Santiago José Barraza Sinning Plataformas II

AKS with Terraform

```
sjbarraza@sjbarraza:~/projects/university/platforms1/terraform$ curl -sL https://aka.ms/InstallAzureCLIDeb | sudo bash
[sudo] password for sjbarraza:
Hit:1 http://archive.ubuntu.com/ubuntu jammy InRelease
Hit:2 http://security.ubuntu.com/ubuntu jammy-security InRelease
Hit:3 http://archive.ubuntu.com/ubuntu jammy-updates InRelease
Hit:4 http://archive.ubuntu.com/ubuntu jammy-backports InRelease
Hit:5 https://download.docker.com/linux/ubuntu jammy InRelease
Reading package lists... Done
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
lsb-release is already the newest version (11.1.0ubuntu4).
lsb-release set to manually installed.
ca-certificates is already the newest version (20240203~22.04.1).
curl is already the newest version (7.81.0-1ubuntu1.20).
gnupg is already the newest version (2.2.27-3ubuntu2.1).
gnupg set to manually installed.
The following NEW packages will be installed:
  apt-transport-https
0 upgraded, 1 newly installed, 0 to remove and 21 not upgraded.
Need to get 1510 B of archives.
After this operation, 170 kB of additional disk space will be used.
Get:1 http://archive.ubuntu.com/ubuntu jammy-updates/universe amd64 apt-transport-https all 2.4.13 [1510 B]
Fetched 1510 B in 0s (3569 B/s)
Selecting previously unselected package apt-transport-https.
(Reading database ... 64721 files and directories currently installed.)
Preparing to unpack .../apt-transport-https_2.4.13_all.deb ...
Unpacking apt-transport-https (2.4.13) ...
Setting up apt-transport-https (2.4.13) ...
Types: deb
URIs: https://packages.microsoft.com/repos/azure-cli/
Suites: jammy
Components: main
Architectures: amd64
Signed-by: /etc/apt/keyrings/microsoft.gpg
Hit:1 http://archive.ubuntu.com/ubuntu jammy InRelease
Hit:2 http://security.ubuntu.com/ubuntu jammy-security InRelease
Get:3 https://packages.microsoft.com/repos/azure-cli jammy InRelease [3596 B]
Hit:4 http://archive.ubuntu.com/ubuntu jammy-updates InRelease
Get:5 https://packages.microsoft.com/repos/azure-cli jammy/main all Packages [1093 B]
Hit:6 http://archive.ubuntu.com/ubuntu jammy-backports InRelease
Get:7 https://packages.microsoft.com/repos/azure-cli jammy/main amd64 Packages [2151 B]
Hit:8 https://download.docker.com/linux/ubuntu jammy InRelease
```

```
sjbarraza@sjbarraza:~/projects/university/platforms1/terraform$ sudo apt-get update && sudo apt-get install -y gnupg software-properties-common
Hit:1 http://archive.ubuntu.com/ubuntu jammy InRelease
Hit:2 http://security.ubuntu.com/ubuntu jammy-security InRelease
Hit:3 http://archive.ubuntu.com/ubuntu jammy-updates InRelease
Hit:4 https://packages.microsoft.com/repos/azure-cli jammy InRelease
Hit:5 http://archive.ubuntu.com/ubuntu jammy-backports InRelease
Hit:6 https://download.docker.com/linux/ubuntu jammy InRelease
Reading package lists... Done
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
gnupg is already the newest version (2.2.27-3ubuntu2.1).
software-properties-common is already the newest version (0.99.22.9).
software-properties-common set to manually installed.
0 upgraded, 0 newly installed, 0 to remove and 21 not upgraded.
sjbarraza@sjbarraza:~/projects/university/platforms1/terraform$ wget -O- https://apt.releases.hashicorp.com/gpg | \
gpg --dearmor | \
sudo tee /usr/share/keyrings/hashicorp-archive-keyring.gpg > /dev/null
--2025-03-03 19:19:54--  https://apt.releases.hashicorp.com/gpg
Resolving apt.releases.hashicorp.com (apt.releases.hashicorp.com)... 3.163.60.116, 3.163.60.73, 3.163.60.42, ...
Connecting to apt.releases.hashicorp.com (apt.releases.hashicorp.com)|3.163.60.116|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 3980 (3.9K) [binary/octet-stream]
Saving to: 'STDOUT'

-                                   100%[=============================================================================>]   3.89K  --.-KB/s    in 0s

2025-03-03 19:19:54 (14.0 MB/s) - written to stdout [3980/3980]

sjbarraza@sjbarraza:~/projects/university/platforms1/terraform$ gpg --no-default-keyring \
--keyring /usr/share/keyrings/hashicorp-archive-keyring.gpg \
--fingerprint
/usr/share/keyrings/hashicorp-archive-keyring.gpg
-------------------------------------------------
pub   rsa4096 2023-01-10 [SC] [expires: 2028-01-09]
      798A EC65 4E5C 1542 8C8E  42EE AA16 FCBC A621 E701
uid           [ unknown] HashiCorp Security (HashiCorp Package Signing) <security+packaging@hashicorp.com>
sub   rsa4096 2023-01-10 [S] [expires: 2028-01-09]

sjbarraza@sjbarraza:~/projects/university/platforms1/terraform$ echo "deb [signed-by=/usr/share/keyrings/hashicorp-archive-keyring.gpg] \
https://apt.releases.hashicorp.com $(lsb_release -cs) main" | \
sudo tee /etc/apt/sources.list.d/hashicorp.list
```

```
sjbarraza@sjbarraza:~/projects/university/platforms1/terraform$ sudo apt-get install terraform
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following NEW packages will be installed:
  terraform
0 upgraded, 1 newly installed, 0 to remove and 21 not upgraded.
Need to get 27.4 MB of archives.
After this operation, 89.6 MB of additional disk space will be used.
Get:1 https://apt.releases.hashicorp.com jammy/main amd64 terraform amd64 1.11.0-1 [27.4 MB]
Fetched 27.4 MB in 5s (5397 kB/s)
Selecting previously unselected package terraform.
(Reading database ... 100027 files and directories currently installed.)
Preparing to unpack .../terraform_1.11.0-1_amd64.deb ...
Unpacking terraform (1.11.0-1) ...
Setting up terraform (1.11.0-1) ...
sjbarraza@sjbarraza:~/projects/university/platforms1/terraform$
sjbarraza@sjbarraza:~/projects/university/platforms1/terraform$ az version
{
  "azure-cli": "2.69.0",
  "azure-cli-core": "2.69.0",
  "azure-cli-telemetry": "1.1.0",
  "extensions": {}
}
sjbarraza@sjbarraza:~/projects/university/platforms1/terraform$ az --version
azure-cli                         2.69.0

core                              2.69.0
telemetry                          1.1.0

Dependencies:
msal                            1.31.2b1
azure-mgmt-resource               23.1.1

Python location '/opt/az/bin/python3'
Config directory '/home/sjbarraza/.azure'
Extensions directory '/home/sjbarraza/.azure/cliextensions'

Python (Linux) 3.12.8 (main, Feb  5 2025, 06:39:23) [GCC 11.4.0]

Legal docs and information: aka.ms/AzureCliLegal
```

