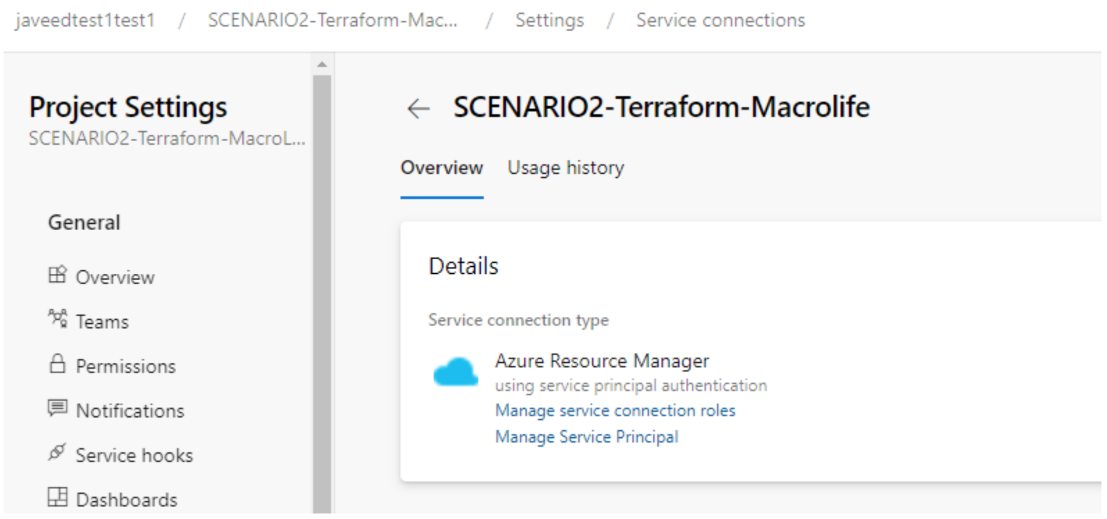
Q2 - SCENARIO

Macro Life, a healthcare company has recently setup the entire Network and Infrastructure on Azure.
The infrastructure has different components such as Virtual N/W, Subnets, NIC, IPs, NSG etc.
The IT team currently has developed PowerShell scripts to deploy each component where all the properties of each resource is set using PowerShell commands.
The business has realized that the PowerShell scripts are growing over period of time and difficult to handover when new admin onboards in the IT.
The IT team has now decided to move to Terraform based deployment of all resources to Azure.
All the passwords are stored in a Azure Service known as key Vault. The deployments needs to be automated using Azure DevOps using IaC(Infrastructure as Code).
1) What are different artifacts you need to create - name of the artifacts and its purpose
2) List the tools you will to create and store the Terraform templates.
3) Explain the process and steps to create automated deployment pipeline.
4) Create a sample Terraform template you will use to deploy Below services:
Vnet
2 Subnet
NSG to open port 80 and 443
1 Window VM in each subnet
1 Storage account
5) Explain how will you access the password stored in Key Vault and use it as Admin Password in the VM Terraform template.

Solution:

Pre-requisites:

1. Service connection (SPN) in azure devops with appropriate access on Azure.



javeedtest1test1 / SCENARIO2-Terraform-Mac... / Settings / Service connections

Project Settings
SCENARIO2-Terraform-MacroL...

