Mandatory Assignment 01 - IaC IIKG3005

- The purpose of each module
- How to use the Terraform scripts
- Any pre-requisites or dependencies

Modules

Network Module

The Network Module is used to provision a VNET, Subnet and Network Security Group. The module takes in the following variables:

> These can also be found in the variables.tf file.

Variable	Description	Type	Default
vnet_rg_name	VNET Resource Group Name	string	vnet-rg
vnet_rg_location	VNET Resource Group Location	string	westeurope
vnet_name	VNET Name	string	vnet
vnet_address_space	VNET Address Space	list	
vnet_dns_servers	VNET DNS Servers	list	
subnet_name	Subnet Name	string	subnet
subnet_address_space	Subnet Address Space	list	
nsg_name	Network Security Group Name	string	nsg
my_ip	Your IP Address, used for NSG later	string	

Storage Account Module

The Storage Account Module is used to create an Azure Storage Account and a Storage Container. The module takes in the following variables:

> These can also be found in the variables.tf file.

Variable	Description	Type	Default
sa_base_name	Base name of the Storage Account	string	sa
sa_rg_name	Storage Account Resource Group Name	string	sa-rg
sa_location	Location of the Storage Account	string	westeurope
sa_container_name	Name of the Storage Container	string	container
sa_account_tier	Storage Account Tier	string	Standard

Variable	Description	Type	Default
sa_replication_type	Storage Account Replication Type	string	GRS

Virtual Machine Module

The Virtual Machine Module is used to provision an Azure Virtual Machine. The module takes in the following variables:

> These can also be found in the variables.tf file.

Variable	Description	Type	Default
vm_name	Virtual Machine Name	string	vm
vm_size	Virtual Machine Size	string	Standard_B2s
vm_rg_name	Virtual Machine Resource Group Name	string	vm-rg
vm_rg_location	Location of the Virtual Machine Resource Group	string	westeurope
vm_nic_name	Virtual Machine NIC Name	string	vm-nic
vm_nic_private_ip_address	Virtual Machine NIC Private IP	string	
pip_name	Public IP Name	string	vm-pip-name
vm_subnet_id	Subnet ID	string	
vm_username	Username for the Virtual Machine	string	
vm_password	Password for the Virtual Machine	string	

Key Vault Module

The Key Vault Module is used to provision Azure Key Vault and store secret. The module takes in the following variables:

> These can also be found in the variables.tf file.

Variable	Description	Type	Default
kv_rgname	Key Vault Resource Group Name	string	kv-rg
kv_location	location of the Key Vault	string	westeurope
kv_base_name	Base name of the Key Vault	string	kv
kv_sku_name	SKU of the Key Vault	string	standard
sa_access_key	Access key for the Storage Account	string	
sa_base_name	Base name of the Storage Account	string	sa
vm_password	Password for the Virtual Machine	string	
vm_username	Username for the Virtual Machine	string	

Dependencies & Prerequisites

- Terraform >= 0.12.0
- Azure CLI >= 2.0.0
- Azure Subscription
- Azure Service Principal

HOWTO

Setup

1. Clone the repository
2. Create a **terraform.tfvars** file in the root of the repository and fill in the variables in the Variables section below.
3. Login to Azure using the Azure CLI
    1. Run **az login**
    2. Follow the instructions on the screen
    3. Alternatively, you can login using a Service Principal
        - Run **az login --service-principal -u -p --tenant**
4. Run **terraform init** to initialize the project
5. Run **terraform plan** to see what resources will be created
6. Run **terraform apply** to create the resources
7. Run **terraform destroy** to destroy the resources

Terraform Plan
Used to see what resources will be created before running **terraform apply**.
Can be used in combination with **-out** to save the plan to a file, which can be used later to apply the plan.
Example: **terraform plan -var-file="terraform.tfvars" -out="terraform.tfplan"**
Using the **-var-file** flag to specify the **terraform.tfvars** file, and the **-out** flag to specify the output file.

Using plan is smart because it will show you what resources will be created, and if there are any errors in the configuration.
You can also keep track of the plan files in a version control system, and use them to apply the plan later.
Example: **terraform apply "terraform.tfplan"**

This way, you can make sure you have a plan to fall back to if something goes wrong, and you can also use the plan to apply the configuration to multiple environments.

Terraform Apply
Used to apply the configuration to the environment.
You can specify a plan file to apply, or you can just run **terraform apply** and it will apply the configuration in the current directory.

Terraform Destroy
Used to destroy the resources created by the configuration.
It is actually an alias for **terraform apply -destroy**, which means that it will apply the configuration, but destroy the resources instead of creating them.
Using this command will prompt you to confirm the destruction of the resources, unless you use the **-auto-**

approve flag.
Example: **terraform destroy -auto-approve**

The main reason for using this command is to make sure that you don't leave any resources running that you don't need.
This saves you money, as well as making sure nothing collides with the resources you create later.
It can also be used to destroy the resources created by a plan file.

