**ACI Infrastructure as code using Terraform**

Below code create the tenant in the ACI.

**Lab1: Creating Tenant in ACI**

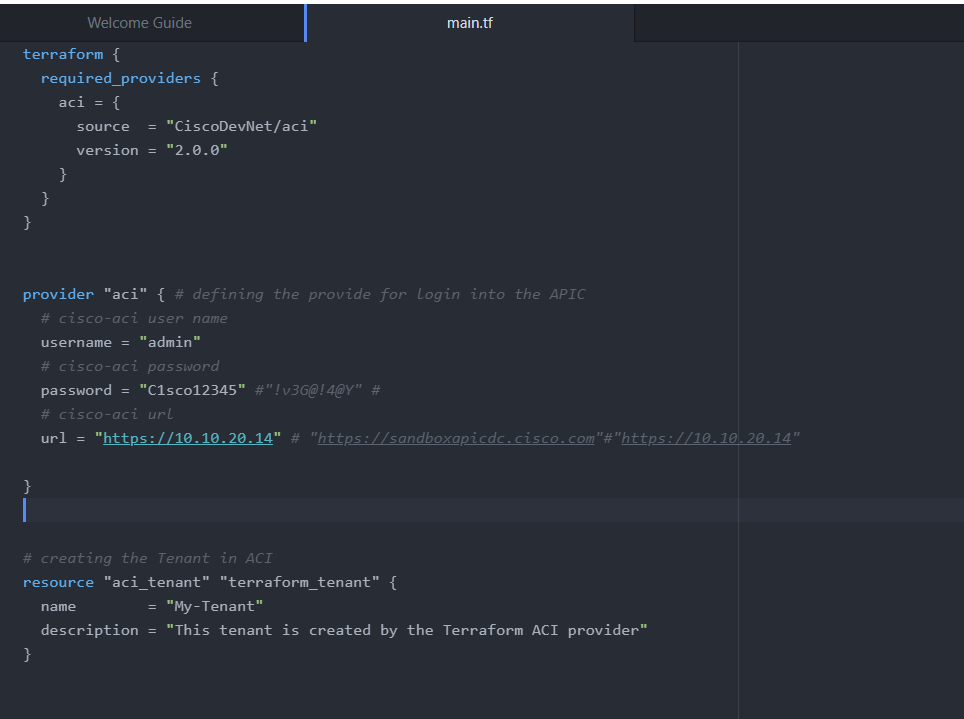
# terraform init

# terraform validate

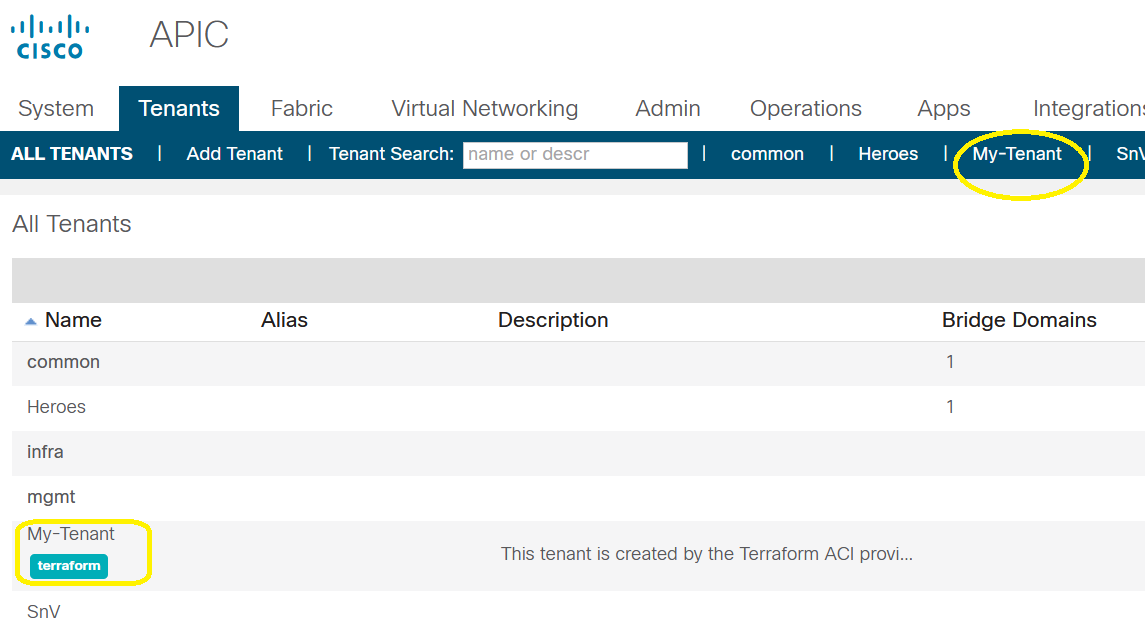
# terraform fmt

# terraform plan

# terraform apply -auto-approve

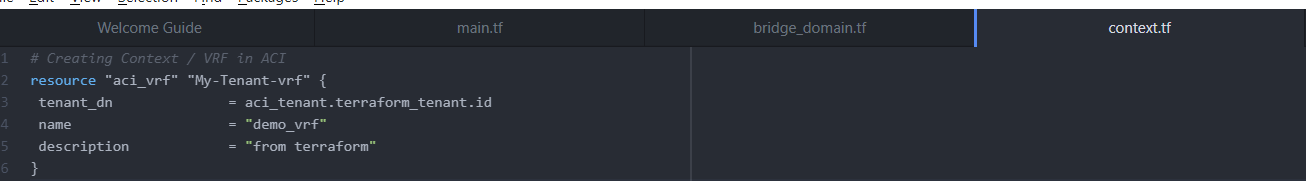


Verifying from the console.

