**Follow these steps to setup a DevOps Pipeline to deploy Terraform in Azure**

**The below creates Resource Group etc required to store Terraform state file**

1. Create Resource Group

#Create Resource Group

Az group create -l eastus2 -n hyperv-migration-landingzone-tf

Output below:-

#Example output

#{

#    "id": "/subscriptions/27f048cd-d37e-4655-8fbe-2e41b14d7327/resourceGroups/hyperv-migration-landingzone-tf",

#    "location": "eastus2",

#    "managedBy": null,

#    "name": "hyperv-migration-landingzone-tf",

#    "properties": {

#      "provisioningState": "Succeeded"

#    },

#    "tags": null,

#    "type": "Microsoft.Resources/resourceGroups"

#  }

1. Create service connection now on Azure DevOps

(after creation give IAM permissions as contributor to the subscription)

Select **Project settings** in Azure DevOps



Select **Service connections**



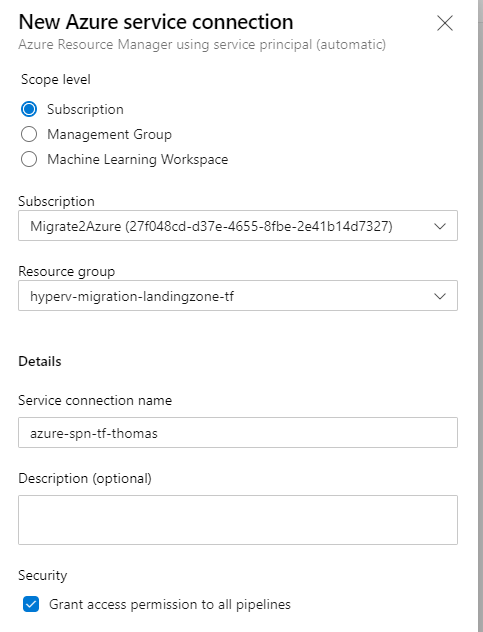
Select **Create service connection** -> **Azure Resource Manager**



Select **Service principal (automatic)**



Select as below – example, change to your specific naming convention(s)



#Create Storage Account to store Terraform State file

az storage account create --resource-group hyperv-migration-landingzone-tf --name thomastfstate --sku Standard\_LRS --encryption-services blob

#Get Storage account key

az storage account keys list --resource-group hyperv-migration-landingzone-tf --account-name thomastfstate

Take copy of key1 value

#Create Storage Account Container

az storage container create --name thomastfstatecontainer --account-name thomastfstate --account-key "op27CuDyTy5eubmwXJRujIfWCsVHp/vOjLdxPO9KqTGFI/facCItMOaDlLbsuJYkNyzHQULIOQJO2h+icBucYg=="

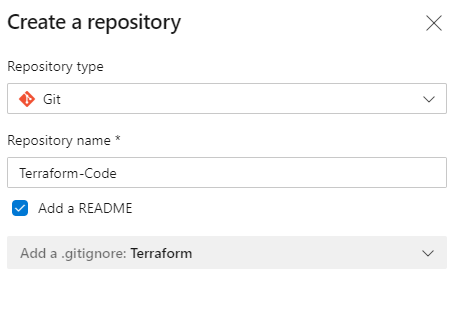
1. Create new Repo in Azure DevOps

Select **Repos**

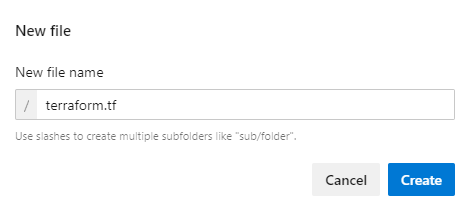


Select icon -> New Repository





Create new terraform.tf example to run pipeline



Example terraform.tf file

*provider* "azurerm" {

    version = "=2.13.0"

    features {}

}

teraform {

    backend "azurerm" {

        resource\_group\_name = "hyperv-migration-landingzone-tf"

        storage\_account\_name = "thomastfstate"

        container\_name = "terraform.tfstate"

