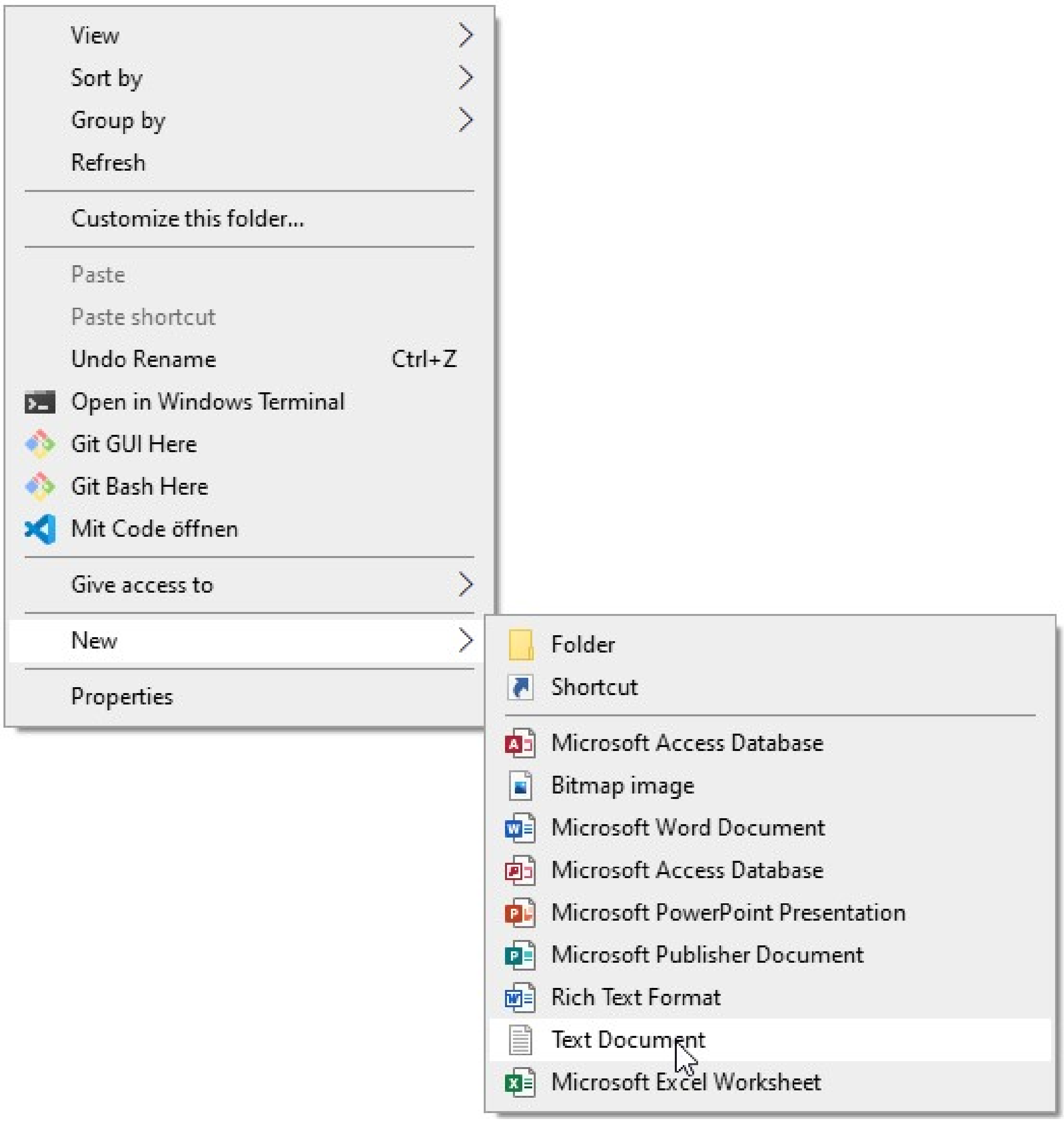
Terraform Training Step-by-Step Documentation

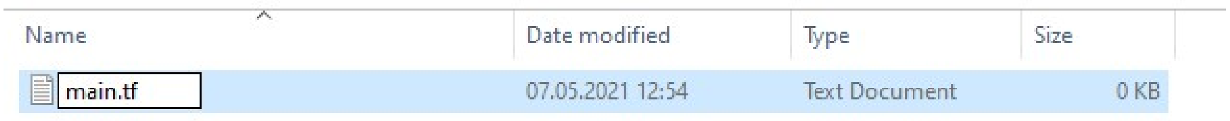
This Document should be used as a Step-by-Step Guide to proceed the Steps of the GKGAB Hands-On Labs within the Terraform Basic Trainings.

Task 1:

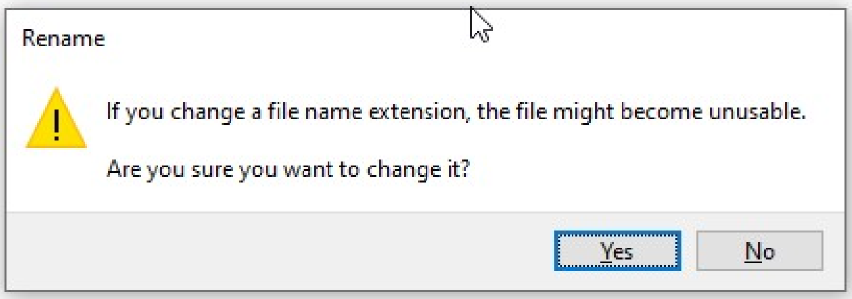
- Create a Folder on your Client (e.g. C:\Terraform\Demo\)
- Navigate to this Folder and create a Text Document



- Rename this Text Document to main.tf



Name	Date modified	Type	Size
main.tf	07.05.2021 12:54	Text Document	0 KB

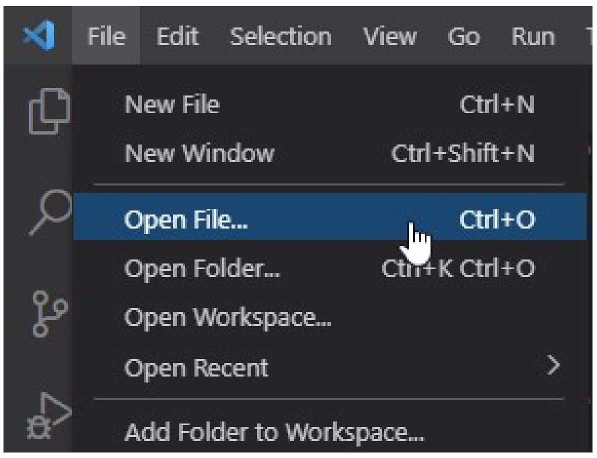


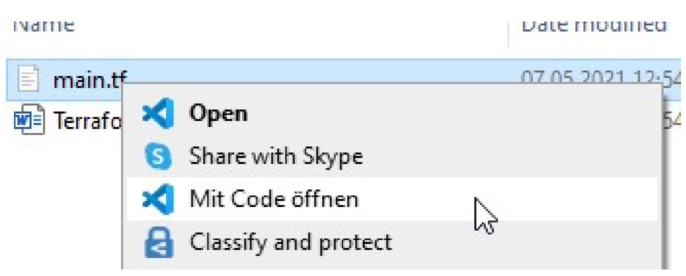
Rename

If you change a file name extension, the file might become unusable.

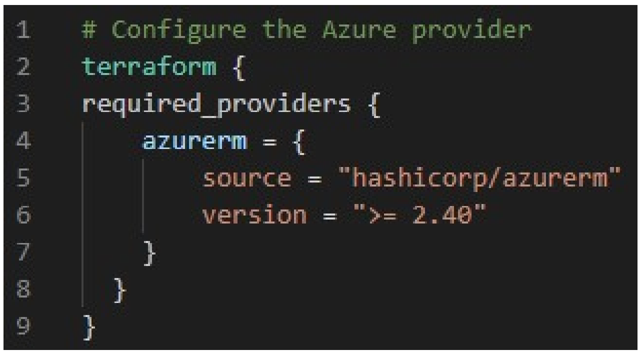
Are you sure you want to change it?

- Open this File with Visual Studio Code



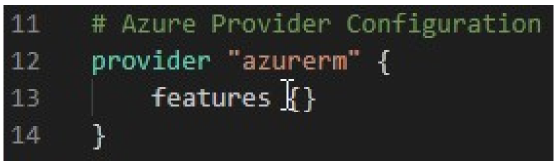


- Start to create the Terraform Provider Configuration Block



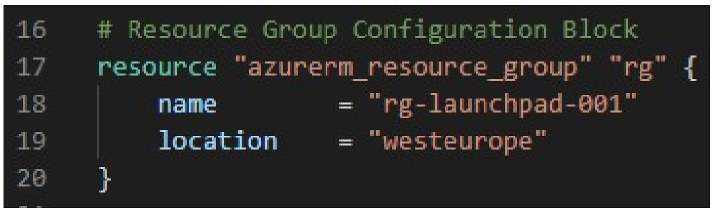
```
# Configure the Azure provider
terraform {
required_providers {
    azurerm = {
        source = "hashicorp/azurerm"
        version = ">= 2.40"
    }
  }
}
```

- Define the Provider



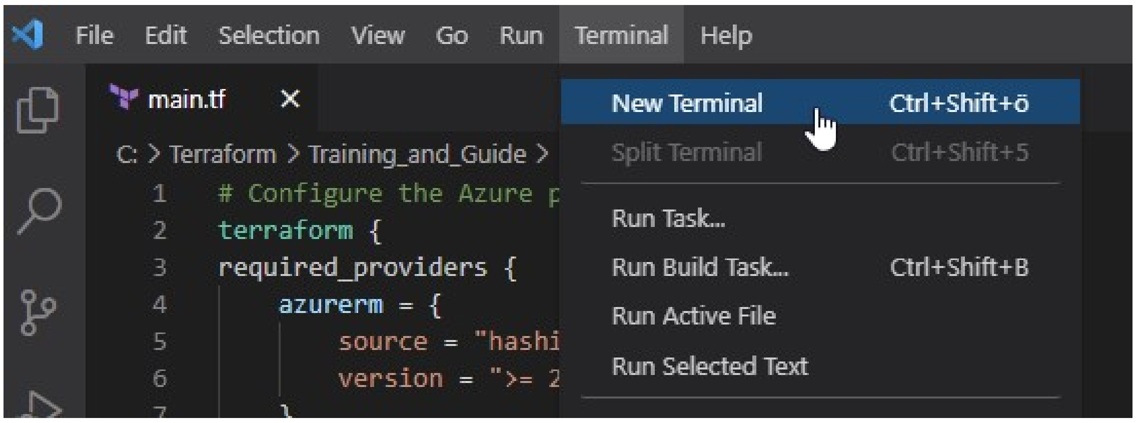
```
# Azure Provider Configuration
provider "azurerm" {
    features {}
}
```