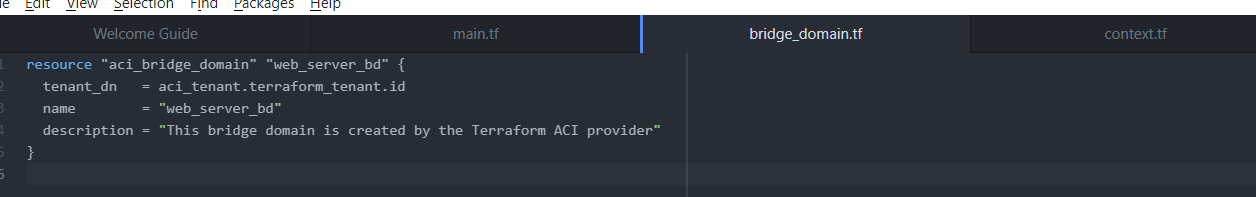


**Lab2: Creating Context/VRF in ACI**

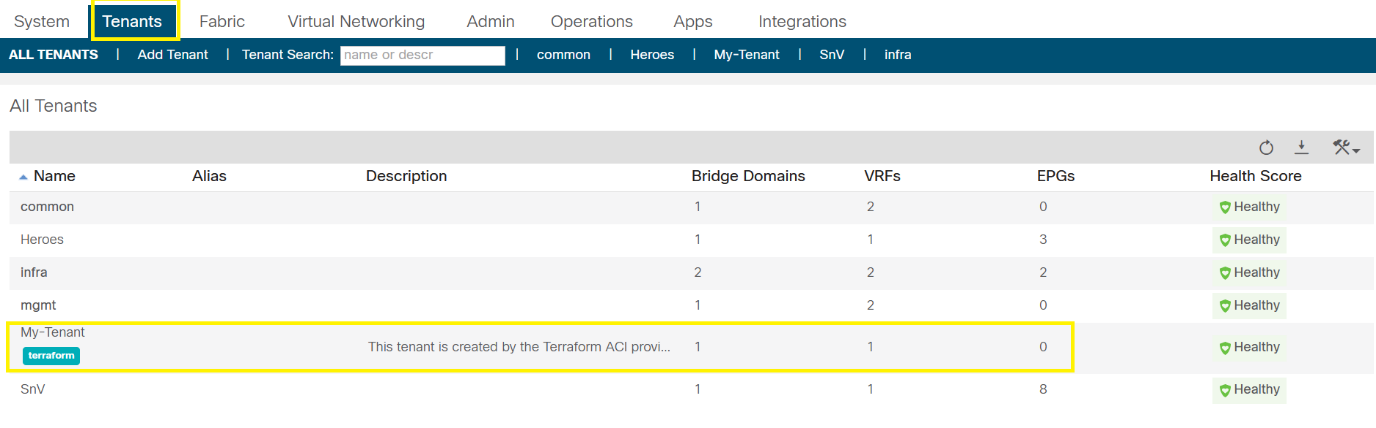
Creating VRF(Context)



