Assignment 28: Authenticate Terraform to Azure using Service Principal + Client Secret

Objective :

Configure Terraform to authenticate to Azure without your personal login by creating and using a Service Principal (SP) with a client secret. You will:

- Create a Service Principal with least-privilege RBAC
- Export credentials as environment variables for Terraform
- Verify end-to-end by provisioning a small Azure resource (RG) using the SP
- Learn secure handling and rotation practices.

What you will do :

- Create an SP scoped to your subscription (role: Contributor by default; adjust if needed)
- Store Client ID, Client Secret, Tenant ID, Subscription ID
- Set ARM_* environment variables Terraform uses
- Run terraform init/plan/apply without Azure CLI login
- Tear down and clean up

Prerequisites :

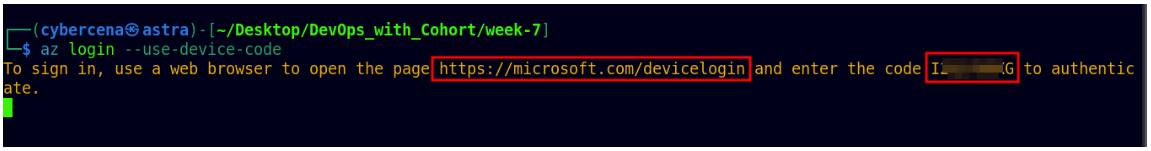
- Active Azure subscription (same as previous assignments:**Azure Free**)
- Azure CLI installed and able to az login (only to create the SP)
- Terraform installed (v1.5+ recommended)
- macOS/Linux shell or Windows PowerShell

Reference for Solution :https://www.youtube.com/watch?v=SVPjxy4em24

Solution:

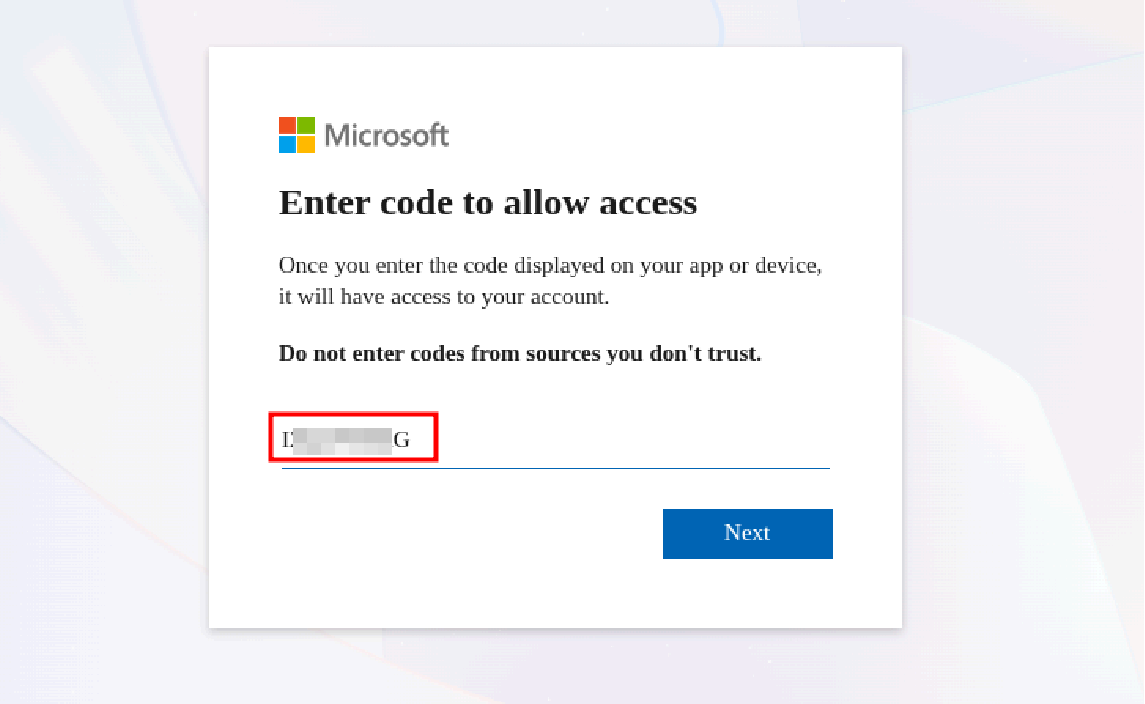
Step 1 : Setup Azure CLI with Service Principal.

