

Asignatura	Datos del alumno	Fecha
Devops & Cloud	Apellidos: Ramón	13/02/2025
	Nombre: David	

Caso Práctico 2: Automatización de despliegues en entornos Cloud

Objetivos de la actividad

Esta actividad tiene los siguientes objetivos:

- ▮ Crear infraestructura de forma automatizada en un proveedor de Cloud pública.
- ▮ Utilizar herramientas de gestión de la configuración para automatizar la instalación y configuración de servicios.
- ▮ Desplegar mediante un enfoque totalmente automatizado aplicaciones en forma de contenedor sobre el sistema operativo.
- ▮ Desplegar mediante un enfoque totalmente automatizado aplicaciones que hagan uso de almacenamiento persistente sobre una plataforma de orquestación de contenedores.

Descripción de la actividad

El caso práctico 2 consiste en desplegar los siguientes elementos:

- ▮ Un (1) repositorio de imágenes de contenedores sobre infraestructura de Microsoft Azure mediante el servicio Azure Container Registry (ACR).
- ▮ Una (1) aplicación en forma de contenedor utilizando Podman sobre una máquina virtual en Azure.
- ▮ Un (1) cluster de Kubernetes como servicio gestionado en Microsoft Azure (AKS).

Asignatura	Datos del alumno	Fecha
Devops & Cloud	Apellidos: Ramón	13/02/2025
	Nombre: David	

- ▮ Una (1) aplicación con almacenamiento persistente sobre el cluster AKS.

Consideraciones

Consideraciones **generales**:

- ▮ El código desarrollado por el alumno deberá estar disponible en un repositorio git público sin modificaciones posteriores a la fecha de entrega del caso práctico. Si el repositorio no es accesible públicamente no se podrá proceder con la evaluación del caso práctico.
- ▮ El código tendrá una estructura específica que será comentada en clase.
- ▮ Todo el código desarrollado por el alumno deberá ser original. Cualquier indicio de copia de otros compañeros o de otras fuentes podrá ser motivo de suspenso.
- ▮ El código deberá estar comentado (en inglés o español) de forma clara y detallada.
- ▮ Toda la infraestructura debe crearse en Azure mediante Terraform como herramienta de infraestructura como código. No se admite la creación de ningún recurso en Azure de forma manual.
- ▮ Toda la configuración de los distintos elementos (Sistema operativo, Kubernetes, etc.,) y el despliegue de las aplicaciones debe hacerse bajo un enfoque automatizado mediante Ansible como herramienta de gestión de la configuración. No se admiten pasos manuales de configuración.

Asignatura	Datos del alumno	Fecha
Devops & Cloud	Apellidos: Ramón	13/02/2025
	Nombre: David	

- ▮ Siempre que sea posible se deberá evitar la utilización de los siguientes módulos de Ansible:
 - Command.
 - Shell.
 - Script.
- ▮ Las aplicaciones que desplegar en Podman y en Kubernetes deben ser distintas.

Consideraciones **específicas del *registry*** desplegado **mediante ACR:**

- ▮ El servicio debe ser accesible desde Internet.
- ▮ El acceso debe ser mediante autenticación.
- ▮ En este repositorio se deben ubicar las imágenes de las dos aplicaciones a desplegar en este caso práctico.
- ▮ Cada imagen de contenedor a utilizar debe encontrarse bajo una carpeta distinta del repositorio.
- ▮ La versión de las imágenes debe ser **casopractico2**.

Consideraciones **específicas** de la aplicación **sobre la máquina virtual:**

- ▮ Debe desplegarse en forma de contenedor de Podman sobre una máquina virtual desplegada en Azure.
- ▮ El sistema operativo a utilizar debe ser Linux, siendo la distribución (Debian, CentOS, Alma Linux, etc) de libre elección.
- ▮ La aplicación se podrá gestionar como un servicio más del sistema operativo, persistente a reinicios y con la opción de detener o iniciar el servicio por un administrador.
- ▮ La imagen de la aplicación se descargará desde el registry desplegado mediante ACR.

Asignatura	Datos del alumno	Fecha
Devops & Cloud	Apellidos: Ramón	13/02/2025
	Nombre: David	

- ▮ La aplicación será un servidor web de libre elección: Apache, Nginx, etc.
- ▮ El servicio Web debe apoyarse en un certificado x.509 autofirmado.
- ▮ El servicio Web debe utilizar al menos autenticación básica (htpasswd).
- ▮ El contenido mostrado por el servicio Web será de libre elección.
- ▮ La aplicación deberá ser accesible desde Internet.
- ▮ La imagen de la aplicación tendrá todos los datos necesarios para el servicio web: contenido a mostrar, certificado x.509 y credenciales de usuarios.

Consideraciones **específicas del cluster de Kubernetes:**

- ▮ El cluster deberá desplegarse como un servicio gestionado en Azure, es decir, mediante AKS.
- ▮ Sólo es necesario desplegar un único worker.
- ▮ El cluster de Kubernetes tendrá conectividad hacia el *registry* privado de ACR.
- ▮ Deberán configurarse las credenciales de acceso al *registry* para que las aplicaciones puedan autenticarse.

Consideraciones **específicas de la aplicación sobre Kubernetes:**

- ▮ La aplicación se descargará una o más imágenes necesarias desde el *registry* privado. La imagen no puede ser la misma utilizada para la aplicación desplegada sobre Podman.
- ▮ El tipo de aplicación a desplegar es de libre elección. Puede ser Apache, Nginx, Jenkins, MongoDB, etc,
- ▮ La aplicación debe utilizar almacenamiento persistente a partir de alguna de las *storage classes* definidas en AKS.
- ▮ La aplicación debe ser accesible desde Internet.

Asignatura	Datos del alumno	Fecha
Devops & Cloud	Apellidos: Ramón	13/02/2025
	Nombre: David	

Entregables

- ▮ Un informe en formato PDF incluyendo los siguientes aspectos:
 - La URL del repositorio git donde está disponible el código con la resolución de este caso práctico.
 - Diagramas mostrando los elementos desplegados y su rol. Estos diagramas deberán ser realizados por el alumno.
 - Descripción del proceso de despliegue, sus requisitos y los recursos resultantes en Azure.
 - Descripción de la aplicación desplegada.
 - Deberás describir los problemas que has encontrado, si los hubiera, cómo se han solucionado y referencias que se hayan utilizado para resolverlas (en formato APA). Si algún problema no se ha podido solucionar, se deberá analizar y proponer soluciones o siguientes pasos.
 - Se deberá indicar la licencia utilizada e indicar las restricciones y el uso que permite la licencia.

Extensión máxima de la actividad: no hay extensión máxima. Fuente Calibri 12 e interlineado 1,5.

Asignatura	Datos del alumno	Fecha
Devops & Cloud	Apellidos: Ramón	13/02/2025
	Nombre: David	

Rúbrica

Automatización de despliegues en entornos Cloud	Descripción	Puntuación máxima (puntos)	Peso %
Criterio 1	Despliegue del <i>registry</i> y subida de las imágenes.	1	10%
Criterio 2	Despliegue de la aplicación en Podman	2	20%
Criterio 3	Despliegue y configuración del cluster de Kubernetes.	2	20%
Criterio 4	Despliegue de la aplicación sobre Kubernetes	2	20%
Criterio 5	Informe.	3	30
		10	100 %

Asignatura	Datos del alumno	Fecha
Devops & Cloud	Apellidos: Ramón	13/02/2025
	Nombre: David	

TAREA AZURE CON TERRAFORM Y ANSIBLE

REPOSITORIO GIT:

<https://github.com/DavidRO71/TareaUnirAzure.git>

AZURE CLI

Instalación del **CLI de Azure** en mi máquina Windows para poder logearme desde la línea de comandos en Windows.

Descargo el instalador desde:

[https://learn.microsoft.com/en-us/cli/azure/install-azure-cli-windows?](https://learn.microsoft.com/en-us/cli/azure/install-azure-cli-windows?pivot=msi)

pivots=msi

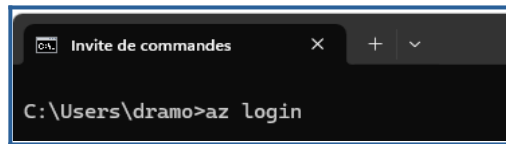
MSI más reciente de la CLI de Azure (64 bits)

Lo instalo.



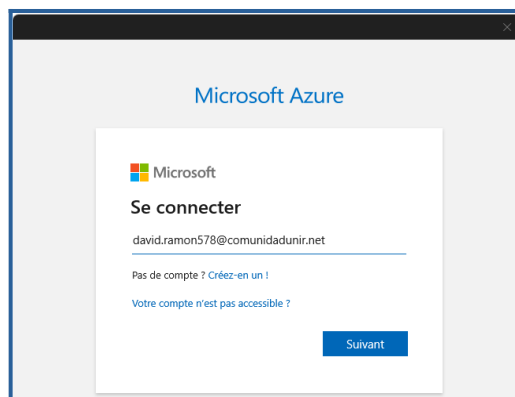
Asignatura	Datos del alumno	Fecha
Devops & Cloud	Apellidos: Ramón	13/02/2025
	Nombre: David	

Abro una nueva terminal y escribo "az login" para logearme en Azure.