Installed Terraform and the Azure CLI.

```
sjbarraza@sjbarraza:~/projects/university/platforms1/terraform$ terraform -help
Usage: terraform [global options] <subcommand> [args]

The available commands for execution are listed below.
The primary workflow commands are given first, followed by
less common or more advanced commands.

Main commands:
  init          Prepare your working directory for other commands
  validate      Check whether the configuration is valid
  plan          Show changes required by the current configuration
  apply         Create or update infrastructure
  destroy       Destroy previously-created infrastructure

All other commands:
  console       Try Terraform expressions at an interactive command prompt
  fmt           Reformat your configuration in the standard style
  force-unlock  Release a stuck lock on the current workspace
  get           Install or upgrade remote Terraform modules
  graph         Generate a Graphviz graph of the steps in an operation
  import        Associate existing infrastructure with a Terraform resource
  login         Obtain and save credentials for a remote host
  logout        Remove locally-stored credentials for a remote host
```

We can see Terraform is installed.

```
sjbarraza@sjbarraza:~/projects/university/platforms1/terraform$ terraform -version
Terraform v1.11.0
on linux_amd64
```

It is in the 1.11.0 version.

```
sjbarraza@sjbarraza:~/projects/university/platforms1/terraform$ az login --use-device-code
To sign in, use a web browser to open the page https://microsoft.com/devicelogin and enter the code ABUW58Y5J to authenticate.

Retrieving tenants and subscriptions for the selection...

[Tenant and subscription selection]

No     Subscription name    Subscription ID                       Tenant
-----  -------------------  ------------------------------------  ------------------
[1] *  Azure for Students   [redacted]                            Universidad Icesi

The default is marked with an *; the default tenant is 'Universidad Icesi' and subscription is 'Azure for Students' ([redacted]).

Select a subscription and tenant (Type a number or Enter for no changes):

Tenant: Universidad Icesi
Subscription: Azure for Students ([redacted])

[Announcements]
With the new Azure CLI login experience, you can select the subscription you want to use more easily. Learn more about it and its configuration at https://go.microsoft.com/fwlink/?linkid=2271236

If you encounter any problem, please open an issue at https://aka.ms/azclibug

[Warning] The login output has been updated. Please be aware that it no longer displays the full list of available subscriptions by default.

sjbarraza@sjbarraza:~/projects/university/platforms1/terraform$
```

