Terraform Task

**Name: Shaik Khaja Basha**

**Batch : Batch 11**

**Date : 26.07.2025**

**Task : Azure CLI**

**Azure CLI**

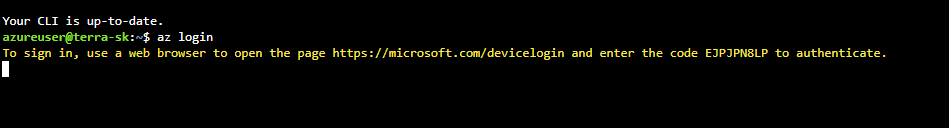
The Azure Command-Line Interface (CLI) is Microsoft's cross-platform command-line tool for managing Azure resources. It allows you to create, manage, and monitor Azure resources from the command line or in scripts.

* Its helps to interact with Azure services to manage resources

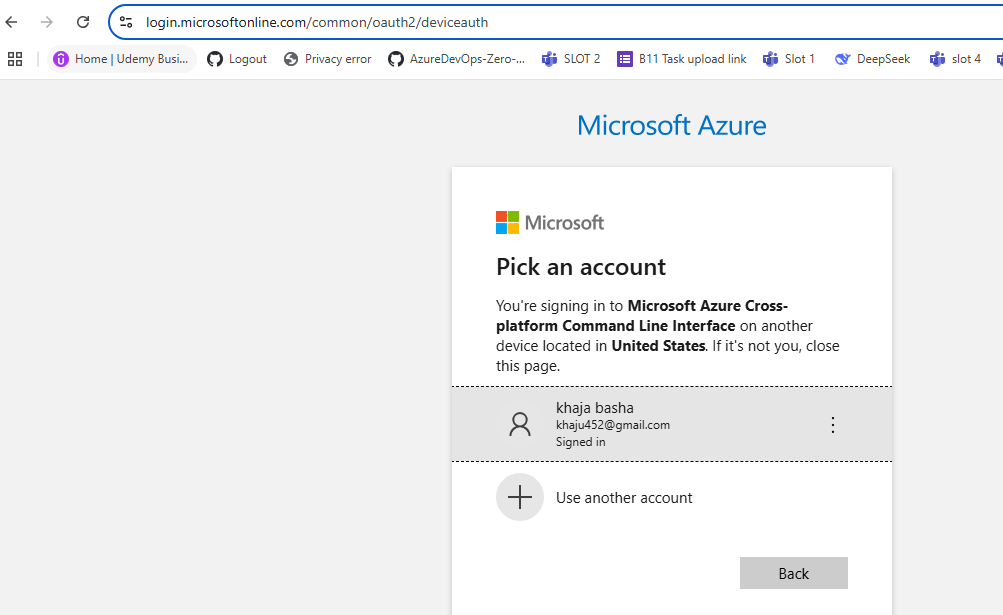
**Basic Commands**

Login