- Add the Resource Group Configuration Block



```
# Resource Group Configuration Block
resource "azurerm_resource_group" "rg" {
    name         = "rg-launchpad-001"
    location     = "westeurope"
}
```

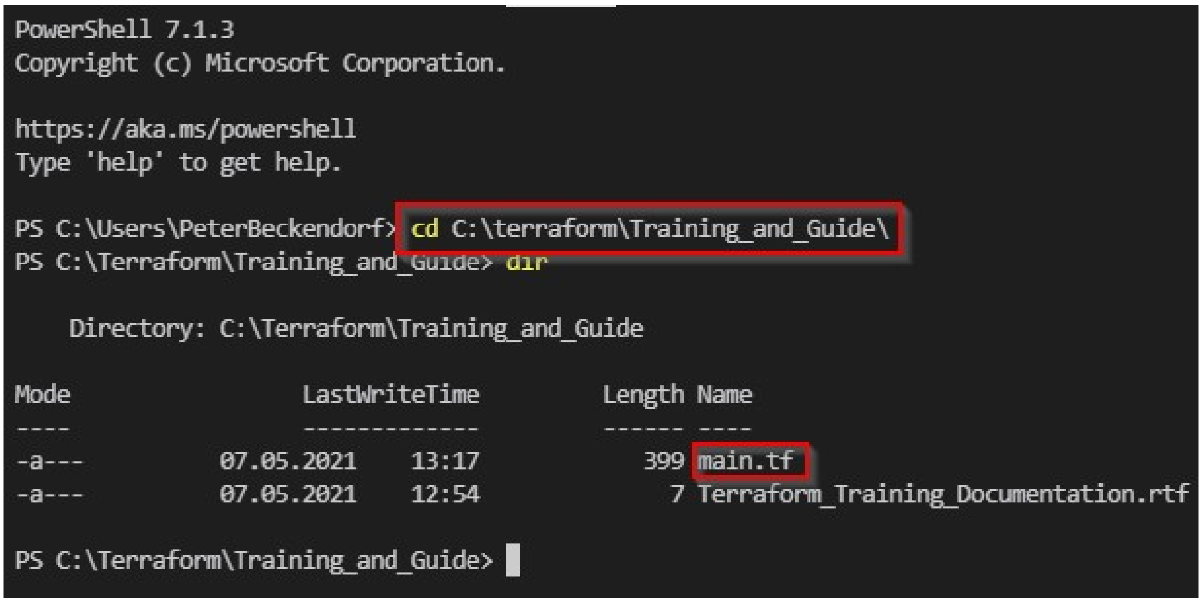
- Save the Configured File -> use STRG+S as Save-Shortcut
- Now open a terminal Session within your Visual Studio Code



File
Edit
Selection
View
Go
Run
Terminal
Help
main.tf
C: > Terraform > Training_and_Guide >
New Terminal Ctrl+Shift+ö
Split Terminal Ctrl+Shift+5
Run Task...
Run Build Task... Ctrl+Shift+B
Run Active File
Run Selected Text

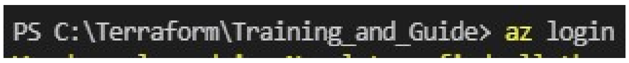
```
# Configure the Azure p
terraform {
required_providers {
    azurerm = {
        source = "hashi
        version = ">= 2
```

- Be sure it is a Powershell Terminal
- Navigate to your location of the Main.tf File

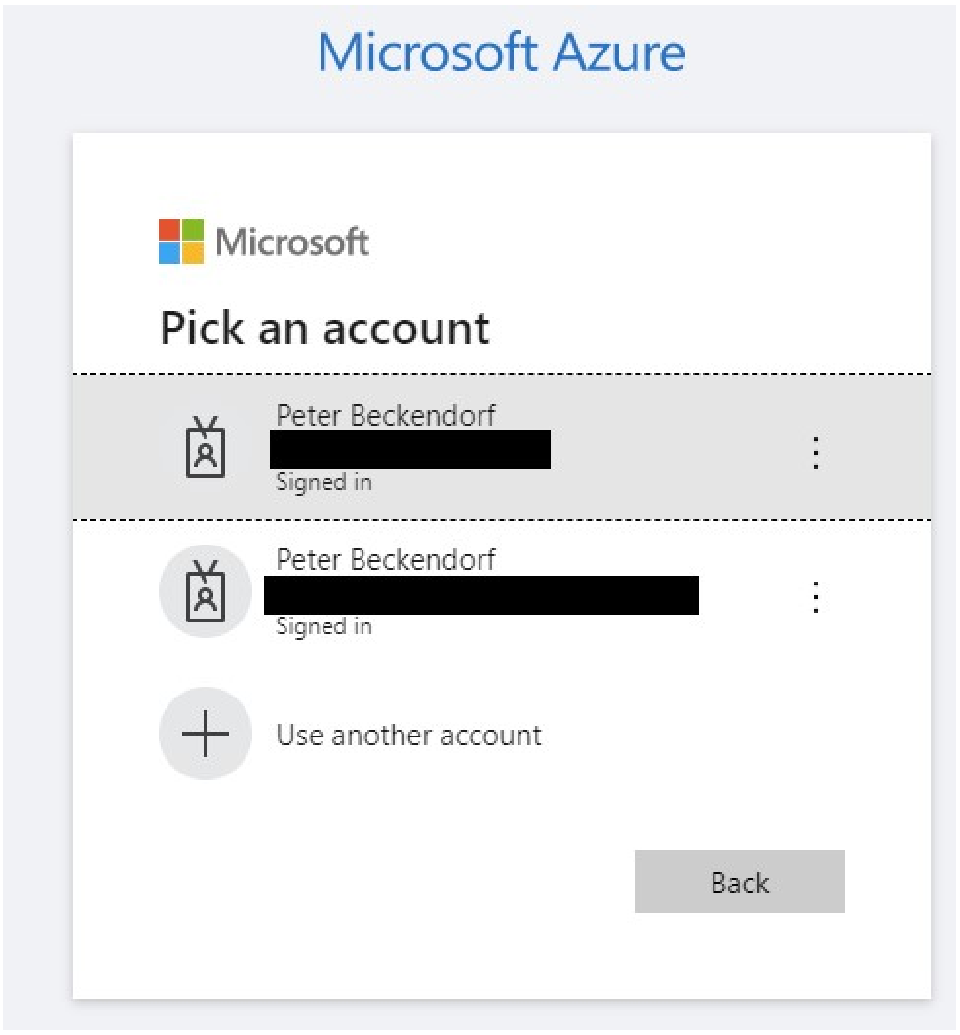


```
PowerShell 7.1.3
Copyright (c) Microsoft Corporation.

https://aka.ms/powershell
Type 'help' to get help.

PS C:\Users\PeterBeckendorf> cd C:\terraform\Training_and_Guide\
PS C:\Terraform\Training_and_Guide> dir

    Directory: C:\Terraform\Training_and_Guide

Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a---          07.05.2021    13:17            399 main.tf
-a---          07.05.2021    12:54              7 Terraform_Training_Documentation.rtf

PS C:\Terraform\Training_and_Guide>
```

- Next you have to login to azure, for this use ***az login***



```
PS C:\Terraform\Training_and_Guide> az login
```