We login into the Azure CLI.

main.tf

```
provider "azurerm" {
  features {}
}

resource "azurerm_resource_group" "labs_plataformas_rg" {
  name     = "labs-plataformas-resources"
  location = "East US"
}

resource "azurerm_kubernetes_cluster" "labs_plataformas_aks" {
  name                = "labs-plataformas-aks1"
  location            = azurerm_resource_group.labs_plataformas_rg.location
  resource_group_name = azurerm_resource_group.labs_plataformas_rg.name
  dns_prefix          = "labs-plataformas-aks1"

  default_node_pool {
    name       = "default"
    node_count = 1
    vm_size    = "Standard_D2_v2"
  }

  identity {
    type = "SystemAssigned"
```

We created a main.tf file for an AKS cluster in Terraform.

```
sjbarraza@sjbarraza:~/projects/university/platforms1/terraform$ terraform fmt
main.tf
sjbarraza@sjbarraza:~/projects/university/platforms1/terraform$ terraform init
Initializing the backend...
Initializing provider plugins...
- Finding latest version of hashicorp/azurerm...
- Installing hashicorp/azurerm v4.21.1...
- Installed hashicorp/azurerm v4.21.1 (signed by HashiCorp)
Terraform has created a lock file .terraform.lock.hcl to record the provider
selections it made above. Include this file in your version control repository
so that Terraform can guarantee to make the same selections by default when
you run "terraform init" in the future.

Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.

If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
```

We format the Terraform code through “terraform fmt” and download the providers trough “terraform init”.

```
sjbarraza@sjbarraza:~/projects/university/platforms1/terraform$ terraform plan

Planning failed. Terraform encountered an error while generating this plan.


Error: `subscription_id` is a required provider property when performing a plan/apply operation

  with provider["registry.terraform.io/hashicorp/azurerm"],
  on main.tf line 1, in provider "azurerm":
   1: provider "azurerm" {
```

Through “terraform plan” we can see that the code has a problem, because it does not have the subscription_id attribute in the provider, so we add it.

```
provider "azurerm" {
  features {}
  subscription_id = 
}
```

```
sjbarraza@sjbarraza:~/projects/university/platforms1/terraform$ terraform plan

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols:
  + create

Terraform will perform the following actions:

  # azurerm_kubernetes_cluster.labs_plataformas_aks will be created
  + resource "azurerm_kubernetes_cluster" "labs_plataformas_aks" {
      + current_kubernetes_version          = (known after apply)
      + dns_prefix                          = "labs-plataformas-aks1"
      + fqdn                                = (known after apply)
      + http_application_routing_zone_name  = (known after apply)
      + id                                  = (known after apply)
      + kube_admin_config                   = (sensitive value)
      + kube_admin_config_raw               = (sensitive value)
      + kube_config                         = (sensitive value)
      + kube_config_raw                     = (sensitive value)
      + kubernetes_version                  = (known after apply)
      + location                            = "eastus"
      + name                                = "labs-plataformas-aks1"
      + node_os_upgrade_channel             = "NodeImage"
      + node_resource_group                 = (known after apply)
```

```
      + node_resource_group                 = (known after apply)
      + node_resource_group_id              = (known after apply)
      + oidc_issuer_url                     = (known after apply)
      + portal_fqdn                         = (known after apply)
      + private_cluster_enabled             = false
      + private_cluster_public_fqdn_enabled = false
      + private_dns_zone_id                 = (known after apply)
      + private_fqdn                        = (known after apply)
      + resource_group_name                 = "labs-plataformas-resources"
      + role_based_access_control_enabled   = true
      + run_command_enabled                 = true
      + sku_tier                            = "Free"
      + support_plan                        = "KubernetesOfficial"
      + tags                                = {
          + "Environment" = "Lab-Plataformas-Production"
        }
      + workload_identity_enabled           = false

      + auto_scaler_profile (known after apply)

      + default_node_pool {
          + kubelet_disk_type    = (known after apply)
```