- Open Terminal in MacOS/Linux or Powershell in Windows and enter the command : az login --use-device-code .



```
┌──(cybercena㉿astra)-[~/Desktop/DevOps_with_Cohort/week-7]
└─$ az login --use-device-code
To sign in, use a web browser to open the page https://microsoft.com/devicelogin and enter the code [redacted] to authentic
ate.
```

- Copy the Link and Visit your Favourite Browser and Enter the code provided in the terminal.



Microsoft

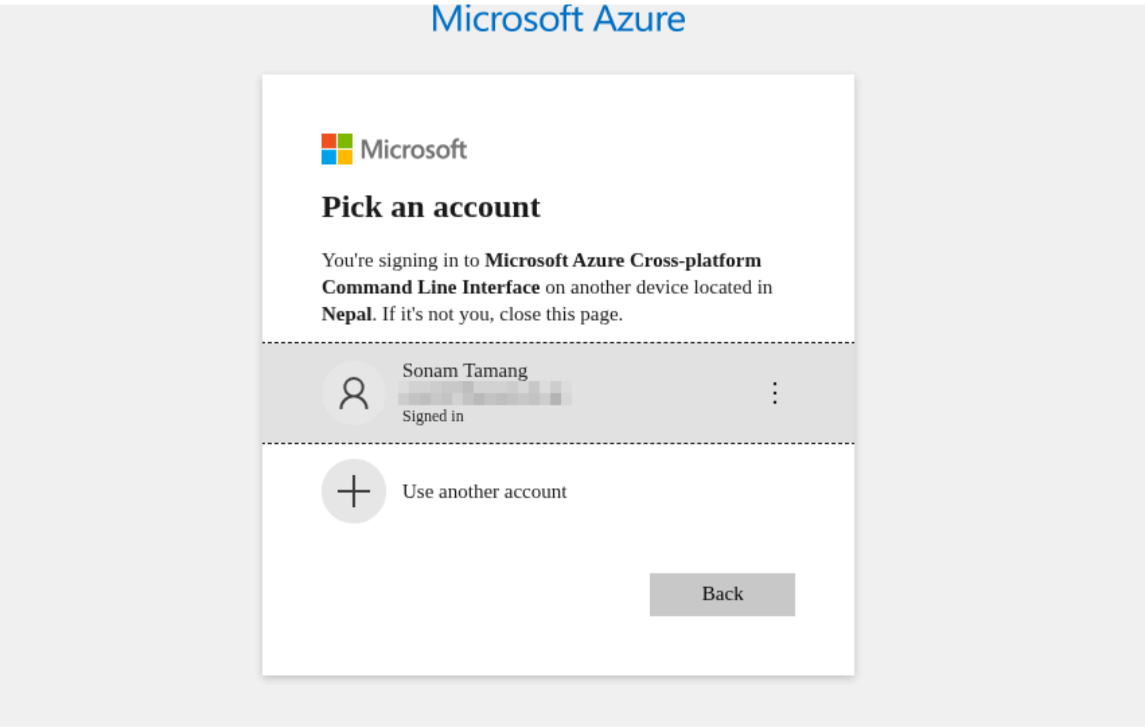
Enter code to allow access

Once you enter the code displayed on your app or device, it will have access to your account.

Do not enter codes from sources you don't trust.

[redacted]

- Select the Azure account you want to use for Project and **confirm** to use Azure CLI.