    }

}

*data* "azurerm\_client\_config" "current" {}

*resource* "azurerm\_resouce\_group" "resourcegroup" {

    name = "thomas-test-rg"

    location = "eastus2"

}

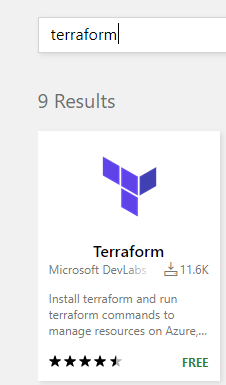
Commit your code



1. Install Terraform extension from <https://marketplace.visualstudio.com/>

Select this **icon** top right -> **browse marketplace**





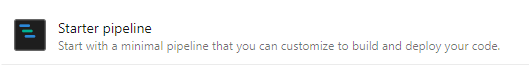
Install Terraform (Follow instructions to install to correct organization)

1. Setup Terraform build

Select **Repos** -> **Setup build**



Select **Starter Pipeline**



Example below of pipeline (to stage and then deploy terraform)

stages :

  - stage: validate

    jobs:

    - job: validate

      continueOnError: false

      steps:

      - task: TerraformInstaller@0

        displayName: 'install'

        inputs:

          terraformVersion: '0.12.3'

      - task: TerraformTaskV1@0

        displayName: 'init'

        inputs:

          provider: 'azurerm'

          command: 'init'

          backendServiceArm: 'azure-spn-tf-thomas'

          backendAzureRmResourceGroupName: 'hyperv-migration-landingzone-tf'

          backendAzureRmStorageAccountName: 'thomastfstate'

          backendAzureRmContainerName: 'thomastfstatecontainer'

          backendAzureRmKey: 'terraform.tfstate'

      - task: TerraformTaskV1@0

        displayName: 'validate'

        inputs:

          provider: 'azurerm'

          command: 'validate'

  - stage: deploy

    jobs:

    - deployment: deploy\_terraform

      continueOnError: false

      environment: 'dev'

      strategy:

       runOnce:

         deploy:

            steps:

              - checkout: self

              - task: TerraformInstaller@0

                displayName: 'install'

                inputs:

                  terraformVersion: '0.12.3'

              - task: TerraformTaskV1@0

                displayName: 'init'

                inputs:

                  provider: 'azurerm'

                  command: 'init'

                  backendServiceArm: 'azure-spn-tf-thomas'

                  backendAzureRmResourceGroupName: 'hyperv-migration-landingzone-tf'

                  backendAzureRmStorageAccountName: 'thomastfstate'

                  backendAzureRmContainerName: 'thomastfstatecontainer'

                  backendAzureRmKey: 'terraform.tfstate'

              - task: TerraformTaskV1@0

                displayName: 'plan'

                inputs:

                  provider: 'azurerm'

                  command: 'plan'

                  environmentServiceNameAzureRM: 'azure-spn-tf-thomas'

              - task: TerraformTaskV1@0

                displayName: 'apply'

                inputs:

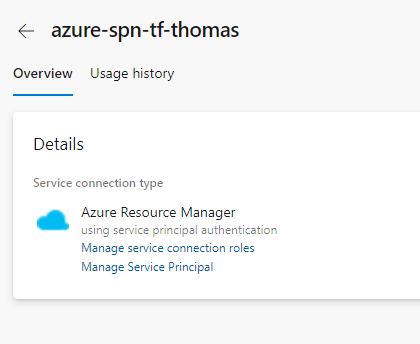
                  provider: 'azurerm'

                  command: 'apply'

                  environmentServiceNameAzureRM: 'azure-spn-tf-thomas'

1. Give Azure Devops SPN contributor access to the subscription(service connection in DevOps)

Select the service connection you previous created and ensure it Is added as **contributor** to the subscription that you will be deploying



1. Deploy using Pipeline

If following steps above, when you deploy code to master branch the pipeline runs automatically between validate & deploy stages.

* Validate stage, validates code and will error if not correct