Variables

The following variables are required to run the configuration:

> These can also be found in the variables.tf file. (Will repeat from the Modules section)

Variable	Description	Type	Default	Note
company	Company name	string	Company	
project	Project name	string	Project	
billing code	Billing code	string	Billing	
kv_rgname	Key Vault Resource Group Name	string	kv-rg	
kv_location	location of the Key Vault	string	westeurope	
kv_base_name	Base name of the Key Vault	string	kv	
sa_base_name	Base name of the Storage Account	string	sa	
sa_rg_name	Storage Account Resource Group Name	string	sa-rg	
sa_location	Location of the Storage Account	string	westeurope	
sa_container_name	Name of the Storage Container	string	container	
sa_account_tier	Storage Account Tier	string	Standard	
sa_replication_type	Storage Account Replication Type	string	GRS	
vnet_rg_name	VNET Resource Group Name	string	vnet-rg	
vnet_rg_location	VNET Resource Group Location	string	westeurope	
vnet_name	VNET Name	string	vnet	

Variable	Description	Type	Default	Note
vnet_address_space	VNET Address Space	list		Needs to be specified
vnet_dns_servers	VNET DNS Servers	list		Needs to be specified
subnet_name	Subnet Name	string	subnet	
subnet_address_space	Subnet Address Space	list		Needs to be specified
nsg_name	Network Security Group Name	string	nsg	
vm_name	Virtual Machine Name	string	vm	
vm_size	Virtual Machine Size	string	Standard_B2s	
vm_rg_name	Virtual Machine Resource Group Name	string	vm-rg	
vm_rg_location	Location of the Virtual Machine Resource Group	string	westeurope	
vm_nic_name	Virtual Machine NIC Name	string	vm-nic	
vm_nic_private_ip_address	Virtual Machine NIC Private IP	string		
pip_name	Public IP Name	string	vm-pip-name	
vm_subnet_id	Subnet ID	string		
vm_username	Username for the Virtual Machine	string		Needs to be specified
vm_password	Password for the Virtual Machine	string		Needs to be specified
kv_sku_name	SKU of the Key Vault	string	standard	
my_ip	Your IP Address, used for NSG later	string		Needs to be specified, or else an NSG won't work

The VM public IP address is not specified in the variables, as it is not needed, it will be created automatically and assigned to the VM.
The VM NIC private IP address is also not specified in the variables, as it is not needed, it will be created automatically and assigned to the VM NIC.
The VM subnet ID is not specified in the variables, as it is not needed, it will be created automatically and assigned to the VM.

Screenshots of Deployment

Terraform

After finishing the configurations and setup of the modules, and running **terraform init** i ran the following command:
terraform plan -var-file="terraform.tfvars" -out="terraform.tfplan"

```
  # module.keyvault.azurerm_key_vault_secret.vm_username will be created
  + resource "azurerm_key_vault_secret" "vm_username" {
      + id              = (known after apply)
      + key_vault_id    = (known after apply)
      + name            = "vm-username"
      + value           = (sensitive value)
      + version         = (known after apply)
      + versionless_id  = (known after apply)
    }

  # module.keyvault.azurerm_resource_group.kv_rg will be created
  + resource "azurerm_resource_group" "kv_rg" {
      + id       = (known after apply)
      + location = "westeurope"
      + name     = "kv-rg-consulit"
    }

  # module.keyvault.random_string.random_string will be created
  + resource "random_string" "random_string" {
      + id          = (known after apply)
      + length      = 12
      + lower       = true
      + min_lower   = 0
      + min_numeric = 0
      + upper       = false
    }

Plan: 21 to add, 0 to change, 0 to destroy.

Saved the plan to: terraform.tfplan
```

After running the plan, i ran the following command:
terraform apply "terraform.tfplan"

