform plan
terraform apply
```

Here we verify that our service is on azure

The screenshot shows the Azure portal interface for a Kubernetes service named 'Juan\_terraform'. The left sidebar contains navigation options like 'Información general', 'Registro de actividad', 'Control de acceso (IAM)', 'Etiquetas', 'Supervisar', 'Diagnosticar y solucionar problemas', 'Microsoft Defender for Cloud', 'Análisis de costos', 'Recursos de Kubernetes', 'Configuración', 'Supervisión', 'Automation', and 'Ayuda'. The main content area is divided into 'Essentials' and 'Propiedades' tabs. The 'Essentials' tab shows key metrics: 'Grupo de recursos' (labTerra), 'Estado de energía' (Running), 'Estado de la operación' (Succeeded), 'Suscripción' (Azure for Students), 'Ubicación' (West Europe), 'Id. de suscripción' (fe2209f4-c434-4a75-bdf3-ea45b7b93ae5), and 'Etiquetas' (Environment: Production). The 'Propiedades' tab shows 'Servicios de Kubernetes' (Tipo de cifrado: Cifrado en reposo con una clave administrada por la plataforma, Grupos de nodos virtuales: No habilitado) and 'Grupos de nodos' (Grupos de nodos: 1 grupo de nodos, Versiones de Kubernetes: 1.30.9). The 'Redes' section shows network configuration details like 'Dirección del servidor de API' (jdr01-o8nyo1a6.hcp.westeurope.azmk8s.io), 'Configuración de red' (Superposición de Azure CNI), 'CIDR de pod' (10.244.0.0/16), 'CIDR del servicio' (10.0.0.0/16), 'Dirección IP del servicio DNS' (10.0.0.10), 'Plano de datos de Cilium' (No habilitado), and 'Directiva de red' (Ninguno).

Now we switch the context

```
juand@Huawei-Laptop:~/terraform$ kubectl config use-context Juan_terraform
Switched to context "Juan_terraform".
```

Now we create the nginx pod

```
juand@Huawei-Laptop:~/terraform$ kubectl run nginx-pod --image=nginx --restart=Never
pod/nginx-pod created
juand@Huawei-Laptop:~/terraform$ kubectl get pods
NAME          READY   STATUS    RESTARTS   AGE
nginx-pod     1/1     Running   0           9s
```

Now we change the type to obtain a port to search in http

```

juand@Huawei-Laptop:~/terraform$ kubectl get svc
NAME         TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)    AGE
kubernetes   ClusterIP   10.0.0.1     <none>        443/TCP    100m
nginx-pod    ClusterIP   10.0.93.203  <none>        80/TCP     114s
juand@Huawei-Laptop:~/terraform$ kubectl describe svc nginx-pod
Name:         nginx-pod
Namespace:    default
Labels:       run=nginx-pod
Annotations:   <none>
Selector:     run=nginx-pod
Type:         ClusterIP
IP Family Policy: SingleStack
IP Families:   IPv4
IP:            10.0.93.203
IPs:           10.0.93.203
Port:         <unset> 80/TCP
TargetPort:    80/TCP
Endpoints:     10.244.0.136:80
Session Affinity: None
Internal Traffic Policy: Cluster
Events:        <none>
juand@Huawei-Laptop:~/terraform$ kubectl get svc nginx-pod -o=jsonpath='{.spec.ports[0].nodePort}'
juand@Huawei-Laptop:~/terraform$ kubectl patch svc nginx-pod -p '{"spec": {"type": "NodePort"}}'
service/nginx-pod patched
juand@Huawei-Laptop:~/terraform$ kubectl get svc nginx-pod
NAME         TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)          AGE
nginx-pod    NodePort     10.0.93.203  <none>        80:30303/TCP     3m10s
juand@Huawei-Laptop:~/terraform$ kubectl get svc nginx-pod -o=jsonpath='{.spec.ports[0].nodePort}'
30303juand@Huawei-Laptop:~/terraform$ minikube ip

```

```

port unable to listen on any of the requested ports! [10000-80]
juand@Huawei-Laptop:~/terraform$ curl http://localhost:8080
C
juand@Huawei-Laptop:~/terraform$ kubectl get svc nginx-pod -o=jsonpath='{.spec.ports[0].nodePort}'
30303juand@Huawei-Laptop:~/terraform$ kubectl proxy --address=0.0.0.0 --port=8001 --accept-hosts='.*'
Starting to serve on [::]:8001

```

Here we:

- Switch kubectl to cluster context
- create nginx pod
- create service and on this amount the nginx so that port 80 is seen
- download lenz in window

Local Kubeconfigs

Events

Helm

Access Control

Custom Resources

WSL Cluster

Overview

Applications

Nodes

Workloads

Overview

Pods

Deployments

Daemon Sets

Stateful Sets

Replica Sets

Replication Controllers

Jobs

Cron Jobs

Config

Network

Services

Endpoints

Ingresses

Ingress Classes

Network Policies

Port Forwarding

Storage

Namespaces

Events

default

Pods (1)

Running: 1

Deployments (0)

Daemon Sets (0)

Stateful Sets (0)

Replica Sets (0)

Jobs (0)

Cron Jobs (0)

default

As Search Events...

19 items

Type	Message	Namespac	Involved Object	Source	Count	Age	Last Se	
Normal	Ensured load balancer	default	Service: nginx-service	service-controller	1	17m	17m	
Normal	NodePort -> LoadBalancer	default	Service: nginx-service	service-controller	1	18m	18m	
Normal	Ensuring load balancer	default	Service: nginx-service	service-controller	1	18m	18m	

Terminal x +