General
Overview
Teams
Permissions
Notifications
Service hooks
Dashboards

SCENARIO2-Terraform-Macrolife

Overview | Usage history

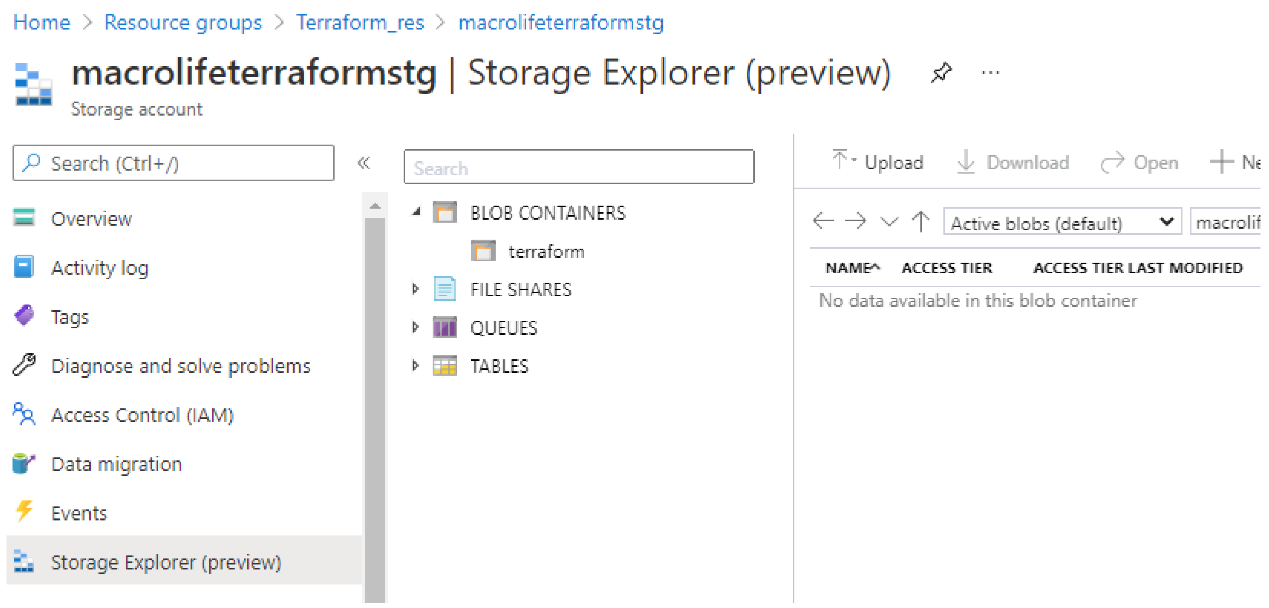
Details

Service connection type

Azure Resource Manager
using service principal authentication
Manage service connection roles
Manage Service Principal

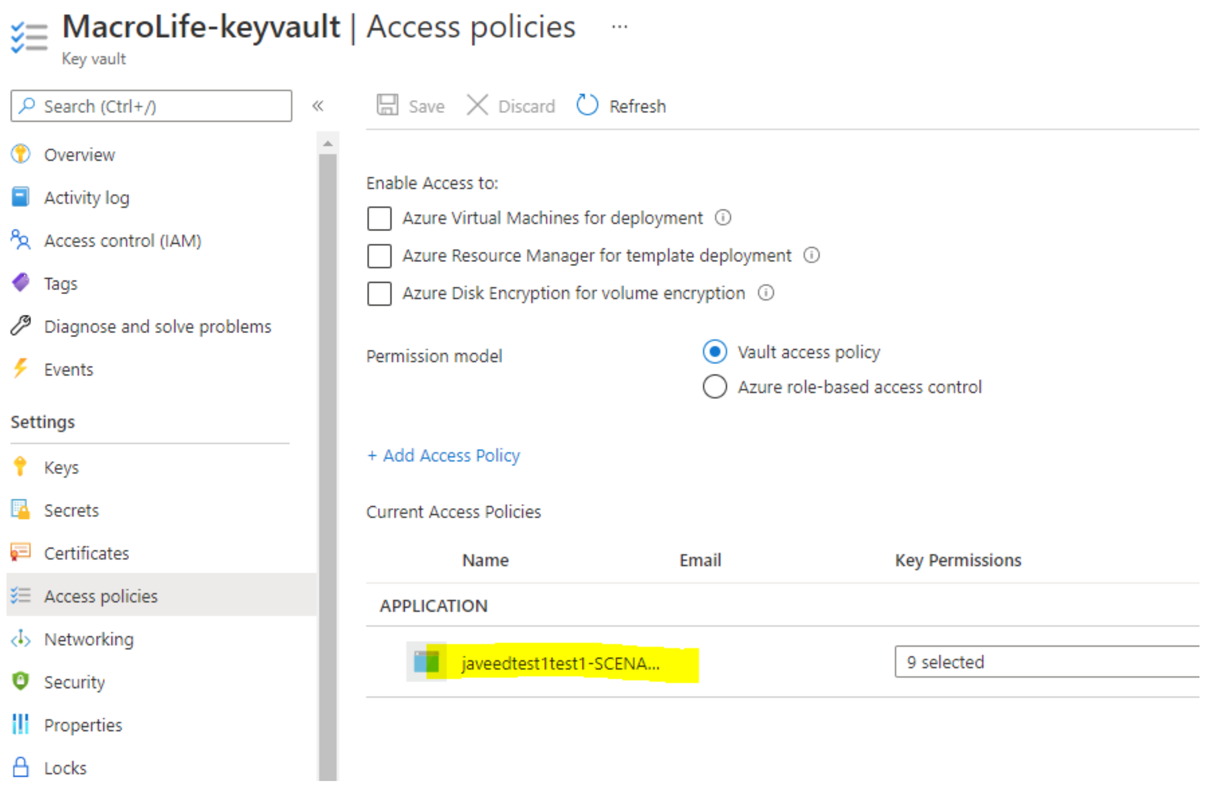
2. Storage Account with a container to store terraform.tfstate for terraform init task during release pipeline or we can also hard code to use storage account to store remote state file. For this demo I am not hard coding remote state file. Example of hard coding remote state file is below, if needed. SPN should also have access on storage account. (We can also create storage account before terraform init with a task but I usually prefer to have it created before hand)

```
terraform {
  backend "azurerm" {
    resource_group_name  = "Terraform_res"
    storage_account_name = "macrolifeterraformstg"
    container_name       = "terraform"
      access_key = "AccessKey"
    }
}
```