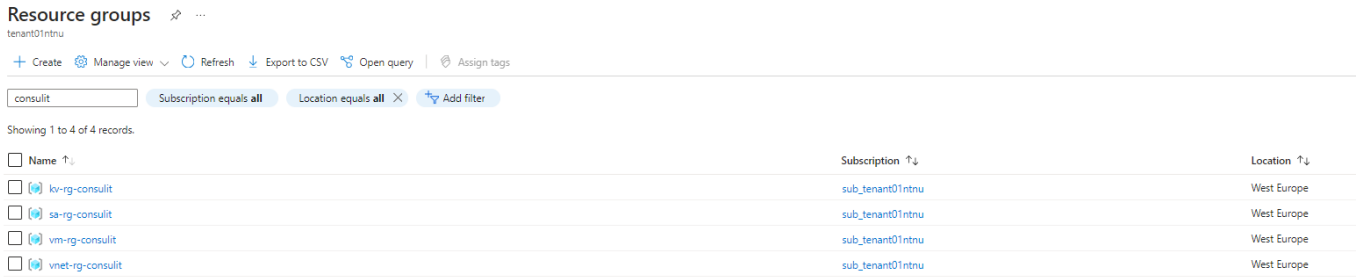
```
module.keyvault.azurerm_key_vault_secret.vm_username: Creation complete after 1s [id=https://kv-
module.keyvault.azurerm_key_vault_secret.vm_password: Creation complete after 1s [id=https://kv-
module.VirtualMachine.azurerm_linux_virtual_machine.linux_vm: Creating...
module.keyvault.azurerm_key_vault_secret.sa_accesskey: Creation complete after 1s [id=https://kv
module.VirtualMachine.azurerm_linux_virtual_machine.linux_vm: Still creating... [10s elapsed]
module.VirtualMachine.azurerm_linux_virtual_machine.linux_vm: Still creating... [20s elapsed]
module.VirtualMachine.azurerm_linux_virtual_machine.linux_vm: Still creating... [30s elapsed]
module.VirtualMachine.azurerm_linux_virtual_machine.linux_vm: Still creating... [40s elapsed]
module.VirtualMachine.azurerm_linux_virtual_machine.linux_vm: Creation complete after 50s [id=/s

Apply complete! Resources: 21 added, 0 changed, 0 destroyed.
PS C:\Prog\Private\IIKG3005\Mandatory Assignments\Assignment01\kristoju-oppg1>
```

Azure Portal

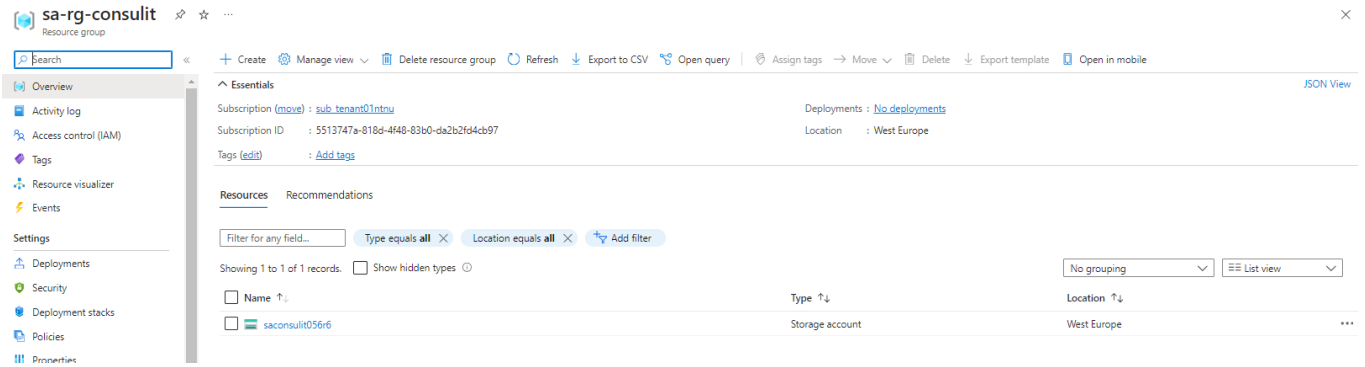
After running the apply command, i went to the Azure Portal to check if the resources were created.