Microsoft Azure

Microsoft

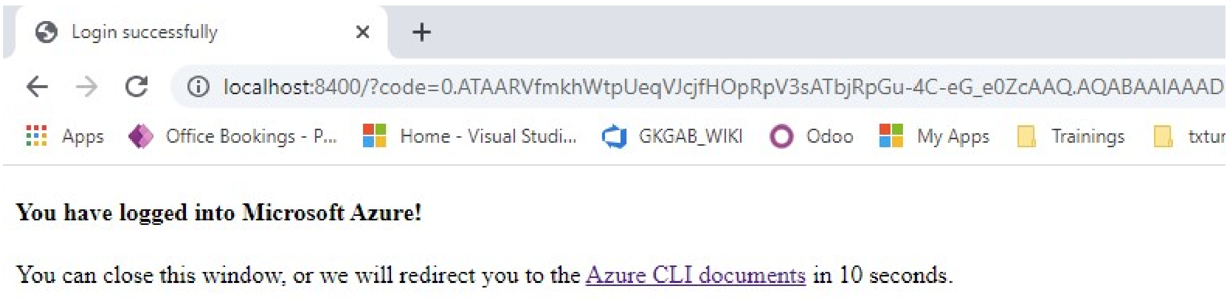
Pick an account

Peter Beckendorf
[redacted]
Signed in

Peter Beckendorf
[redacted]
Signed in

Use another account

Back



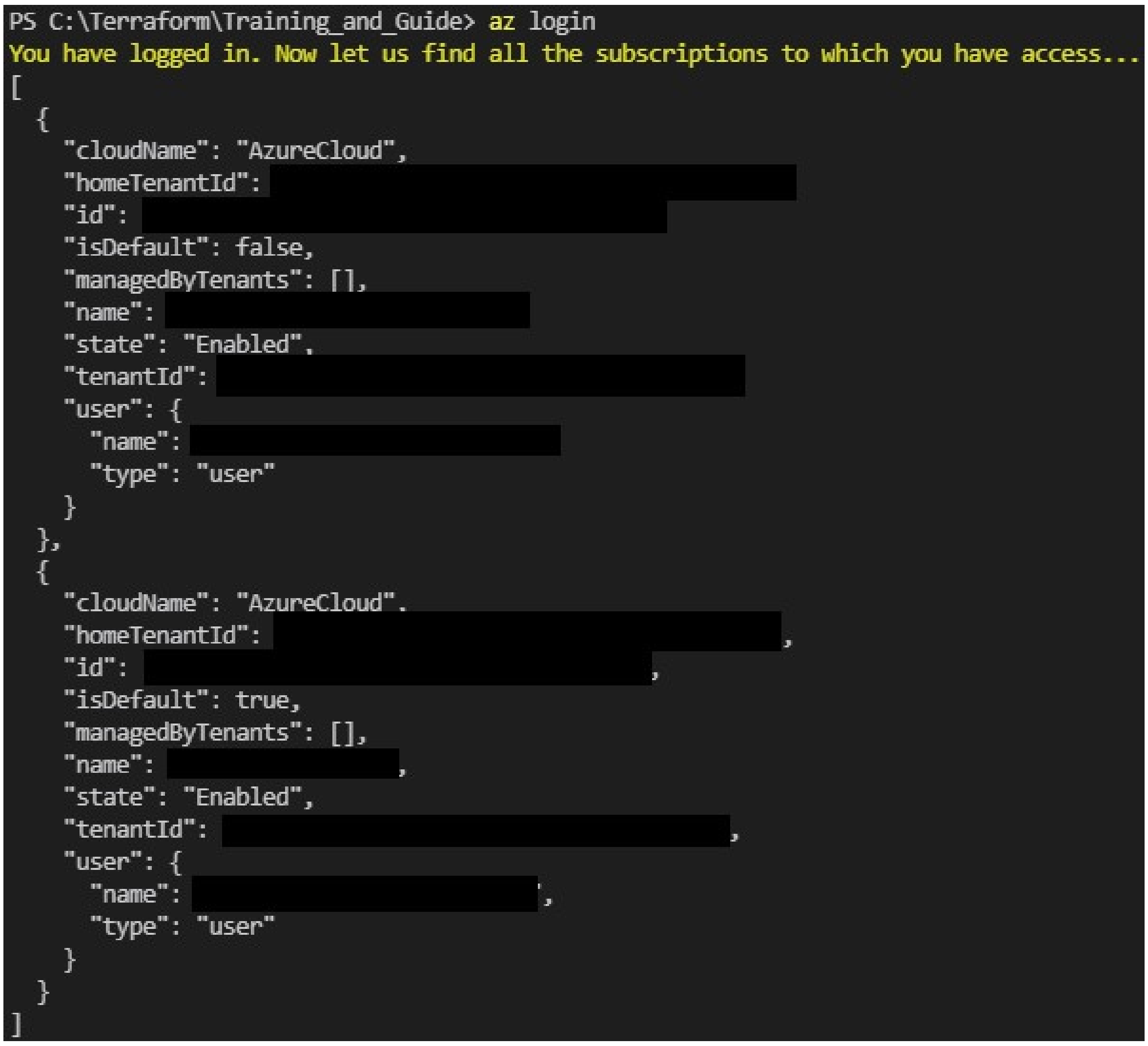
Login successfully

localhost:8400/?code=0.ATAARVfmkhWtpUeqVJcjfHOpRpV3sATbjRpGu-4C-eG_e0ZcAAQ.AQABAAIAAAD

Apps Office Bookings - P... Home - Visual Studi... GKGAB_WIKI Odoo My Apps Trainings txtur

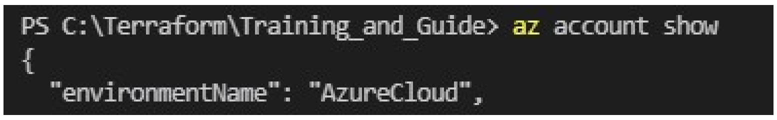
You have logged into Microsoft Azure!

You can close this window, or we will redirect you to the Azure CLI documents in 10 seconds.



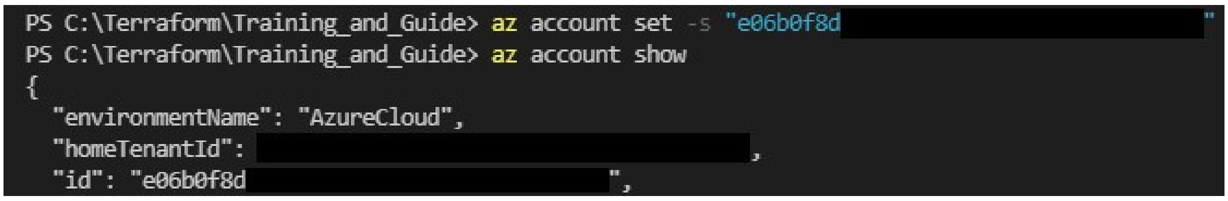
```
PS C:\Terraform\Training_and_Guide> az login
You have logged in. Now let us find all the subscriptions to which you have access...
[
  {
    "cloudName": "AzureCloud",
    "homeTenantId": [redacted]
    "id": [redacted]
    "isDefault": false,
    "managedByTenants": [],
    "name": [redacted]
    "state": "Enabled",
    "tenantId": [redacted]
    "user": {
      "name": [redacted]
      "type": "user"
    }
  },
  {
    "cloudName": "AzureCloud",
    "homeTenantId": [redacted],
    "id": [redacted],
    "isDefault": true,
    "managedByTenants": [],
    "name": [redacted],
    "state": "Enabled",
    "tenantId": [redacted],
    "user": {
      "name": [redacted],
      "type": "user"
    }
  }
]
```

- Check if your in the correct subsriction with ***az account show***



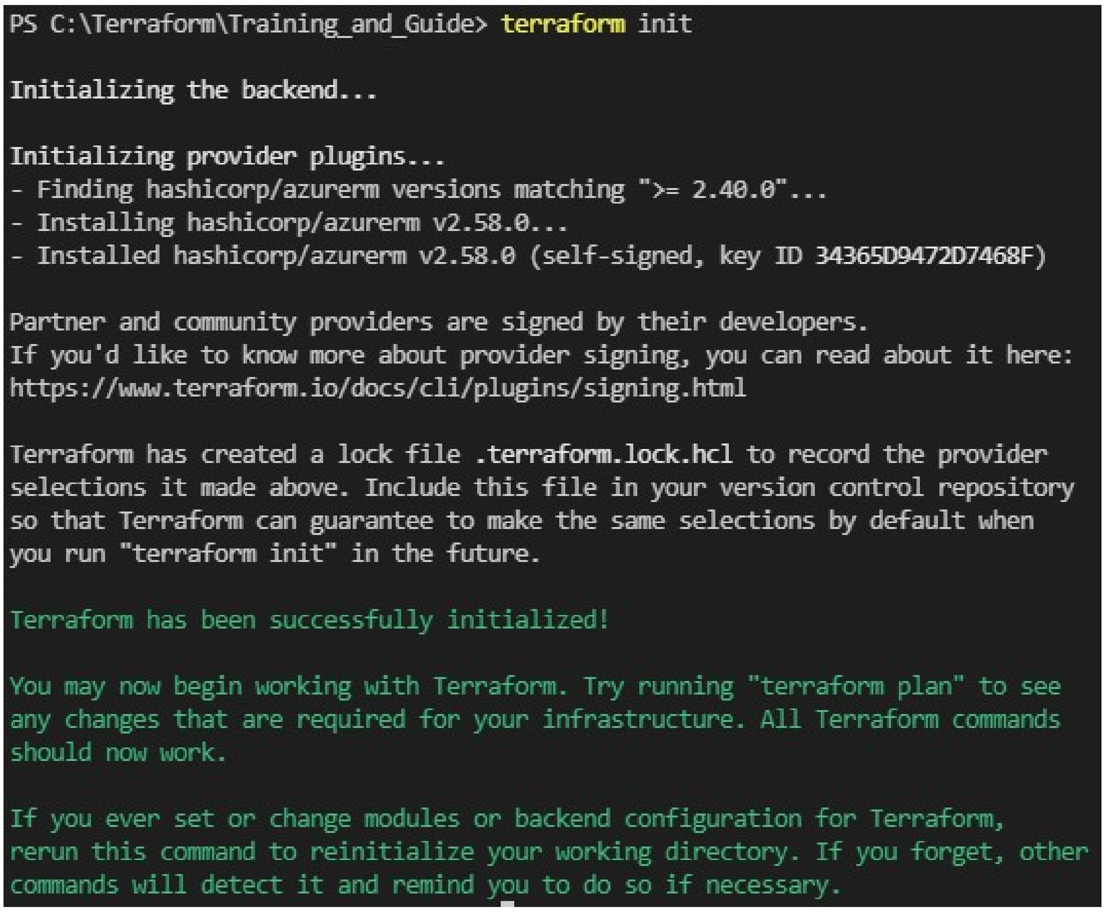
```
PS C:\Terraform\Training_and_Guide> az account show
{
  "environmentName": "AzureCloud",
```

- If your not tied to the right subscription, you can change this with ***az account set -s "***" ->*** **<u>put in the Subscription ID or Subscription Name</u>**



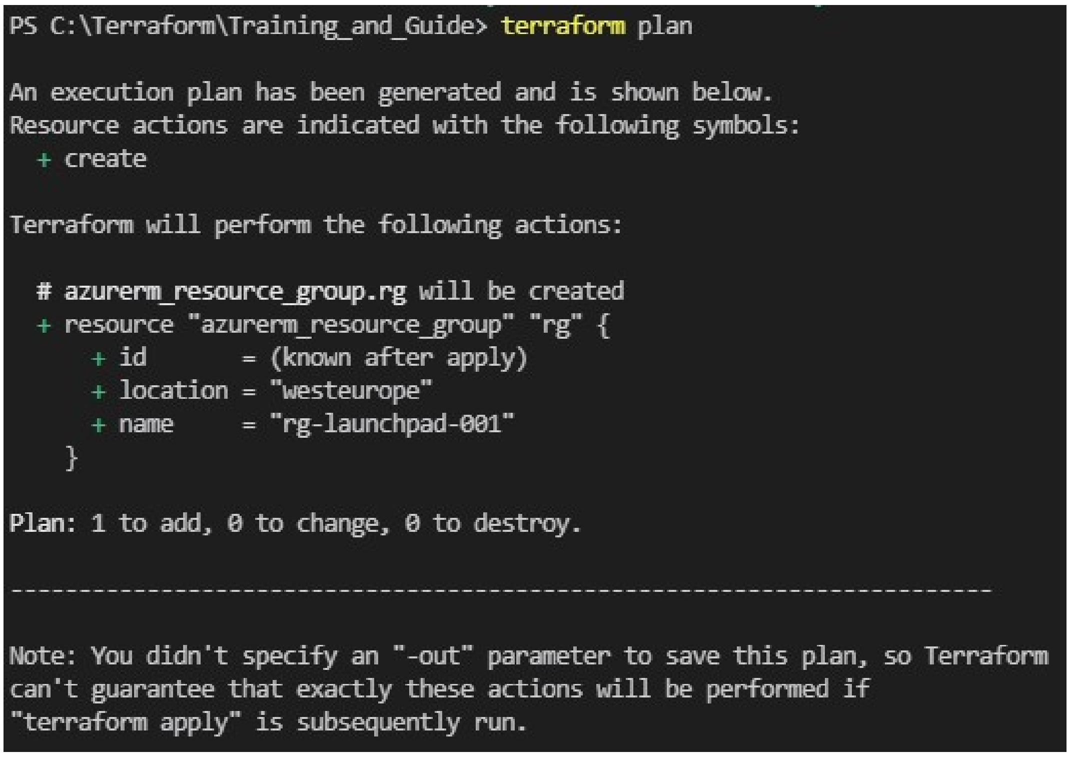
```
PS C:\Terraform\Training_and_Guide> az account set -s "e06b0f8d[redacted]"
PS C:\Terraform\Training_and_Guide> az account show
{
  "environmentName": "AzureCloud",
  "homeTenantId": [redacted],
  "id": "e06b0f8d[redacted]",
```