**Lab3: Creating Bridge Domain**

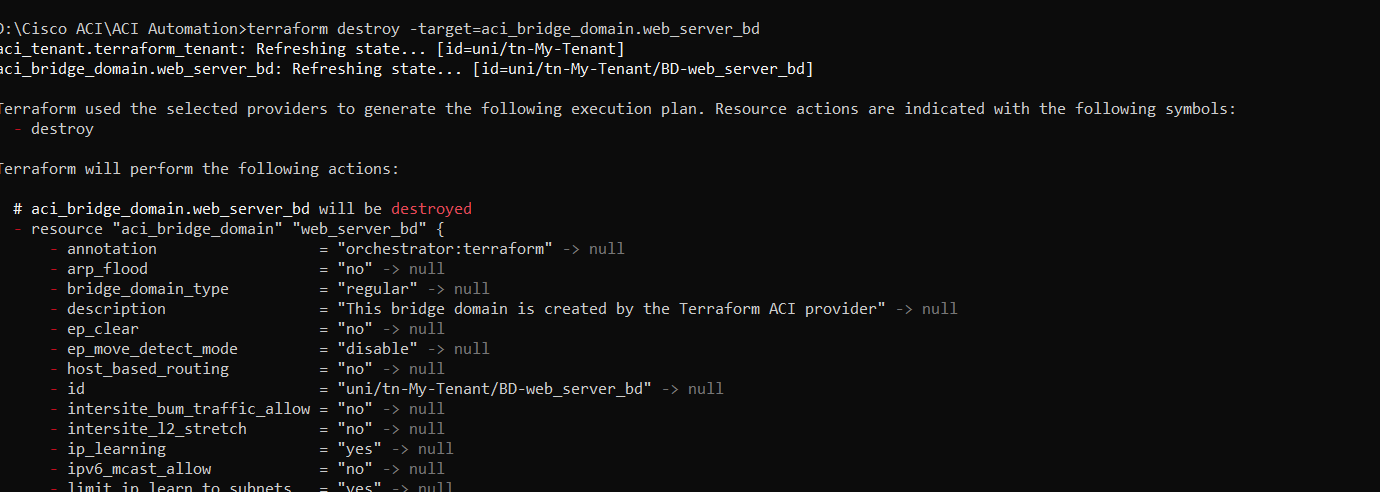
Creating Bridge Domain

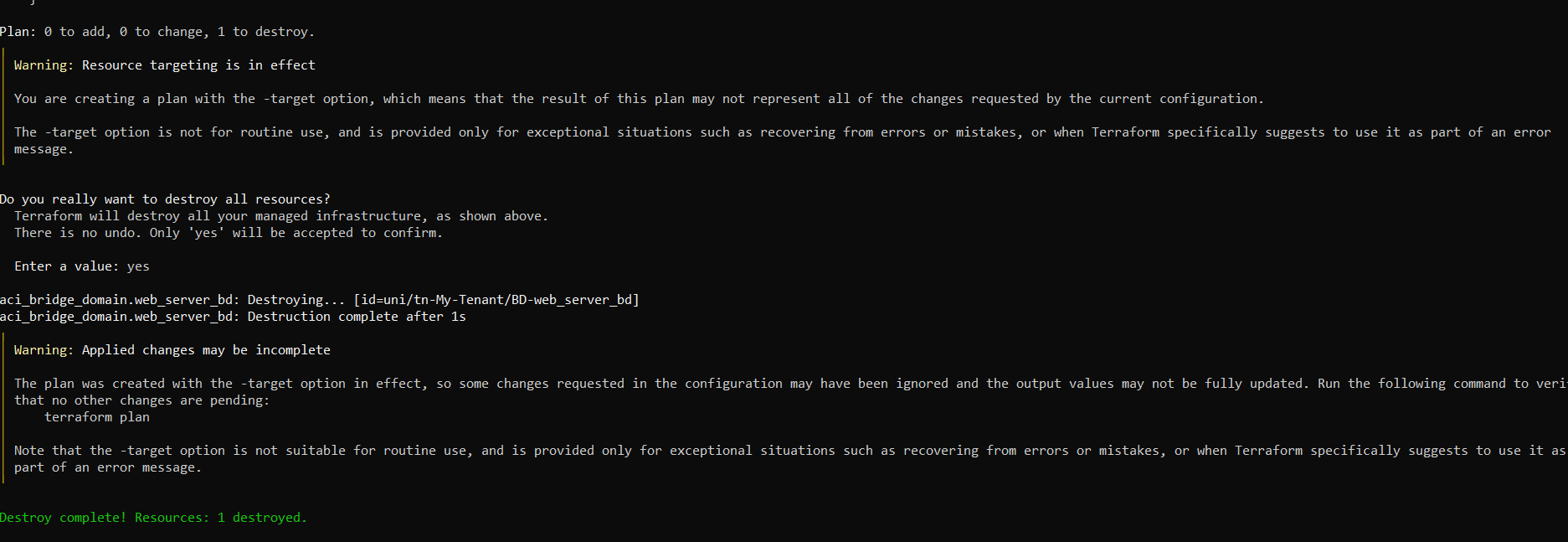
Verification from console



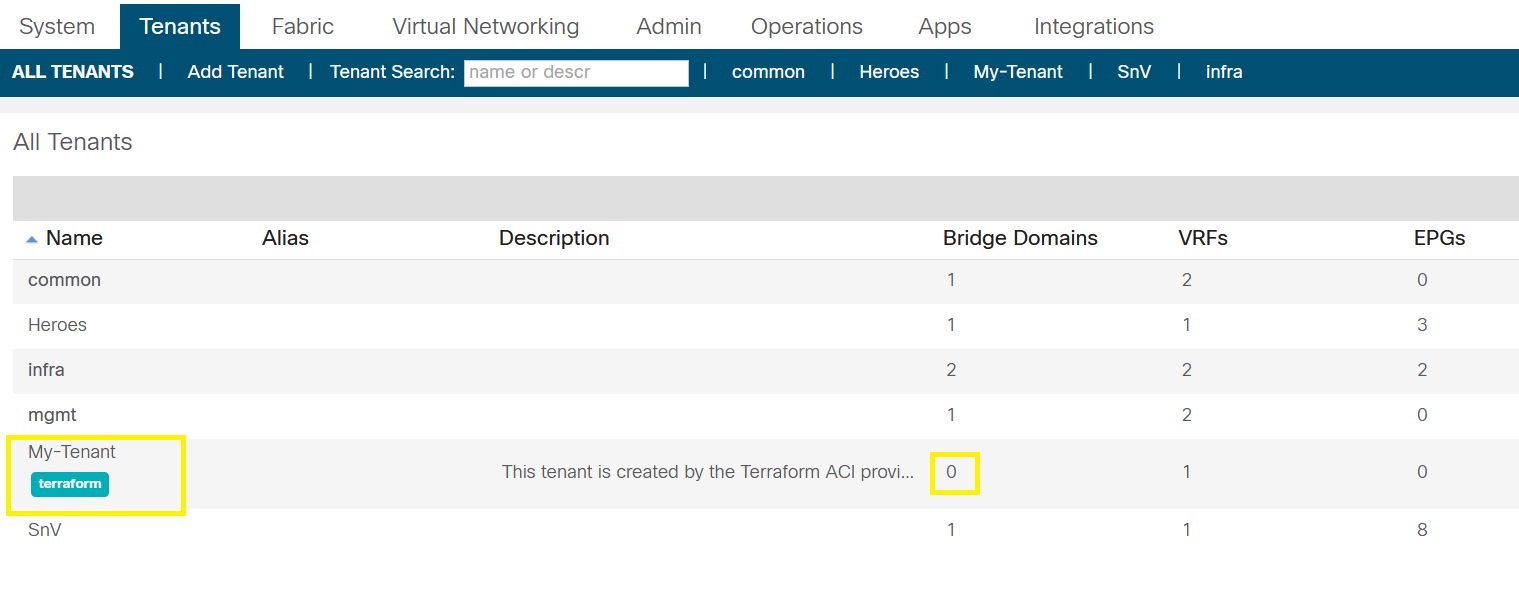
**Lab4: Deleting Bridge Domain**

Deleting bridge domain using -target=aci\_bridge\_domain.web\_server\_bd resource name.





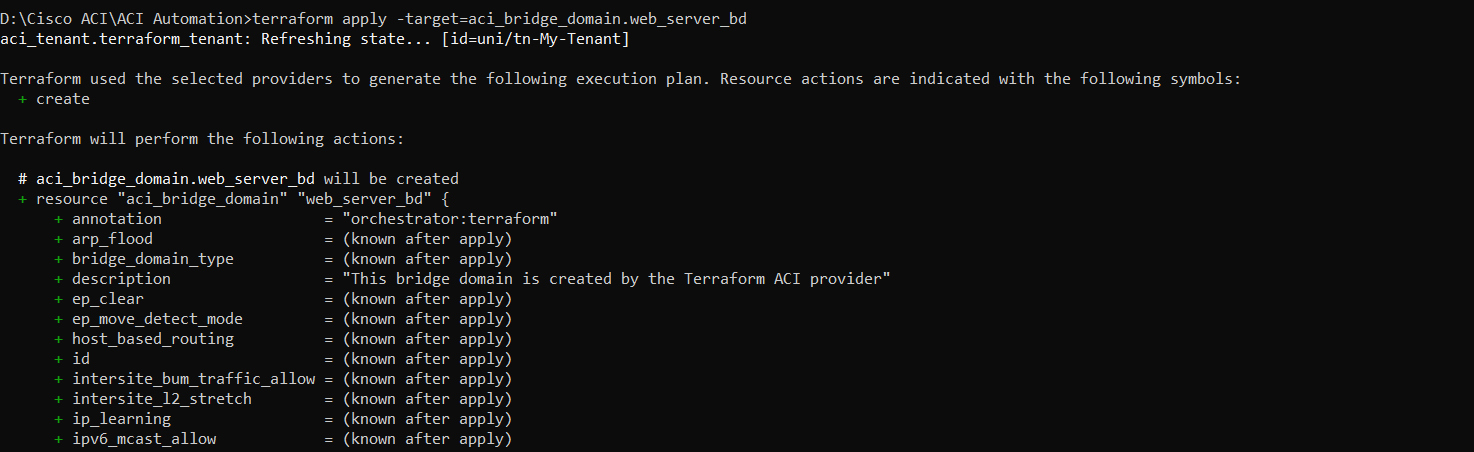
Verifying from console



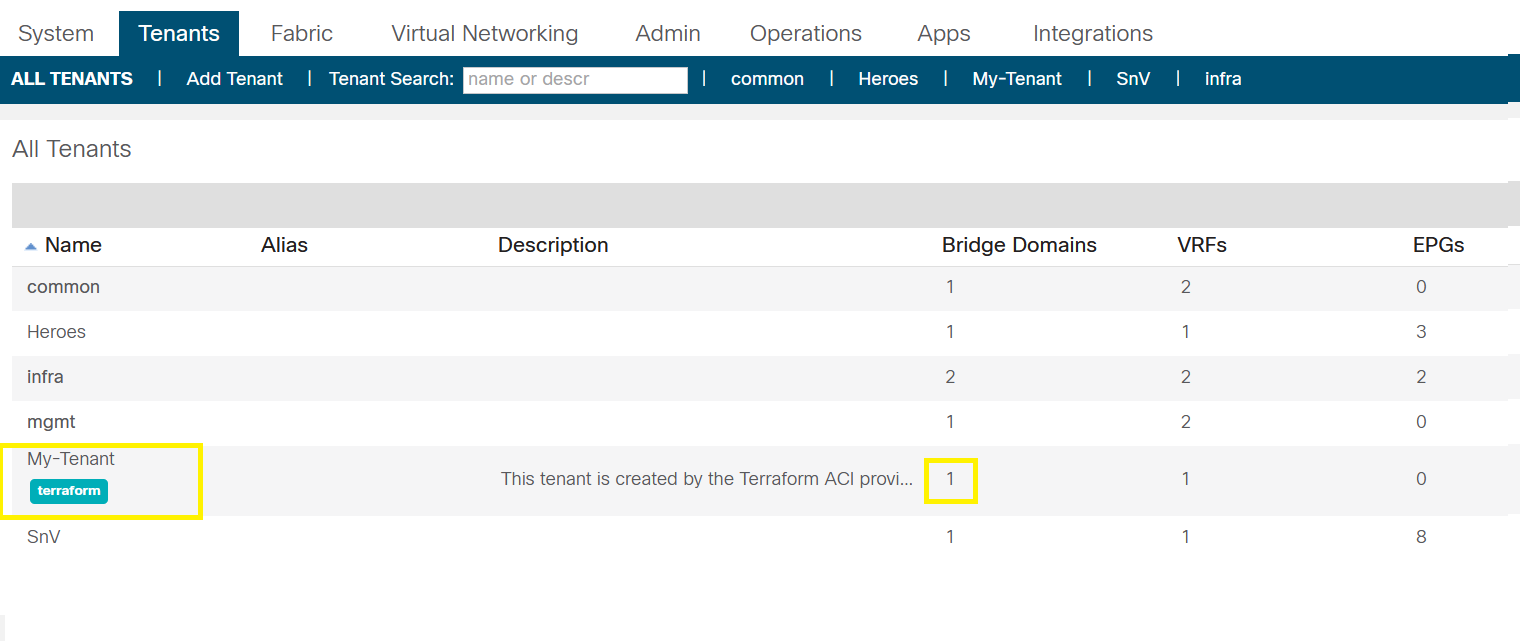
**Lab5: Adding Bridge Domain using target**