No data available in this blob container

3. Key vault with VM password secret, service connection (SPN) should be allowed in key vault access policies and request source IP should be added in firewall.(MS trusted services should be allowed)



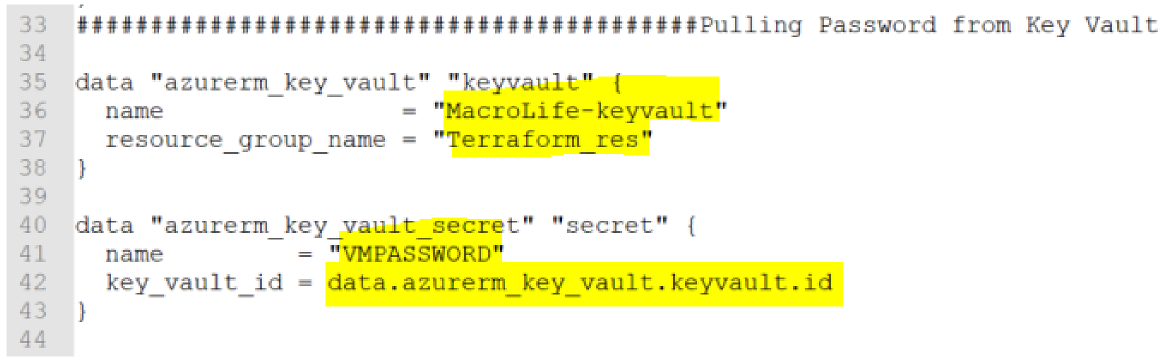
Enable Access to:
- ☐ Azure Virtual Machines for deployment
- ☐ Azure Resource Manager for template deployment
- ☐ Azure Disk Encryption for volume encryption

Permission model: [✓ Vault access policy] Azure role-based access control

Current Access Policies

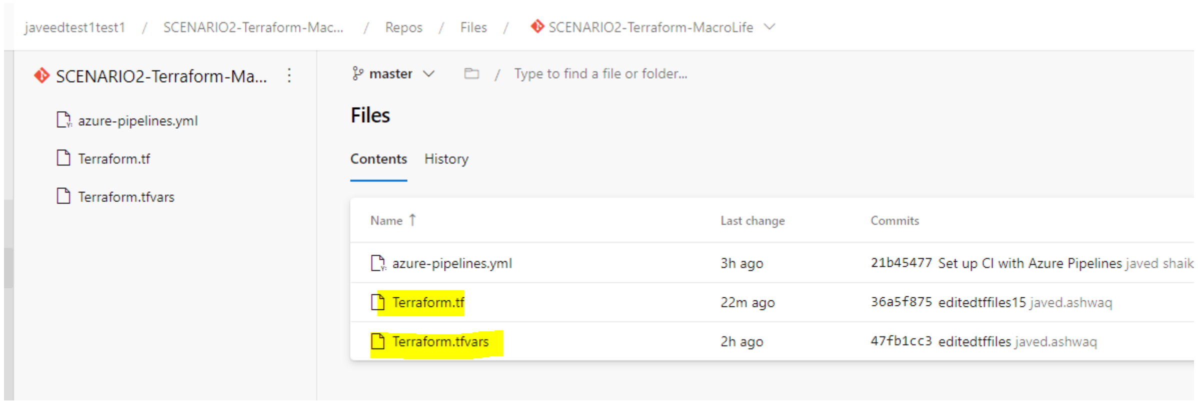
Name	Email	Key Permissions
APPLICATION		
javeedtest1test1-SCENA...		9 selected

4. Update key vault details in Terraform.tf file (between line 35 to 43).as show below.


```
33 ###########################################Pulling Password from Key Vault
34
35 data "azurerm_key_vault" "keyvault" {
36   name                = "MacroLife-keyvault"
37   resource_group_name = "Terraform_res"
38 }
39
40 data "azurerm_key_vault_secret" "secret" {
41   name         = "VMPASSWORD"
42   key_vault_id = data.azurerm_key_vault.keyvault.id
43 }
44
```