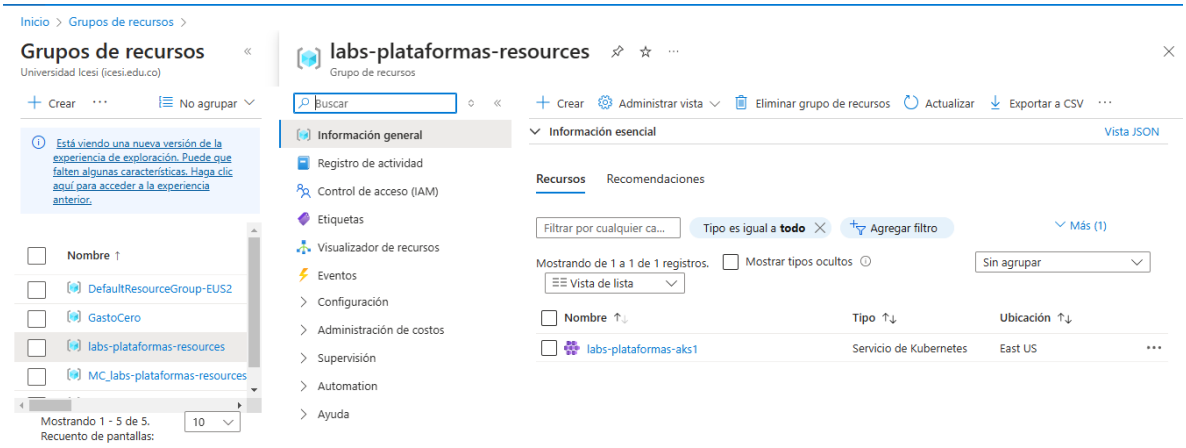
```
      + default_node_pool {
          + kubelet_disk_type    = (known after apply)
          + max_pods             = (known after apply)
          + name                 = "default"
          + node_count           = 1
          + node_labels          = (known after apply)
          + orchestrator_version = (known after apply)
          + os_disk_size_gb      = (known after apply)
          + os_disk_type         = "Managed"
          + os_sku               = (known after apply)
          + scale_down_mode      = "Delete"
          + type                 = "VirtualMachineScaleSets"
          + ultra_ssd_enabled    = false
          + vm_size              = "Standard_D2_v2"
          + workload_runtime     = (known after apply)
        }

      + identity {
          + principal_id = (known after apply)
          + tenant_id    = (known after apply)
          + type         = "SystemAssigned"
        }
```

```
      + kubelet_identity (known after apply)

      + network_profile (known after apply)

      + windows_profile (known after apply)
    }

  # azurerm_resource_group.labs_plataformas_rg will be created
  + resource "azurerm_resource_group" "labs_plataformas_rg" {
      + id       = (known after apply)
      + location = "eastus"
      + name     = "labs-plataformas-resources"
    }

Plan: 2 to add, 0 to change, 0 to destroy.

Changes to Outputs:
  + client_certificate = (sensitive value)
  + kube_config        = (sensitive value)
```

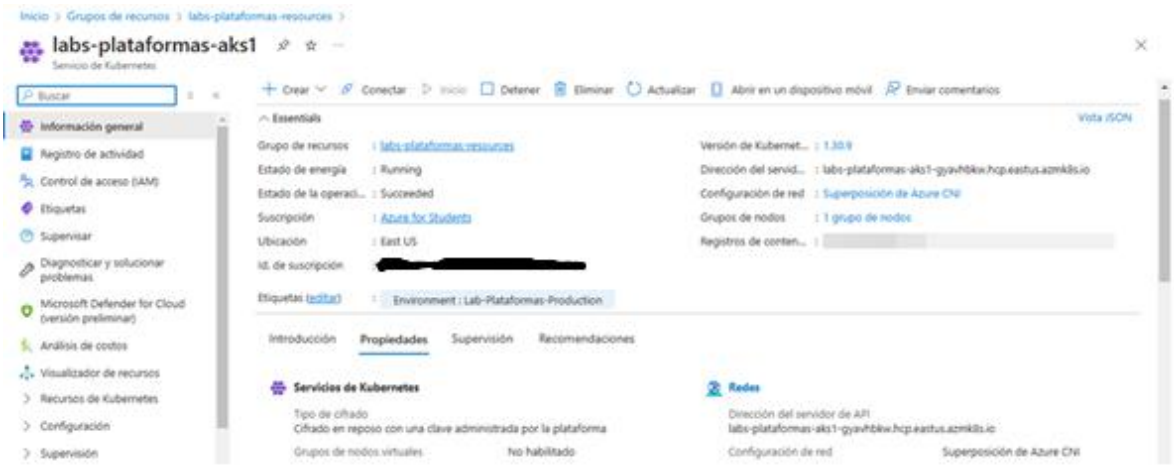
We can see that the “terraform plan” worked correctly.

```
sjbarraza@sjbarraza:~/projects/university/platforms1/terraform$ terraform apply

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols:
  + create

Terraform will perform the following actions:

  # azurerm_kubernetes_cluster.labs_plataformas_aks will be created
  + resource "azurerm_kubernetes_cluster" "labs_plataformas_aks" {
      + current_kubernetes_version          = (known after apply)
      + dns_prefix                          = "labs-plataformas-aks1"
      + fqdn                                = (known after apply)
      + http_application_routing_zone_name  = (known after apply)
      + id                                  = (known after apply)
      + kube_admin_config                   = (sensitive value)
      + kube_admin_config_raw               = (sensitive value)
      + kube_config                         = (sensitive value)
```