Adding again Bridge Domain

**terraform apply -target=aci\_bridge\_domain.web\_server\_bd**



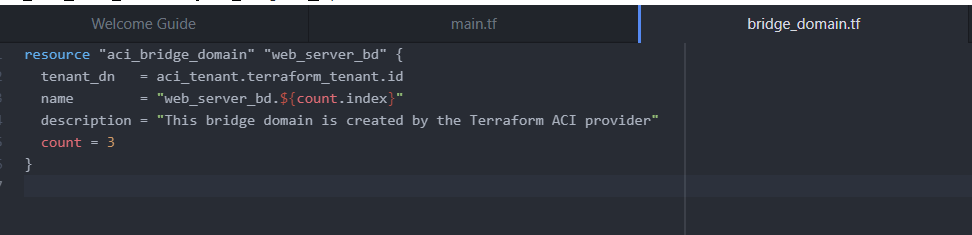
Verifying from Console



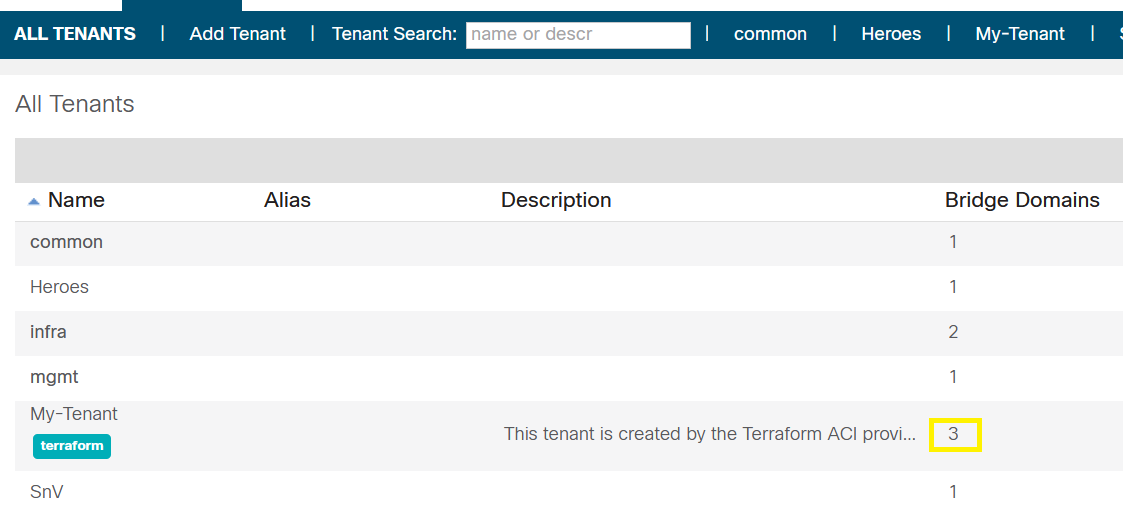
**Lab6: Adding multiple Bridge Domain using Count**

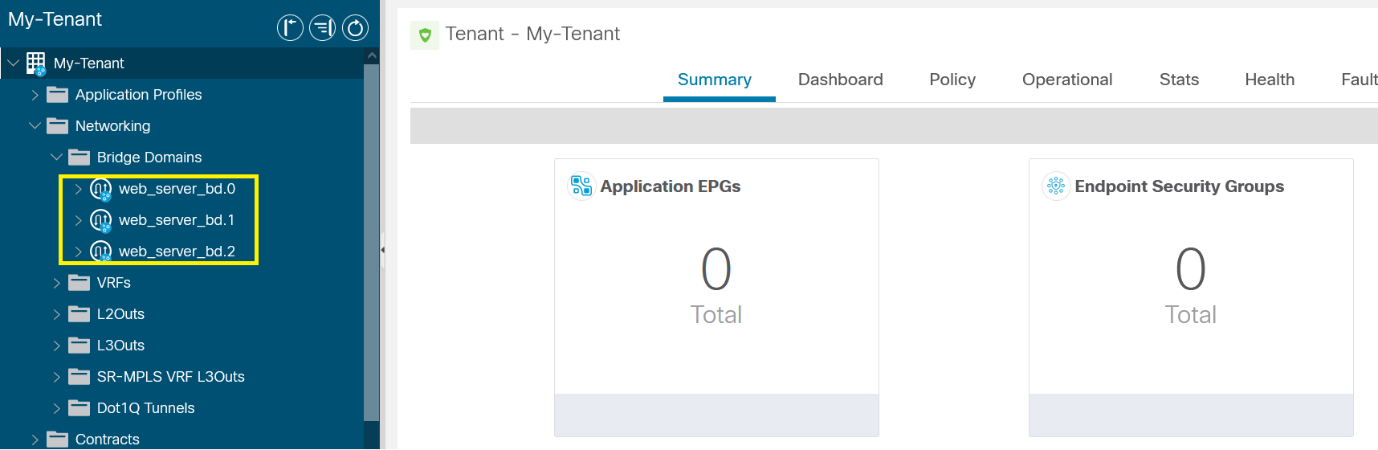
Adding three bridge domain using count.index

terraform apply -target=aci\_bridge\_domain.web\_server\_bd

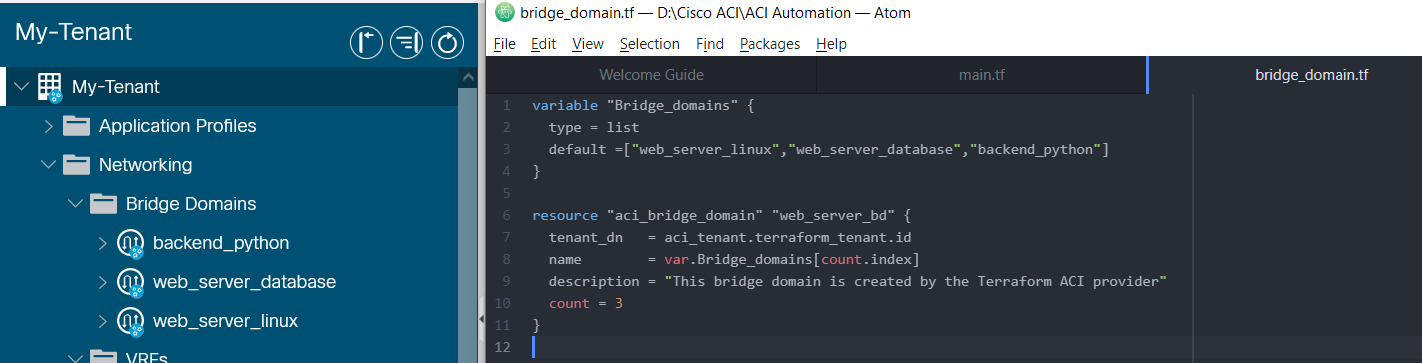


Verifying from Console





The above approach is not feasible as we don’t have name like web\_server.0. The below code create the three bridge domain using the list. **terraform apply -target=aci\_bridge\_domain.web\_server\_bd**



Adding Bridge domain into the context, here depends\_on is use to create the context first and then create the bridge domain.