Configuration

1. Create a repo with terraform files.(you can ignore the YAML file below, it was created as I already ran build pipeline).



javeedtest1test1 / SCENARIO2-Terraform-Mac... / Repos / Files / SCENARIO2-Terraform-MacroLife

SCENARIO2-Terraform-Ma...
- azure-pipelines.yml
- Terraform.tf
- Terraform.tfvars

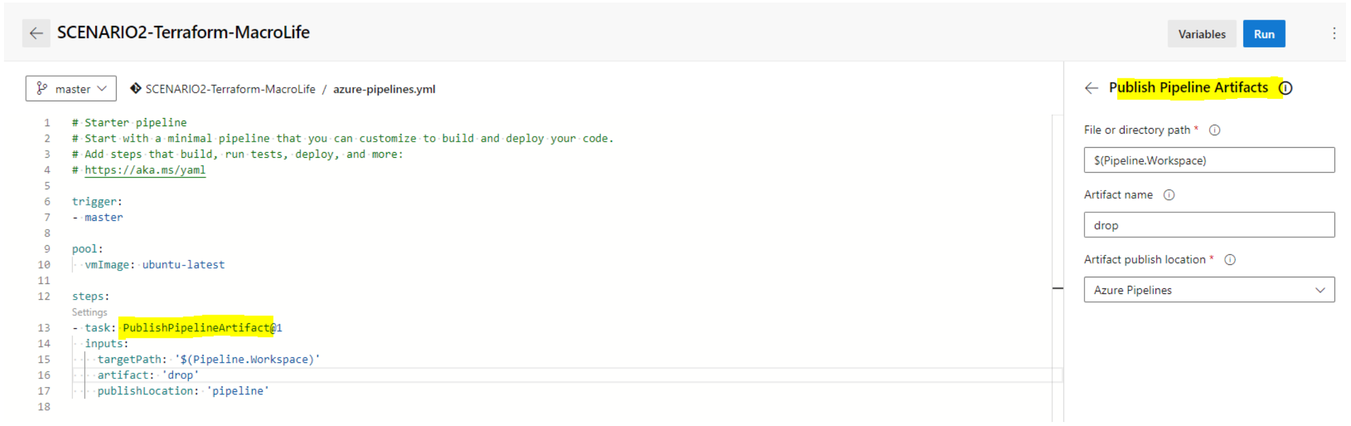
master / Type to find a file or folder...

Files

Contents History

Name ↑	Last change	Commits
azure-pipelines.yml	3h ago	21b45477 Set up CI with Azure Pipelines javed shaik
Terraform.tf	22m ago	36a5f875 editedtffiles15 javed.ashwaq
Terraform.tfvars	2h ago	47fb1cc3 editedtffiles javed.ashwaq

2. Since there is nothing to compile or build, I am just using release pipeline to drop repo files to artifacts. I used 'publish pipeline artifacts' to create artifacts.



SCENARIO2-Terraform-MacroLife

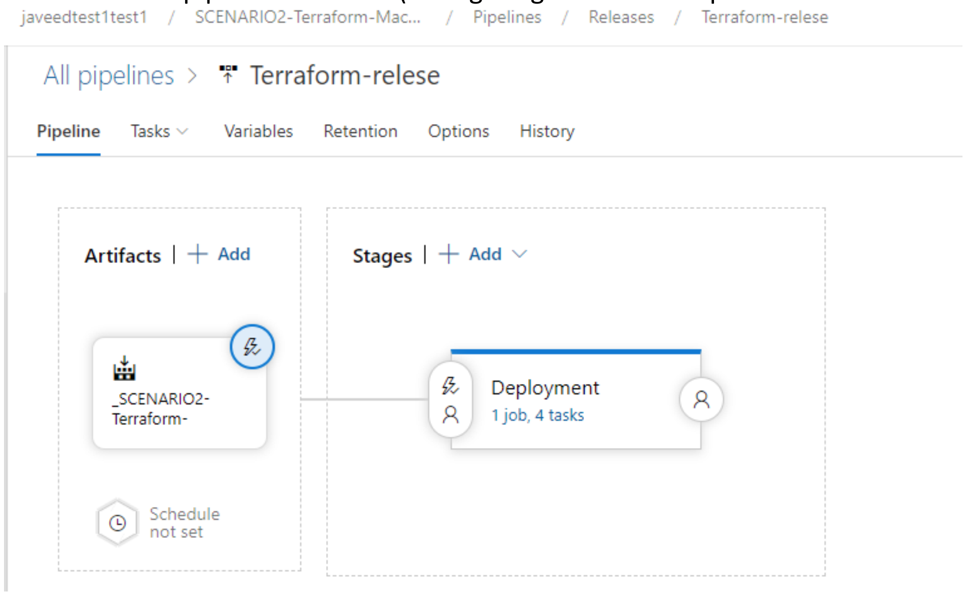
master SCENARIO2-Terraform-MacroLife / azure-pipelines.yml

```
# Starter pipeline
# Start with a minimal pipeline that you can customize to build and deploy your code.
# Add steps that build, run tests, deploy, and more:
# https://aka.ms/yaml

trigger:
- master

pool:
  vmImage: ubuntu-latest

steps:
Settings
- task: PublishPipelineArtifact@1
  inputs:
    targetPath: '$(Pipeline.Workspace)'
    artifact: 'drop'
    publishLocation: 'pipeline'
```