Now we do “terraform apply” to upload these changes to our Azure infrastructure.



Once it is finished, we can se that the resource group is successfully created, and it has the Kubernetes cluster inside.



We can see that the Kubernetes cluster is running.

```
sjbarraza@sjbarraza:~/projects/university/platforms1/terraform/credentials$ az aks get-credentials --resource-group labs-plataformas-resources --name labs-plataformas
-aks1 --overwrite-existing
Merged "labs-plataformas-aks1" as current context in /home/sjbarraza/.kube/config
sjbarraza@sjbarraza:~/projects/university/platforms1/terraform/credentials$
```

Now we download the credentials.

```
sjbarraza@sjbarraza:~/projects/university/platforms1/terraform/credentials$
sjbarraza@sjbarraza:~/projects/university/platforms1/terraform/credentials$ kubectl config current-context
labs-plataformas-aks1
sjbarraza@sjbarraza:~/projects/university/platforms1/terraform/credentials$
```

We see that the current context is the AKS cluster (it was minikube before).

```
sjbarraza@sjbarraza:~/projects/university/platforms1/terraform/credentials$ kubectl config get-contexts
CURRENT   NAME                  CLUSTER               AUTHINFO                                                NAMESPACE
*         labs-plataformas-aks1   labs-plataformas-aks1   clusterUser_labs-plataformas-resources_labs-plataformas-aks1
          minikube              minikube              minikube                                                default
sjbarraza@sjbarraza:~/projects/university/platforms1/terraform/credentials$
```

We can se our contexts, in this case, AKS and minikube.

```
sjbarraza@sjbarraza:~/projects/university/platforms1/terraform/credentials$ kubectl config use-context labs-plataformas-aks1
Switched to context "labs-plataformas-aks1".
sjbarraza@sjbarraza:~/projects/university/platforms1/terraform/credentials$
sjbarraza@sjbarraza:~/projects/university/platforms1/terraform/credentials$
sjbarraza@sjbarraza:~/projects/university/platforms1/terraform/credentials$
```

This command is used to switch to the AKS cluster, but we were already in it.

```
sjbarraza@sjbarraza:~/projects/university/platforms1/terraform/credentials$ kubectl get nodes
NAME                                STATUS   ROLES    AGE     VERSION
aks-default-18997406-vmss000000   Ready    <none>   8m17s   v1.30.9
sjbarraza@sjbarraza:~/projects/university/platforms1/terraform/credentials$
sjbarraza@sjbarraza:~/projects/university/platforms1/terraform/credentials$
```

We can see the default node.

```
sjbarraza@sjbarraza:~/projects/university/platforms1/terraform/credentials$ kubectl get ns
NAME              STATUS   AGE
default           Active   11m
kube-node-lease   Active   11m
kube-public       Active   11m
kube-system       Active   11m
sjbarraza@sjbarraza:~/projects/university/platforms1/terraform/credentials$
```

