



Terraform Exercises

Task 1: Install terraform and Azure CLI.

To install Terraform, you can follow this:

- Visit the Terraform website at <https://www.terraform.io/downloads.html> and download the appropriate package for your operating system. Terraform is available for Windows, Mac, and Linux.
- Once the download is complete, extract the package to a directory of your choice.
- Add the directory to your system's PATH environment variable. This will allow you to run the terraform command from any directory.
- To test your installation, open a command prompt or terminal window and type terraform version. This should display the installed version of Terraform.

To install Azure CLI on your system, you can follow this:

- Go to the official Azure CLI installation page at <https://docs.microsoft.com/en-us/cli/azure/install-azure-cli> and choose your operating system.
- Follow the instructions provided for your operating system to install Azure CLI.
- Once installed, open a command prompt or terminal window and type: “az login” to authenticate with your Azure account. This will open a browser window and prompt you to log in to your Azure account.
- After successfully logging in, you can start using Azure CLI to manage your Azure resources.



To connect my azure account to terraform I did this:

- First, you need to create an Azure service principal. This is a type of identity that Terraform can use to authenticate with Azure. To create a service principal, you can follow the instructions in the official Azure documentation.
- Once you have created a service principal, you will need to obtain the following information:
 - Subscription ID: This is the unique identifier for your Azure subscription. You can find this in the Azure portal under the "Overview" section of your subscription.
 - Tenant ID: This is the identifier for your Azure Active Directory tenant. You can find this in the Azure portal under the "Properties" section of your Azure Active Directory.
 - Client ID: This is the identifier for your service principal. You can find this in the Azure portal under the "App registrations" section of your Azure Active Directory.
 - Client secret: This is the password for your service principal. You should have created this when you created the service principal.
- Next, you need to create a Terraform configuration file that includes the Azure provider and the information you obtained in the previous step.
- Here is my configuration file:

```
provider "azurerm" {  
  subscription_id = "subscription_id"  
  tenant_id      = "tenant_id"  
  client_id      = "client_id"  
  client_secret  = "client_secret"  
}
```
- Replace subscription_id, tenant_id, client_id, and client_secret with the corresponding values you obtained in the previous step.
- Save the configuration file with a .tf extension, such as main.tf.



- Open a command prompt or terminal window and navigate to the directory where you saved the configuration file.
- Run the “terraform init” command to initialize the Terraform working directory and download the Azure provider.

```
C:\Users\stani\Desktop\DevOpsAcademy\ScaleFocus-Academy-Homework\20.1.Terraform and Azure Active Directory>terraform init

Initializing the backend...

Initializing provider plugins...
- Reusing previous version of hashicorp/azurerm from the dependency lock file
- Using previously-installed hashicorp/azurerm v3.51.0

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.
```

- Once the initialization is complete, you can run other terraform commands, such as terraform plan or terraform apply.

```
C:\Users\stani\Desktop\DevOpsAcademy\ScaleFocus-Academy-Homework\20.1.Terraform and Azure Active Directory>terraform plan

No changes. Your infrastructure matches the configuration.

Terraform has compared your real infrastructure against your configuration and found no differences, so no changes are needed.

C:\Users\stani\Desktop\DevOpsAcademy\ScaleFocus-Academy-Homework\20.1.Terraform and Azure Active Directory>terraform apply

No changes. Your infrastructure matches the configuration.

Terraform has compared your real infrastructure against your configuration and found no differences, so no changes are needed.

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

C:\Users\stani\Desktop\DevOpsAcademy\ScaleFocus-Academy-Homework\20.1.Terraform and Azure Active Directory>
```



```
C:\Users\stani\Desktop\DevOpsAcademy\ScaleFocus-Academy-Homework>cd "20.1.Terraform and Azure Active Directory"
C:\Users\stani\Desktop\DevOpsAcademy\ScaleFocus-Academy-Homework\20.1.Terraform and Azure Active Directory>terraform init

Initializing the backend...

Initializing provider plugins...
- Reusing previous version of hashicorp/azurerm from the dependency lock file
- Using previously-installed hashicorp/azurerm v3.51.0

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.

C:\Users\stani\Desktop\DevOpsAcademy\ScaleFocus-Academy-Homework\20.1.Terraform and Azure Active Directory>terraform plan

No changes. Your infrastructure matches the configuration.

Terraform has compared your real infrastructure against your configuration and found no differences, so no changes are needed.

C:\Users\stani\Desktop\DevOpsAcademy\ScaleFocus-Academy-Homework\20.1.Terraform and Azure Active Directory>terraform apply

No changes. Your infrastructure matches the configuration.

Terraform has compared your real infrastructure against your configuration and found no differences, so no changes are needed.

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

C:\Users\stani\Desktop\DevOpsAcademy\ScaleFocus-Academy-Homework\20.1.Terraform and Azure Active Directory>
```