Publish Pipeline Artifacts

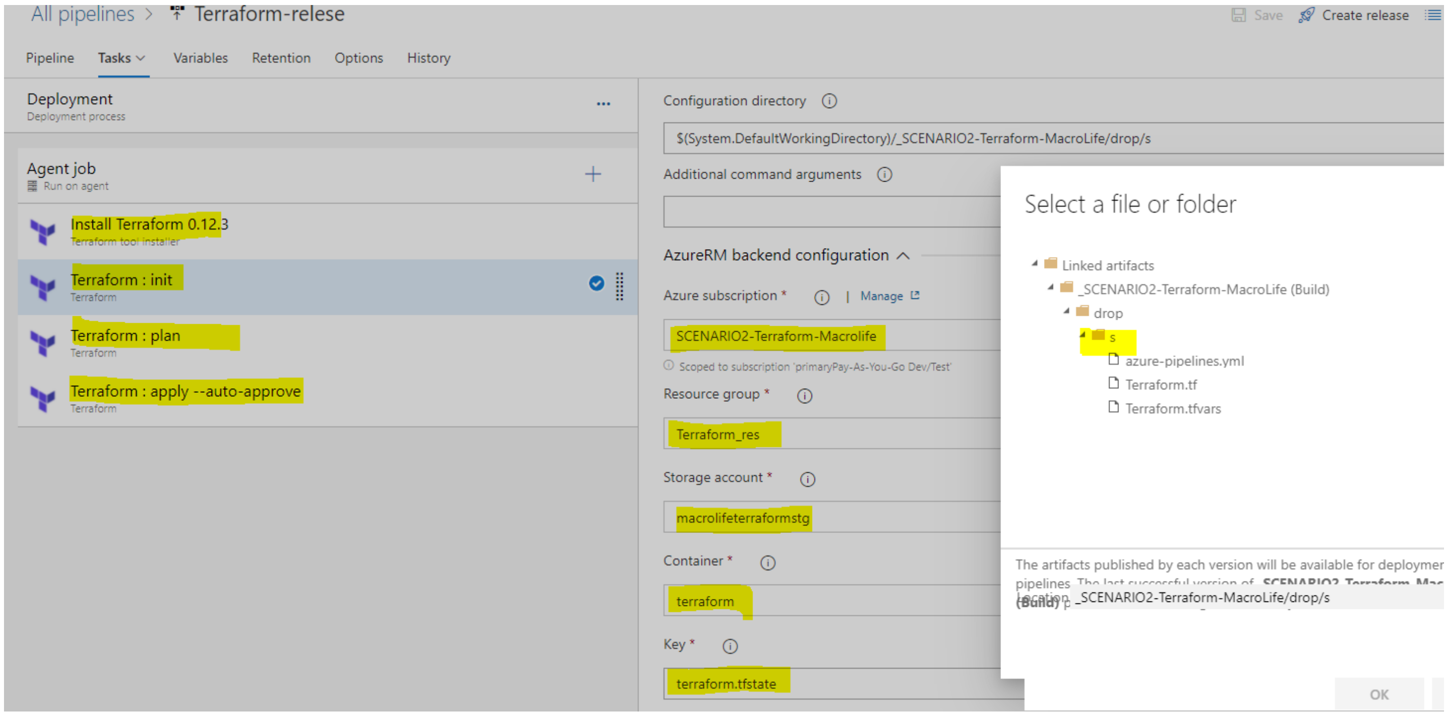
File or directory path *	$(Pipeline.Workspace)
Artifact name	drop
Artifact publish location *	Azure Pipelines