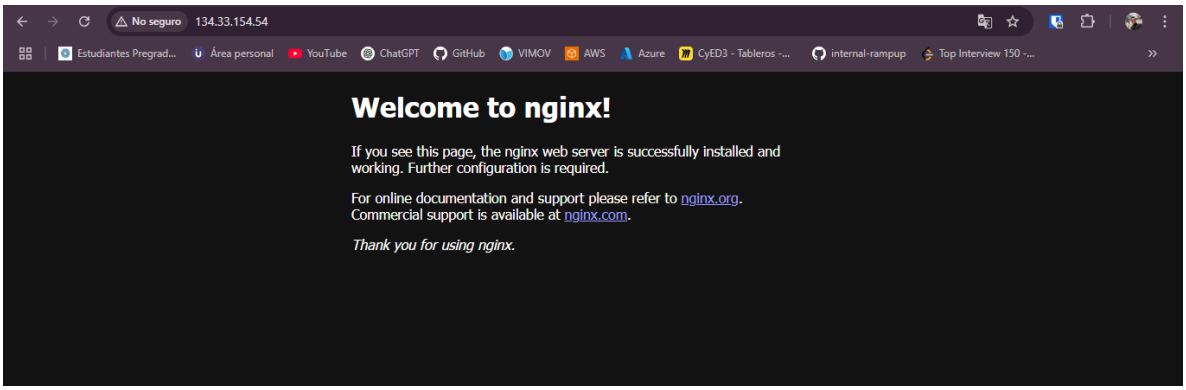
These are the namespaces of the AKS cluster.

```
sjbarraza@sjbarraza:~/projects/university/platforms1/terraform/credentials$ kubectl run server --image=nginx
pod/server created
sjbarraza@sjbarraza:~/projects/university/platforms1/terraform/credentials$ kubectl get pods
NAME     READY   STATUS    RESTARTS   AGE
server   1/1     Running   0          7s
sjbarraza@sjbarraza:~/projects/university/platforms1/terraform/credentials$
```

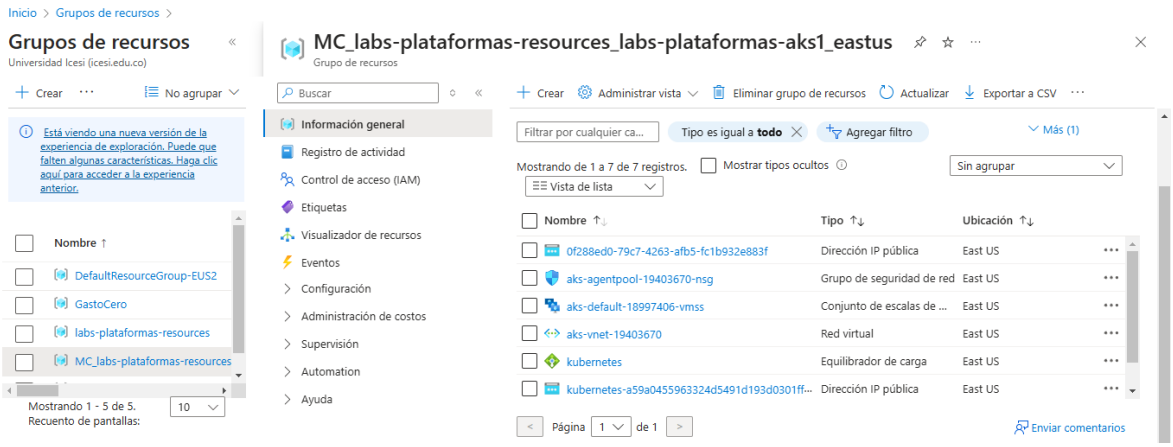
We run an Ngnix pod.

```
sjbarraza@sjbarraza:~/projects/university/platforms1/terraform/credentials$ kubectl expose pod server --type=LoadBalancer --port=80 --target-port=80
service/server exposed
sjbarraza@sjbarraza:~/projects/university/platforms1/terraform/credentials$
sjbarraza@sjbarraza:~/projects/university/platforms1/terraform/credentials$ kubectl get services
NAME         TYPE           CLUSTER-IP    EXTERNAL-IP   PORT(S)        AGE
kubernetes   ClusterIP      10.0.0.1      <none>        443/TCP        16m
server       LoadBalancer   10.0.179.68   <pending>     80:32750/TCP   7s
sjbarraza@sjbarraza:~/projects/university/platforms1/terraform/credentials$ kubectl get services
NAME         TYPE           CLUSTER-IP    EXTERNAL-IP   PORT(S)        AGE
kubernetes   ClusterIP      10.0.0.1      <none>        443/TCP        16m
server       LoadBalancer   10.0.179.68   <pending>     80:32750/TCP   11s
sjbarraza@sjbarraza:~/projects/university/platforms1/terraform/credentials$ kubectl get services
NAME         TYPE           CLUSTER-IP    EXTERNAL-IP     PORT(S)        AGE
kubernetes   ClusterIP      10.0.0.1      <none>          443/TCP        16m
server       LoadBalancer   10.0.179.68   134.33.154.54   80:32750/TCP   14s
sjbarraza@sjbarraza:~/projects/university/platforms1/terraform/credentials$
```

We expose the pod as a load balancer because it needs to have an external ip address to us to access it through internet.



We can see that the service is correctly accessible from internet.