Microsoft Azure

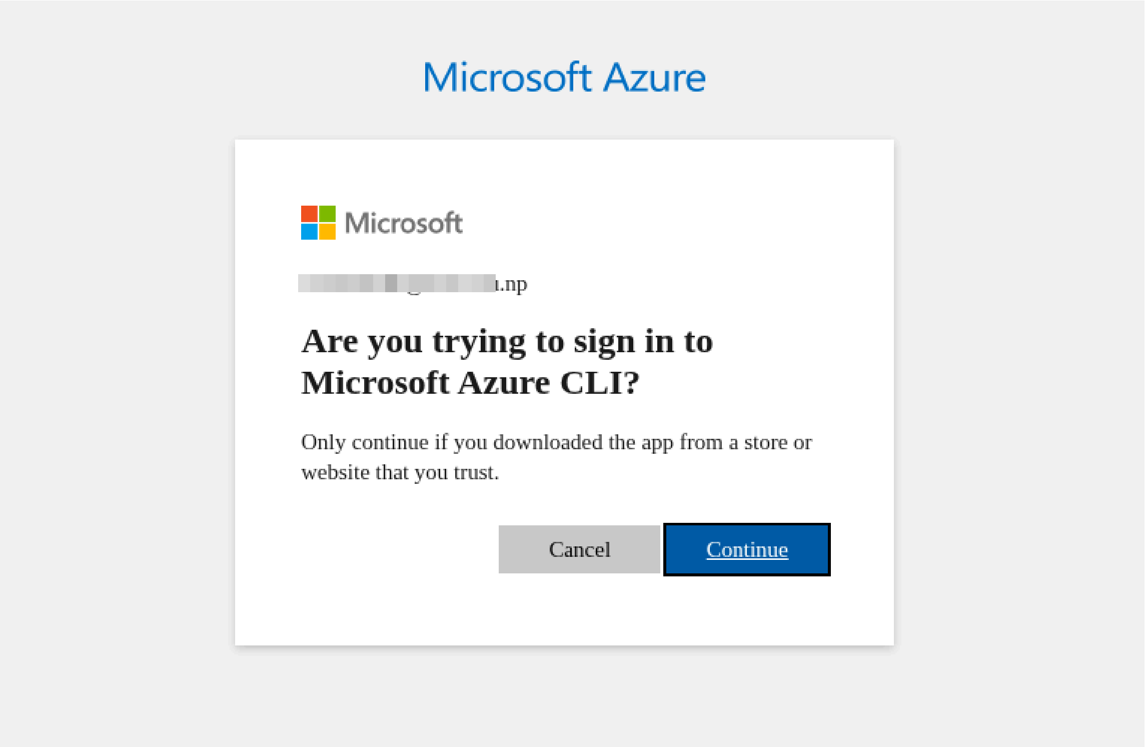
Microsoft

Pick an account

You're signing in to **Microsoft Azure Cross-platform Command Line Interface** on another device located in **Nepal**. If it's not you, close this page.

Sonam Tamang
[redacted]
Signed in

Use another account



Microsoft Azure

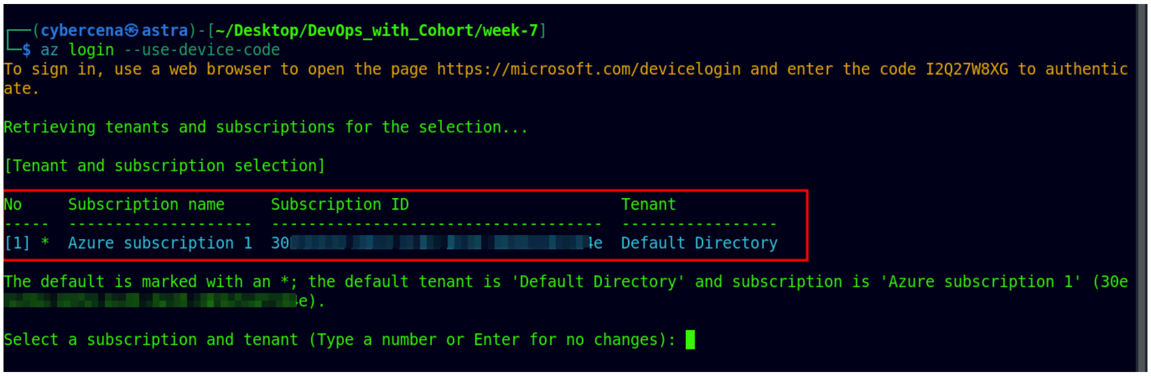
Microsoft

[redacted].np

Are you trying to sign in to Microsoft Azure CLI?