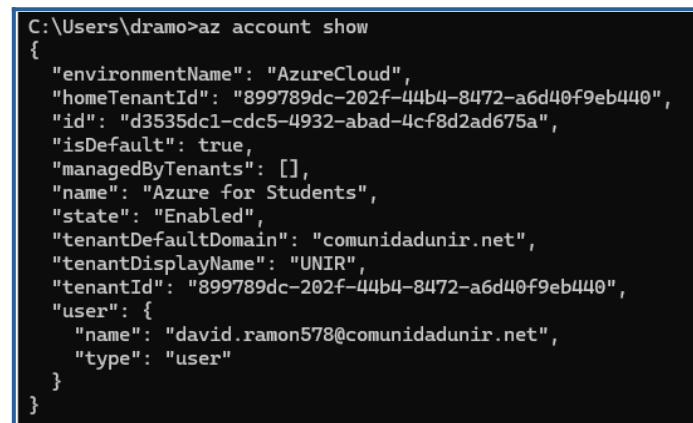


```
C:\Users\dramo>az login
```

Elijo mi cuenta de estudiante.



Una vez conectado, puedo ver mis datos:



```
C:\Users\dramo>az account show
{
  "environmentName": "AzureCloud",
  "homeTenantId": "899789dc-202f-44b4-8472-a6d40f9eb440",
  "id": "d3535dc1-cdc5-4932-abad-4cf8d2ad675a",
  "isDefault": true,
  "managedByTenants": [],
  "name": "Azure for Students",
  "state": "Enabled",
  "tenantDefaultDomain": "comunidadunir.net",
  "tenantDisplayName": "UNIR",
  "tenantId": "899789dc-202f-44b4-8472-a6d40f9eb440",
  "user": {
    "name": "david.ramon578@comunidadunir.net",
    "type": "user"
  }
}
```


Asignatura	Datos del alumno	Fecha
Devops & Cloud	Apellidos: Ramón	13/02/2025
	Nombre: David	

PODMAN

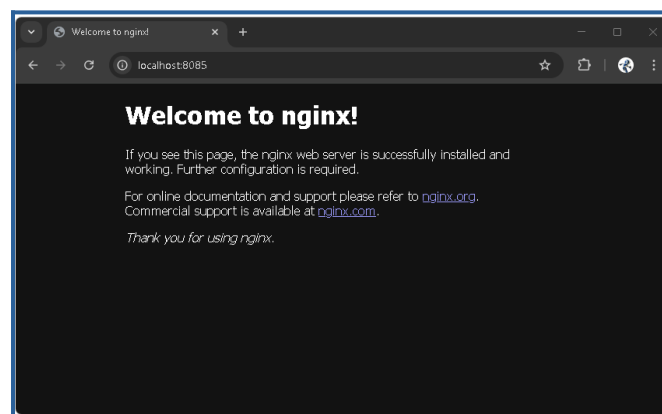
Me bajo la imagen de nginx versión 1.27.4 en alpine para que sea más ligera en local para comprobar que Podman se ha instalado bien en mi **maquina local**.

podman pull nginx nginx:1.27.4-alpine-slim

La ejecuto en **local**:

podman run -d -p 8085:80 --name nginx-container nginx

Abro el navegador y voy a la dirección siguiente **http://localhost:8085/** para comprobar que funciona correctamente



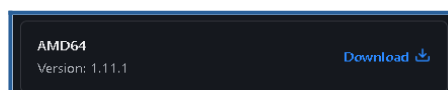
TERRAFORM

Me bajo el archivo comprimido que contiene **Terraform** desde:

<https://developer.hashicorp.com/terraform/install>

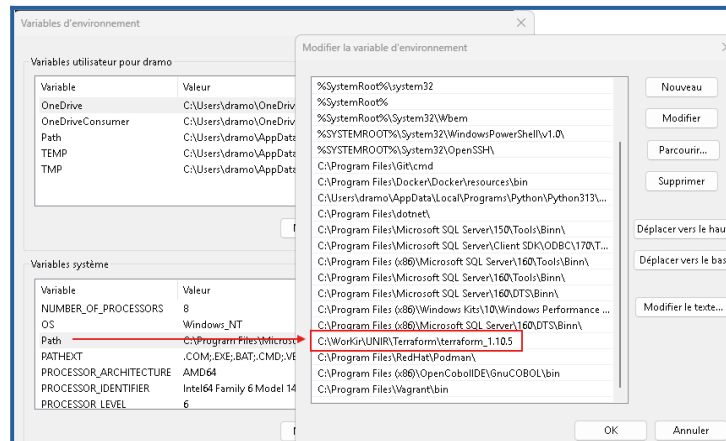
Como estoy en Windows 64bits, me bajo el archivo:

(terraform_1.10.5_windows_386.zip) desde:



Asignatura	Datos del alumno	Fecha
Devops & Cloud	Apellidos: Ramón	13/02/2025
	Nombre: David	

Copio la ruta de mi carpeta "**C:\WorKir\UNIR\Terraform\terraform_1.10.5**" y pego en el **PATH** en las variables de entorno:



SSH

Creación de la **clave SSH** desde el **CMD**:

```
ssh-keygen -t rsa -b 4096 -C "david.ramon578@comunidadunir.net"
```

Comprobacion que se ha generado correctamente:

```
cat ~/.ssh/id_rsa.pub
```

Resultado:

No pongo la clave completa, solo el principio y el final

```
ssh-rsa AAAAB3.....CcwfdPPxIKow==
```

Asignatura	Datos del alumno	Fecha
Devops & Cloud	Apellidos: Ramón	13/02/2025
	Nombre: David	

VISUAL STUDIO CODE

El repositorio de **GIT**:

<https://github.com/DavidRO71/TareaUnirAzure.git>

Estructura de mi proyecto en **Visual Studio Code**:

TAREAAZURE

Ansible

Salidas

Salida_imagen_acr_playbook.txt

Salida_imagen_redis_azure_vote_front_ACR_playbook.txt

Salida_imagen_redis_azure_vote_front.txt

Salida_podman_vm_playbook.txt

acr-secret.yaml

deployment.yaml

imagen_acr_playbook.yml

imagen_redis_azure_vote_front_ACR_playbook.yml

imagen_redis_azure_vote_front_playbook.yml

podman_vm_playbook.yml

inventory.ini

variables.yml

service.yaml

Terraform

main.tf

variables.tf

Salida_Init.txt

Salida_Plan.txt

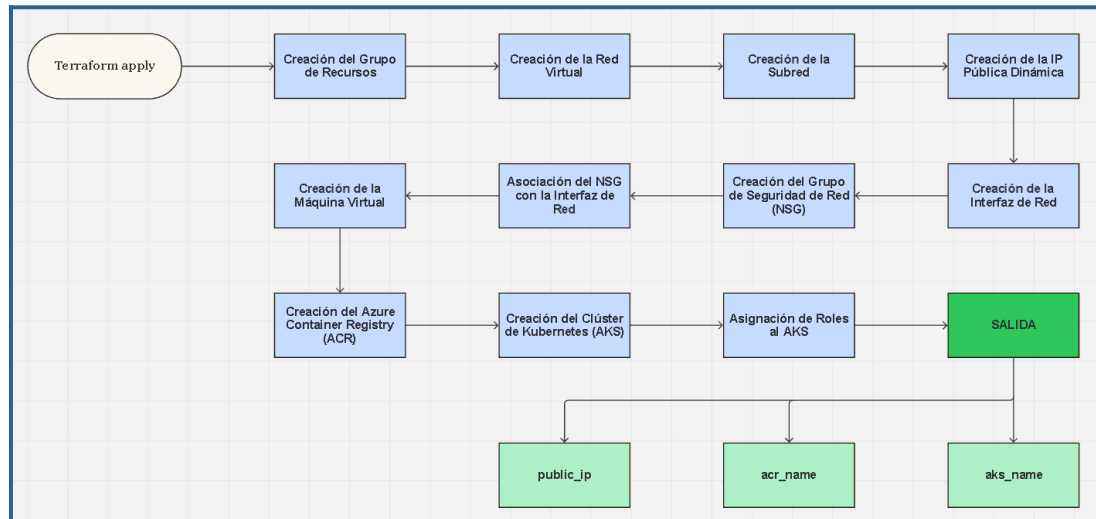
Salida_Apply.txt

Salida_Destroy.txt

Asignatura	Datos del alumno	Fecha
Devops & Cloud	Apellidos: Ramón	13/02/2025
	Nombre: David	

TERRAFORM

Diagrama de los pasos que vamos a realizar con Terraform:



Con los 2 ficheros (**main.tf** y **variables.tf**) que se encuentran en la carpeta de Terraform establecemos una infraestructura básica en Azure que abarca un grupo de recursos, un contenedor de imágenes de Docker (ACR), una red virtual, una máquina virtual con Ubuntu, un clúster de Kubernetes (AKS), un grupo de seguridad para permitir SSH y una dirección IP pública.

Los 4 ficheros (**Salida_Init.txt**, **Salida_Plan.txt**, **Salida_Apply.txt** y **Salida_Destroy.txt**) son los ficheros con los resultados de las ejecuciones de los comandos de Terraform.

Salida_Init.txt => Terraform init
Salida_Plan.txt => Terraform plan
Salida_Apply.txt => Terraform apply
Salida_Destroy.txt => Terraform destroy

Asignatura	Datos del alumno	Fecha
Devops & Cloud	Apellidos: Ramón	13/02/2025
	Nombre: David	

Terraform init

terraform init es un comando fundamental para preparar y configurar un entorno de trabajo en Terraform. Este comando garantiza que todos los proveedores, módulos y configuraciones del backend estén listos para ser utilizados en la creación, modificación o eliminación de recursos.

Terraform plan

Terraform plan es una herramienta esencial en Terraform que permite previsualizar y verificar los cambios propuestos en la infraestructura. Es una manera segura de revisar lo que Terraform tiene previsto hacer antes de llevar a cabo esos cambios, lo cual es crucial para gestionar entornos de forma efectiva y evitar errores inesperados.