- Now proceed with the terraform deployment and start with ***terraform init***



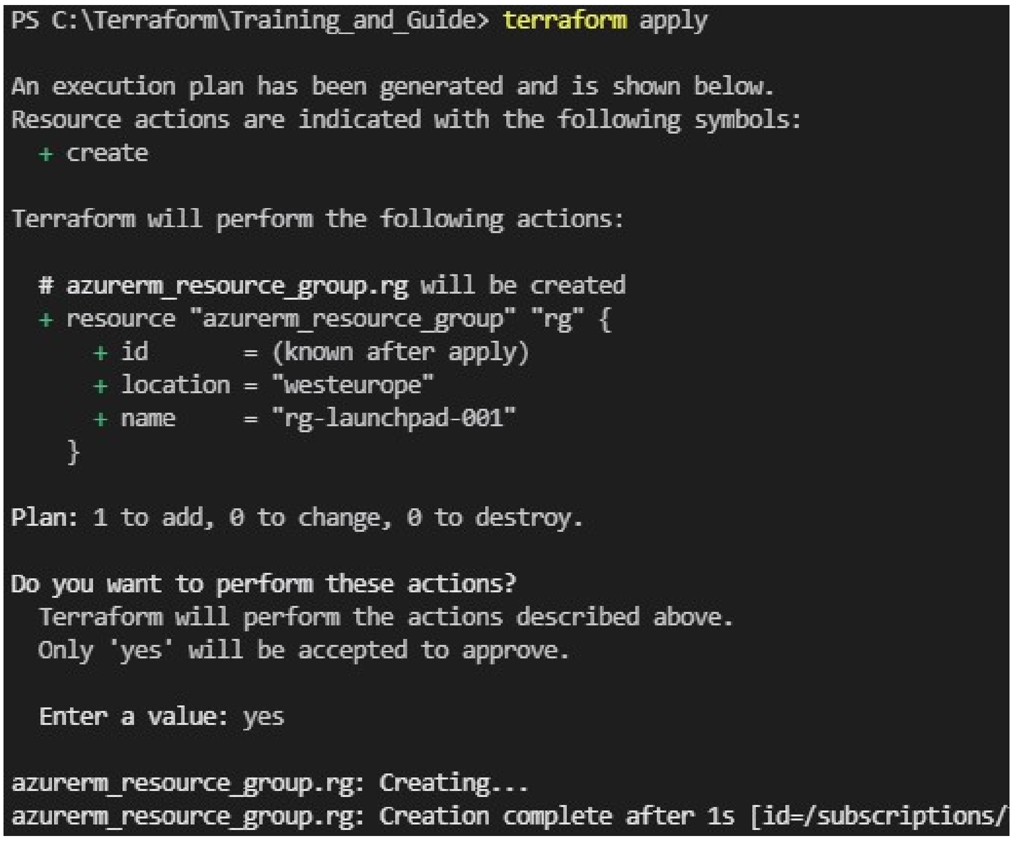
```
PS C:\Terraform\Training_and_Guide> terraform init

Initializing the backend...

Initializing provider plugins...
- Finding hashicorp/azurerm versions matching ">= 2.40.0"...
- Installing hashicorp/azurerm v2.58.0...
- Installed hashicorp/azurerm v2.58.0 (self-signed, key ID 34365D9472D7468F)

Partner and community providers are signed by their developers.
If you'd like to know more about provider signing, you can read about it here:
https://www.terraform.io/docs/cli/plugins/signing.html

Terraform has created a lock file .terraform.lock.hcl to record the provider
selections it made above. Include this file in your version control repository
so that Terraform can guarantee to make the same selections by default when
you run "terraform init" in the future.

Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.

If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
```

- Next will be a ***terraform plan*** to see what will happen when deploying the code



```
PS C:\Terraform\Training_and_Guide> terraform plan

An execution plan has been generated and is shown below.
Resource actions are indicated with the following symbols:
  + create

Terraform will perform the following actions:

  # azurerm_resource_group.rg will be created
  + resource "azurerm_resource_group" "rg" {
      + id       = (known after apply)
      + location = "westeurope"
      + name     = "rg-launchpad-001"
    }

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

------------------------------------------------------------------------

Note: You didn't specify an "-out" parameter to save this plan, so Terraform
can't guarantee that exactly these actions will be performed if
"terraform apply" is subsequently run.
```

- Now do a ***terraform apply*** to get the code deployed and acknowledge to deploy it with a ***yes***

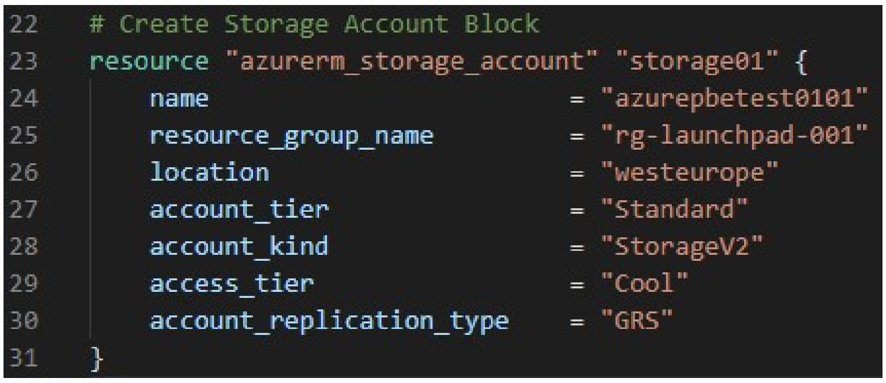


```
PS C:\Terraform\Training_and_Guide> terraform apply

An execution plan has been generated and is shown below.
Resource actions are indicated with the following symbols:
  + create

Terraform will perform the following actions:

  # azurerm_resource_group.rg will be created
  + resource "azurerm_resource_group" "rg" {
      + id       = (known after apply)
      + location = "westeurope"
      + name     = "rg-launchpad-001"
    }

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

Do you want to perform these actions?
  Terraform will perform the actions described above.
  Only 'yes' will be accepted to approve.

  Enter a value: yes

azurerm_resource_group.rg: Creating...
azurerm_resource_group.rg: Creation complete after 1s [id=/subscriptions/
```

- Now the Resources are created and cen be also found in the Azure Portal

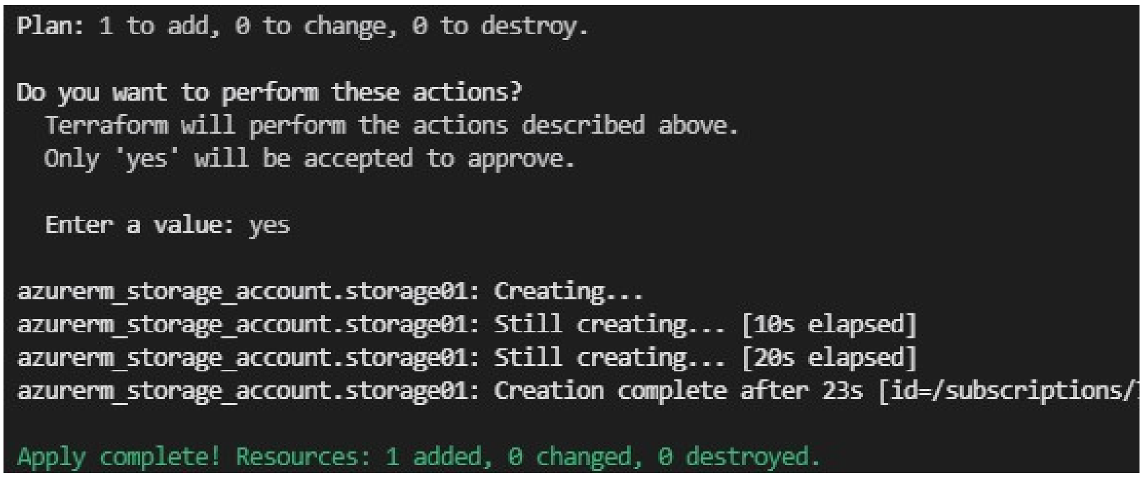
Task 2

- Now to add the Storage Account Configuration, return to the Script within VS Code
- Add another Block for the Storage Account



```
# Create Storage Account Block
resource "azurerm_storage_account" "storage01" {
    name                        = "azurepbetest0101"
    resource_group_name         = "rg-launchpad-001"
    location                    = "westeurope"
    account_tier                = "Standard"
    account_kind                = "StorageV2"
    access_tier                 = "Cool"
    account_replication_type    = "GRS"
}
```

- and again perform a ***terraform plan*** and after that a ***terraform apply***