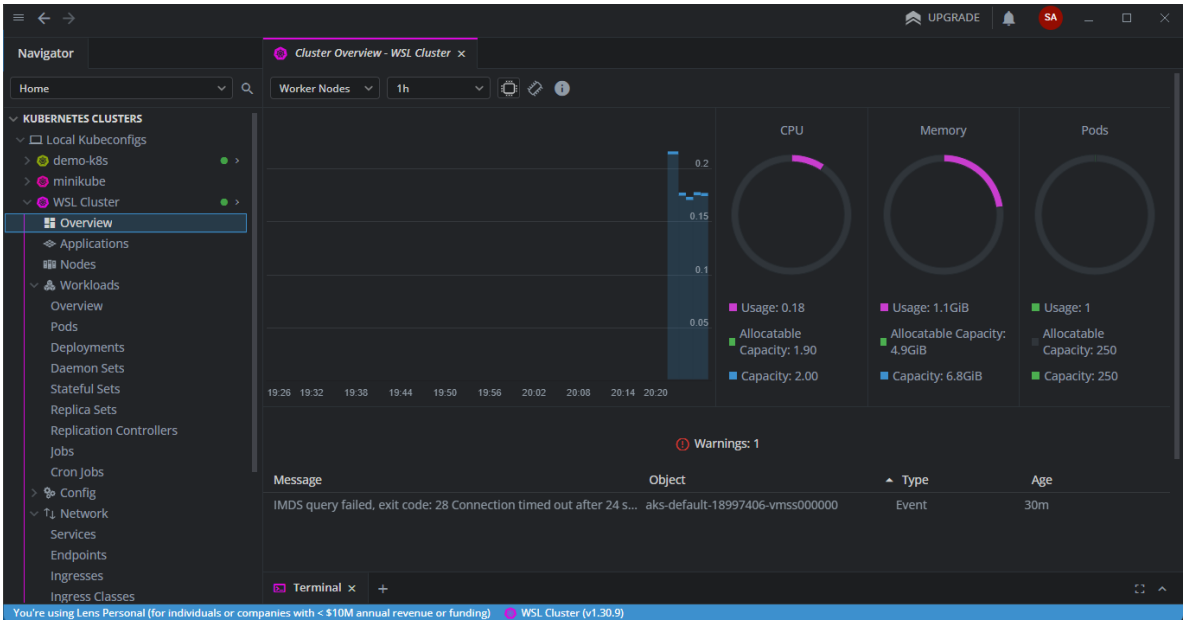
We can see the base resources that AKS created to make itself capable of creating a Kubernetes cluster.

```
sjbarraza@sjbarraza:~/projects/university/platforms1/terraform/credentials$ kubectl proxy --port 8001 --reject-paths "^/api/./pods/./attach"
Starting to serve on 127.0.0.1:8001
```

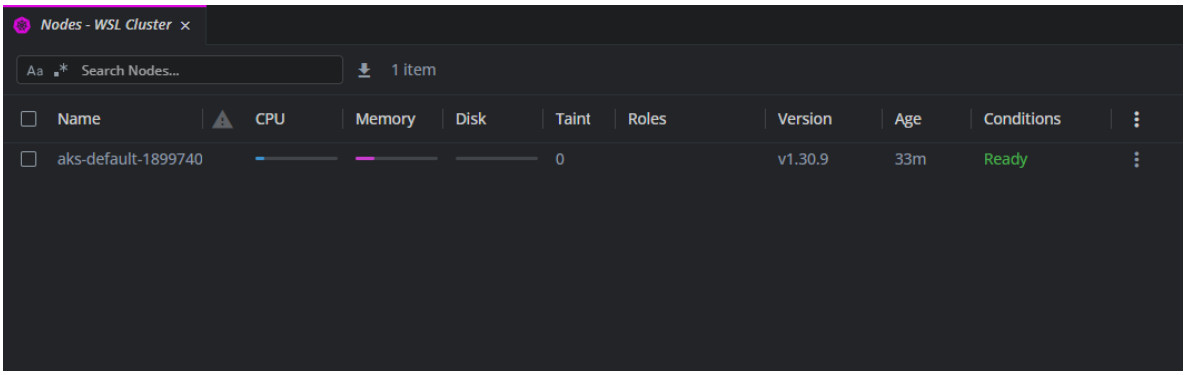
Now we expose our Kubernetes cluster access that WSL has to the windows host machine through port 8001.

```yaml
apiVersion: v1
kind: Config
clusters:
  - name: "WSL Cluster"
    cluster:
      server: http://localhost:8001
users:
  - name: nouser
contexts:
  - name: "WSL Cluster"
    context:
      cluster: "WSL Cluster"
      user: nouser
current-context: "WSL Cluster"
preferences: {}
```