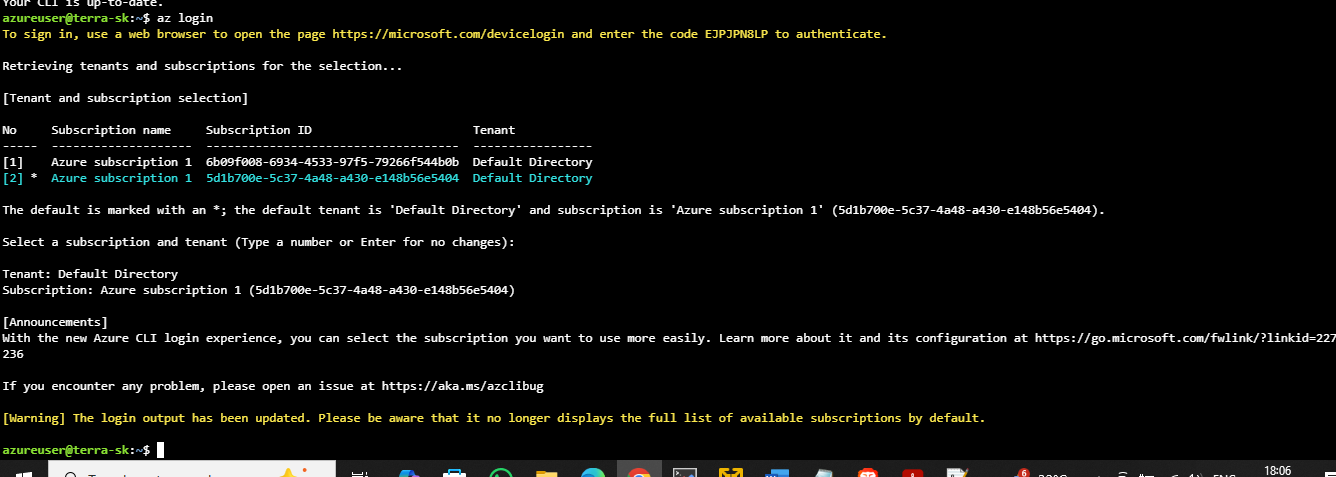
az login



* When we write az login it will create a link with code
* Copy the code and click the URL it will open a window and login with credentials



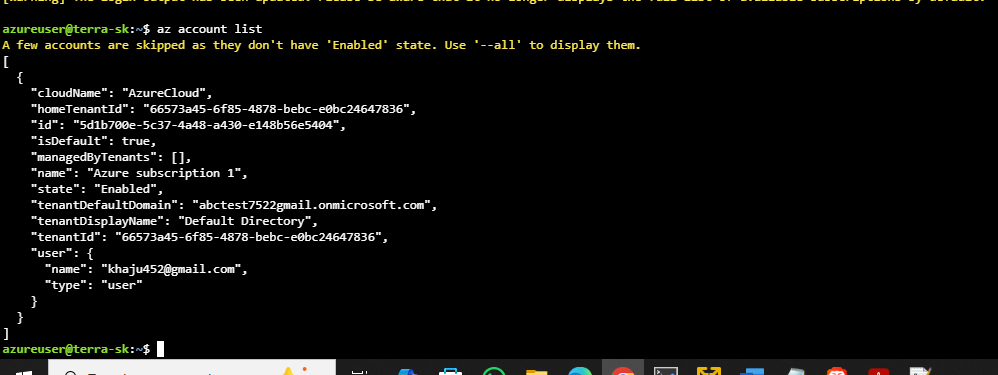
As soon as login it will give me the tenants which will be there



List subscriptions

az account list

which will show what are the account subscription

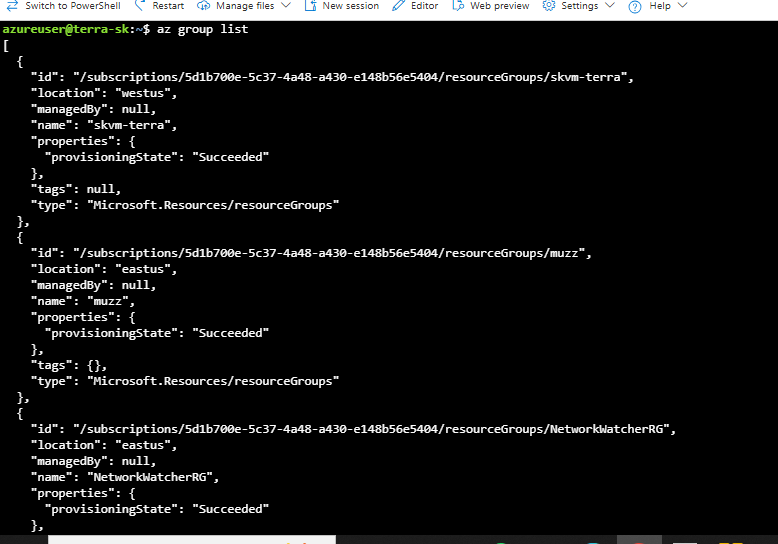


Set default subscription

az account set --subscription "My Subscription Name"