3. Create release pipeline as below.(Configuring artifacts as input with continuous trigger)



javeedtest1test1 / SCENARIO2-Terraform-Mac... / Pipelines / Releases / Terraform-relese

4. Create 4 tasks as below in deployment stage,



All pipelines > Terraform-relese

Pipeline | Tasks | Variables | Retention | Options | History

Deployment
Deployment process

Agent job
Run on agent

- Install Terraform 0.12.3 — Terraform tool installer
- Terraform : init — Terraform
- Terraform : plan — Terraform
- Terraform : apply --auto-approve — Terraform

Configuration directory: $(System.DefaultWorkingDirectory)/_SCENARIO2-Terraform-MacroLife/drop/s

Additional command arguments:

AzureRM backend configuration

Azure subscription *: SCENARIO2-Terraform-Macrolife
Scoped to subscription 'primaryPay-As-You-Go Dev/Test'

Resource group *: Terraform_res

Storage account *: macrolifeterraformstg

Container *: terraform

Key *: terraform.tfstate

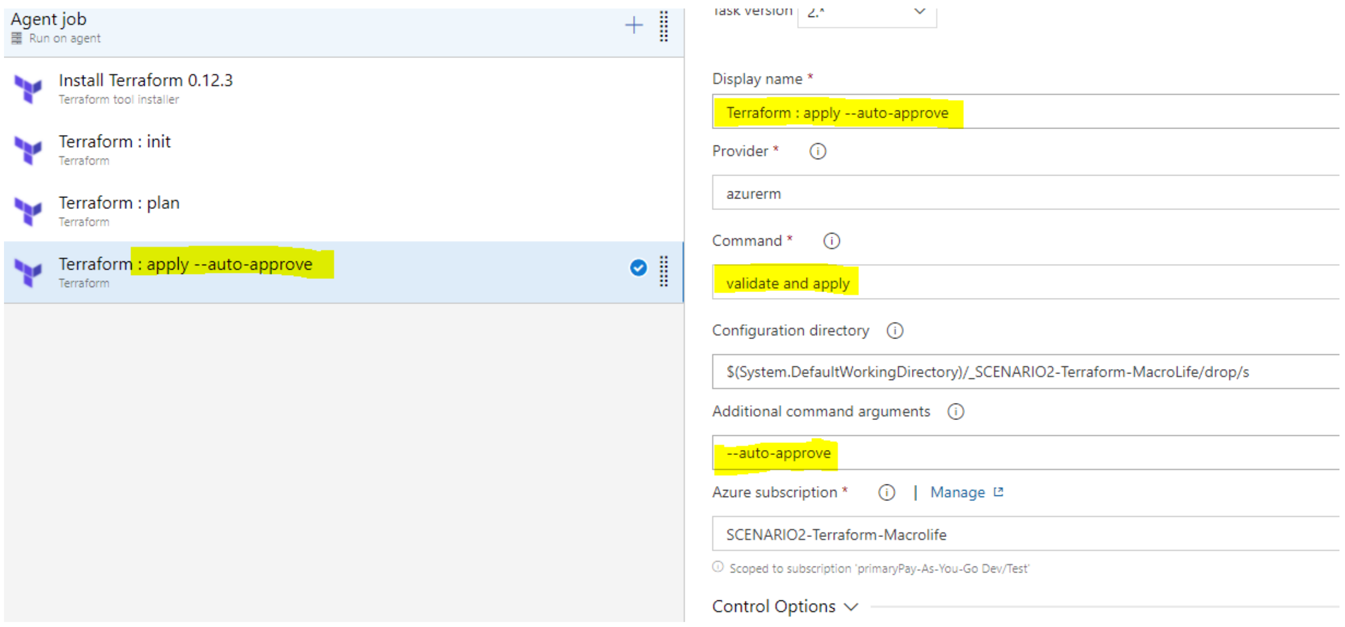
Select a file or folder

```
Linked artifacts
  _SCENARIO2-Terraform-MacroLife (Build)
    drop
      s
        azure-pipelines.yml
        Terraform.tf
        Terraform.tfvars
```

The artifacts published by each version will be available for deploymer pipelines. The last successful version of _SCENARIO2-Terraform-Mac (Build) p

Location: _SCENARIO2-Terraform-MacroLife/drop/s

1. Terraform installer - To install Terraform.
2. Terraform Init - Fill the storage account, container details and configure directory should be the folder which has terraform files (.tf).
3. Terraform plan - configure directory should be the folder which has terraform files (.tf).
4. Terraform apply – should also have the same configuration directory as above, in additional arguments give –auto-approve so that during the deployment it will not prompt for confirmation.