Terraform apply

Terraform apply es el comando que realmente aplica los cambios a la infraestructura definidos en los archivos de configuración de Terraform. Después de previsualizar los cambios con terraform plan, terraform apply lleva a cabo la creación, modificación o eliminación de recursos en el entorno de infraestructura. Es fundamental revisar los cambios antes de aplicar y usar terraform apply con precaución para asegurar que los cambios sean correctos y seguros.

Terraform destroy

Terraform destroy se utiliza para eliminar todos los recursos que Terraform ha gestionado y que fueron creados a partir de un conjunto de archivos de configuración. Es una manera de limpiar la infraestructura que has creado anteriormente, asegurándote de que no queden recursos huérfanos o innecesarios que sigan consumiendo recursos en tu cuenta en la nube (como en AWS, Azure, Google Cloud, etc.).

Asignatura	Datos del alumno	Fecha
Devops & Cloud	Apellidos: Ramón	13/02/2025
	Nombre: David	

Explicación de mi fichero variables.tf

Este archivo de **variables de Terraform** establece la configuración que son esenciales para manejar recursos en Azure. Aquí defino las variables con **su tipo, su valor y una descripción** para saber a que hace referencia cada una de ella.

Estas variables se utilizarán en el fichero **main.tf**. Cuando se procese este ultimo, **se sustituirá la variable por su valor**.

Con las declaraciones de estas en este archivo se puede **reutilizar fácilmente**, ya que puedes ajustar sus valores para diferentes entornos o proyectos sin tener que modificar el código en el fichero **main.tf de Terraform**.

variable "subscription_id"

```
variable "subscription_id" {
  description = "ID de la suscripción"
  type      = string
  default   = "d3535dc1-cdc5-4932-abad-4cf8d2ad675a"
}
```

- **description:** Pongo que es el ID de la suscripción de Azure donde se crearán los recursos.

- **type:** El tipo de dato es string porque el valor es una cadena de caracteres.

- **default:** Pongo el ID de la suscripción.

variable "client_id"

```
variable "client_id" {
  description = "ID de cliente"
  type      = string
  default   = "Azure for Students"
}
```

- **description:** Pongo que es el ID del cliente vinculado a la suscripción de Azure.

- **type:** El tipo de dato es string porque el valor es una cadena de caracteres.

- **default:** Pongo "Azure for Students", que la cuenta de estudiante en Azure.

Asignatura	Datos del alumno	Fecha
Devops & Cloud	Apellidos: Ramón	13/02/2025
	Nombre: David	

variable "tenant_id"

```
variable "tenant_id" {
  description = "ID de tenant"
  type       = string
  default    = "899789dc-202f-44b4-8472-a6d40f9eb440"
}
```

- **description:** Pongo el ID de tenant de Azure es el identificador único asociado a mi suscripción.

- **type:** El tipo de dato es string porque el valor es una cadena de caracteres.

- **default:** Pongo el ID de tenant.

variable "resource_group_name"

```
variable "resource_group_name" {
  description = "El nombre del grupo de recursos"
  type       = string
  default    = "davidrgn"
}
```

- **description:** Pongo el nombre del grupo de recursos en Azure.

- **type:** El tipo de dato es string porque el valor es una cadena de caracteres.

- **default:** Pongo "davidrgn", que es el nombre del grupo de mi recurso.

Asignatura	Datos del alumno	Fecha
Devops & Cloud	Apellidos: Ramón	13/02/2025
	Nombre: David	

variable "resource_group_location"

```
variable "resource_group_location" {
  description = "La localizacion donde se crearan mis recursos"
  type       = string
  default    = "Spain Central"
}
```

- **description:** Pongo la región geográfica donde se desplegarán mis recursos dentro del grupo de recursos.

- **type:** El tipo de dato es string porque el valor es una cadena de caracteres.

- **default:** Pongo "Spain Central", que es una región de Azure que utilizaré.

variable "azurerm_container_registry"

```
variable "azurerm_container_registry" {
  description = "El nombre de mi ACR"
  type       = string
  default    = "davidacr"
}
```

- **description:** Pongo el nombre del Azure Container Registry (ACR), que es un servicio de Azure para almacenar imágenes de contenedores.

- **type:** El tipo de dato es string porque el valor es una cadena de caracteres.

- **default:** Pongo "davidacr", que es el nombre de mi registro.

Asignatura	Datos del alumno	Fecha
Devops & Cloud	Apellidos: Ramón	13/02/2025
	Nombre: David	

variable "location_container_registry"

```
variable "location_container_registry" {
  description = "La localizacion de mi ACR"
  type       = string
  default    = "Spain Central"
}
```

- **description:** Pongo la ubicación geográfica donde se creará el Azure Container Registry.

- **type:** El tipo de dato es string porque el valor es una cadena de caracteres.

- **default:** Pongo "Spain Central", porque es la misma región que la de los recursos.

variable "admin_ssh_key"

```
variable "admin_ssh_key" {
  description = "Clave SSH pública para poder acceder a la maquina virtual"
  type       = string
  default    = "ssh-rsa AAAAB3NzaC.....david.ramon578@comunidadunir.net"
}
```

description: Pongo la clave SSH pública (que he creado anteriormente), que se utilizará para acceder a las máquinas virtuales de Azure.

type: El tipo de dato es string porque el valor es una cadena de caracteres.

default: Pongo mi clave SSH pública, que me permite acceder a una máquina virtual sin necesidad de contraseña. Está asociada al correo electrónico david.ramon578@comunidadunir.net.

Asignatura	Datos del alumno	Fecha
Devops & Cloud	Apellidos: Ramón	13/02/2025
	Nombre: David	

Explicación de mi fichero main.tf

Declaración de proveedores y versiones:

```
terraform {
  required_providers {
    azurerm = {
      source = "hashicorp/azurerm"
      version = ">= 4.21.1"
    }
  }
}
```

Aquí indico que vamos a utilizar el proveedor **azurerm de HashiCorp** en este proyecto. Este proveedor facilita la interacción con los recursos de Azure y debe ser cualquier versión compatible de **4.21.x**.

Configuración del proveedor de Azure:

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

Aquí estoy configurando el proveedor de **Azure (azurerm)**. La configuración establece parámetros de autenticación como el `subscription_id`, `client_id` y `tenant_id`, que son variables necesarias para la autenticación en Azure, están definidas en el fichero **variables.tf**. y se referencian con la palabra **var.<nombre_variable>**.

El bloque `features {}`: es obligatorio y se deja vacío.

Asignatura	Datos del alumno	Fecha
Devops & Cloud	Apellidos: Ramón	13/02/2025
	Nombre: David	

Creación de un grupo de recursos:

```
resource "azurerm_resource_group" "arg" {
  name     = var.resource_group_name
  location = var.resource_group_location
}
```

Este recurso crea un **grupo de recursos en Azure**, que es un contenedor para otros recursos.

Creación de una red virtual (VNet):

```
resource "azurerm_virtual_network" "avnet" {
  name                = "miAvnet"
  location            = azurerm_resource_group.arg.location
  resource_group_name = azurerm_resource_group.arg.name
  address_space       = ["10.0.0.0/16"]
}
```

Aquí estoy estableciendo una **red virtual (VNet)** que abarca el espacio de direcciones IP de la red con un rango de direcciones 10.0.0.0/16.

Creación de una subred:

```
resource "azurerm_subnet" "asubnet" {
  name                = "miAsubnet"
  resource_group_name = azurerm_resource_group.arg.name
  virtual_network_name = azurerm_virtual_network.avnet.name
  address_prefixes     = ["10.0.1.0/24"]
}
```

Aquí estoy estableciendo una **subred dentro de la VNet** que se creó anteriormente, utilizando el prefijo de direcciones IP 10.0.1.0/24.

Asignatura	Datos del alumno	Fecha
Devops & Cloud	Apellidos: Ramón	13/02/2025
	Nombre: David	

Creación del Azure Container Registry (ACR):

```
resource "azurerm_container_registry" "acr" {
  name                = var.azurerm_container_registry
  location            = var.location_container_registry
  resource_group_name = var.resource_group_name
  sku                 = "Basic"
  admin_enabled       = true
}
```

Aquí estoy creando un **Azure Container Registry (ACR)**, que es un servicio diseñado para almacenar imágenes de contenedores. Se establece el nombre del ACR utilizando la variable **var.azurerm_container_registry**, así como la ubicación y el grupo de recursos. Además, se habilita la autenticación de administrador configurando **admin_enabled** en true.

Creación de un clúster de Kubernetes (AKS):

```
resource "azurerm_kubernetes_cluster" "aks" {
  name                = "miAKS"
  location            = azurerm_resource_group.location
  resource_group_name = azurerm_resource_group.name
  dns_prefix          = "myaks"
  default_node_pool {
    name           = "default"
    node_count     = 1
    vm_size        = "Standard_B2s"
  }
  identity {
    type = "SystemAssigned"
  }
}
```

Aquí estoy creando un **clúster de Kubernetes (AKS)**. Configuro el clúster con un solo nodo y elijo el tipo de **máquina virtual (Standard_B2s)**. También se le asigna una identidad gestionada por Azure a este clúster (**SystemAssigned**).

Asignatura	Datos del alumno	Fecha
Devops & Cloud	Apellidos: Ramón	13/02/2025
	Nombre: David	

Creación de la IP pública dinámica:

```
resource "azurerm_public_ip" "apublic_ip" {
  name                = "milIPPublic"
  location            = azurerm_resource_group.arg.location
  resource_group_name = azurerm_resource_group.arg.name
  allocation_method   = "Static"
  sku                 = "Basic"
  tags = {
    environment = "dev"
  }
}
```

Aquí estoy creando la **IP pública** que se asignará de forma estática. Esta IP sera utilizada por la interfaz de red y la máquina virtual.