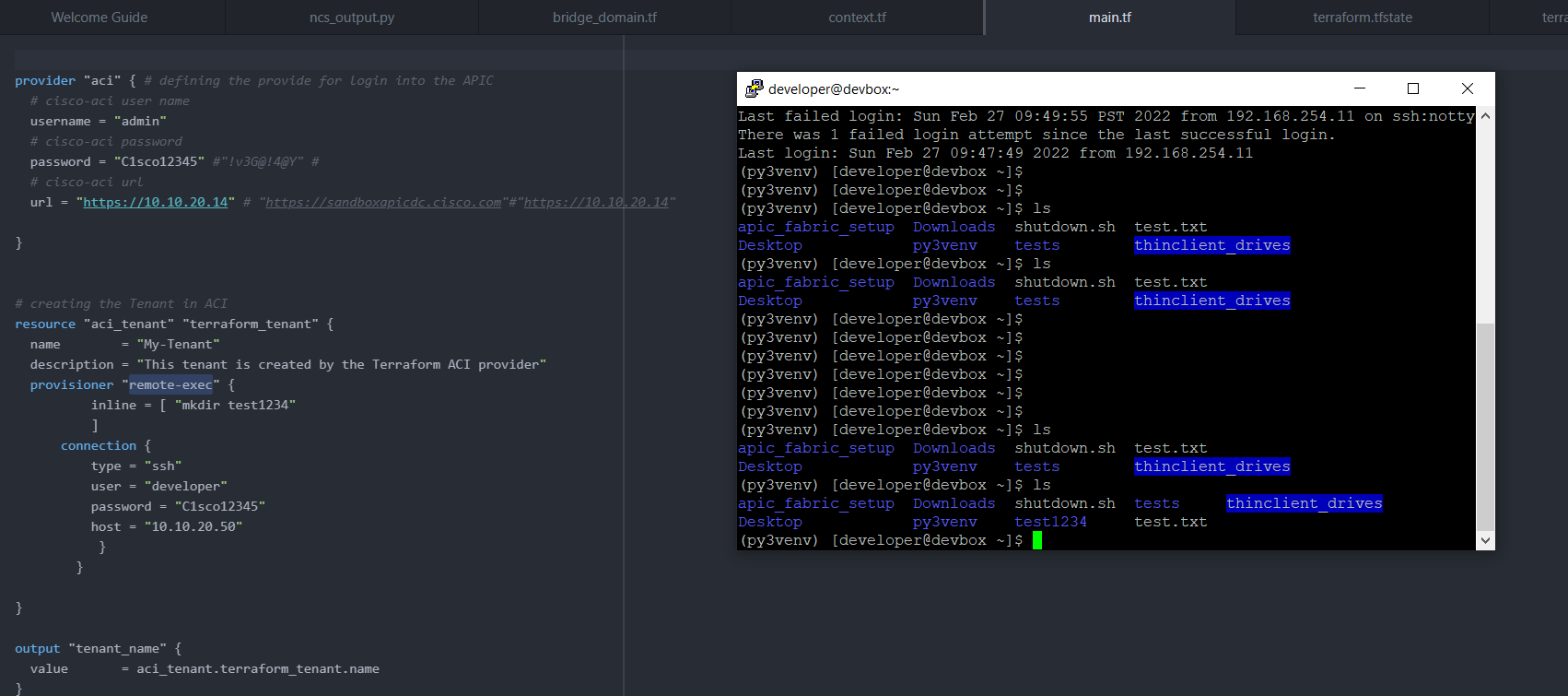
**Lab7: Adding multiple Bridge Domain using under the VRF and using depends\_on:**

Graphical user interface, text

Description automatically generated

Text

Description automatically generated



terraform {

required\_providers {

aci = {

source = "CiscoDevNet/aci"

version = "2.0.0"

}

}

}

provider "aci" { # defining the provide for login into the APIC

# cisco-aci user name

username = "admin"

# cisco-aci password

password = "C1sco12345" #"!v3G@!4@Y" #

# cisco-aci url

url = "https://10.10.20.14" # "https://sandboxapicdc.cisco.com"#"https://10.10.20.14"

}

# creating the Tenant in ACI

resource "aci\_tenant" "terraform\_tenant" {

name = "My-Tenant"

description = "This tenant is created by the Terraform ACI provider"

provisioner "remote-exec" {

inline = [ "mkdir test1234"

]

connection {

type = "ssh"

user = "developer"

password = "C1sco12345"

host = "10.10.20.50"