Resource Groups

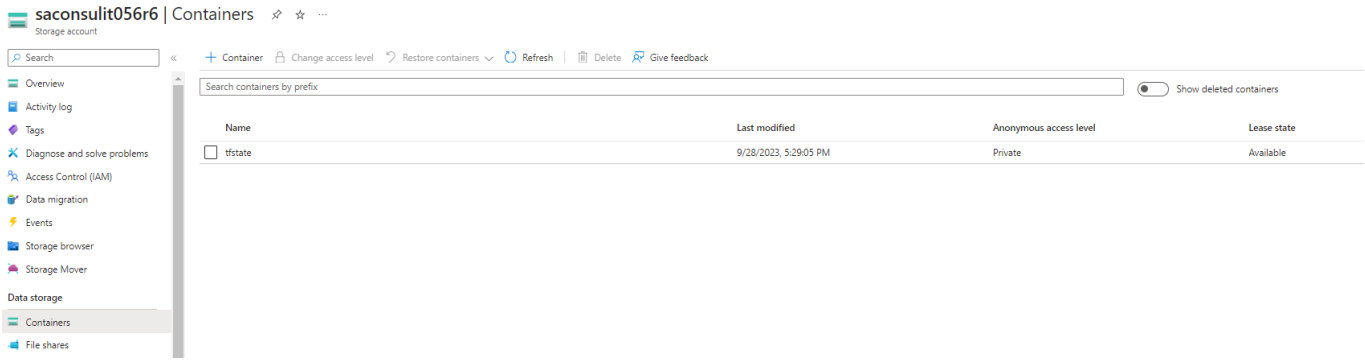


Storage Account

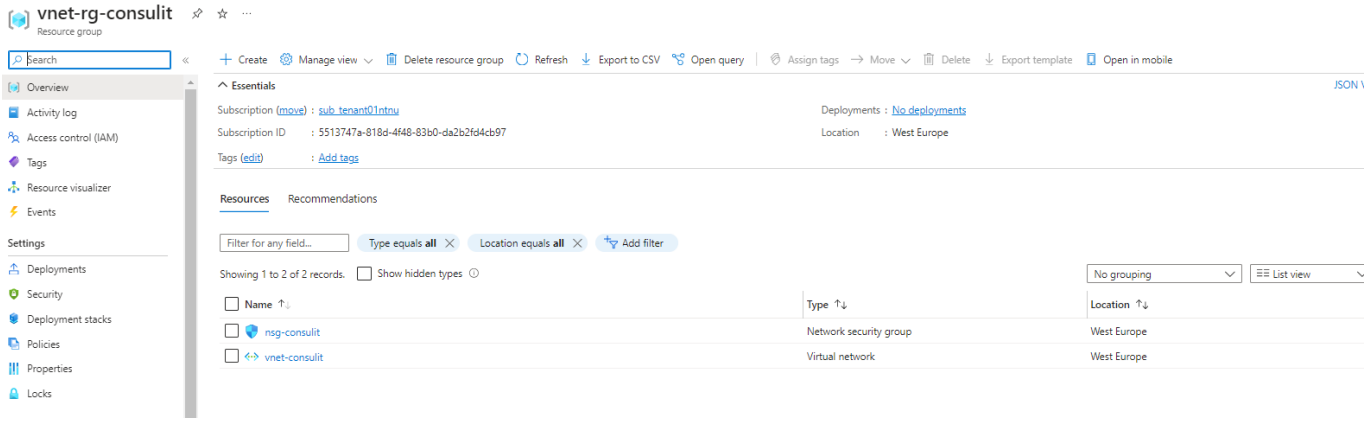
Storage Account:



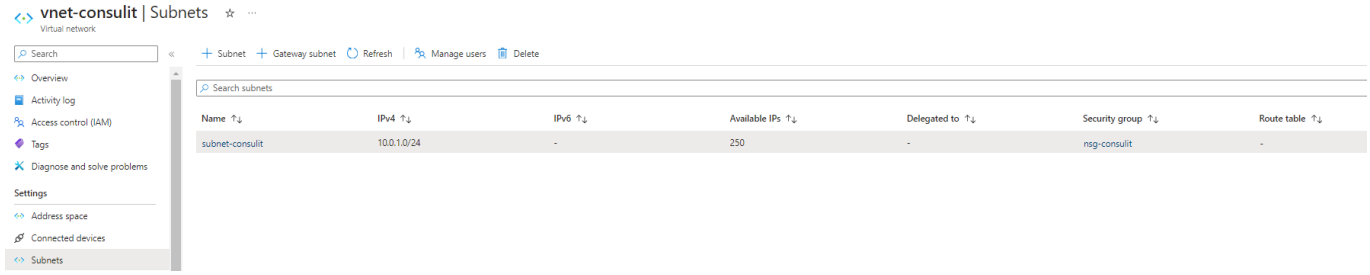
Container:



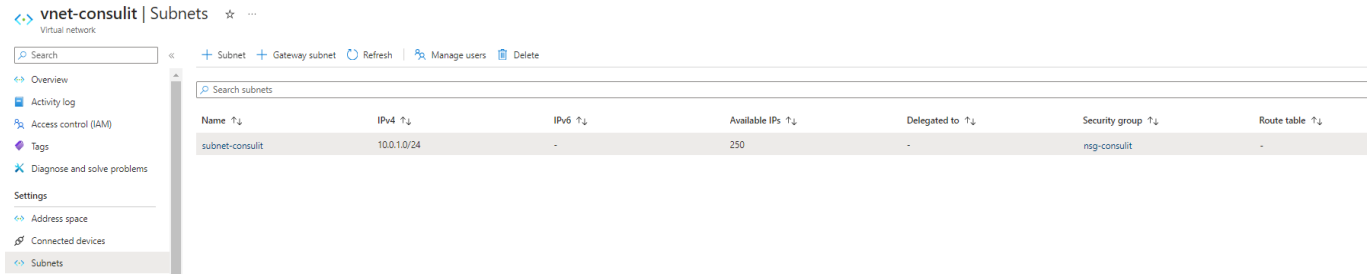
Network



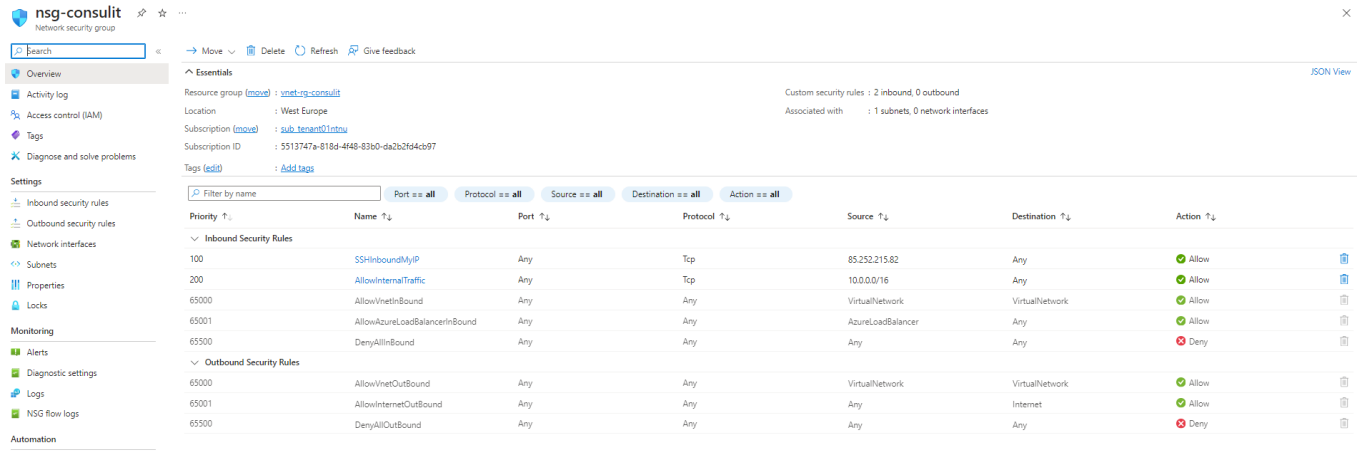
VNET:



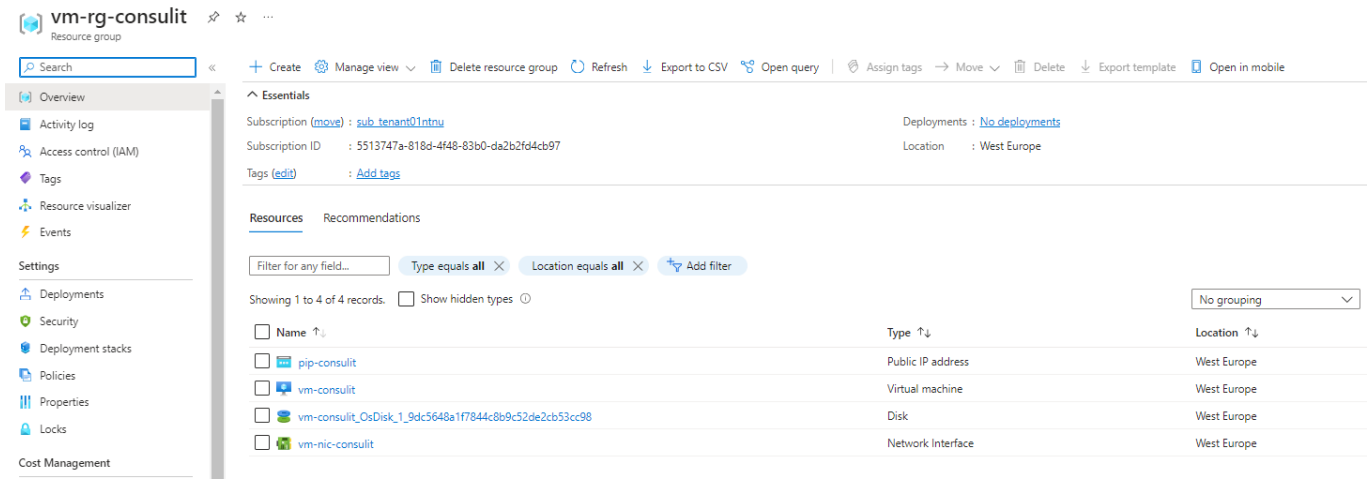
Subnet:



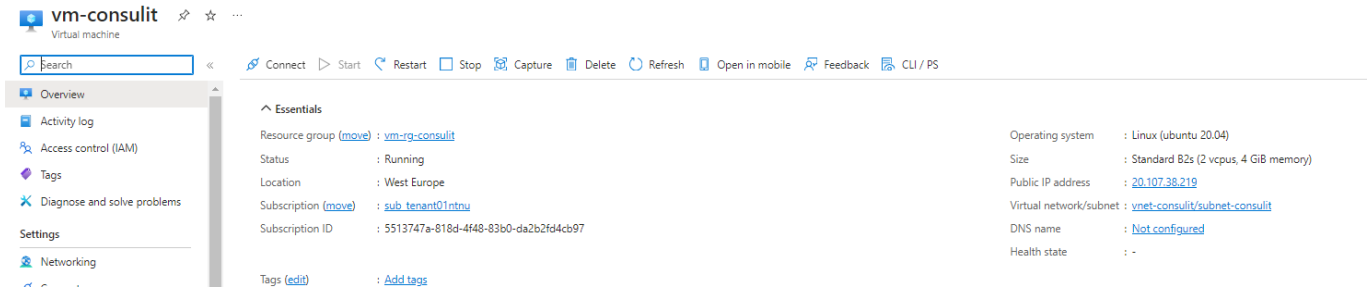
NSG:



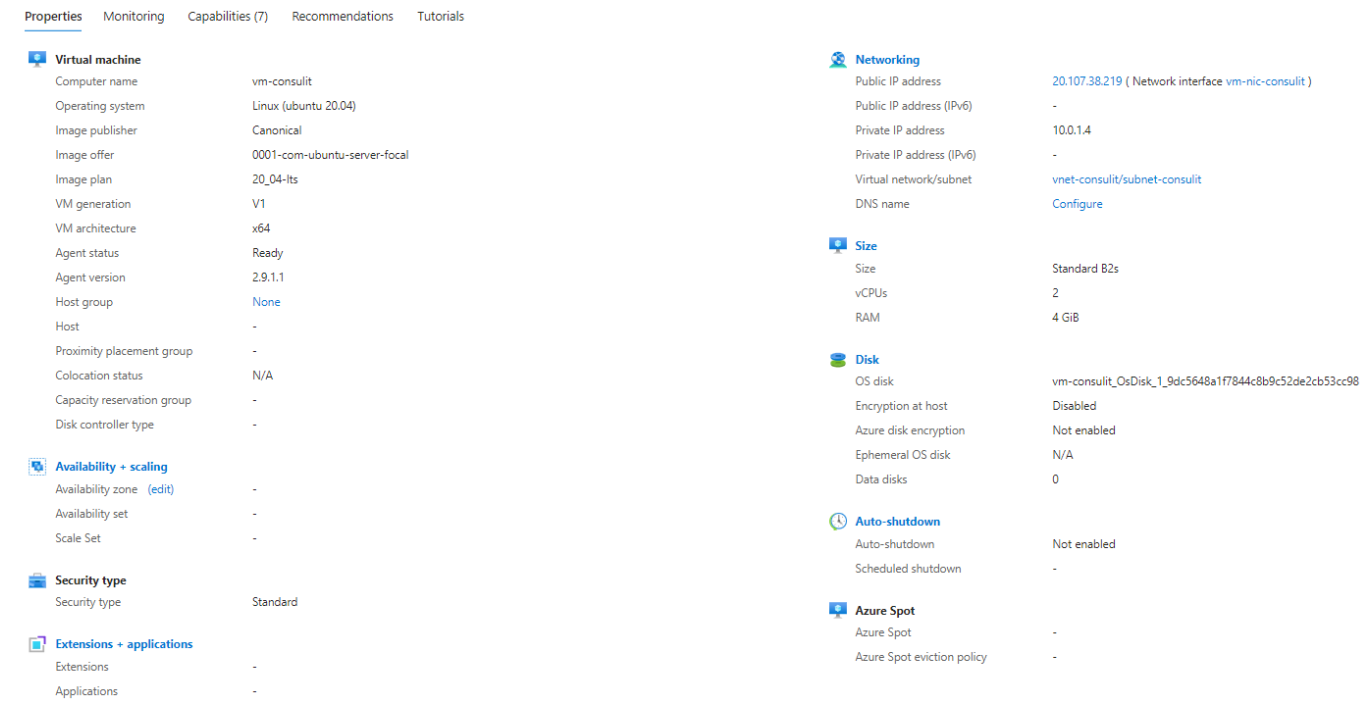
Virtual Machine



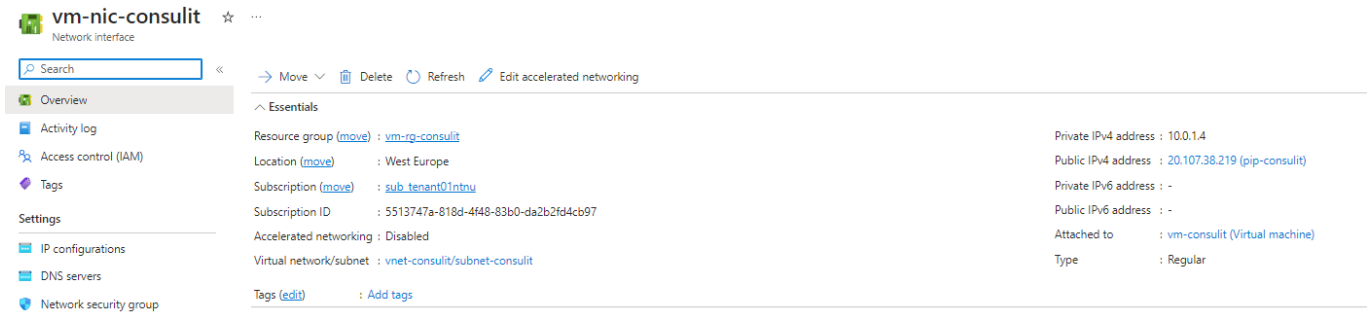
VM:



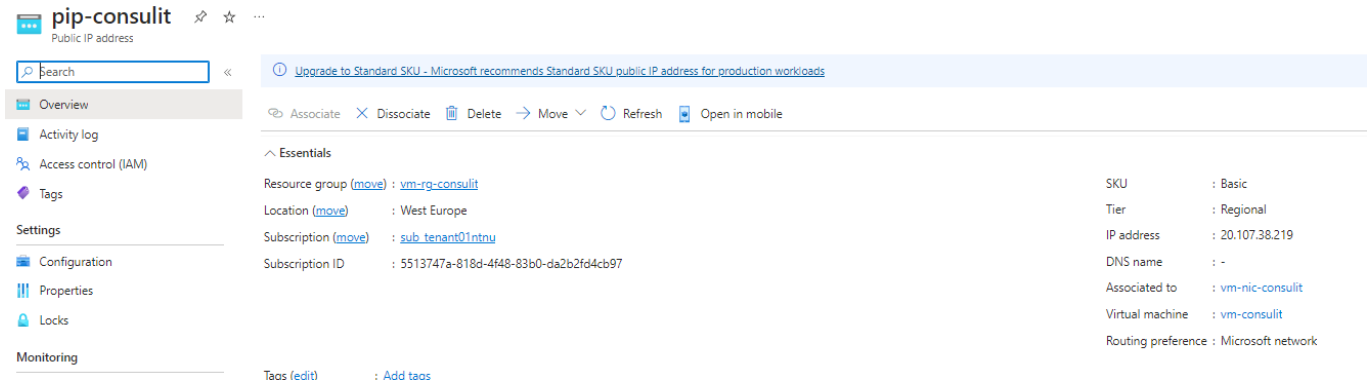
VM Details:



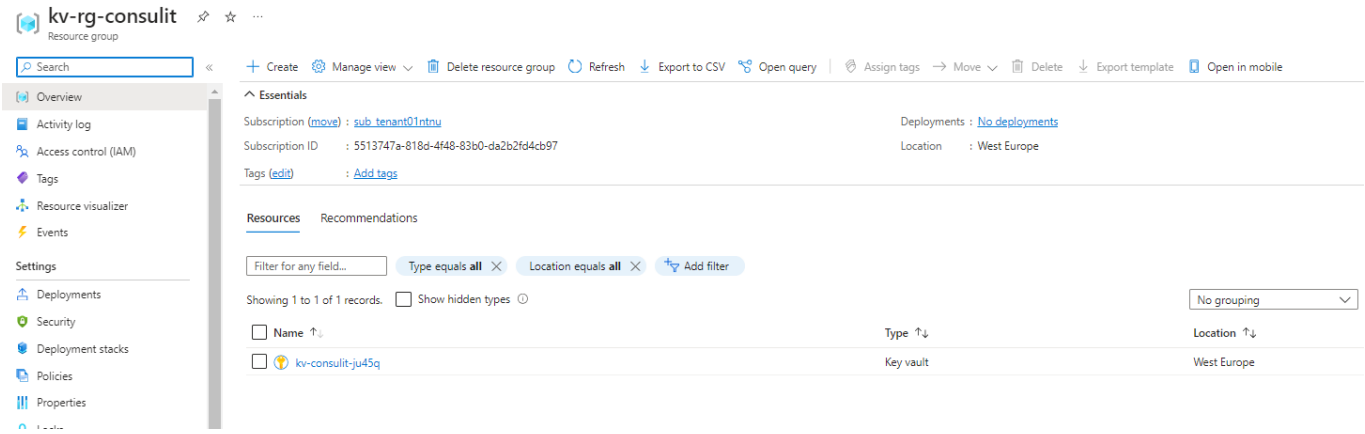
VM NIC:



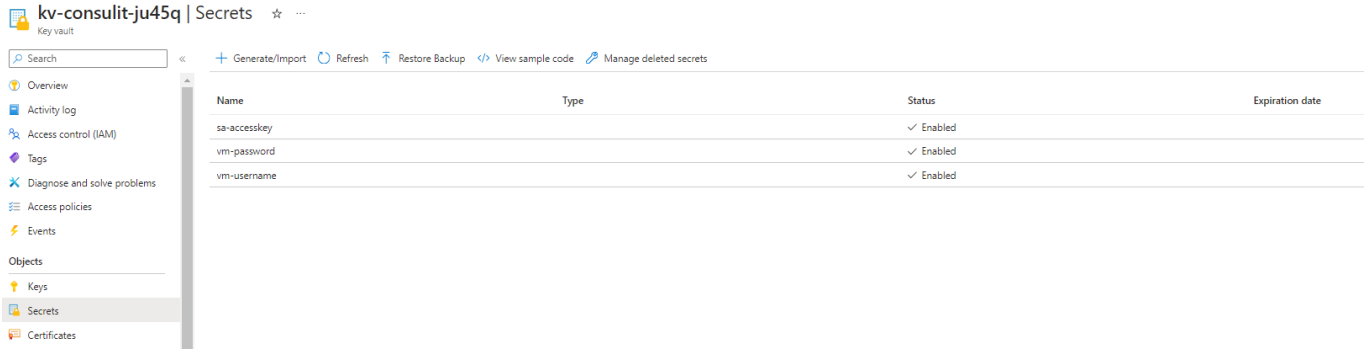
Public IP:



Key Vault



Keyvault Secrets:



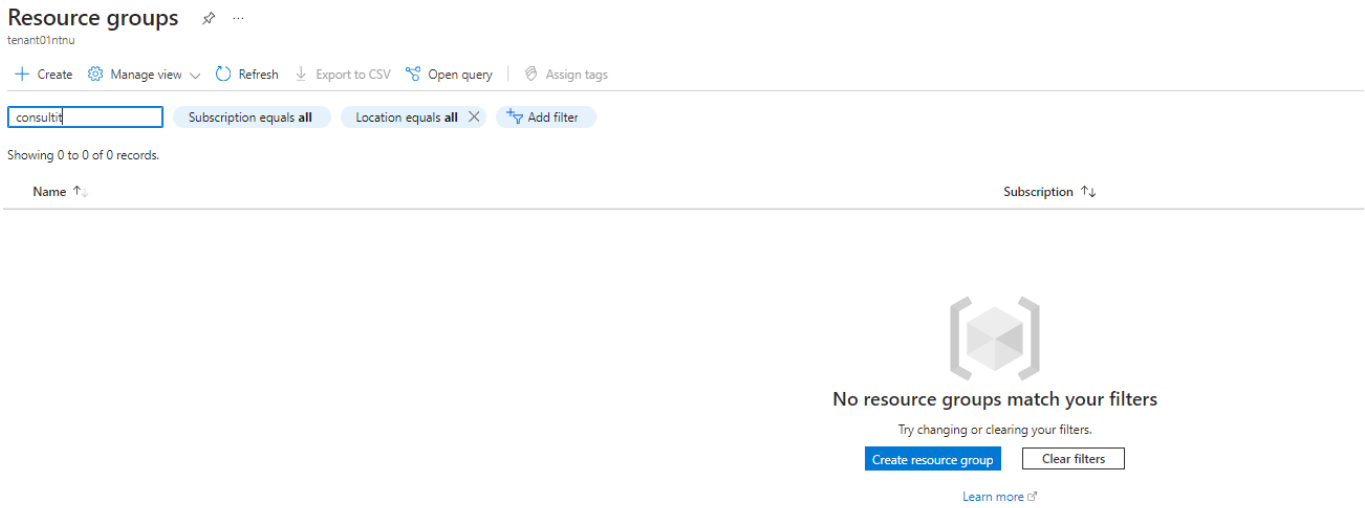
> **Note**: To be able to view these secrets in the Azure Portal, you need to have access to the Key Vault.
> By default we only allowed access for our application, so we need to add access for our user account to see it.
> This can be done by going to the Key Vault, and then going to **Access Policies** and adding your user account with the **Get**, and **List** permission.
> You will not be able to see the values directly.

Terraform

After finishing with the resources, i ran the following command:
terraform destroy

```
Destroy complete! Resources: 21 destroyed.
PS C:\Prog\Private\IIKG3005\Mandatory Assignments\Assignment01\kristoju-oppg1>
```

After running the destroy command, i went to the Azure Portal to check if the resources were deleted.



And then we are done.