Creación de una interfaz de red para la VM:

```
resource "azurerm_network_interface" "anic" {
  name                = "anic-ubuntu"
  location            = var.resource_group_location
  resource_group_name = var.resource_group_name
  ip_configuration {
    name                = "milIPConf"
    subnet_id           = azurerm_subnet.asubnet.id
    public_ip_address_id = azurerm_public_ip.apublic_ip.id
    private_ip_address_allocation = "Dynamic"
  }
}

depends_on = [azurerm_public_ip.apublic_ip]
```

Aquí estoy configurando una **interfaz de red** para la máquina virtual, asociándola a una subred y a una IP pública dinámica.

Asignatura	Datos del alumno	Fecha
Devops & Cloud	Apellidos: Ramón	13/02/2025
	Nombre: David	

Aquí tuve otro problema porque no le daba tiempo a que se creara la IP pública así que tuve que añadir el `depends_on = [azurerm_public_ip.apublic_ip]` para garantizar que la interfaz de red no se cree hasta que la IP pública esté disponible.

Creación de una máquina virtual Ubuntu:

```
resource "azurerm_linux_virtual_machine" "avml" {
  name          = "avml-ubuntu"
  location      = var.resource_group_location
  resource_group_name = var.resource_group_name
  size          = "Standard_B1s"
  admin_username = "ubuntuadmin"
  disable_password_authentication = true
  network_interface_ids = [azurerm_network_interface.anic.id]
  source_image_reference {
    publisher = "Canonical"
    offer     = "0001-com-ubuntu-server-jammy"
    sku       = "22_04-lts"
    version   = "latest"
  }
  os_disk {
    caching          = "ReadWrite"
    storage_account_type = "Standard_LRS"
  }
  admin_ssh_key {
    username   = "ubuntuadmin"
    public_key = var.admin_ssh_key
  }
  tags = {
    environment = "dev"
  }
}
```

Aquí estoy configurando una **máquina virtual Linux (Ubuntu) en Azure**.

Asignatura	Datos del alumno	Fecha
Devops & Cloud	Apellidos: Ramón	13/02/2025
	Nombre: David	

El tamaño de la maquina virtual es **Standard_B1s**, el Acceso SSH lo establezco mediante una clave SSH, utilizando la variable **var.admin_ssh_key** para la clave pública.

Asignación de los permisos

```
resource "azurerm_role_assignment" "acr_owner" {
  principal_id      = azurerm_kubernetes_cluster.aks.identity[0].principal_id
  role_definition_name = "Owner" # Asigno el rol con los permisos completos
  scope             = azurerm_container_registry.acr.id
}
```

Aqui estoy dando los permisos para poder interactuar con mi **ACR**, lo que me permitirá subir imágenes desde la VM usando Podman. Le he dado como permiso **"Owner"** para no tener problemas a la hora de subir mi imagen.

Configuración del grupo de seguridad de red (NSG) para habilitar SSH:

```
resource "azurerm_network_security_group" "ansg" {
  name                = "ansg-test"
  location            = var.resource_group_location
  resource_group_name = var.resource_group_name
  depends_on          = [azurerm_resource_group.arg]
  security_rule {
    name                = "AllowSSH"
    priority            = 1000
    direction          = "Inbound"
    access              = "Allow"
    protocol            = "Tcp"
    source_port_range   = "*"
    destination_port_range = "22"
    source_address_prefix = "*"
    destination_address_prefix = "*"
  }
}
```

Asignatura	Datos del alumno	Fecha
Devops & Cloud	Apellidos: Ramón	13/02/2025
	Nombre: David	

```
security_rule {
  name          = "AllowTCP"
  priority      = 1001
  direction     = "Inbound"
  access        = "Allow"
  protocol      = "Tcp"
  source_port_range = "*"
  destination_port_range = "80"
  source_address_prefix = "*"
  destination_address_prefix = "*"
}
```

Aquí estoy configurando un **grupo de seguridad de red (NSG)** que permite el acceso **SSH** por el **puerto 22** a las máquinas virtuales que se encuentran dentro de la red.

Asociar el NSG con la interfaz de red de la VM:

```
resource "azurerm_network_interface_security_group_association" "nsg_association" {
  network_interface_id      = azurerm_network_interface.anic.id
  network_security_group_id = azurerm_network_security_group.ansg.id
}
```

Aquí estoy vinculando el **grupo de seguridad de red (NSG)** a la interfaz de red de la máquina virtual, permitiendo o restringiendo el tráfico de red de acuerdo con las reglas establecidas.

Variables de salida:

Solamente he puesto 3 variables de salida.

```
output "public_ip" {
  value          = azurerm_public_ip.apublic_ip.ip_address
  description    = "Salida para mostrar mi IP pública y dinámica de la máquina virtual"
}
```


Asignatura	Datos del alumno	Fecha
Devops & Cloud	Apellidos: Ramón	13/02/2025
	Nombre: David	

```
output "acr_name" {
  value      = azurerm_container_registry.acr.name
  description = "Salida para mostrar el nombre de mi ACR"
}
```

```
output "aks_name" {
  value      = azurerm_kubernetes_cluster.aks.name
  description = "Salida para mostrar el nombre de mi AKS"
}
```

Asignatura	Datos del alumno	Fecha
Devops & Cloud	Apellidos: Ramón	13/02/2025
	Nombre: David	

Recursos creados en Azure:

Microsoft Azure

Buscar recursos, servicios y documentos (G+J)

Copilot

Inicio >

Grupos de recursos

UNIR

+ Crear

Administrar vista

Actualizar

Exportar a CSV

Abrir consulta

Asignar etiquetas

Está viendo una nueva versión de la experiencia de exploración. Puede que falten algunas características. Haga clic aquí para acceder a la experiencia anterior.

Filtrar por cualquier ca...

Suscripción es igual a **todo**

Ubicación es igual a **todo**

+ Agregar filtro

<input type="checkbox"/>	Nombre ↑	Suscripción	Ubicación
<input type="checkbox"/>	davidrgn	Azure for Students	Spain Central
<input type="checkbox"/>	MC_davidrgn_miAKS_spaincentral	Azure for Students	Spain Central
<input type="checkbox"/>	NetworkWatcherRG	Azure for Students	Spain Central

davidrgn

Grupo de recursos

Buscar

+ Crear

Administrar grupo de recursos

Eliminar grupo de recursos

Actualizar

Exportar a CSV

Abrir consulta

Asignar etiquetas

Mover

Eliminar

Información general

Registro de actividad

Control de acceso (IAM)

Etiquetas

Visualizador de recursos

Eventos

Configuración

Administración de costos

Supervisión

Automación

Ayuda

Información esencial

Suscripción (mover) : Azure for Students

Id. de suscripción : d3535dc1-cdc5-4932-abad-4cf8d2ad675a

Etiquetas (editar) : Agregar etiquetas

Implementaciones : Sin implementaciones

Ubicación : Spain Central

Recursos

Recomendaciones

Filtrar por cualquier ca...

Tipo es igual a **todo**

Ubicación es igual a **todo**

+ Agregar filtro

Mostrando de 1 a 8 de 8 registros.

Mostrar tipos ocultos

Sin agrupar

<input type="checkbox"/>	Nombre ↑	Tipo ↑	Ubicación ↑
<input type="checkbox"/>	anic-ubuntu	Interfaz de red	Spain Central
<input type="checkbox"/>	ansg-test	Grupo de seguridad de red	Spain Central
<input type="checkbox"/>	avml-ubuntu	Máquina virtual	Spain Central
<input type="checkbox"/>	avml-ubuntu_OsDisk_1_b6c72d69fee04ad99ae2fef9c26d9f90	Disco	Spain Central
<input type="checkbox"/>	davidacr	Registro de contenedor	Spain Central
<input type="checkbox"/>	miAKS	Servicio de Kubernetes	Spain Central
<input type="checkbox"/>	miAvnet	Red virtual	Spain Central
<input type="checkbox"/>	miIPPublic	Dirección IP pública	Spain Central

Actualizar a la SKU estándar: Microsoft recomienda una dirección IP pública de SKU estándar para las cargas de trabajo de producción.

Asociar

Desasociar

Eliminar

Mover

Actualizar

Abrir en un dispositivo móvil

Enviar comentarios

Essentials

Grupo de recursos (mover) : davidrgn

Ubicación (mover) : Spain Central

Suscripción (mover) : Azure for Students

Id. de suscripción : d3535dc1-cdc5-4932-abad-4cf8d2ad675a

SKU : Basic

Nivel : Regional

Dirección IP : 68.221.132.111

Nombre DNS : -

Ámbito de etiqueta de nombre d... : -

Asociado a : anic-ubuntu

Máquina virtual : avml-ubuntu

Preferencia de enrutamiento : Red de Microsoft

Asignatura	Datos del alumno	Fecha
Devops & Cloud	Apellidos: Ramón	13/02/2025
	Nombre: David	

ANSIBLE

Ahora que he creada toda mi **infraestructura** con **Terraform**, utilizaré **Ansible** para configurar e instalar las aplicaciones.

Ansible es una herramienta de automatización de TI que facilita la gestión y configuración de sistemas, aplicaciones y redes de forma eficiente. Con Ansible, podemos llevar a cabo tareas como instalar software, administrar configuraciones, desplegar aplicaciones, orquestar procesos en servidores y dispositivos de manera remota.