Agent job
Run on agent

- Install Terraform 0.12.3 — Terraform tool installer
- Terraform : init — Terraform
- Terraform : plan — Terraform
- Terraform : apply --auto-approve — Terraform

Task version: 2.*

Display name *: Terraform : apply --auto-approve

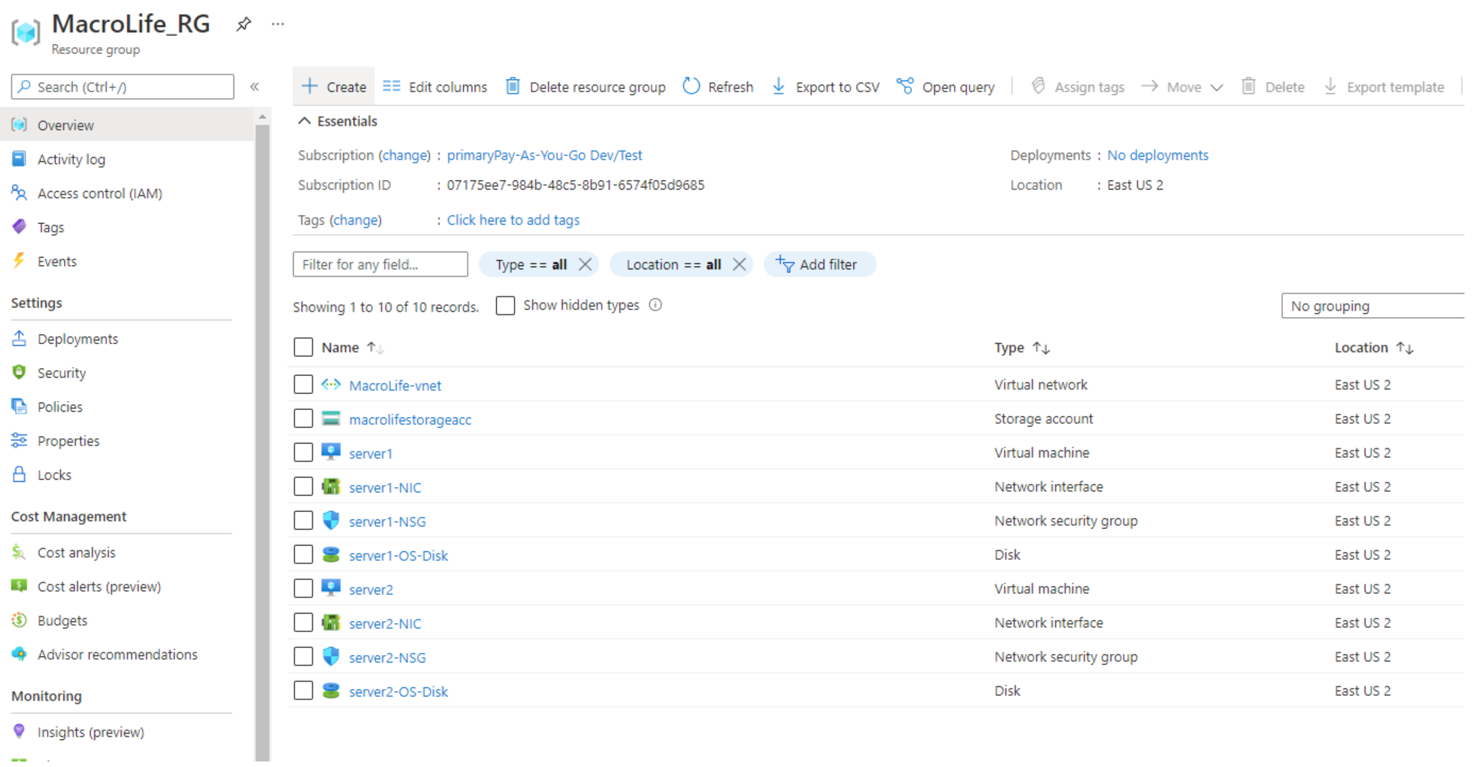
Provider *: azurerm

Command *: validate and apply

Configuration directory: $(System.DefaultWorkingDirectory)/_SCENARIO2-Terraform-MacroLife/drop/s

Additional command arguments: --auto-approve

Azure subscription * | Manage: SCENARIO2-Terraform-Macrolife
Scoped to subscription 'primaryPay-As-You-Go Dev/Test'