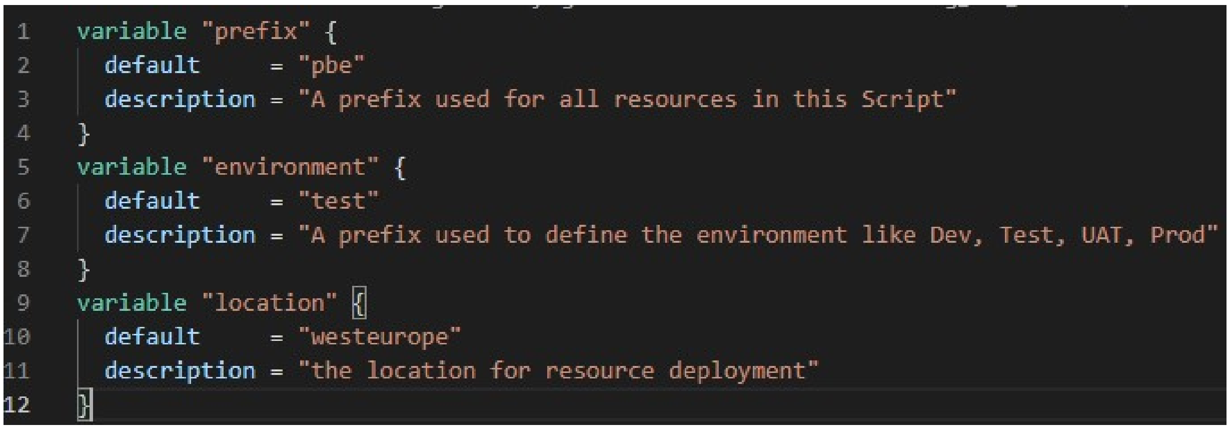


```
Plan: 1 to add, 0 to change, 0 to destroy.

Do you want to perform these actions?
  Terraform will perform the actions described above.
  Only 'yes' will be accepted to approve.

  Enter a value: yes

azurerm_storage_account.storage01: Creating...
azurerm_storage_account.storage01: Still creating... [10s elapsed]
azurerm_storage_account.storage01: Still creating... [20s elapsed]
azurerm_storage_account.storage01: Creation complete after 23s [id=/subscriptions/

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

- those changes can now again also be found in the Azure Portal

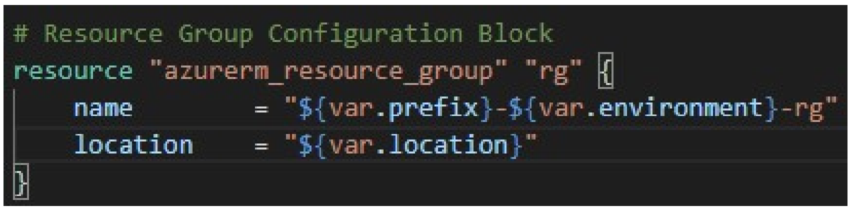
Task 3

- in the Folder where the main.tf file is located, create a ***variables.tf*** file and open this one with Visual Studio Code
- Add the following variables configurations to this File (prefix will be your personal prefix)



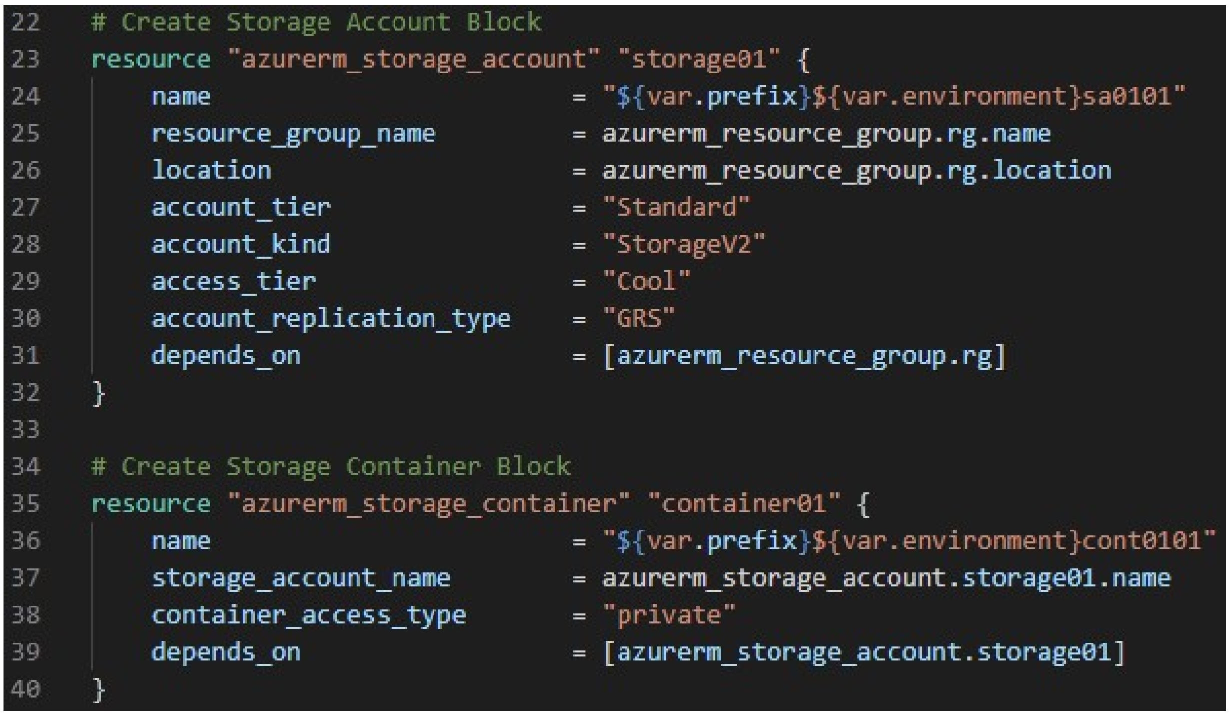
```
variable "prefix" {
  default     = "pbe"
  description = "A prefix used for all resources in this Script"
}
variable "environment" {
  default     = "test"
  description = "A prefix used to define the environment like Dev, Test, UAT, Prod"
}
variable "location" {
  default     = "westeurope"
  description = "the location for resource deployment"
}
```

- Then change to the main.tf file and adjust the resource group section
- reference to the variables file



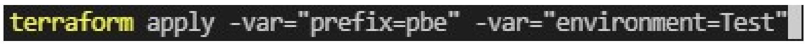
```
# Resource Group Configuration Block
resource "azurerm_resource_group" "rg" {
    name         = "${var.prefix}-${var.environment}-rg"
    location     = "${var.location}"
}
```

- you should now also add the storage container block, take care about the globally unique name and also about the dependency between the resources



```
# Create Storage Account Block
resource "azurerm_storage_account" "storage01" {
    name                         = "${var.prefix}${var.environment}sa0101"
    resource_group_name          = azurerm_resource_group.rg.name
    location                     = azurerm_resource_group.rg.location
    account_tier                 = "Standard"
    account_kind                 = "StorageV2"
    access_tier                  = "Cool"
    account_replication_type     = "GRS"
    depends_on                   = [azurerm_resource_group.rg]
}

# Create Storage Container Block
resource "azurerm_storage_container" "container01" {
    name                         = "${var.prefix}${var.environment}cont0101"
    storage_account_name         = azurerm_storage_account.storage01.name
    container_access_type        = "private"
    depends_on                   = [azurerm_storage_account.storage01]
}
```

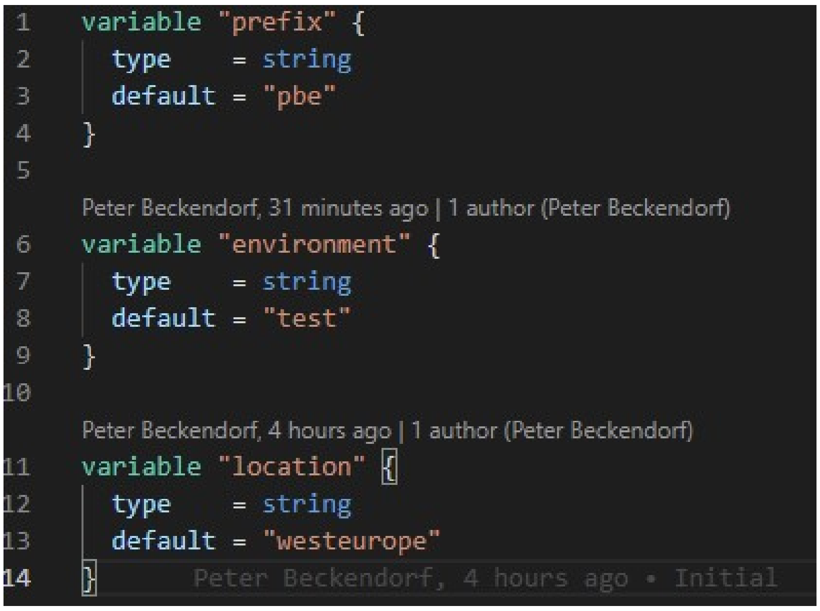
- you can also pass variables while doing the ***terraform apply*** command, which would look like



```
terraform apply -var="prefix=pbe" -var="environment=Test"
```

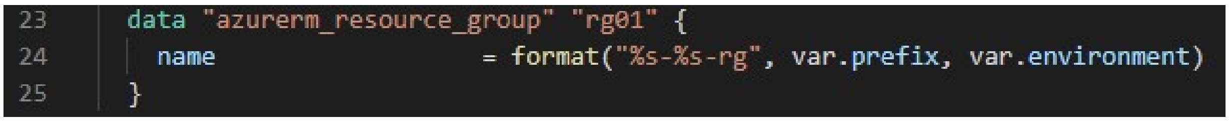
Task 4

- first check the variables defined in the ***variables.tf***



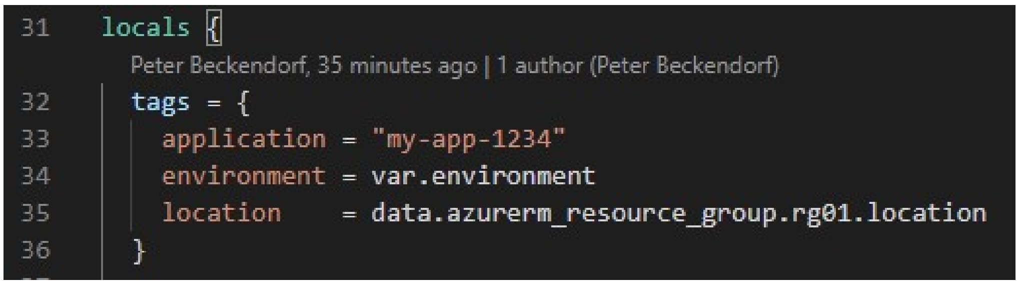
```
variable "prefix" {
  type    = string
  default = "pbe"
}

Peter Beckendorf, 31 minutes ago | 1 author (Peter Beckendorf)
variable "environment" {
  type    = string
  default = "test"
}