}

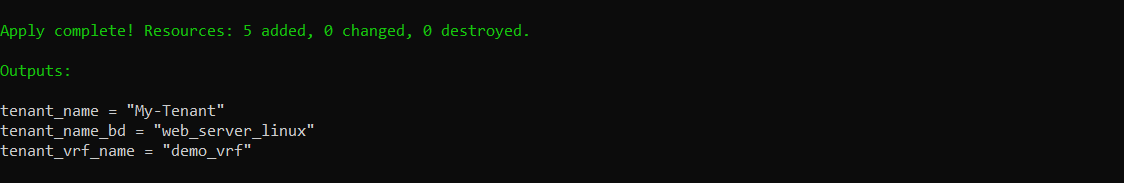
}

}

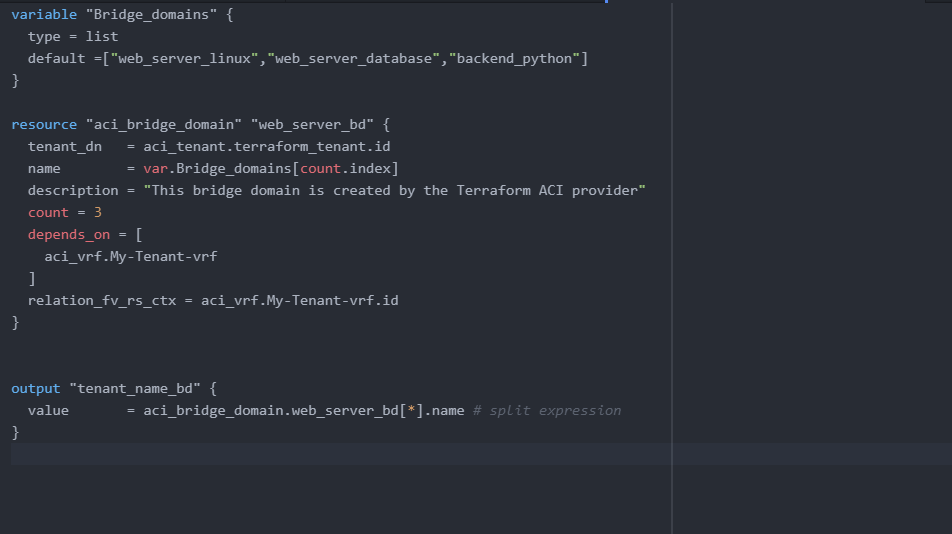
output "tenant\_name" {

value = aci\_tenant.terraform\_tenant.name

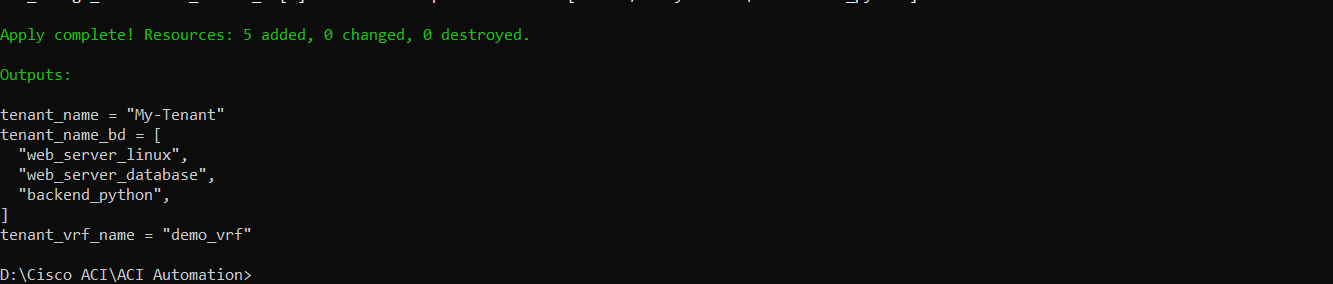
}



**Lab8: Using Splat function to display the list in output variable**

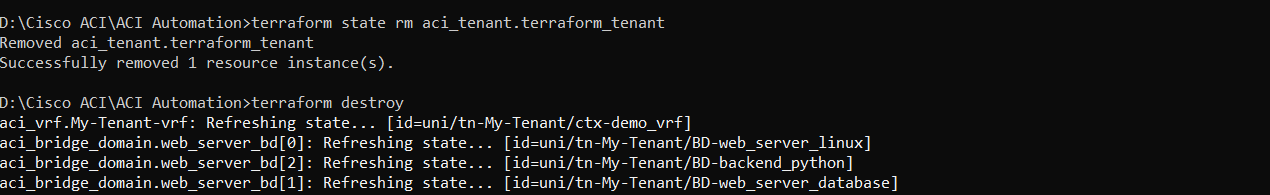


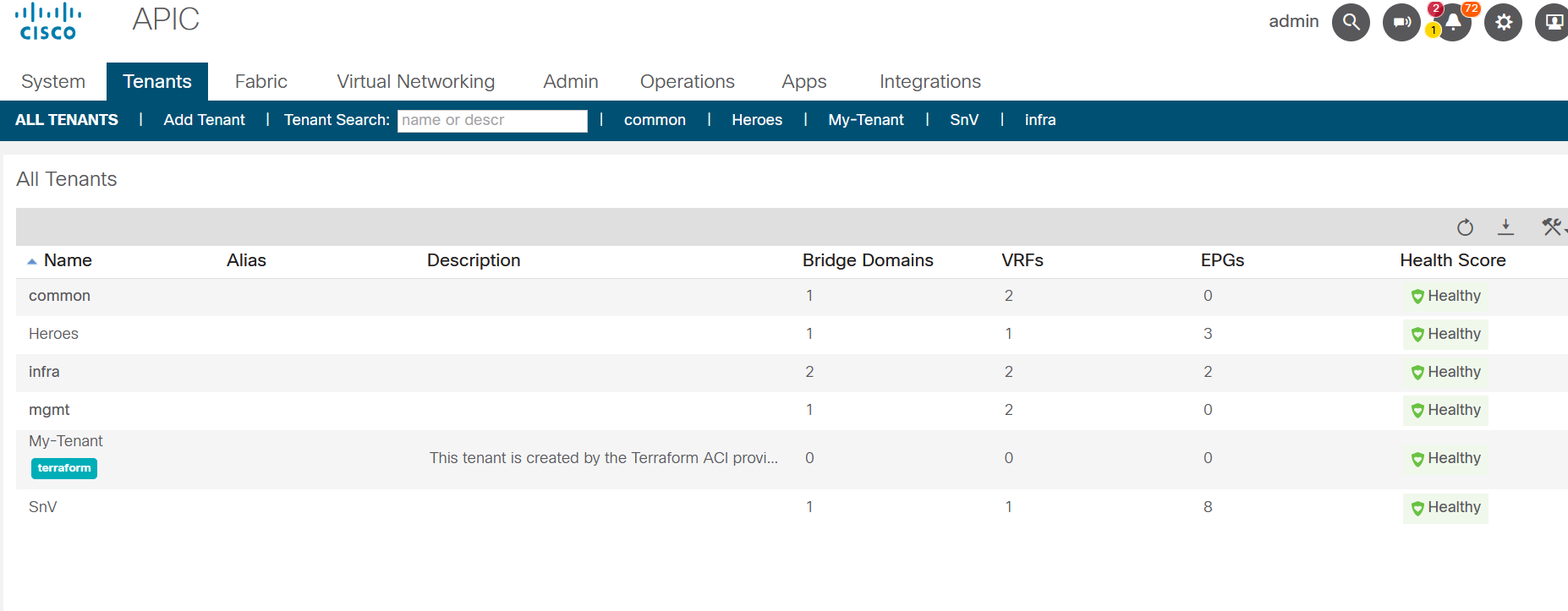
Output:



**Lab 9: Terraform state rm command**

Terraform state rm is handy command it allow you to remove the specific resource from the state file, once you run the terraform delete this resource will not be deleted.