Instalación de Ansible en Windows 11

En mi powershell pego el comando:

```
wsl --install
```

```
david@PC-DAVID: ~
PS C:\Users\dramo> wsl --install
Installation : Ubuntu
Ubuntu a été installé.
Lancement de Ubuntu...
Installing, this may take a few minutes...
Please create a default UNIX user account. The username does not need to match your Windows username.
For more information visit: https://aka.ms/wslusers
Enter new UNIX username: david
New password:
Retype new password:
passwd: password updated successfully
Installation successful!
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

Welcome to Ubuntu 24.04.1 LTS (GNU/Linux 5.15.167.4-microsoft-standard-WSL2 x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/pro

System information as of Tue Mar 11 16:00:40 CET 2025

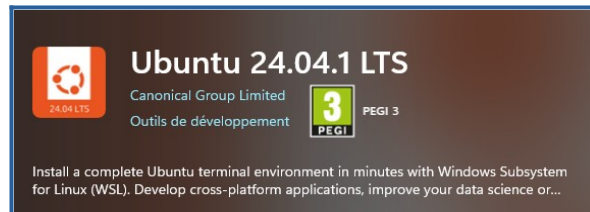
System load:  0.99           Processes:            48
Usage of /:   0.1% of 1006.85GB Users logged in:      0
Memory usage: 9%            IPv4 address for eth0: 192.168.198.126
Swap usage:   0%

This message is shown once a day. To disable it please create the
/home/david/.hushlogin file.
david@PC-DAVID:~$
```

El usuario es: **david** y la contraseña súper segura es: **123456**

Asignatura	Datos del alumno	Fecha
Devops & Cloud	Apellidos: Ramón	13/02/2025
	Nombre: David	

Me voy a la **tienda de Microsoft** y instalo **Ubuntu** en su versión **24.04.1 LTS**



Hago un **sudo apt update** para actualizar la lista de paquetes disponibles desde los repositorios configurados en el sistema.

```
david@PC-DAVID: ~
david@PC-DAVID:~$ sudo apt update
[sudo] password for david:
```

Después hago **sudo apt install software-properties-common** para actualizar las dependencias necesarias.

```
david@PC-DAVID: ~
david@PC-DAVID:~$ sudo apt install software-properties-common
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
software-properties-common is already the newest version (0.99.49.1).
software-properties-common set to manually installed.
0 upgraded, 0 newly installed, 0 to remove and 130 not upgraded.
david@PC-DAVID:~$
```

Ahora agrego el repositorio de Ansible con **sudo add-apt-repository ppa:ansible/ansible**

Asignatura	Datos del alumno	Fecha
Devops & Cloud	Apellidos: Ramón	13/02/2025
	Nombre: David	

```
david@PC-DAVID: ~$ sudo add-apt-repository ppa:ansible/ansible
Repository: 'Types: deb
URIs: https://ppa.launchpadcontent.net/ansible/ansible/ubuntu/
Suites: noble
Components: main
'
Description:
Ansible is a radically simple IT automation platform that makes your applications and systems easier to deploy. Avoid wr
iting scripts or custom code to deploy and update your applications- automate in a language that approaches plain Englis
h, using SSH, with no agents to install on remote systems.
http://ansible.com/
If you face any issues while installing Ansible PPA, file an issue here:
https://github.com/ansible-community/ppa/issues
More info: https://launchpad.net/~ansible/+archive/ubuntu/ansible
Adding repository.
Press [ENTER] to continue or Ctrl-c to cancel.
Hit:1 http://archive.ubuntu.com/ubuntu noble InRelease
Hit:2 http://security.ubuntu.com/ubuntu noble-security InRelease
Hit:3 http://archive.ubuntu.com/ubuntu noble-updates InRelease
Get:4 https://ppa.launchpadcontent.net/ansible/ansible/ubuntu noble InRelease [17.8 kB]
Hit:5 http://archive.ubuntu.com/ubuntu noble-backports InRelease
Get:6 https://ppa.launchpadcontent.net/ansible/ansible/ubuntu noble/main amd64 Packages [776 B]
Get:7 https://ppa.launchpadcontent.net/ansible/ansible/ubuntu noble/main Translation-en [472 B]
Fetched 19.1 kB in 1s (37.1 kB/s)
Reading package lists... Done
david@PC-DAVID: ~$
```

Nuevamente actualizo los paquetes con el comando **sudo apt update** y finalmente instalo Ansible con el comando **sudo apt install ansible**.

```
david@PC-DAVID: ~$ sudo apt install ansible
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  ansible-core python3-jmespath python3-kerberos python3-nacl python3-requests-ntlm python3-resolvelib python3-winrm python3-x
Suggested packages:
  python-nacl-doc python3-gssapi python3-invoke
```

Y finalmente, una vez terminado el proceso, compruebo que todo se haya instalado correctamente con el comando **ansible --version**.

```
david@PC-DAVID: ~$ ansible --version
ansible [core 2.17.9]
  config file = /etc/ansible/ansible.cfg
  configured module search path = ['/home/david/.ansible/plugins/modules', '/usr/share/ansible/plugins/modules']
  ansible python module location = /usr/lib/python3/dist-packages/ansible
  ansible collection location = /home/david/.ansible/collections:/usr/share/ansible/collections
  executable location = /usr/bin/ansible
  python version = 3.12.3 (main, Nov 6 2024, 18:32:19) [GCC 13.2.0] (/usr/bin/python3)
  jinja version = 3.1.2
  libyaml = True
david@PC-DAVID: ~$
```

Asignatura	Datos del alumno	Fecha
Devops & Cloud	Apellidos: Ramón	13/02/2025
	Nombre: David	

Ahora creo el fichero **inventory.ini** desde **Visual Studio Code**:

En este fichero tenemos los datos de las pruebas que he hecho en local y en Azure.

[demo]

Mi grupo demo (Localhost), con la IP 10.171.190.69

10.171.190.69

[miHost]

Mi grupo miHost (AZURE) La IP cambia cada vez que creo los recursos

68.221.196.2

[all:vars]

Variables para los 2 grupos

ansible_ssh_common_args='-o StrictHostKeyChecking=no'

[demo:vars]

Variables para el grupo demo

ansible_user=ubuntu

ansible_ssh_private_key_file=/home/david/.ssh/id_rsa

[miHost:vars]

Variables para el grupo miHost

ansible_ssh_user=ubuntuadmin

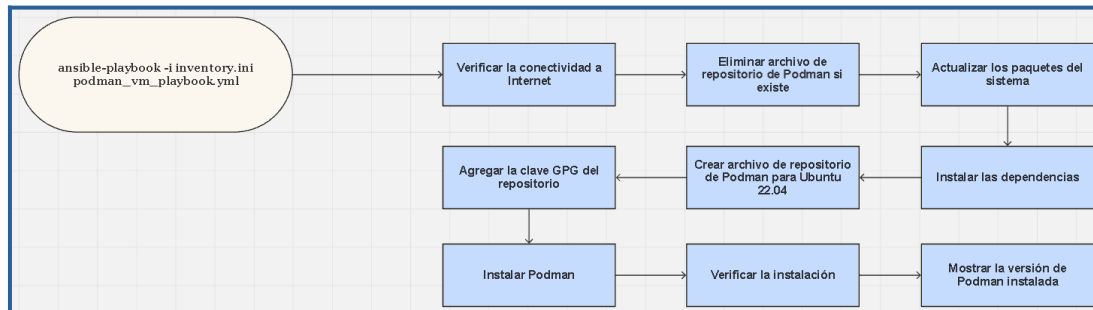
ansible_python_interpreter=/usr/bin/python3.10

ansible_ssh_private_key_file=/home/david/.ssh/id_rsa

Las salidas de las ejecuciones de los playbook de Ansible se encuentran en la carpeta Ansible\Salidas

Asignatura	Datos del alumno	Fecha
Devops & Cloud	Apellidos: Ramón	13/02/2025
	Nombre: David	

Instalación de PODMAN en mi maquina virtual



Ahora voy a **instalar PODMAN** en mi maquina virtual con Ubuntu para poder posteriormente poder ejecutar contenedores.

Tengo que actualizar los paquetes del sistema, agregar la instalación de las dependencias, agregar el repositorio de Podman, importar la clave GPG para el repositorio de Podman, instalar Podman, verificar que la instalación sea ha realizado correctamente y mostrar la versión de Podman que acabamos de instalar.

Utilizo el **Playbook podman_vm_playbook.yml**

Explicación de mi fichero Playbook podman_vm_playbook.yml

Verificar la conectividad a Internet.

Este paso lo he puesto porque he tenido problemas de conexión a Internet.

- name: Verificar conectividad a Internet

uri:

url: "https://www.google.com"

return_content: no

register: google_check

failed_when: google_check.status != 200

Asignatura	Datos del alumno	Fecha
Devops & Cloud	Apellidos: Ramón	13/02/2025
	Nombre: David	

Eliminar archivo de repositorio de Podman si existe.

He tenido también problemas con este fichero, no cogía correctamente, es por eso que lo he añadido el principio para que lo borrara y lo volviera a crear más abajo.

- name: Eliminar archivo de repositorio de Podman si existe

file:

path: /etc/apt/sources.list.d/packages_podman_io_debian_ubuntu_22_04.list

state: absent

Actualizar los paquetes del sistema.

Aquí me aseguro que los paquetes del sistema están actualizados antes de instalar Podman.

- name: Actualizar los paquetes del sistema

apt:

update_cache: yes

upgrade: yes

Instalar las dependencias.

Instalo las dependencias necesarias.

- name: Instalación de dependencias

apt:

name:

- curl

- gnupg2

- lsb-release

- software-properties-common

- apt-transport-https

state: present

update_cache : yes

curl: Sirve para descargar el script de instalación o archivos necesarios.

gnupg2: Sirve para manejar las claves GPG necesarias para validar los repositorios.

lsb-release: Sirve para determinar la versión de la distribución y ajustar los repositorios de manera adecuada.

Asignatura	Datos del alumno	Fecha
Devops & Cloud	Apellidos: Ramón	13/02/2025
	Nombre: David	

software-properties-common: Para permitir la gestión de los repositorios.

apt-transport-https: Sirve para permitir la instalación de paquetes desde repositorios HTTPS.

update_cache: yes: Sirve para actualizar la caché de APT.