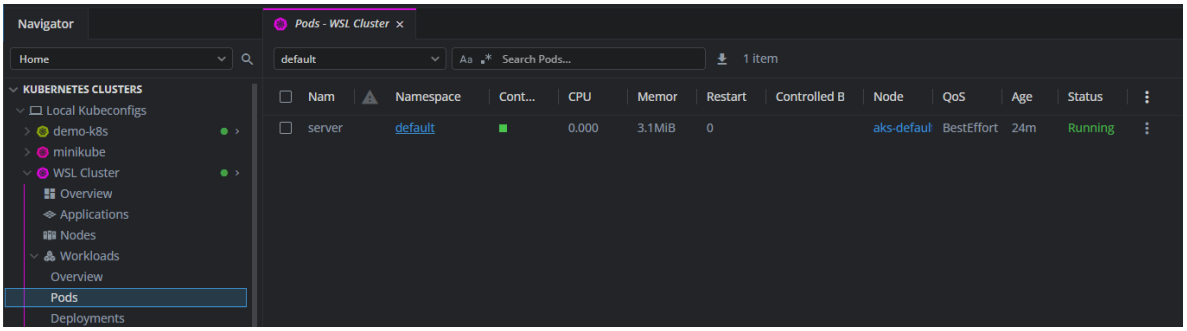
We add this kubeconfig template to lens, so it can access to the Kubernetes cluster that WSL is using (in this case, AKS cluster).



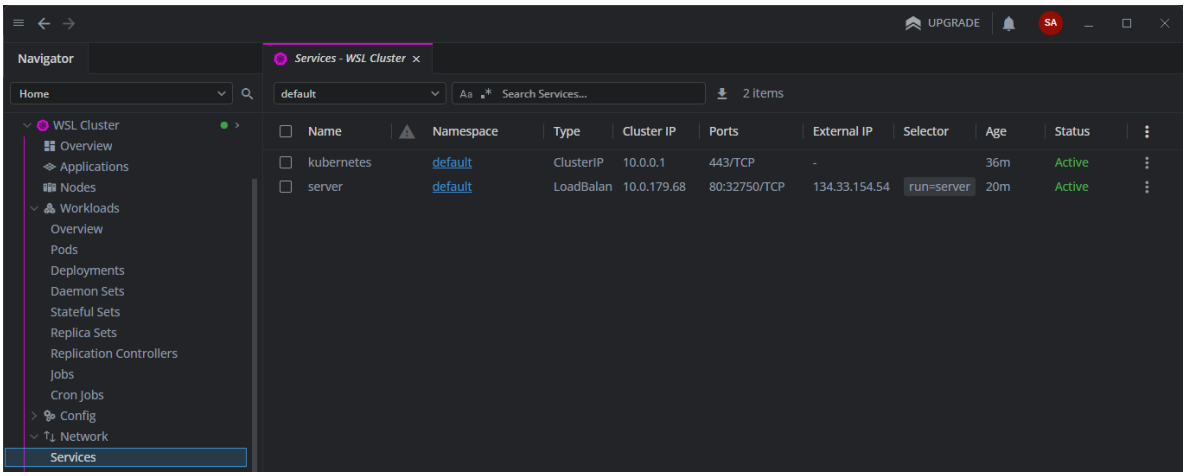
We can see the CPU, memory and other metrics of the cluster.



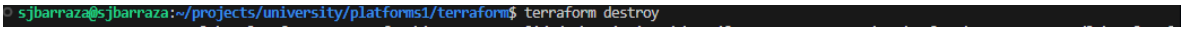
We can see the nodes.



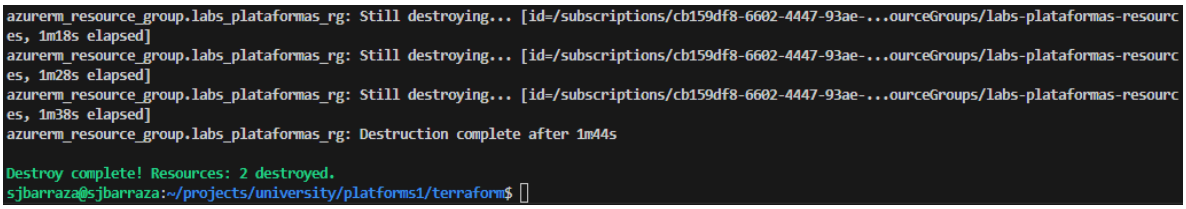
We can see the server pod that is running Nginx.



We can see the service that is exposing the Nginx server.



Now, we destroy the infrastructure.



And the infrastructure is destroyed successfully.