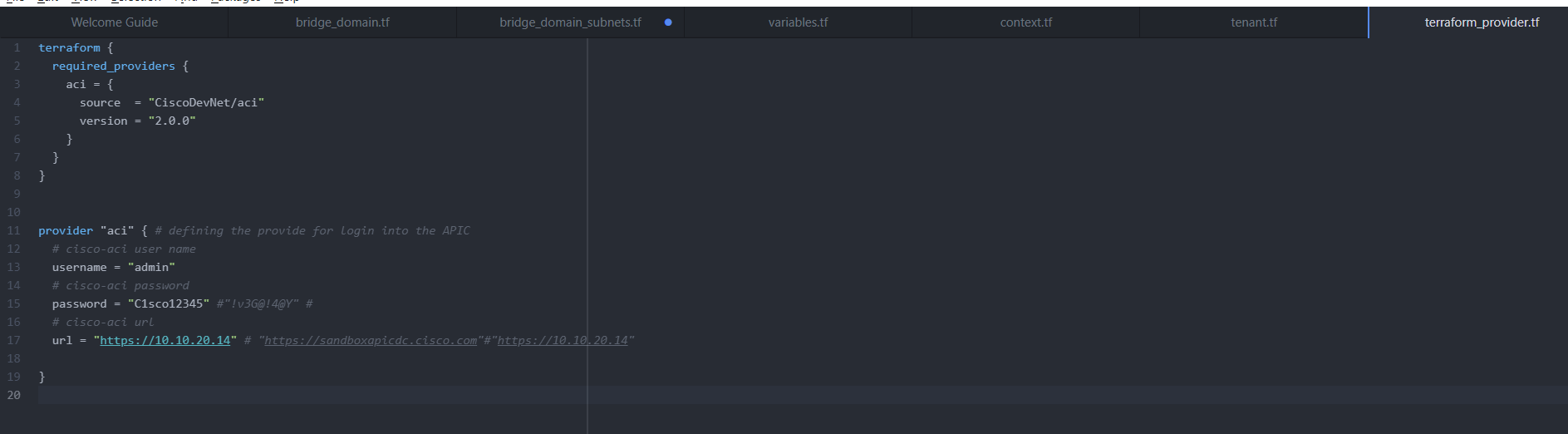




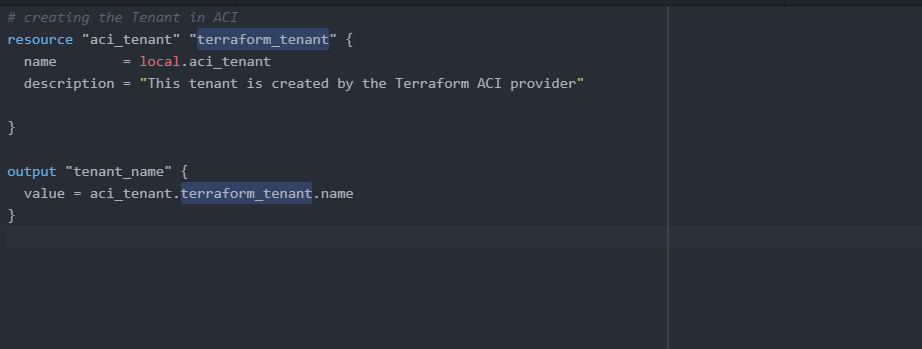
**Lab10: Create the tenant,vrf,bridge domain, 3 subnets inside bridge domain use local to define the aci resources and delete one subnet**

We created six terraform files separate for provider, tenant, bridge domain, context, and variable.