Crear archivo de repositorio de Podman para Ubuntu 22.04

- name: Crear archivo de repositorio de Podman para Ubuntu 22.04

copy:

content: |

deb [arch=amd64]

https://download.opensuse.org/repositories/devel:/kubic:/libcontainers:/stable/xUbuntu_22.04/ /

dest: "/etc/apt/sources.list.d/podman_ubuntu_22_04.list"

owner: root

group: root

mode: '0644'

copy: Este módulo lo utilizo para copiar archivos a la máquina remota.

content: Aquí defino el contenido que se copiará al archivo.

dest: Especifico la ruta donde se guardará el archivo en la máquina remota.

owner, group, mode: Establezco el propietario, el grupo y los permisos del archivo.

Agregar la clave GPG del repositorio.

- name: Agregar la clave GPG del repositorio

apt_key:

url: "https://download.opensuse.org/repositories/devel:/kubic:/libcontainers:/stable/xUbuntu_22.04/Release.key"

state: present

apt_key: Este módulo sirve para gestionar las claves GPG de APT.

url: Esta es la URL desde la cual se descargará la clave GPG para autenticar el repositorio.

state: present: Este paso indica que la clave debe ser añadida al sistema.

Asignatura	Datos del alumno	Fecha
Devops & Cloud	Apellidos: Ramón	13/02/2025
	Nombre: David	

Instalar Podman.

Utilizo el módulo apt para instalar el paquete podman .

- name: Instalar Podman

apt:

name: podman

state: present

Verificar la instalación.

Ejecuto el comando podman --version para verificar que la instalación se ha realizado correctamente.

- name: Verificar la instalación de Podman

command: podman --version

register: podman_version

changed_when: false

Mostrar la versión de Podman instalada.

Al Final muestro un mensaje con la versión de Podman se acaba de instalada.

- name: Mostrar la versión de Podman instalada

debug:

msg: "La versión de Podman instalada es: {{ podman_version.stdout }}"

Salida de la ejecución:

Aquí muestro la parte de la salida de la ejecución del playbook de Ansible (podman_vm_playbook.yml) donde dice que Podman se ha instalado correctamente en la VM de Azure y la recapitulación de la ejecución.

La salida completa se encuentra en la carpeta Ansible\Salidas.

TASK [Mostrar la versión de Podman instalada]

ok: [68.221.196.2] => {

"msg": "La versión de Podman instalada es: podman version 3.4.4"

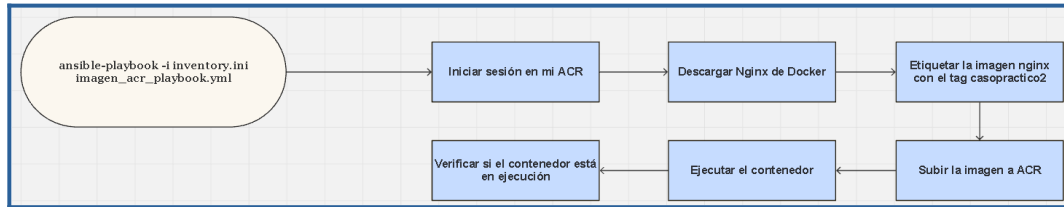
}

PLAY RECAP

68.221.196.2 : ok=10 changed=5 unreachable=0 failed=0 skipped=0 rescued=0 ignored=0

Asignatura	Datos del alumno	Fecha
Devops & Cloud	Apellidos: Ramón	13/02/2025
	Nombre: David	

Creación de un nuevo contenedor con PODMAN desde mi VM



Voy a crear un **nuevo contenedor** con Podman desde mi maquina virtual. Voy a escoger un servidor de **NGINX**. Después de tagearlo como **casopractico2** lo ejecuto.

Importo mi fichero de variables.

En el fichero **imagen_acr_playbook.yml**, para poder utilizar mis variables, tengo que importar mi fichero donde están definidas.

vars_files:

- variables.yml

Descripción de mi fichero variables.yml.

En este fichero tengo las variables que utilizaré en los playbooks de Ansible.

mi_acr: "davidacr"

mi_usuario: "davidacr.azurecr.io"

mi_contrasenya: "xlb5lpoWoZgZFoQ4xZWjc6feC2iuvm0ttaZlSWSSKi+ACRCIXUEX"

image_name: "nginx:latest"

tagged_image: "casopractico2"

img_redis_name: "redis"

#Variables para el AKS

aks_cluster_name: "miAKS"

aks_resource_group: "davidrgn"

acr_name: "davidacr"

redis_image_name: "redis" ACR

redis_version: "latest"

namespace: "default"

dns_prefix: "myaks"

Asignatura	Datos del alumno	Fecha
Devops & Cloud	Apellidos: Ramón	13/02/2025
	Nombre: David	

Aquí he tenido unos problemas porque se me olvido cambiar la contraseña al volver a crear los recursos y me daba error al iniciar la sesión en mi ACR desde Ansible:

TASK [Iniciar sesión en ACR]

fatal: [68.221.196.2]: FAILED! => {"changed": false, "failed_when_result": true, "msg":

"Unable to gather info for davidacr.azurecr.io: Error: error logging into

\\"davidacr.azurecr.io\\": invalid username/password\\n"}

Iniciar sesión en ACR.

Inicio la sesión en mi ACR con mis credenciales.

- name: Iniciar sesión en ACR

containers.podman.podman_login:

registry: "{{ mi_usuario }}"

username: "{{ mi_acr }}"

password: "{{ mi_contrasenya }}"

register: podman_login

failed_when: podman_login.failed

containers.podman.podman_login: Es un módulo de Ansible que se usa para iniciar sesión en el ACR con Podman.

registry: La URL de tu ACR que se encuentra en la variable mi_usuario

username: El nombre de usuario para el ACR que se encuentra en la variable mi_acr

password: La contraseña para el ACR que se encuentra en la variable mi_contrasenya

register: Guarda el resultado de esta tarea en la variable podman_login.

Asignatura	Datos del alumno	Fecha
Devops & Cloud	Apellidos: Ramón	13/02/2025
	Nombre: David	

Descargar la imagen nginx desde Docker Hub.

Descargo la imagen de repositorio de Docker

- name: Descargar imágenes desde Docker Hub

```
containers.podman.podman_image:
```

```
  name: "{{ item.name }}"
```

```
  tag: "{{ item.tag | default('latest') }}"
```

```
  state: present
```

```
  pull: yes
```

```
  loop:
```

```
    - { name: "docker.io/nginx", tag: "latest" }
```

containers.podman.podman_image: Este módulo permite manejar imágenes de contenedores con Podman.

name: Especifico el nombre de la imagen que voy descargar. El valor se obtiene de la lista en el loop, donde cada item.name es una imagen específica. En este caso solamente tengo **nginx**.

tag: La etiqueta de la imagen que usaré es latest para descargar la versión más reciente.

state: present: Esto asegura que la imagen esté presente en el sistema. Si no está, se descargará.

pull: yes: Esta opción indica que las imágenes deben descargarse haciendo un pull.

loop: El loop se usa para iterar sobre una lista de imágenes, descargándola.

Solamente tengo la imagen de:

- nginx:latest

Asignatura	Datos del alumno	Fecha
Devops & Cloud	Apellidos: Ramón	13/02/2025
	Nombre: David	

Etiquetar la imagen nginx con el tag casopractico2.

Etiqueto la imagen que acabo de descargar con el tag **casopractico2**.

- name: Etiquetar imágenes descargadas para ACR

```
containers.podman.podman_tag:
  image: "{{ item.name }}"
  target_names: "{{ mi_usuario }}/{{ item.name }}:{{ tagged_image }}"
loop:
  - { name: "nginx" }
```

containers.podman.podman_tag: Este módulo permite tagear la imagen descargada.

image: El nombre de la imagen que se quiere etiquetar.

target_names: Este es nuevo es nuevo nombre y tag para la imagen.

loop: El loop aplica el tageado casopractico2 en la imagen:

- nginx

Subir la imagen a ACR:

- name: Subir imágenes a ACR

```
containers.podman.podman_image:
  name: "{{ mi_usuario }}/{{ item.name }}"
  tag: "{{ tagged_image }}"
  push: yes
loop:
  - { name: "nginx" }
```

containers.podman.podman_image: Este módulo permite gestionar imágenes de contenedores aquí vamos a subirlas a mi ACR.

name: El nombre completo de la imagen que se va a subir con mi mi_usuario y item.name.

tag: El tag de la imagen que se está subiendo que he definido anteriormente.

push: yes: Indica a Podman que hacer un push de la imagen al ACR.

loop: El loop sube la imagen a mi ACR de :

- nginx

Asignatura	Datos del alumno	Fecha
Devops & Cloud	Apellidos: Ramón	13/02/2025
	Nombre: David	

Ejecutar el contenedor.

Utilizo el módulo **podman_container** para ejecutar el contenedor con la imagen etiquetada. Con **state: started** iniciamos el contenedor, y el parámetro **detach: yes** lo ejecuta en segundo plano. El puerto 80 del contenedor está mapeado al puerto 80 de la máquina virtual.

- name: Ejecutar el contenedor nginx desde ACR

```
containers.podman.podman_container:
  name: nginx_container
  image: "{{ mi_usuario }}/nginx:{{ tagged_image }}"
  state: started
  restart_policy: always
  ports:
    - "80:80"
```

containers.podman.podman_container: Este módulo se utiliza para gestionar contenedores en Podman.

name: Especifico el nombre del contenedor que voy a crear (nginx_container).

image: Especifico la imagen que se voy a ejecutar. Es la imagen que acabo de subir a mi ACR.

state: started: Indica que el contenedor debe iniciarse. Si ya está en ejecución, no hace nada.

restart_policy: always: Esto asegura que el contenedor se reinicie automáticamente si se detiene por alguna razón.

ports: Mapeo el puerto 80 dentro del contenedor al puerto 80 del host.