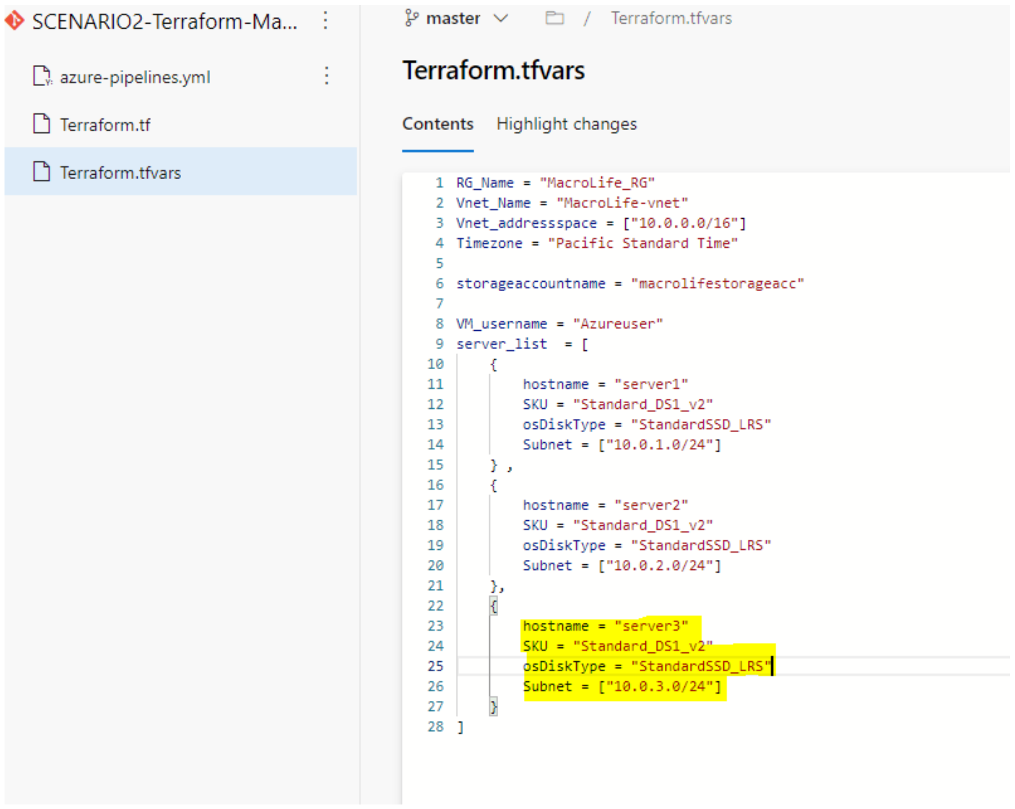
Subscription (change) : primaryPay-As-You-Go Dev/Test
Subscription ID : 07175ee7-984b-48c5-8b91-6574f05d9685
Tags (change) : Click here to add tags
Deployments : No deployments
Location : East US 2

Showing 1 to 10 of 10 records.

Name	Type	Location
MacroLife-vnet	Virtual network	East US 2
macrolifestorageacc	Storage account	East US 2
server1	Virtual machine	East US 2
server1-NIC	Network interface	East US 2
server1-NSG	Network security group	East US 2
server1-OS-Disk	Disk	East US 2
server2	Virtual machine	East US 2
server2-NIC	Network interface	East US 2
server2-NSG	Network security group	East US 2
server2-OS-Disk	Disk	East US 2

Notes on Terraform.tfvars File

1. Unlike hard coding all the values in Terraform.tf file I have separated it in to Terraform.tfvars file so that it would be easy to change values if needed.
2. Also, I configured VM deployments keeping scaling in mind. So, if you want to deploy 3 server or only 1 server. Just server list variables need to be updated with appropriate values. According to that it would deploy 3rd server, subnet & NSG along with it.



```
RG_Name = "MacroLife_RG"
Vnet_Name = "MacroLife-vnet"
Vnet_addressspace = ["10.0.0.0/16"]
Timezone = "Pacific Standard Time"

storageaccountname = "macrolifestorageacc"

VM_username = "Azureuser"
server_list  = [
    {
        hostname = "server1"
        SKU = "Standard_DS1_v2"
        osDiskType = "StandardSSD_LRS"
        Subnet = ["10.0.1.0/24"]
    } ,
    {
        hostname = "server2"
        SKU = "Standard_DS1_v2"
        osDiskType = "StandardSSD_LRS"
        Subnet = ["10.0.2.0/24"]
    },
    {
        hostname = "server3"
        SKU = "Standard_DS1_v2"
        osDiskType = "StandardSSD_LRS"
        Subnet = ["10.0.3.0/24"]
    }
]
```