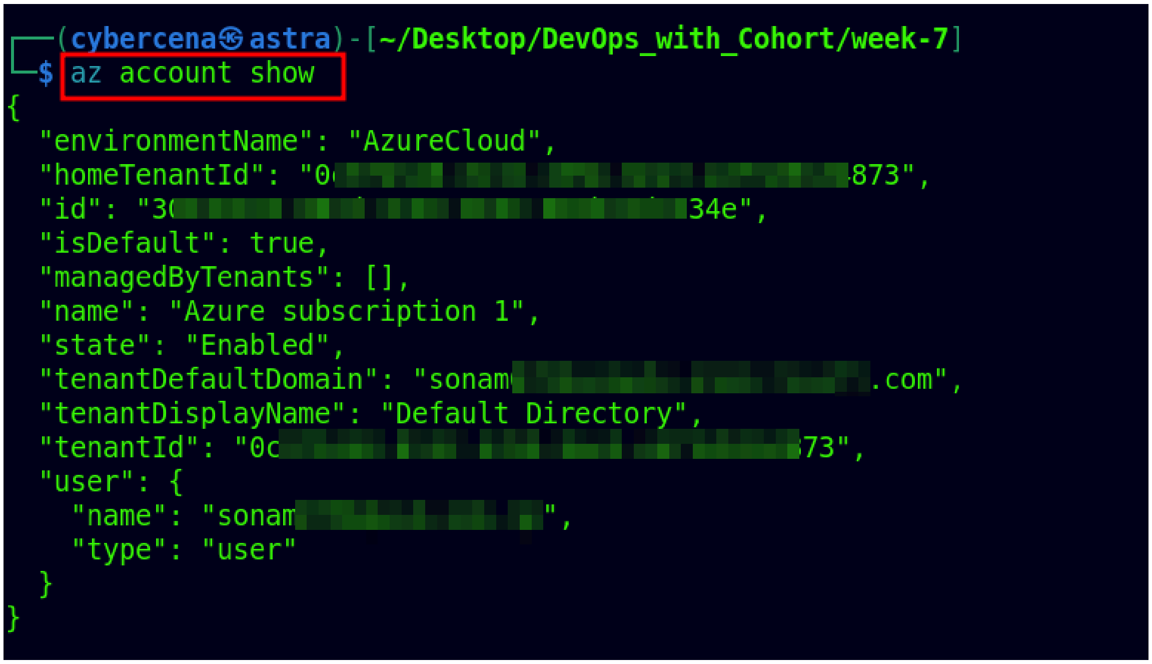
Only continue if you downloaded the app from a store or website that you trust.

- Check the Terminal, you will see the Subscription details if you are logged in.



```
┌──(cybercena㉿astra)-[~/Desktop/DevOps_with_Cohort/week-7]
└─$ az login --use-device-code
To sign in, use a web browser to open the page https://microsoft.com/devicelogin and enter the code I2Q27W8XG to authentic
ate.

Retrieving tenants and subscriptions for the selection...

[Tenant and subscription selection]

No     Subscription name     Subscription ID                             Tenant
-----  --------------------  ------------------------------------------  -----------------
[1] *  Azure subscription 1  30[redacted]4e  Default Directory

The default is marked with an *; the default tenant is 'Default Directory' and subscription is 'Azure subscription 1' (30e
[redacted]e).

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

- Select the subscription or press 'Enter' to select by default.
- Enter the command **az account show** in the terminal to check the account info and copy the **subscription id** .



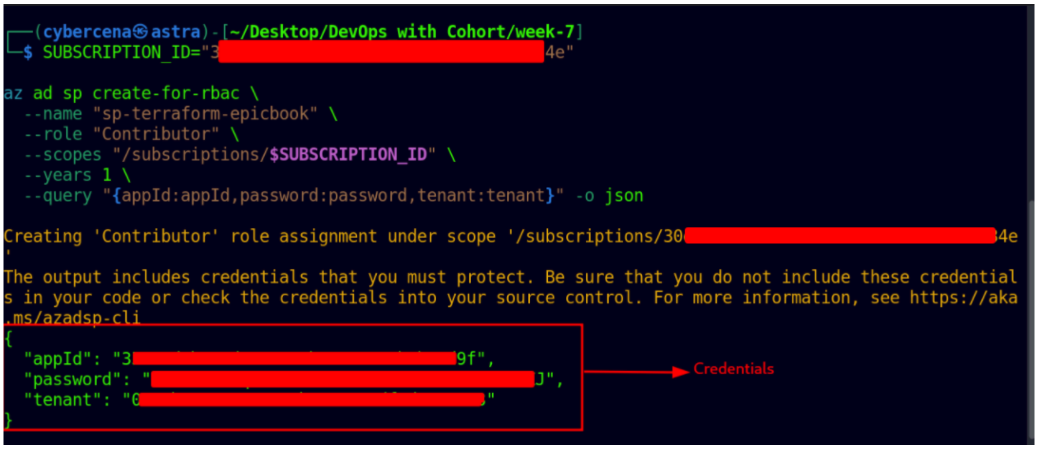
```
┌──(cybercena㉿astra)-[~/Desktop/DevOps_with_Cohort/week-7]
└─$ az account show
{
  "environmentName": "AzureCloud",
  "homeTenantId": "0[redacted]873",
  "id": "3[redacted]34e",
  "isDefault": true,
  "managedByTenants": [],
  "name": "Azure subscription 1",
  "state": "Enabled",
  "tenantDefaultDomain": "sonam[redacted].com",
  "tenantDisplayName": "Default Directory",
  "tenantId": "0c[redacted]73",
  "user": {
    "name": "sonam[redacted]",
    "type": "user"
  }
}
```

Step 2 : Create a Service Principal with RBAC.

- Copy the Subscription id and prepare a command to create a service principal with RBAC (Role Based Access Control).