Peter Beckendorf, 4 hours ago | 1 author (Peter Beckendorf)
variable "location" {
  type    = string
  default = "westeurope"
}       Peter Beckendorf, 4 hours ago • Initial
```

- next move to ***main.tf*** file and add a data Block to reference to the in Task 3 generated Resource Group (https://www.terraform.io/docs/language/functions/format.html)



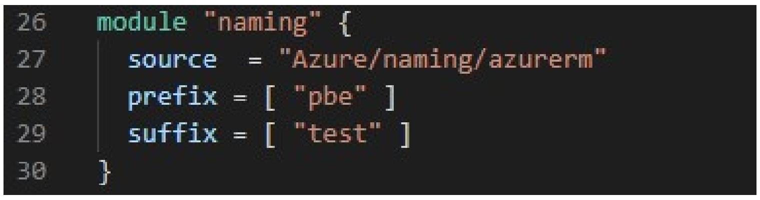
```
data "azurerm_resource_group" "rg01" {
  name                    = format("%s-%s-rg", var.prefix, var.environment)
}
```

- now you can use data source from the previously created resource group like the location (https://registry.terraform.io/providers/hashicorp/azurerm/latest/docs/data-sources/resource_group)



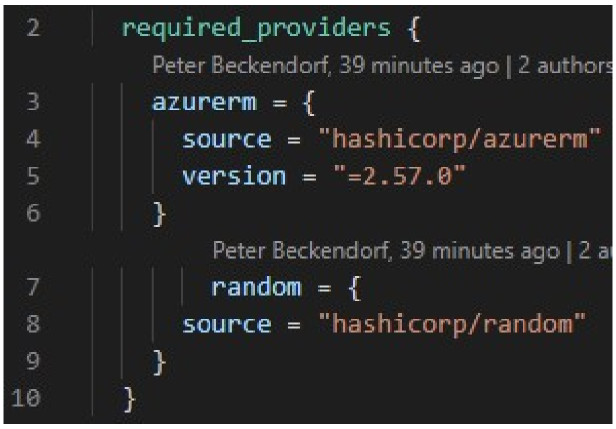
```
locals {
  Peter Beckendorf, 35 minutes ago | 1 author (Peter Beckendorf)
  tags = {
    application = "my-app-1234"
    environment = var.environment
    location    = data.azurerm_resource_group.rg01.location
  }
```

- next you should add the naming module (https://registry.terraform.io/modules/Azure/naming/azurerm/latest)



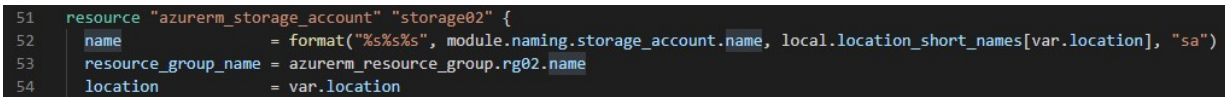
```
module "naming" {
  source  = "Azure/naming/azurerm"
  prefix = [ "pbe" ]
  suffix = [ "test" ]
}
```

- And add the random provider as this would otherwise be reported as a failure



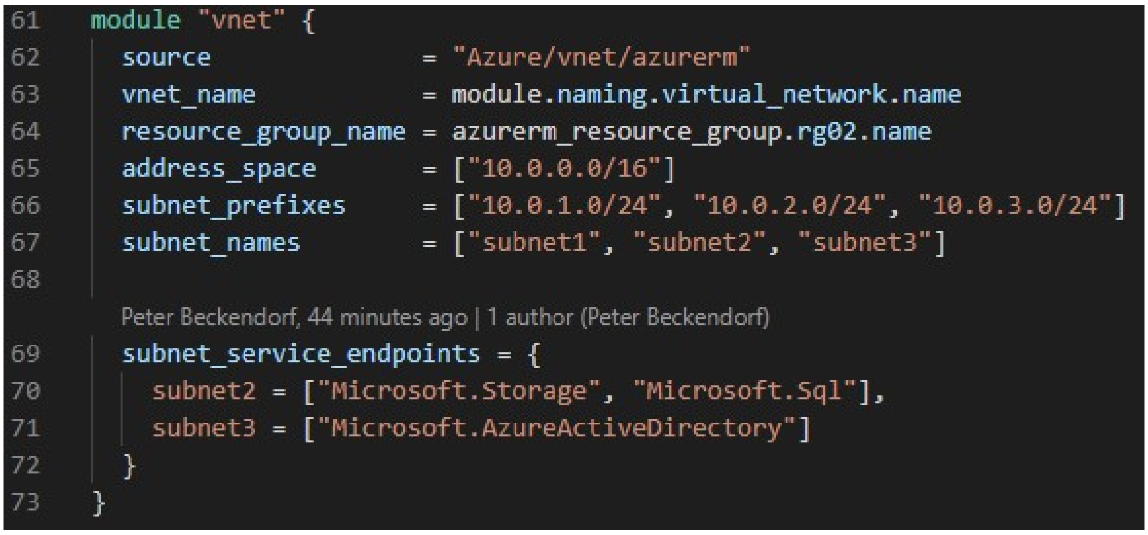
```
  required_providers {
    Peter Beckendorf, 39 minutes ago | 2 authors
    azurerm = {
      source = "hashicorp/azurerm"
      version = "=2.57.0"
    }
        Peter Beckendorf, 39 minutes ago | 2 a
        random = {
      source = "hashicorp/random"
    }
  }
```

- After this you can use this naming module for the name variable creation



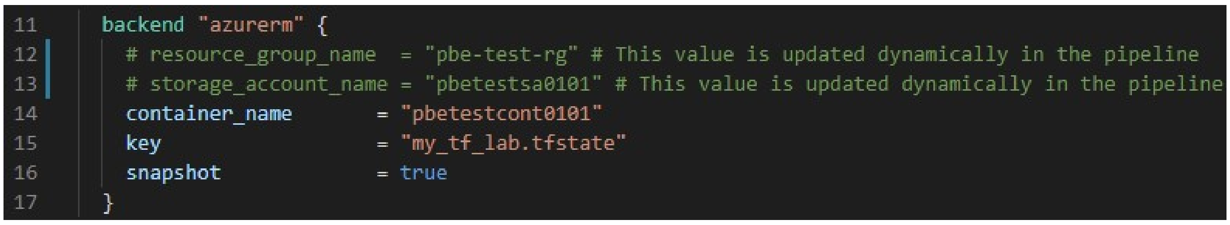
```
resource "azurerm_storage_account" "storage02" {
  name                = format("%s%s%s", module.naming.storage_account.name, local.location_short_names[var.location], "sa")
  resource_group_name = azurerm_resource_group.rg02.name
  location            = var.location
```

- As next step the VNet Module shall be integrated, you can again use the naming module to create the name of the VNet



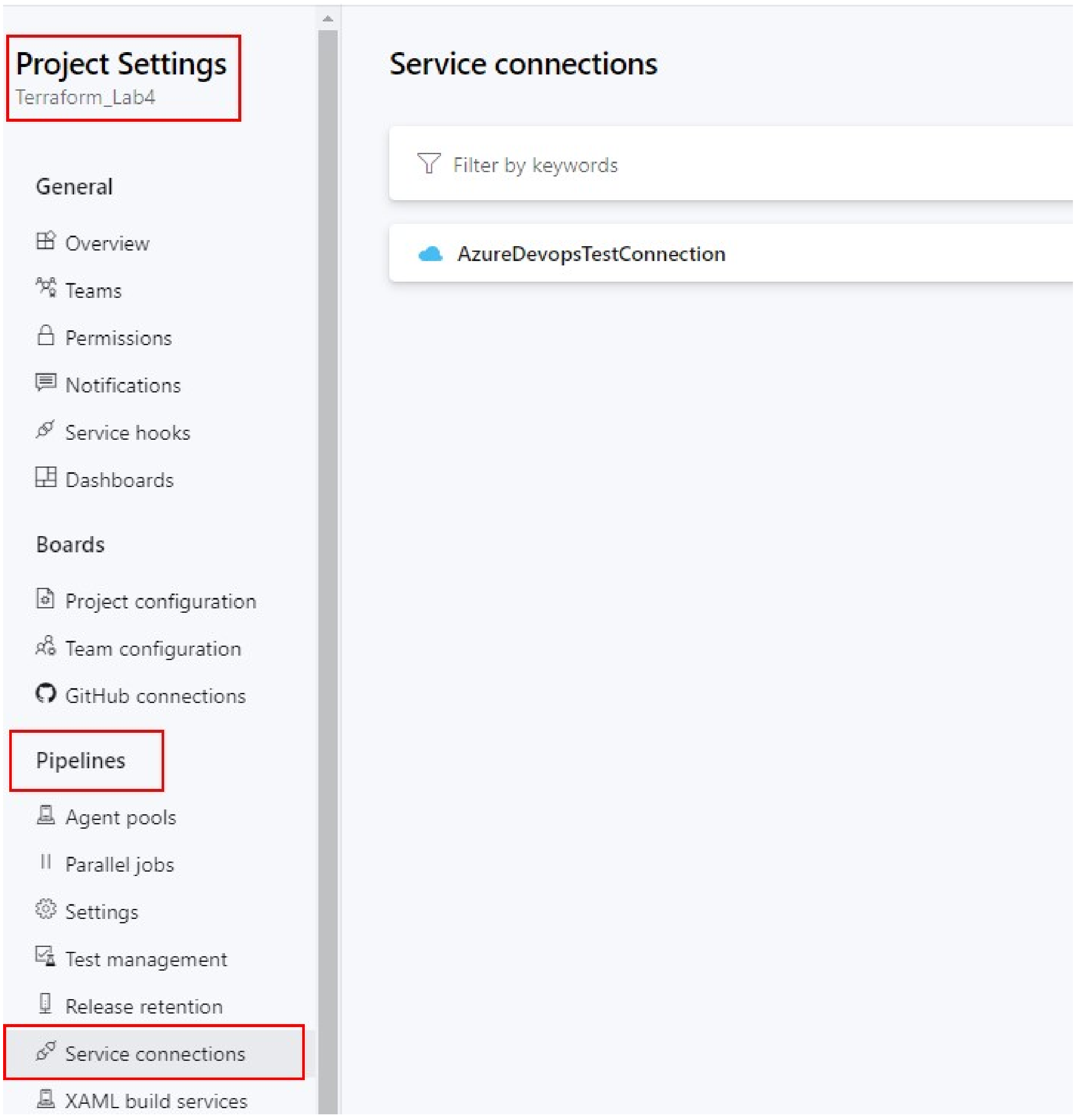
```
module "vnet" {
  source              = "Azure/vnet/azurerm"
  vnet_name           = module.naming.virtual_network.name
  resource_group_name = azurerm_resource_group.rg02.name
  address_space       = ["10.0.0.0/16"]
  subnet_prefixes     = ["10.0.1.0/24", "10.0.2.0/24", "10.0.3.0/24"]
  subnet_names        = ["subnet1", "subnet2", "subnet3"]

  Peter Beckendorf, 44 minutes ago | 1 author (Peter Beckendorf)
  subnet_service_endpoints = {
    subnet2 = ["Microsoft.Storage", "Microsoft.Sql"],
    subnet3 = ["Microsoft.AzureActiveDirectory"]
  }
}
```

- When this is done, go ahead and adjust your project as described in the ***README.md***
- Task 1 should be allready done in the Steps above
- Task 2 means to adjust the container_name variable in the backend block, the name can be looked up in the azure portal

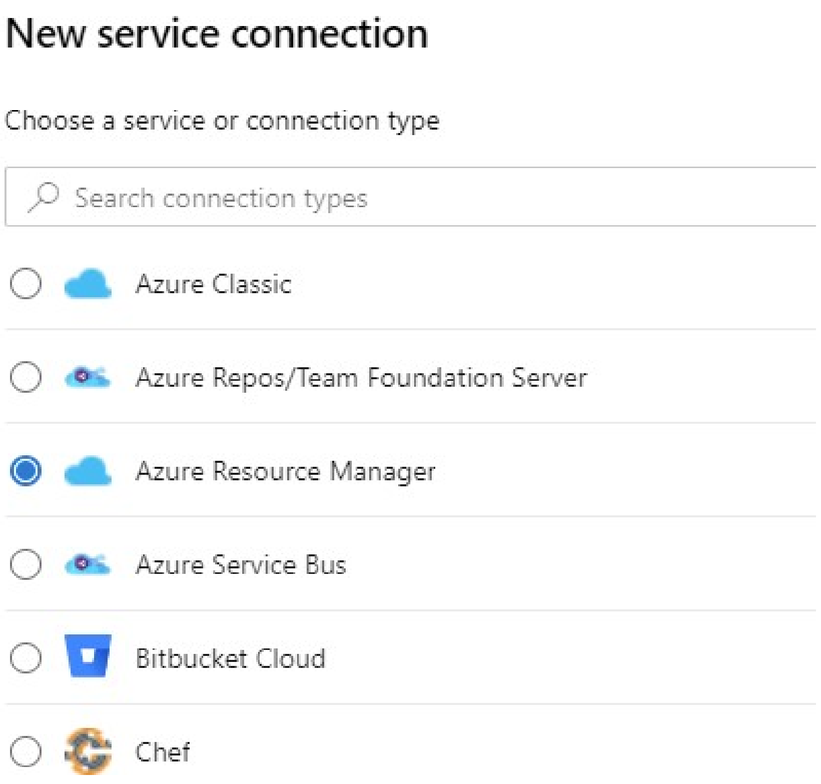