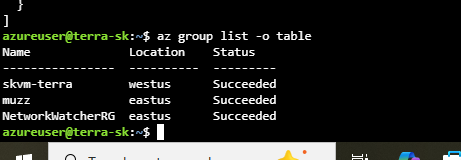
List resource groups

az group list



Which will give me as a JSON format if I want to see as table then

Az group list -o table

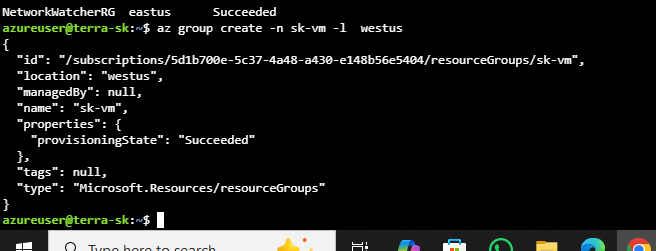


Which is easy for understanding

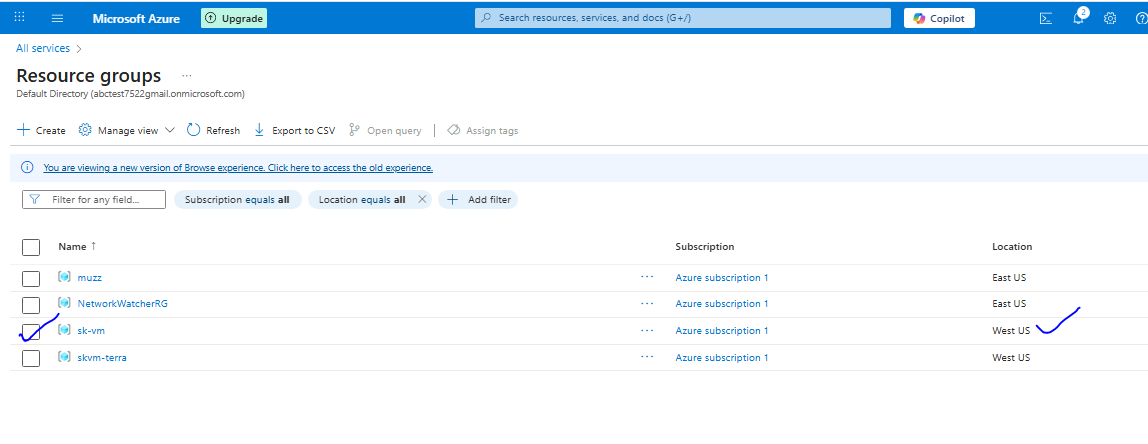
**Create a resource group**

az group create --name MyResourceGroup --location eastus

az group create -n sk-vm -l westus

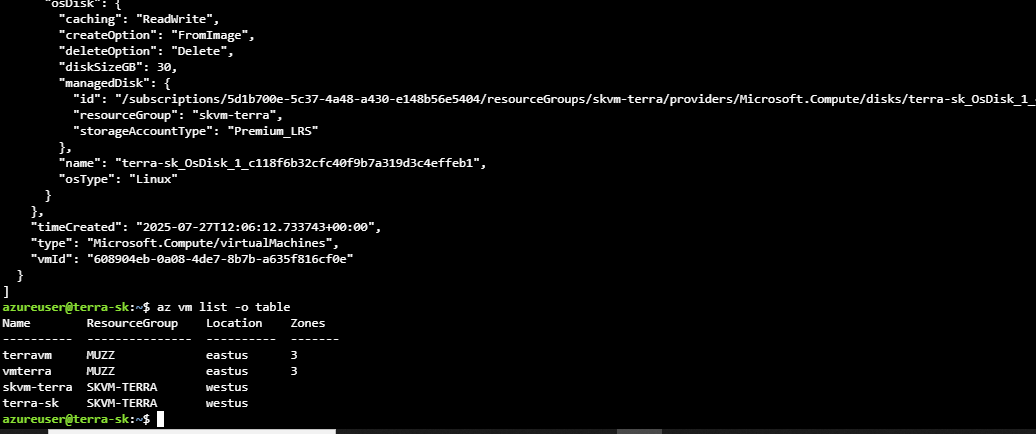


We can check whether its created or not though GUI



List virtual machines

az vm list



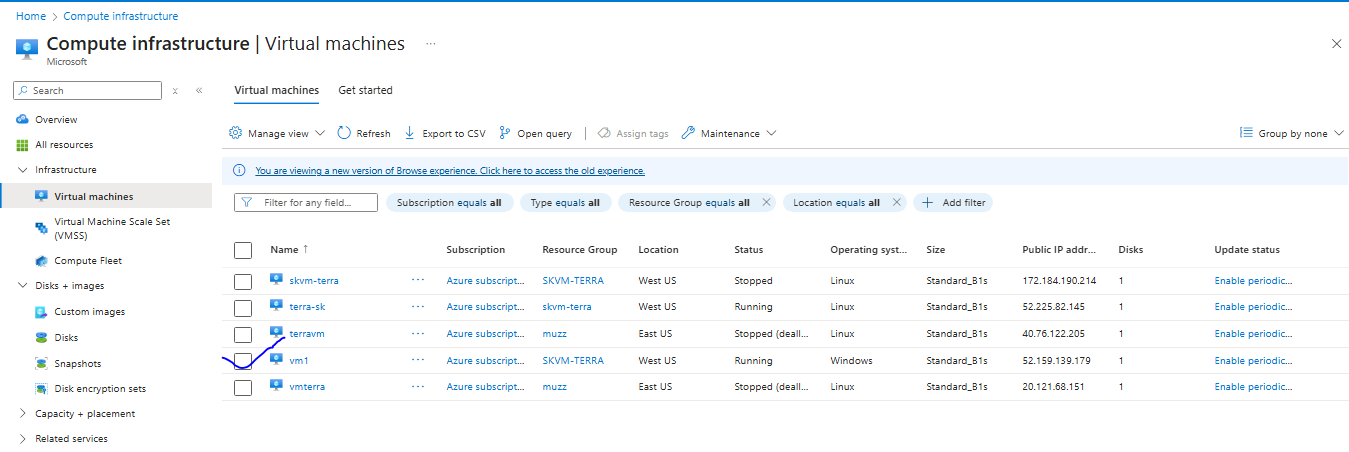
We can start or stop the vm through cli

az vm start -g <rgname> -n <vmname>

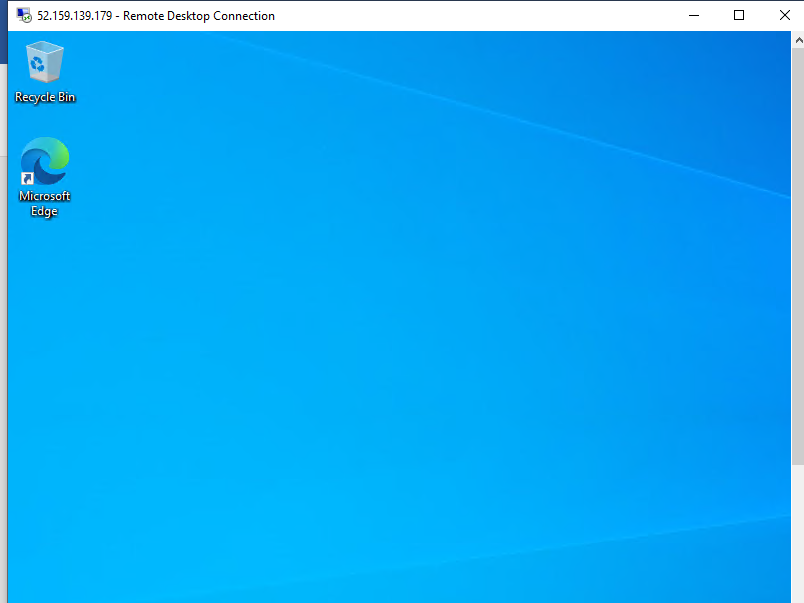
az vm show -g <rgname> -n <vmname>

**creating a vm**

az vm create -g test -n vm1



We can see a vm is created by using azure cli



**Create azure active directory(AAD) or entra id**

Checking Existing Directories

az ad signed-in-user show --query 'userPrincipalName'

az account list --query '[].tenantId' -o tsv

**lets connect azure to terraform**

az ad sp create-for-rbac --role="Contributor" --scopes="/subscriptions/5d1b700e-5c37-4a48-a430-e148b56e5404"

if its get fail we can use export command with required details

{

"appId": " 1ba6dbc4-1815-4d1d-b576-ef9060254d0a ",

"displayName": "5d1b700e-5c37-4a48-a430-e148b56e5404",

"password": " BsT8Q~knW3dH2~C5wlZDf8aibZdIel7xEUocSbYN ",

"tenant": "66573a45-6f85-4878-bebc-e0bc24647836"

}

subscription\_id = " 5d1b700e-5c37-4a48-a430-e148b56e5404"

client\_id = " 1ba6dbc4-1815-4d1d-b576-ef9060254d0a"

client\_secret = " BsT8Q~knW3dH2~C5wlZDf8aibZdIel7xEUocSbYN"