```
SUBSCRIPTION_ID="<your-subscription-id>"

az ad sp create-for-rbac \
  --name "sp-terraform-epicbook" \
  --role "Contributor" \
  --scopes "/subscriptions/$SUBSCRIPTION_ID" \
  --years 1 \
  --query "{appId:appId,password:password,tenant:tenant}" -o json
```

- Using the above command will generate the appId, tenant Id and Password.



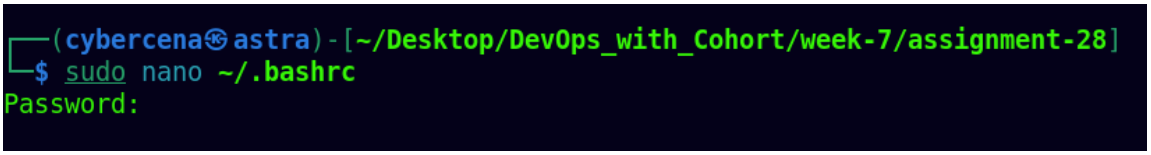
```
┌──(cybercena㉿astra)-[~/Desktop/DevOps with Cohort/week-7]
└─$ SUBSCRIPTION_ID="3[redacted]4e"

az ad sp create-for-rbac \
  --name "sp-terraform-epicbook" \
  --role "Contributor" \
  --scopes "/subscriptions/$SUBSCRIPTION_ID" \
  --years 1 \
  --query "{appId:appId,password:password,tenant:tenant}" -o json

Creating 'Contributor' role assignment under scope '/subscriptions/30[redacted]4e
'
The output includes credentials that you must protect. Be sure that you do not include these credential
s in your code or check the credentials into your source control. For more information, see https://aka
.ms/azadsp-cli
{
  "appId": "3[redacted]9f",
  "password": "[redacted]J",
  "tenant": "0[redacted]"
}
```

Step 3 : Save the credentials as an environment variable.

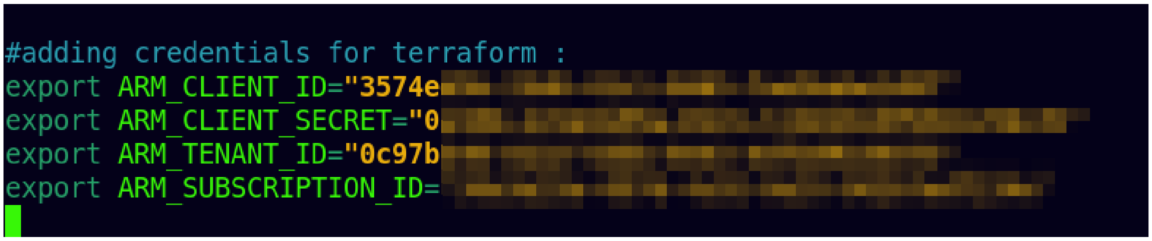
- If you are using Linux, add the below content with real credentials to **~/.bashrc** file.

  Command : nano ~/.bashrc

```
export ARM_CLIENT_ID="<appId>"
export ARM_CLIENT_SECRET="<password>"
export ARM_TENANT_ID="<tenant>"
export ARM_SUBSCRIPTION_ID="$SUBSCRIPTION_ID"
```



```
┌──(cybercena㉿astra)-[~/Desktop/DevOps_with_Cohort/week-7/assignment-28]
└─$ sudo nano ~/.bashrc
Password:
```

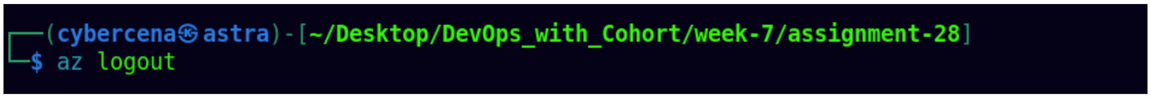