```
backend "azurerm" {
  # resource_group_name  = "pbe-test-rg" # This value is updated dynamically in the pipeline
  # storage_account_name = "pbetestsa0101" # This value is updated dynamically in the pipeline
  container_name       = "pbetestcont0101"
  key                  = "my_tf_lab.tfstate"
  snapshot             = true
}
```

- For Task 3 you need to create a Service Connection in the Azure DevOps if not existing
- Navigate to Project settings and then under the Pipelines section you will find Service Connections



- If no Connection is available, create a new Service Connection (upper Right Side) and choose Azure Resource Manager as the connection type



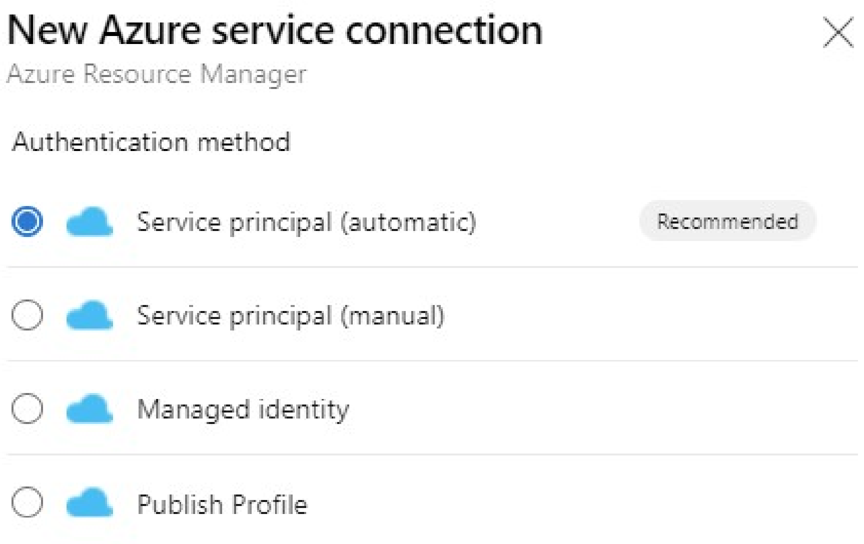
New service connection

Choose a service or connection type

Search connection types

- ( ) Azure Classic
- ( ) Azure Repos/Team Foundation Server
- (●) Azure Resource Manager
- ( ) Azure Service Bus
- ( ) Bitbucket Cloud
- ( ) Chef

- Select Service principal (automatic) as Authentication method



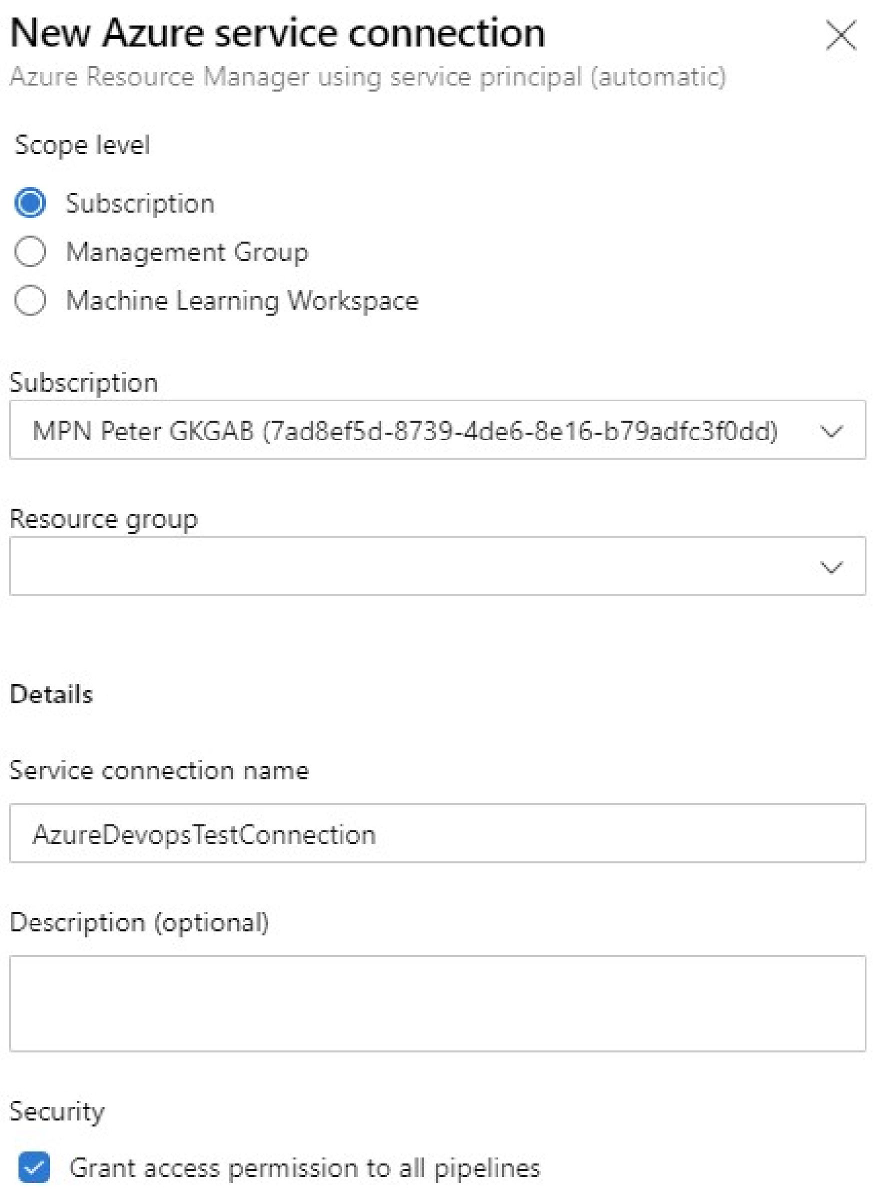
New Azure service connection

Azure Resource Manager

Authentication method

- (●) Service principal (automatic) Recommended
- ( ) Service principal (manual)
- ( ) Managed identity
- ( ) Publish Profile

- Next you choose the right Subscription, leave Resource Group empty and provide a Service Connection Name that will then be inserted to the ***deploy.yml*** file for the pipeline


New Azure service connection

Azure Resource Manager using service principal (automatic)

Scope level

[✓ Subscription] Management Group Machine Learning Workspace

Subscription: MPN Peter GKGAB (7ad8ef5d-8739-4de6-8e16-b79adfc3f0dd)

Resource group:

Details

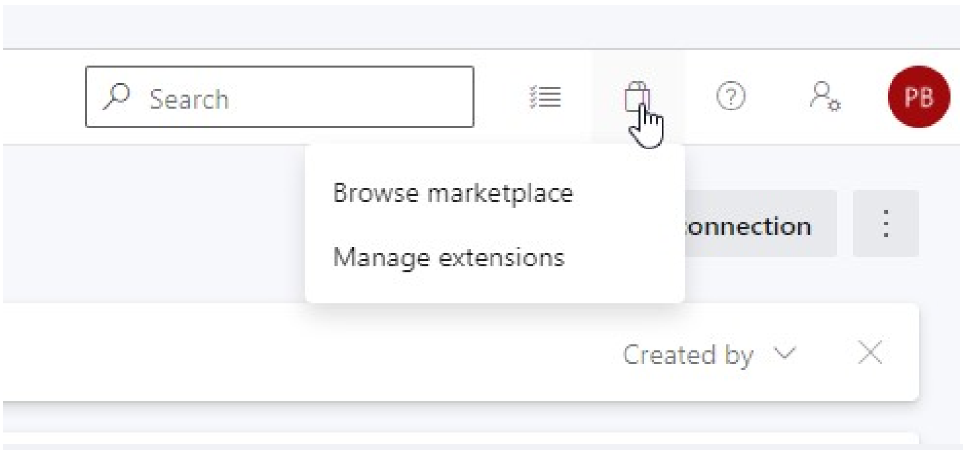
Service connection name: AzureDevopsTestConnection

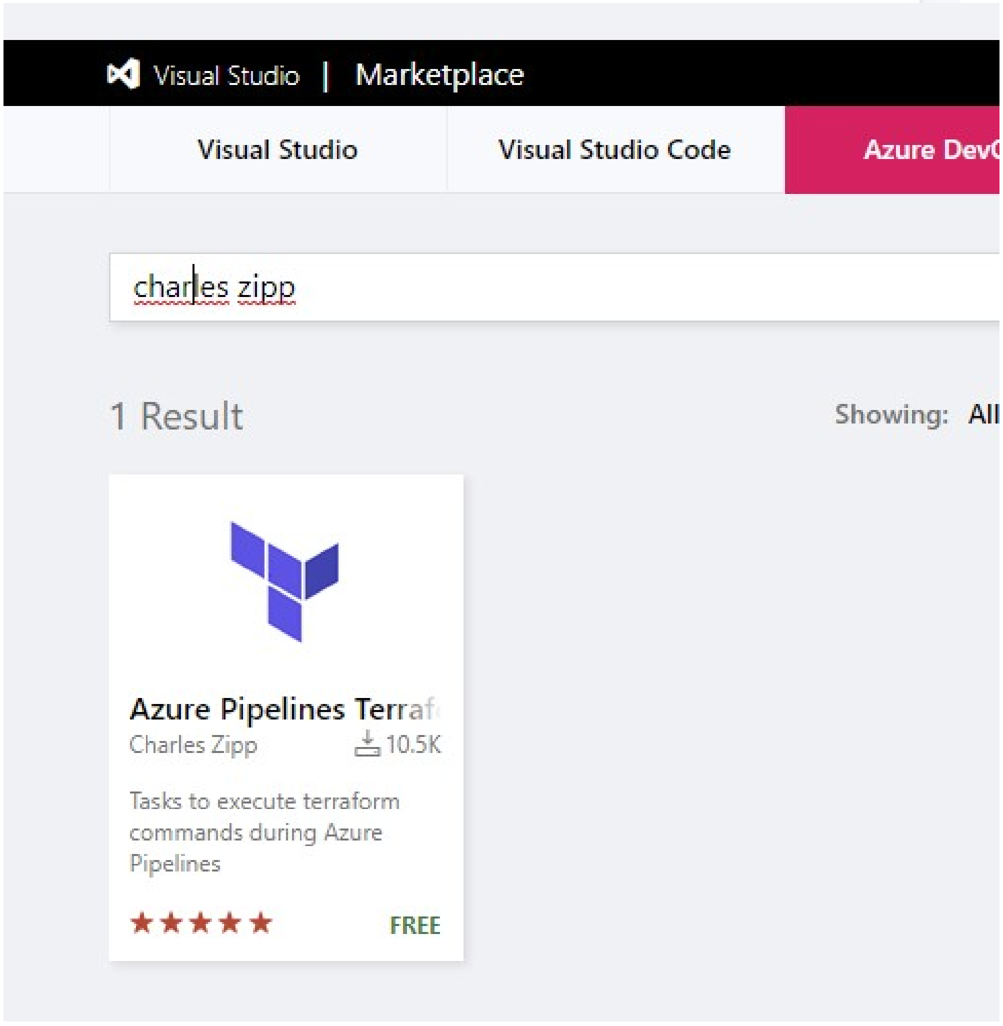
Description (optional):

Security

✓ Grant access permission to all pipelines

- When this is done, it might be required to install the pipeline extension from the marketplace