Salida de la ejecución:

Aquí muestro la parte de la salida de la ejecución del playbook de Ansible (imagen_acr_playbook.yml) donde dice que he descargado y subido correctamente mi imagen nginx a mi ACR.

La salida completa se encuentra en la carpeta Ansible\Salidas

TASK [Ejecutar el contenedor nginx desde ACR]

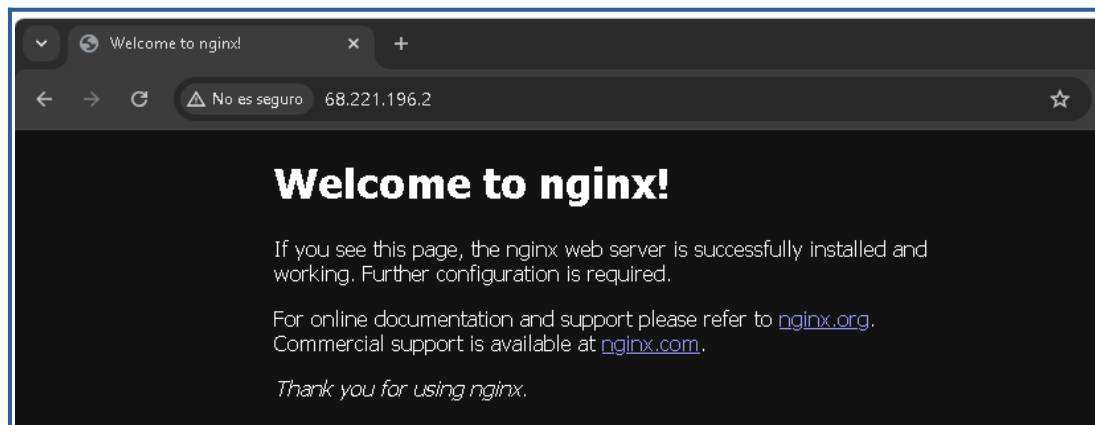
changed: [68.221.196.2]

Asignatura	Datos del alumno	Fecha
Devops & Cloud	Apellidos: Ramón	13/02/2025
	Nombre: David	

PLAY RECAP

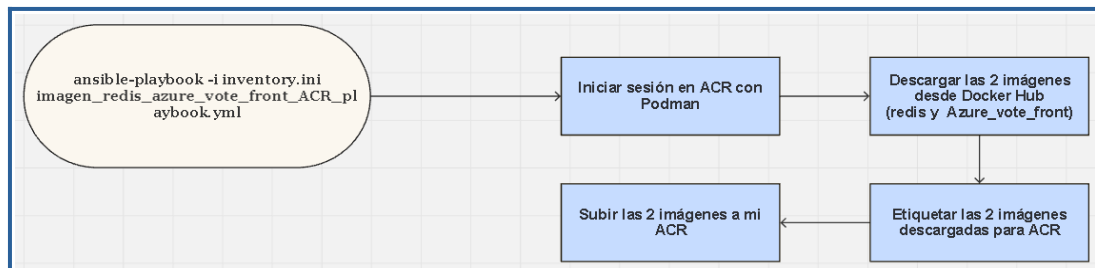
68.221.196.2 : ok=6 changed=5 unreachable=0 failed=0 skipped=0 rescued=0 ignored=0

Voy a mi navegador para confirmar que mi servidor nginx esta funcionando correctamente:



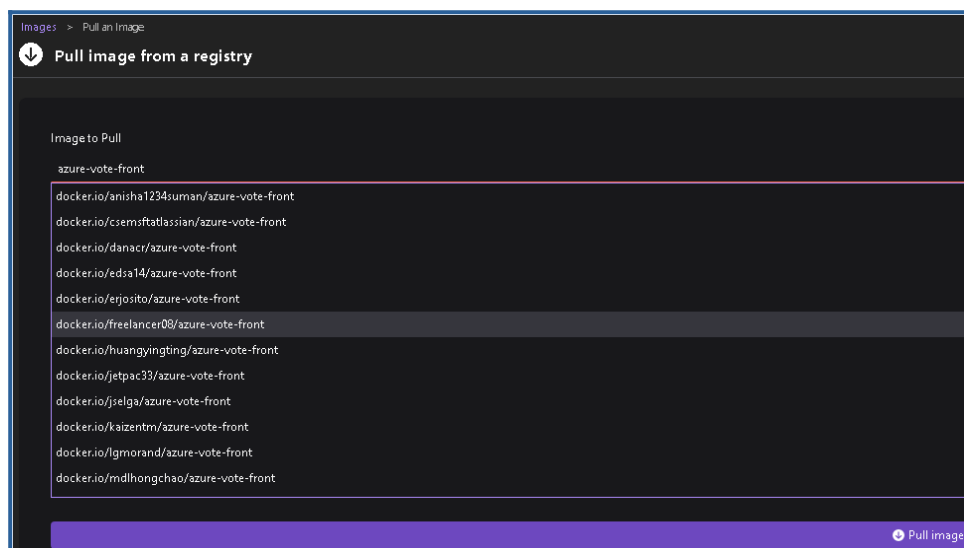
Asignatura	Datos del alumno	Fecha
Devops & Cloud	Apellidos: Ramón	13/02/2025
	Nombre: David	

Para la aplicación con persistencia, he decido utilizar **Nginx**, **Redis** y **Azure_Vote_Front** como mencionaste durante las clases.



Como la imagen de **nginx** ya la tengo en mi ACR, solamente descargo, tagueo y subo a mi ACR las imagenes de **Redis** y **Azure_Vote_Front**.

La imagen de **azure_vote_front** no le he encontrado en **Docker** y al buscarla en **Podman** me han salido varios usuarios que la tienen. He optado por uno de ellos (**freelancer08**).



Asignatura	Datos del alumno	Fecha
Devops & Cloud	Apellidos: Ramón	13/02/2025
	Nombre: David	

Explicación del fichero imagen_redis_azure_vote_front_ACR_playbook.yml

Iniciar sesión en ACR con Podman.

Inicio la sesión con mis credenciales que se encuentran en mi fichero de variables para poder subir mis imágenes a mi ACR.

- name: Iniciar sesión en ACR con Podman

containers.podman.podman_login:

registry: "{{ mi_usuario }}"

username: "{{ mi_acr }}"

password: "{{ mi_contrasenya }}"

register: podman_login

failed_when: podman_login.failed

containers.podman.podman_login: Este módulo permite autenticarte en el ACR utilizando Podman con los datos de registry, username y password. Estos son los parámetros necesarios para la autenticación.

register: podman_login: Guarda el resultado en la variable podman_login.

failed_when: podman_login.failed: Si la tarea de login falla, la ejecución del playbook se detendrá con un error.

Descargar imágenes desde Docker Hub (repositorio público):

Me descargo las 2 imágenes de Docker.

- name: Descargar imágenes desde Docker Hub (repositorio público)

containers.podman.podman_image:

name: "{{ item.name }}"

tag: "{{ item.tag | default('latest') }}"

state: present

pull: yes

loop:

- { name: "docker.io/library/redis", tag: "latest" }

- { name: "docker.io/freelancer08/azure-vote-front", tag: "latest" }

Asignatura	Datos del alumno	Fecha
Devops & Cloud	Apellidos: Ramón	13/02/2025
	Nombre: David	

containers.podman.podman_image: Este módulo permite manejar imágenes de contenedores con Podman.

name: Especifico el nombre de la imagen que voy descargar. El valor se obtiene de la lista en el loop, donde cada item.name es una imagen específica.

tag: La etiqueta de la imagen que usaré es latest para descargar la versión más reciente.

state: present: Esto asegura que la imagen esté presente en el sistema. Si no está, se descargará.

pull: yes: Esta opción indica que las imágenes deben descargarse haciendo un pull.

loop: El loop se usa para iterar sobre una lista de imágenes, descargándolas. Las 2 imágenes que se descargaran son:

- redis:latest
- azure-vote-front:latest

Etiquetar imágenes descargadas para ACR

Aquí tagueo mis imágenes con **casopractico2**.

- name: Etiquetar imágenes descargadas para ACR

containers.podman.podman_tag:

image: "{{ item.name }}"

target_names: "{{ mi_usuario }}/{{ item.name }}:{{ tagged_image }}"

loop:

- { name: "redis" }

- { name: "azure-vote-front" }

containers.podman.podman_tag: Este módulo permite taguear las imágenes descargadas.

image: El nombre de la imagen que se quiere etiquetar.

target_names: Este es nuevo es nuevo nombre y tag para la imagen.

loop: El loop aplica el tagueado **casopractico2** a las tres imágenes:

- redis,
- azure-vote-front.

Asignatura	Datos del alumno	Fecha
Devops & Cloud	Apellidos: Ramón	13/02/2025
	Nombre: David	

Subir imágenes a ACR.

Aquí voy a subir las 2 imágenes creadas a mi ACR.

- name: Subir imágenes a ACR

containers.podman.podman_image:

name: "{{ mi_usuario }}/{{ item.name }}"

tag: "{{ tagged_image }}"

push: yes

loop:

- { name: "redis" }

- { name: "azure-vote-front" }

containers.podman.podman_image: Este módulo permite gestionar imágenes de contenedores aquí vamos a subirlas a mi ACR.

name: El nombre completo de la imagen que se va a subir con mi mi_usuario y item.name.

tag: El tag de la imagen que se está subiendo que he definido anteriormente.

push: yes: Indica a Podman que hacer un push de la imagen al ACR.

loop: El loop sube las tres imágenes a mi ACR:

- redis,
- azure-vote-front

Asignatura	Datos del alumno	Fecha
Devops & Cloud	Apellidos: Ramón	13/02/2025
	Nombre: David	

Creación del fichero de secretos:

Este archivo es un manifiesto de Kubernetes que establece un recurso Secret, diseñado para guardar de manera segura las credenciales que se necesitan para acceder a un repositorio privado de imágenes Podman. En este caso, el repositorio se encuentra en Azure Container Registry (ACR).