```
#adding credentials for terraform :
export ARM_CLIENT_ID="3574e[redacted]
export ARM_CLIENT_SECRET="0[redacted]
export ARM_TENANT_ID="0c97b[redacted]
export ARM_SUBSCRIPTION_ID=[redacted]
```

- **Run the command : source ~/.bashrc**

Step 4 : Log out Azure CLI (to prove Terraform uses SP)

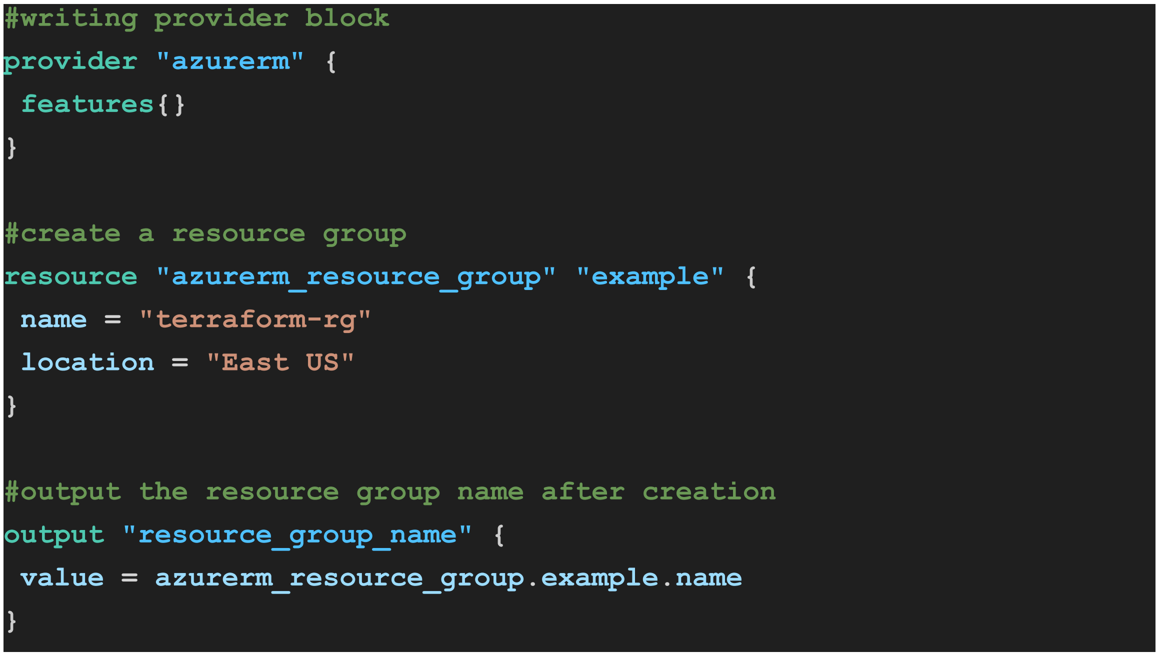
- **Command : az logout**



```
┌──(cybercena㉿astra)-[~/Desktop/DevOps_with_Cohort/week-7/assignment-28]
└─$ az logout
```

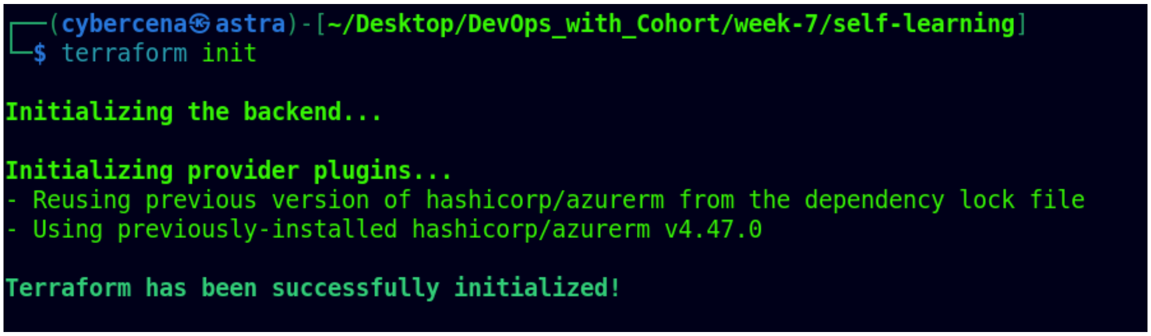
Step 5 : Test if the service principal is working or not.

- **Create a Terraform script with .tf extension.**



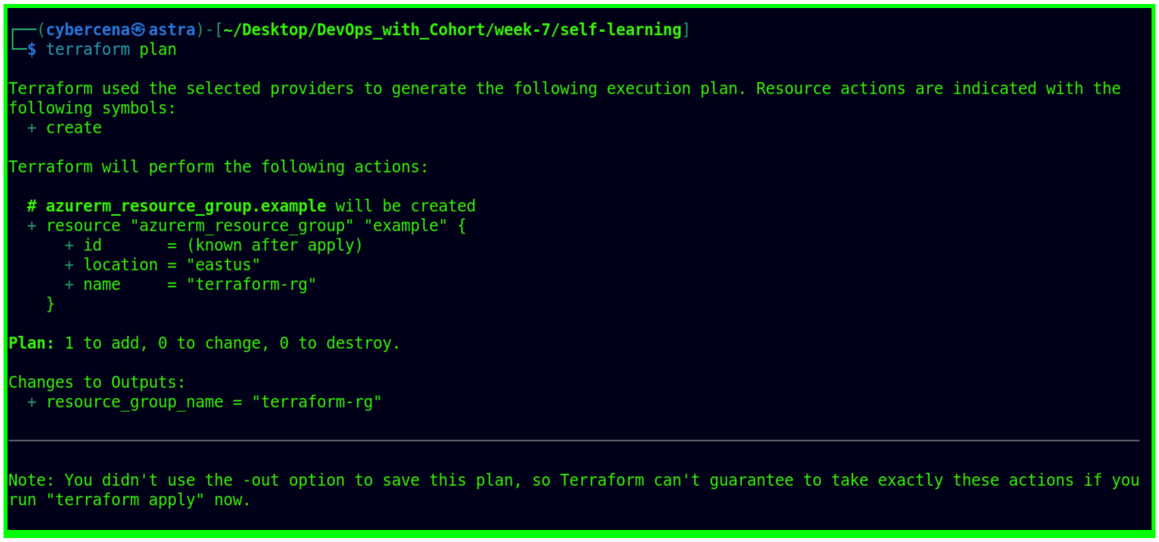
```
#writing provider block
provider "azurerm" {
 features{}
}