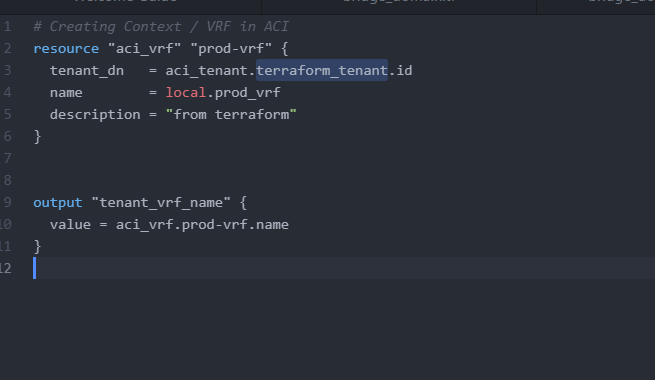
Terraform\_provide.tf



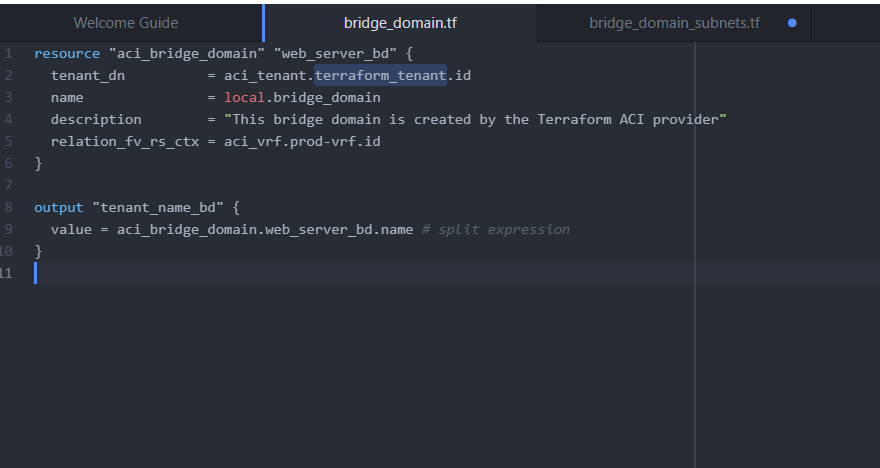
tenant.tf



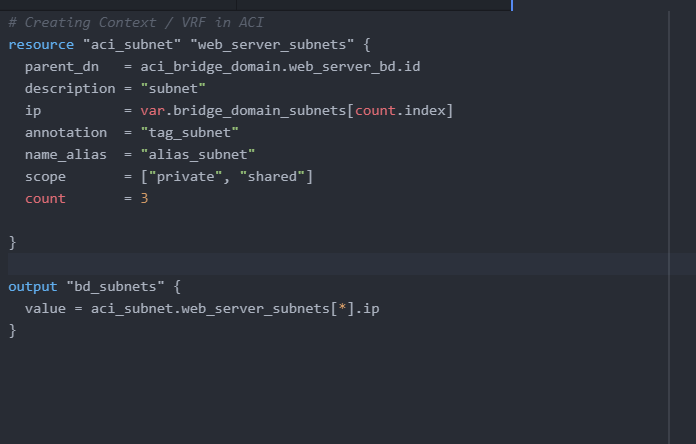
Context.tf



Bridge\_domain.tf



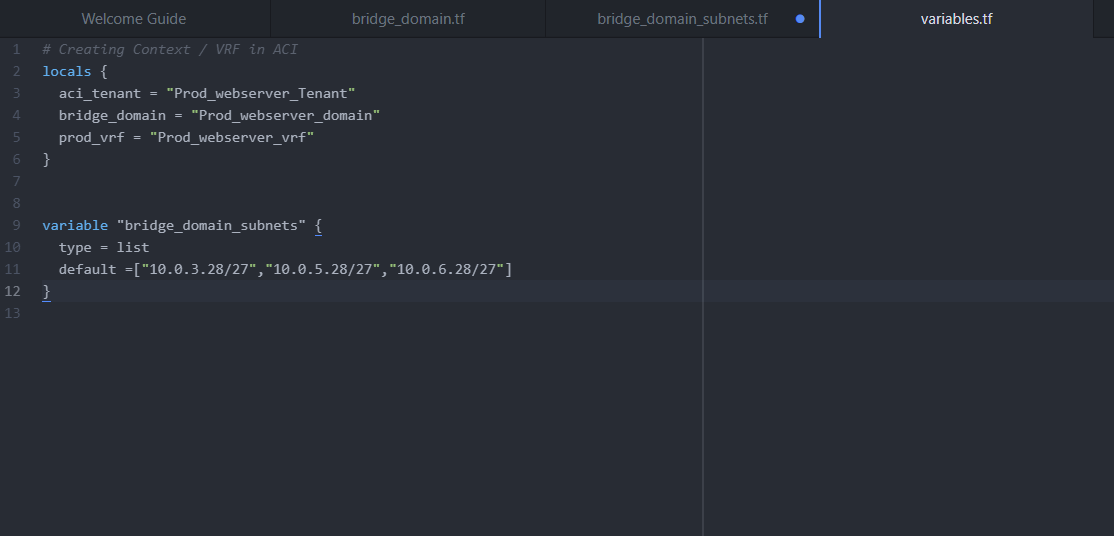
Bridge\_domain\_subnets.tf



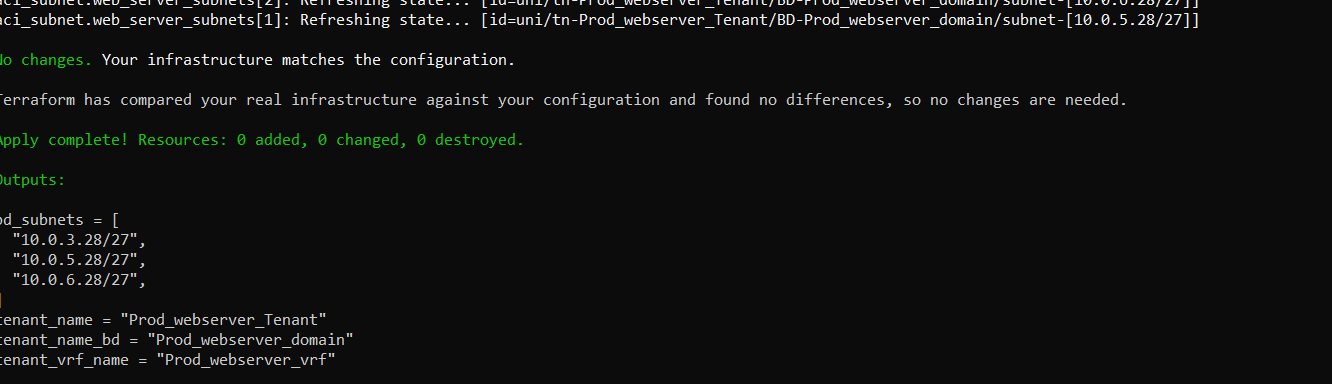
Variable.tf

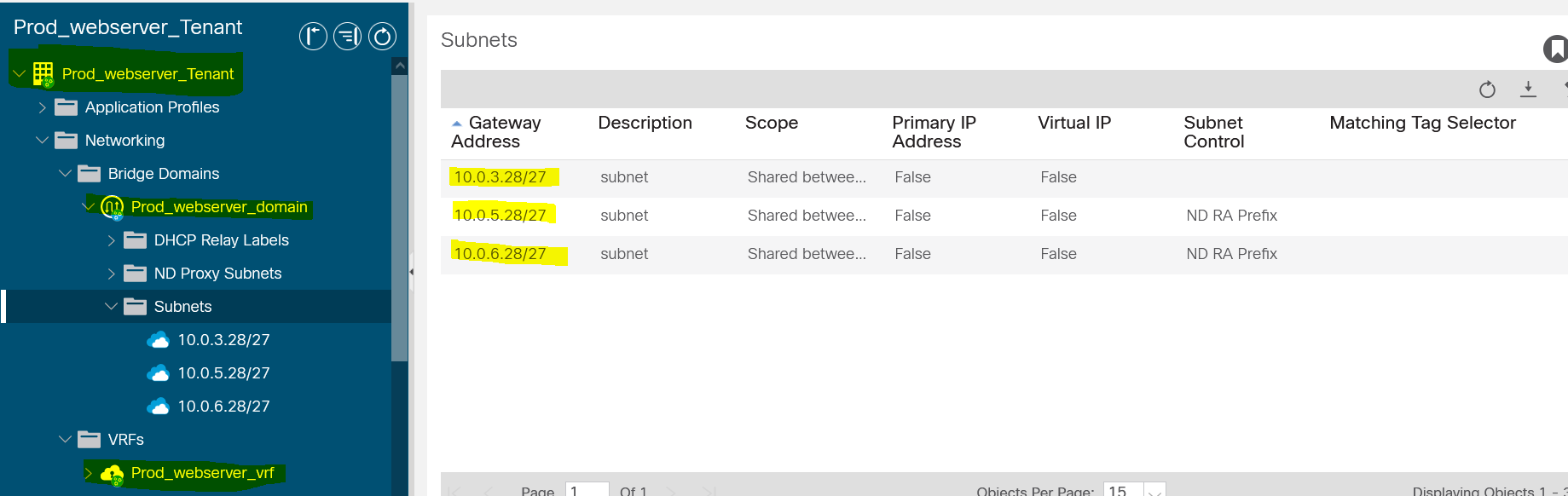
It is best practice is to create the separate variable.tf file and define variable. Here we use locals

Local are used in terraform don’t repeat the code.



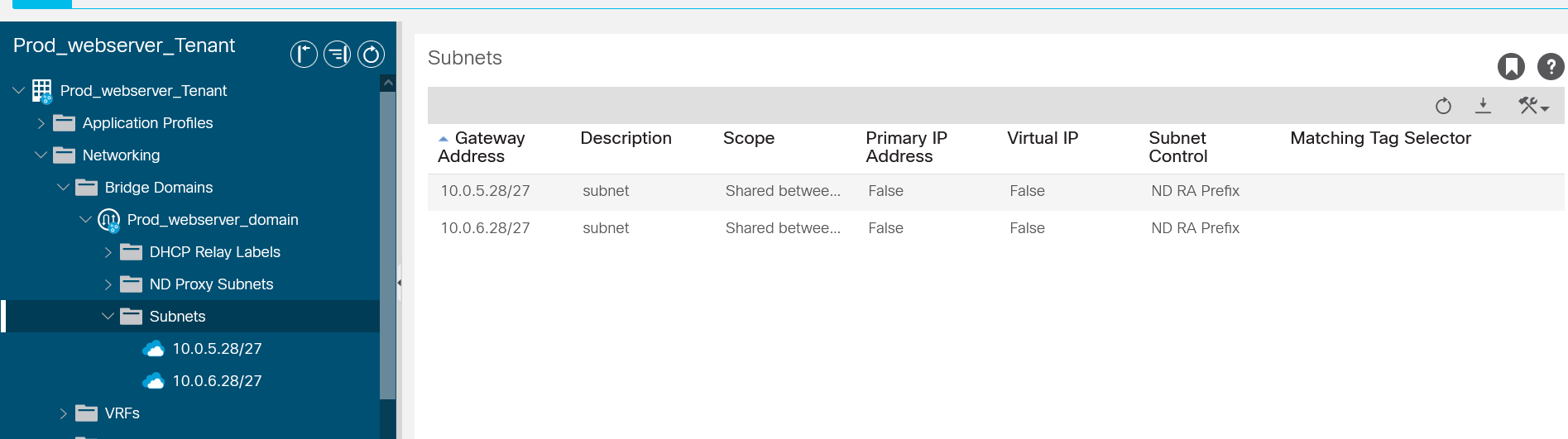
Output





**Delete subnet**

**terraform destroy -target=aci\_subnet.web\_server\_subnets[0]**



Text

Description automatically generated