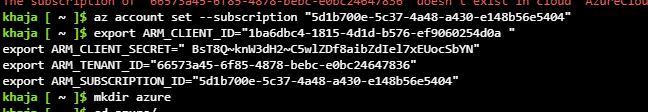
tenant\_id = " 66573a45-6f85-4878-bebc-e0bc24647836"

export ARM\_CLIENT\_ID="1ba6dbc4-1815-4d1d-b576-ef9060254d0a "

export ARM\_CLIENT\_SECRET=" BsT8Q~knW3dH2~C5wlZDf8aibZdIel7xEUocSbYN"

export ARM\_TENANT\_ID="66573a45-6f85-4878-bebc-e0bc24647836"

export ARM\_SUBSCRIPTION\_ID="5d1b700e-5c37-4a48-a430-e148b56e5404"



Create a directory mkdir azure

Create a resource provider file vi res.tf

terraform {

required\_providers {

azurerm = {

source = "hashicorp/azurerm"

version = "4.37.0"

}

}

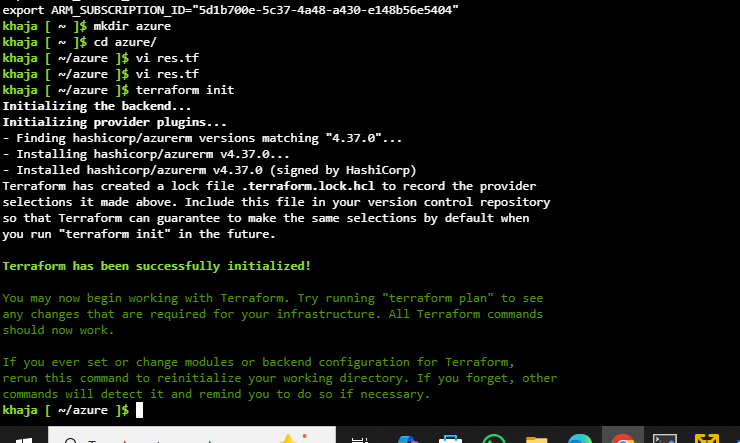
}

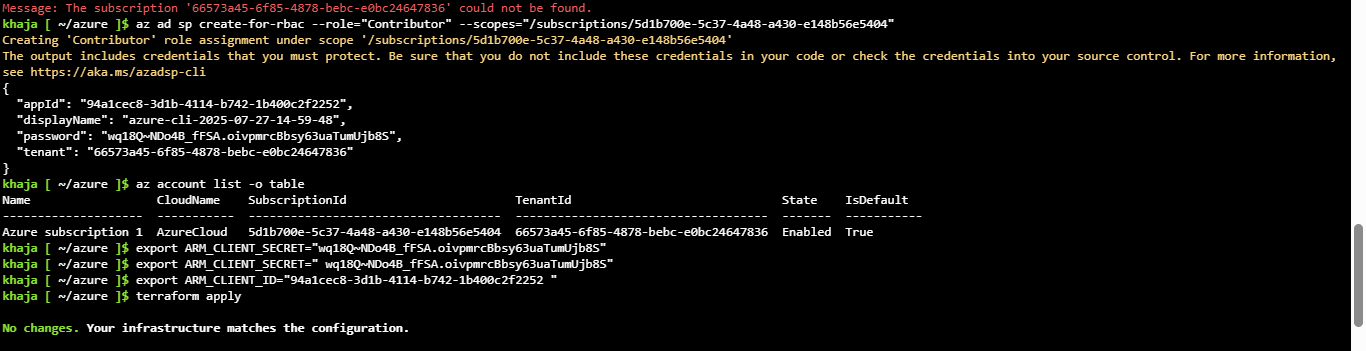
provider "azurerm" {