#create a resource group
resource "azurerm_resource_group" "example" {
 name = "terraform-rg"
 location = "East US"
}

#output the resource group name after creation
output "resource_group_name" {
 value = azurerm_resource_group.example.name
}
```

- **Initialize a new Terraform directory by using command : terraform init**



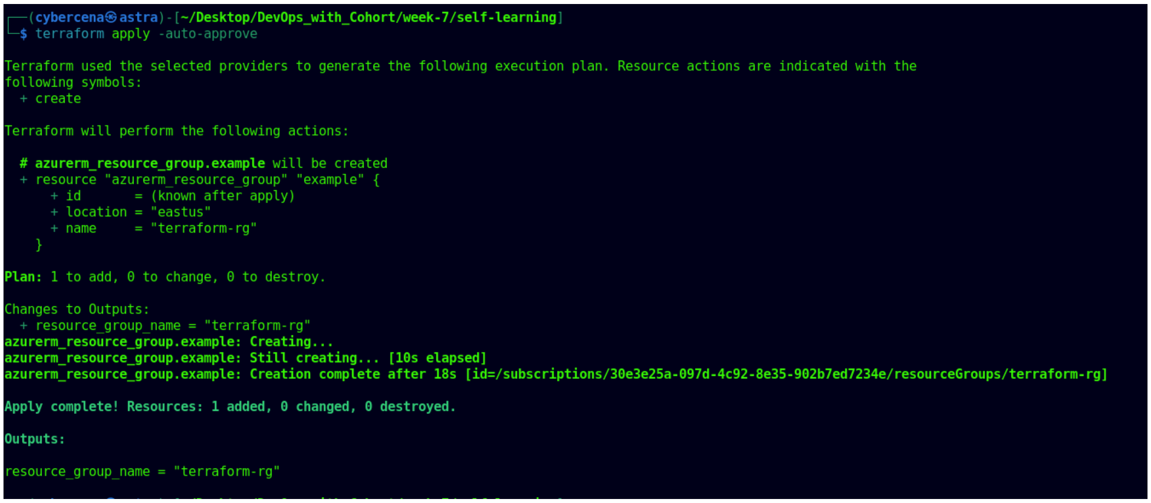
```
┌──(cybercena㉿astra)-[~/Desktop/DevOps_with_Cohort/week-7/self-learning]
└─$ terraform init

Initializing the backend...

Initializing provider plugins...
- Reusing previous version of hashicorp/azurerm from the dependency lock file
- Using previously-installed hashicorp/azurerm v4.47.0

Terraform has been successfully initialized!
```

- **Create an execution plan by using the command : terraform plan**

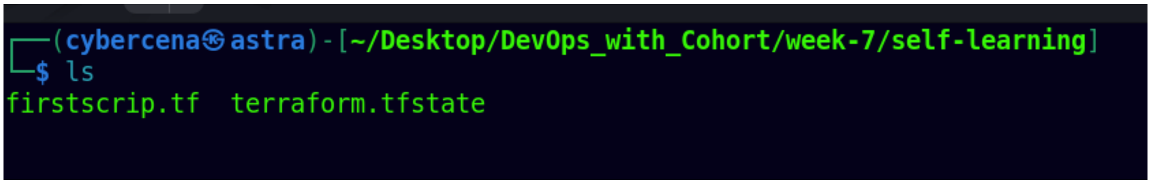


```
┌──(cybercena㉿astra)-[~/Desktop/DevOps_with_Cohort/week-7/self-learning]
└─$ terraform plan

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_resource_group.example will be created
  + resource "azurerm_resource_group" "example" {
      + id       = (known after apply)
      + location = "eastus"
      + name     = "terraform-rg"
    }

Plan: 1 to add, 0 to change, 0 to destroy.