Los datos se codifican en Base64 y se guardan en el fichero de secretos.

Usuario:

```
echo -n "davidacr.azurecr.io" | base64
```

Contraseña:

```
echo -n "xlb5lpoWoZgZFoQ4xZWjc6feC2iuvm0ttaZISWSSKi+ACRCIXUEX" | base64
```

Explicación del fichero acr-secret.yaml

```
apiVersion: v1
```

```
kind: Secret
```

```
metadata:
```

```
  name: acr-secret
```

```
  namespace: default
```

```
type: kubernetes.io/dockerconfigjson
```

```
data:
```

```
  .dockerconfigjson: |
```

```
{
```

```
  "auths": {
```

```
    "davidacr.azurecr.io": {
```

```
      "username": "ZGF2aWRhY3luYXp1cmVjci5pbw==",
```

```
      "password":
```

```
"eElINWxwb1dvWmdaRm9RNHhaV2pjNmZlQzJpdXZtMHR0YVpsU1dTU0tpK0FDUKNsWFVF
WA==",
```

```
      "email": "david.ramon578@comunidadunir.net"
```

```
    }
```

```
  }
```

```
}
```

Asignatura	Datos del alumno	Fecha
Devops & Cloud	Apellidos: Ramón	13/02/2025
	Nombre: David	

apiVersion: v1 Especifica la versión de la API de Kubernetes que se está utilizando.

kind: Secret Define que el tipo de recurso que estoy creando es un Secret.

Metadata Esta sección proporciona metadatos para el recurso. En este caso:

- **name: acr-secret:** El nombre del Secret.
- **namespace: default:** El Secret con el namespace default.

type: kubernetes.io/dockerconfigjson Este campo especifica el tipo de Secret que estoy creando. El tipo kubernetes.io/dockerconfigjson se utiliza para guardar las credenciales necesarias para acceder a un registro de Docker. En mi caso, estoy accediendo al ACR.

Data Esta sección contiene los datos reales del Secret en formato base64. En Kubernetes, los datos sensibles se almacenan de forma codificada en base64.

.dockerconfigjson Aquí defino el contenido del archivo de configuración dockerconfig.json utilizado por Podman para autenticar al cliente Podman con un registro privado.

auths: Aquí se especifican las configuraciones de autenticación para los registros Podman.

- **davidacr.azurecr.io:** Es el nombre del registro privado del ACR.
 - **username:** Es el nombre de usuario para autenticarse en el registro. Está en base64.
 - **password:** La contraseña también está codificada en base64
 - **email:** El correo electrónico asociado a la cuenta no está codificado y es visible directamente.

Aplicar el Secret a Kubernetes

Con este comando aplicamos el **Secret** a Kubernetes:

kubectl apply -f acr-secret.yaml

Asignatura	Datos del alumno	Fecha
Devops & Cloud	Apellidos: Ramón	13/02/2025
	Nombre: David	

Verificar que el Secret se ha creado correctamente

Nos podemos asegurar de que el **Secret** se ha creado correctamente con el comando siguiente:

```
kubect! get secrets acr-secret --namespace=default
```

Creación de un fichero para despliegue.

Creo el fichero **deployment.yaml** que utilizará el Secret que hemos creado para autenticar el pull de imágenes desde tu Azure Container Registry (ACR).

Aplicar el archivo Deployment.yaml al clúster de Kubernetes

Para aplicar el fichero **Deployment.yaml** al clúster de Kubernetes, utilizamos este comando:

```
kubect! apply -f Deployment.yaml
```

Explicación de mi fichero deployment.yaml

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: mi-aplicacion
spec:
  replicas: 1
  selector:
    matchLabels:
      app: mi-aplicacion
  template:
    metadata:
      labels:
        app: mi-aplicacion
    spec:
      containers:
        - name: redis
```

Asignatura	Datos del alumno	Fecha
Devops & Cloud	Apellidos: Ramón	13/02/2025
	Nombre: David	

image: davidacr.azurecr.io/redis:casopractico2

ports:

- containerPort: 6379

- name: azure-vote-front

image: davidacr.azurecr.io/azure-vote-front:casopractico2

ports:

- containerPort: 80

imagePullSecrets:

- name: acr-secret

apiVersion: apps/v1: Define la versión de la API de Kubernetes que se está utilizando.

kind: Deployment: Define que el tipo de recurso que estás creando es un **Deployment**, que es un objeto que asegura la disponibilidad y el manejo de las réplicas de los contenedores en los Pods.

metadata: Contiene metadatos sobre el recurso:

- **name: mi-aplicacion:** El nombre del Deployment que es mi-aplicacion

spec: Aquí defino la especificación del Deployment.

- **replicas: 1:** Este campo define cuántas réplicas de los contenedores se deben ejecutar. Solamente pongo 1.

selector: Este campo es fundamental para que Kubernetes pueda identificar correctamente los Pods que están bajo la gestión de este Deployment. Es importante que los Pods tengan una etiqueta que coincida con app: mi-aplicacion.

template: Define el **Pod** que se creará dentro del Deployment. Los Pods son la unidad básica de ejecución de contenedores en Kubernetes.

- **metadata:** Aquí defino las etiquetas del Pod. En este caso, se aplica la etiqueta app: mi-aplicacion, lo que hace que el Deployment pueda gestionar este Pod.
- **spec:** Define la especificación de los contenedores dentro del Pod.
 - **containers:** Especifica los contenedores que estarán en el Pod.
 - **name: redis:** 1er contenedor es **Redis**.

Asignatura	Datos del alumno	Fecha
Devops & Cloud	Apellidos: Ramón	13/02/2025
	Nombre: David	

- **image: davidacr.azurecr.io/redis:casopractico2:** Se especifica la imagen de Redis que se usará, la cual está almacenada en mi ACR.
 - **ports:** Expongo el puerto 6379 dentro del contenedor, que es el puerto por defecto de Redis.
- **name: azure-vote-front:** 2º contenedor es **azure-vote-front**.
- **image: davidacr.azurecr.io/azure-vote-front:casopractico2:** Se especifica la imagen de azure-vote-front que se utilizará, almacenada en el mismo ACR.
 - **ports:** Exponemos el puerto 80 dentro del contenedor.
- **imagePullSecrets:** Esta parte es clave porque le indica a Kubernetes que debe usar un Secret para autenticar la descarga de imágenes desde un repositorio privado. En este contexto:
 - **name: acr-secret:** El secreto llamado **acr-secret** tiene las credenciales que necesitas para acceder a el ACR y descargar las 2 imágenes.

Exponer el servicio.

Mi aplicación necesita ser accesible desde fuera del clúster de Kubernetes en mi navegador para eso, tengo que exponer un servicio.

Para esta prueba, creo un Service de tipo LoadBalancer.

Explicación del archivo service.yaml

```

apiVersion: v1
kind: Service
metadata:
  name: mi-aplicacion-service
spec:
  selector:
    app: mi-aplicacion
  ports:
    - protocol: TCP
      port: 80
      targetPort: 80
  type: LoadBalancer

```

Asignatura	Datos del alumno	Fecha
Devops & Cloud	Apellidos: Ramón	13/02/2025
	Nombre: David	

apiVersion: v1 Especifica la versión de la API de Kubernetes que estoy utilizando para crear el recurso.

kind: Service Este recurso que estoy creando es del tipo Service, y su función es ofrecer una capa de abstracción que permite a los pods comunicarse entre sí. Así, las aplicaciones pueden intercambiar información a través de una dirección IP fija y un puerto.

Metadata:

- **name:** Es el nombre del servicio que estoy creando.

Spec: La sección spec define la configuración y las especificaciones de cómo el servicio debería funcionar.

- **Selector:** El **selector** le dice al servicio a qué **pods** se debe dirigir el tráfico.
 - **app: mi-aplicacion** es por donde el servicio enviará el tráfico a todos los pods donde **label app** sea igual a mi-aplicacion.
- **Ports:** Esta sección define los puertos en los que el servicio estará disponible.
 - **protocol: TCP** Define el protocolo que usará el servicio es el TCP.
 - **port:** El servicio escuchará en el puerto 80 dentro del clúster de Kubernetes.
 - **targetPort:** Es el puerto en el que el contenedor está escuchando dentro de los pods. .

type: LoadBalancer El tipo de servicio que se creará es un LoadBalancer.

Asignatura	Datos del alumno	Fecha
Devops & Cloud	Apellidos: Ramón	13/02/2025
	Nombre: David	

LINKS UTILIZADOS

Azure cli:

<https://learn.microsoft.com/en-us/cli/azure/install-azure-cli-windows?pivots=msi>

Podman:

<https://podman.io/docs/installation>

<https://docs.podman.io/en/latest/Commands.html>

Terraform:

<https://developer.hashicorp.com/terraform/install>

<https://developer.hashicorp.com/terraform/docs>

Git:

<https://github.com/DavidRO71/TareaUnirAzure.git>

Microsoft Store:

Microsoft Store para instalar **Ubuntu** en **Windows**

Ansible:

<https://docs.ansible.com/>

Kubernetes:

<https://kubernetes.io/docs/home/>

<https://kubernetes.io/docs/concepts/configuration/secret/>

Diagrama:

<https://miro.com/>