- When this is done, insert the variables to your ***deploy.yml*** file as also described in the ***Readme.md*** File

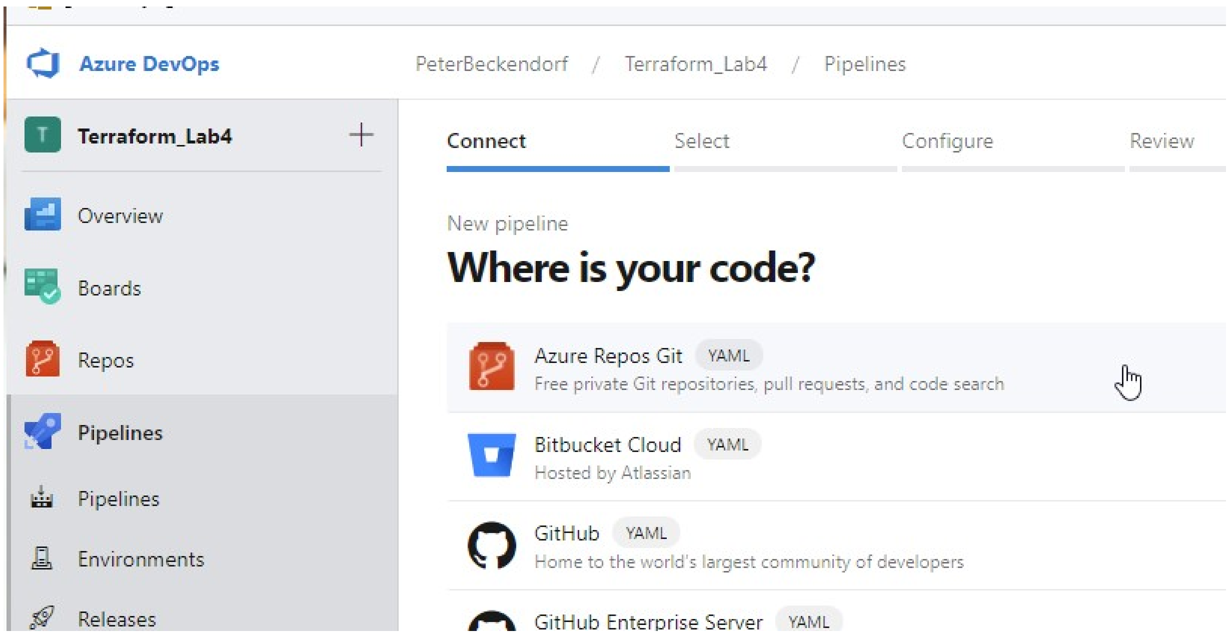
  - (service connection name) Should be replaced with the service connection name as instructed

  - (lab subscription id) Should be replaced with the subscription id as instructed

- (your resource group name) Should be replaced with the resource group name you created in the previous lab

- (your storage account name) Should be replaced with the Storage Account name you created in the previous lab

- When this all is done and fine, and the latest version of the ***deploy.yml*** is pushed to the Repository, you can create the Pipeline -> Azure Repos Git



New pipeline

Where is your code?

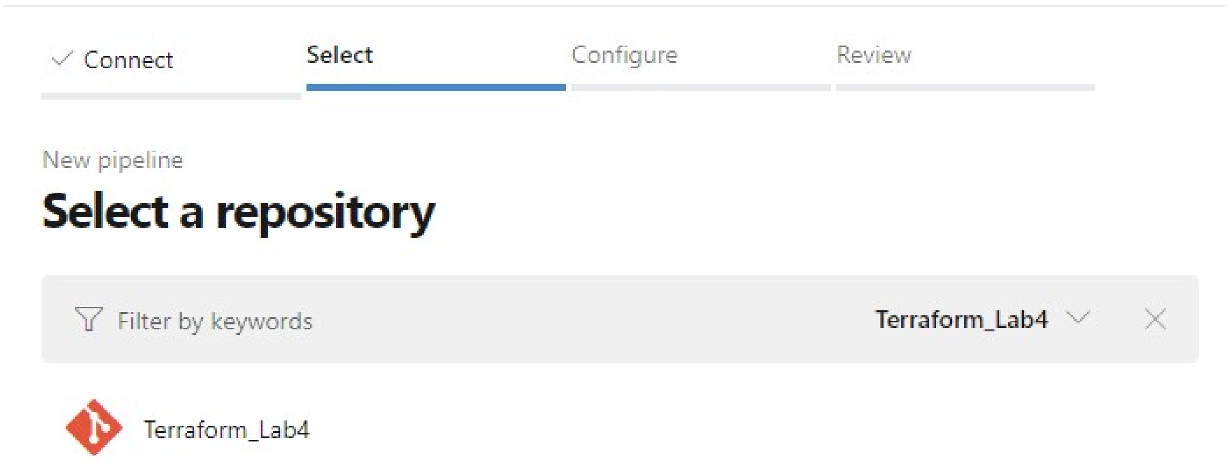
Azure Repos Git YAML
Free private Git repositories, pull requests, and code search

Bitbucket Cloud YAML
Hosted by Atlassian

GitHub YAML
Home to the world's largest community of developers

GitHub Enterprise Server YAML

- Choose your Repo



New pipeline

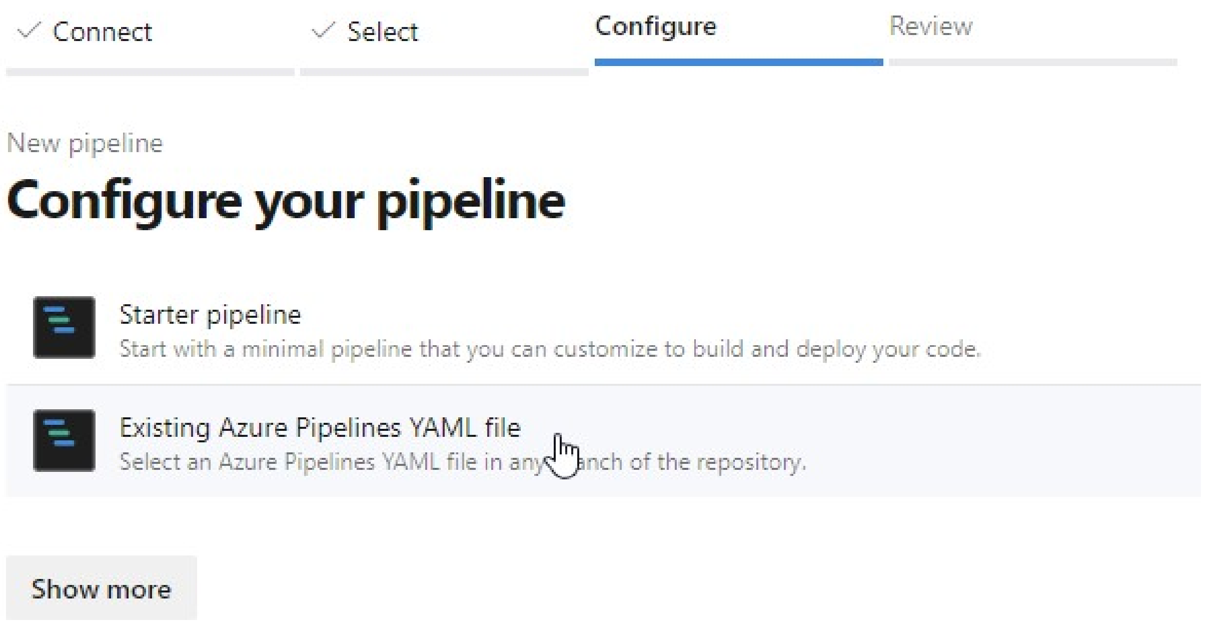
Select a repository

Filter by keywords

Terraform_Lab4

Terraform_Lab4

- Then choose “Existing Azure Pipelines YAML file”

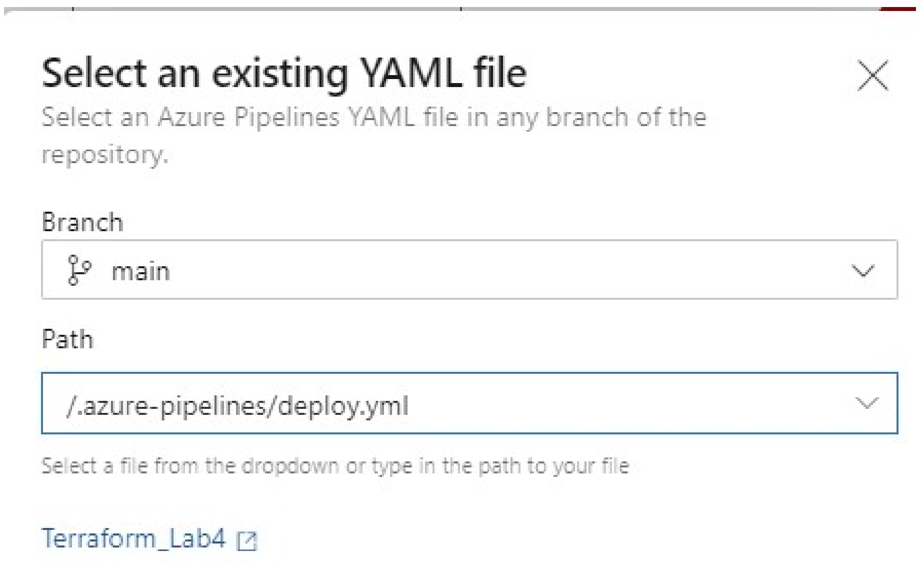


New pipeline

Configure your pipeline

Starter pipeline
Start with a minimal pipeline that you can customize to build and deploy your code.

Existing Azure Pipelines YAML file
Select an Azure Pipelines YAML file in any branch of the repository.


Select an existing YAML file
Select an Azure Pipelines YAML file in any branch of the repository.

Branch: main

Path: /.azure-pipelines/deploy.yml

Select a file from the dropdown or type in the path to your file

- Now you can review it once again if all is fine and if so, continue to run this pipeline (press Run on the upper Right Corner above the Code Window
- If all was configured right, you should have your code deployed without any issues
- If any Problems come up, review and fix them in your terraform configuration Files