Changes to Outputs:
  + resource_group_name = "terraform-rg"

─────────────────────────────────────────────────────────────────────────────────────────────────────────────────────────

Note: You didn't use the -out option to save this plan, so Terraform can't guarantee to take exactly these actions if you
run "terraform apply" now.
```

- **Apply the changes defined in plan by using the command : terraform apply -auto-approve .**

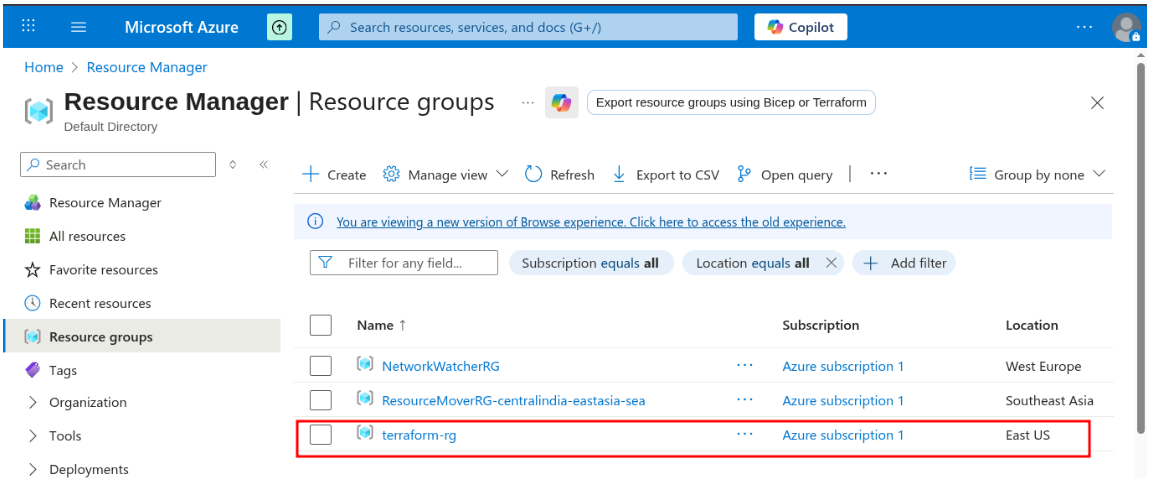


```
┌──(cybercena㉿astra)-[~/Desktop/DevOps_with_Cohort/week-7/self-learning]
└─$ terraform apply -auto-approve

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_resource_group.example will be created
  + resource "azurerm_resource_group" "example" {
      + id       = (known after apply)
      + location = "eastus"
      + name     = "terraform-rg"
    }

Plan: 1 to add, 0 to change, 0 to destroy.

Changes to Outputs:
  + resource_group_name = "terraform-rg"
azurerm_resource_group.example: Creating...
azurerm_resource_group.example: Still creating... [10s elapsed]
azurerm_resource_group.example: Creation complete after 18s [id=/subscriptions/30e3e25a-097d-4c92-8e35-902b7ed7234e/resourceGroups/terraform-rg]

Apply complete! Resources: 1 added, 0 changed, 0 destroyed.

Outputs:

resource_group_name = "terraform-rg"
```



```
┌──(cybercena㉿astra)-[~/Desktop/DevOps_with_Cohort/week-7/self-learning]
└─$ ls
firstscrip.tf  terraform.tfstate
```

- **Check in Azure Portal if you want,**



Resource Manager | Resource groups

You are viewing a new version of Browse experience. Click here to access the old experience.

Name ↑	Subscription	Location
NetworkWatcherRG	Azure subscription 1	West Europe
ResourceMoverRG-centralindia-eastasia-sea	Azure subscription 1	Southeast Asia
terraform-rg	Azure subscription 1	East US

Step 6 : Rotate / show / delete secret (reference)

- **Rotate secret (create a new password) :**

  Command : az ad sp credential reset –name "<appId>" –years 1

- **Delete SP (cleanup when done with labs):**

  Command : az ad sp delete –id "<appId>"