features {}

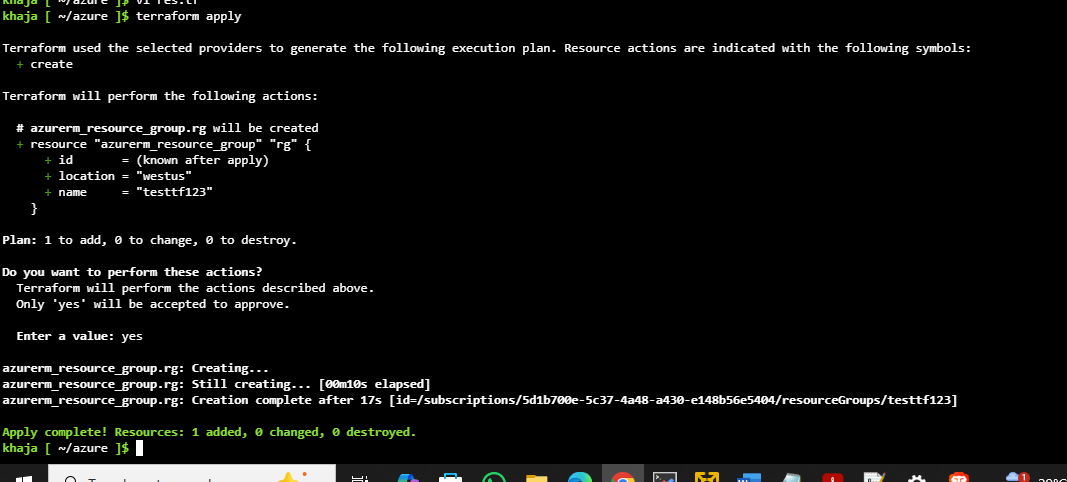
}

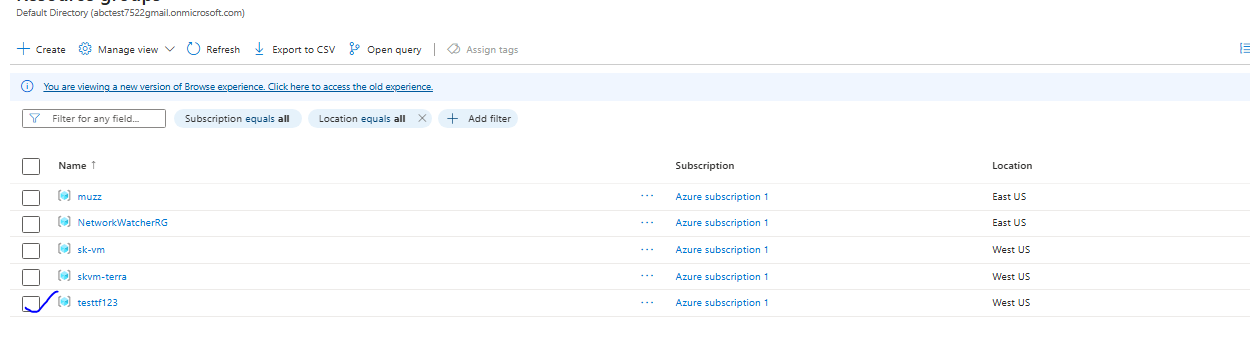
Then we initialized the terraform init





When we do terraform apply the resources are created





Here we can see the resource has been created

If I don’t want to give export everytime we can provide in our configuration once

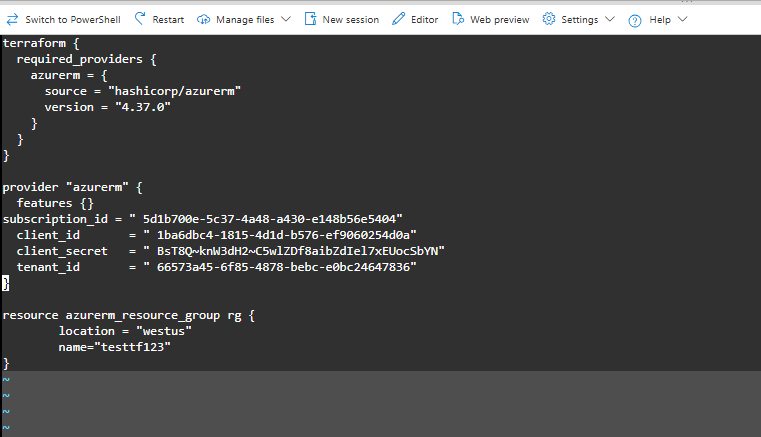
subscription\_id = "bccafdc0-1217-4202-883e-2c0dbce2e3cb"

client\_id = "90638e0f-524e-45ea-b751-7f200d04a9c9"

client\_secret = "35M8Q~sGvaujHKbwi0jDgLxkMvj7VX0PqnSYgabq"

tenant\_id = "c9947813-2c41-49f2-937e-98e030d55bae"

}



Which is not preferable always use environmental variables