- The output from the commands shows that there are no changes, because I already connected my Azure account with Terraform, and I didn't change my configuration file(*.tf)
- Here I am providing a screenshot for the version of terraform and azurerm:

```
C:\Users\stani\Desktop\DevOpsAcademy\ScaleFocus-Academy-Homework\20.1.Terraform and Azure Active Directory>terraform version
Terraform v1.4.4
on windows_386
+ provider registry.terraform.io/hashicorp/azurerm v3.51.0

C:\Users\stani\Desktop\DevOpsAcademy\ScaleFocus-Academy-Homework\20.1.Terraform and Azure Active Directory>
```

- Here is Azure CLI output of the current subscription:

```
C:\Program Files\Microsoft SDKs\Azure\NET SDK\v2.9>az account show
{
  "environmentName": "AzureCloud",
  "homeTenantId": "ee42abc4-d671-4367-83c8-a65253dcec36",
  "id": "c983dec5-cde0-4991-9469-c26f8cf60056",
  "isDefault": true,
  "managedByTenants": [],
  "name": "Azure Pass - Sponsorship",
  "state": "Enabled",
  "tenantId": "ee42abc4-d671-4367-83c8-a65253dcec36",
  "user": {
    "name": "StanislavNikolovNew1@outlook.com",
    "type": "user"
  }
}

C:\Program Files\Microsoft SDKs\Azure\NET SDK\v2.9>
```



Task 2: Define your first terraform infrastructure code.

```
C:\Users\stani\Desktop\DevOpsAcademy\ScaleFocus-Academy-Homework\20.1.Terraform and Azure Active Directory>cd Task_2

C:\Users\stani\Desktop\DevOpsAcademy\ScaleFocus-Academy-Homework\20.1.Terraform and Azure Active Directory\Task_2>terraform init

Initializing the backend...

Initializing provider plugins...
- Finding latest version of hashicorp/azurerm...
- Installing hashicorp/azurerm v3.51.0...
- Installed hashicorp/azurerm v3.51.0 (signed by HashiCorp)

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.
```

- Initializing terraform.

```
C:\Users\stani\Desktop\DevOpsAcademy\ScaleFocus-Academy-Homework\20.1.Terraform and Azure Active Directory\Task_2>terraform plan

No changes. Your infrastructure matches the configuration.
```

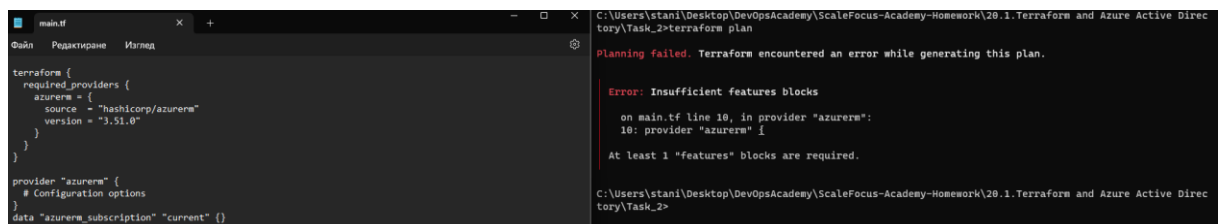
-

```
C:\Users\stani\Desktop\DevOpsAcademy\ScaleFocus-Academy-Homework\20.1.Terraform and Azure Active Directory\Task_2>terraform plan
data.azurerm_subscription.current: Reading...
data.azurerm_subscription.current: Read complete after 0s [id=/subscriptions/c983dec5-cde0-4991-9469-c26f8cf60056]

No changes. Your infrastructure matches the configuration.

Terraform has compared your real infrastructure against your configuration and found no differences, so no changes are needed.
C:\Users\stani\Desktop\DevOpsAcademy\ScaleFocus-Academy-Homework\20.1.Terraform and Azure Active Directory\Task_2>
```

- Terraform plan, which is successful.



The screenshot shows a code editor with a file named `main.tf` containing the following Terraform configuration:

```
terraform {
  required_providers {
    azurerm = {
      source  = "hashicorp/azurerm"
      version = "3.51.0"
    }
  }
}

provider "azurerm" {
  # Configuration options
}

data "azurerm_subscription" "current" {}
```

The terminal window shows the command `C:\Users\stani\Desktop\DevOpsAcademy\ScaleFocus-Academy-Homework\20.1.Terraform and Azure Active Directory\Task_2>terraform plan` and the following error message:

```
Planning failed. Terraform encountered an error while generating this plan.

Error: Insufficient features blocks
on main.tf line 10, in provider "azurerm":
10: provider "azurerm" {
   At least 1 "features" blocks are required.
```

- This time it will throw you an error, because azurerm needs a required arguments in order to run properly.



```
main.tf
terraform {
  required_providers {
    azure = {
      source = "hashicorp/azure"
      version = "~>3.0.0"
    }
  }
}

# Configure the Microsoft Azure Provider
provider "azure" {
  features {}
}

data "azurerm_subscription" "current" {}
```

```
C:\Users\stani\Desktop\DevOpsAcademy\ScaleFocus-Academy-Homework\20.1.Terraform and Azure Active Directory\Task_2>terraform plan

Error: Inconsistent dependency lock file

The following dependency selections recorded in the lock file are inconsistent with the current configuration:
- provider registry.hashicorp.azure: locked version selection 3.0.0 doesn't match the updated version constraints "~>3.0.0"

To update the locked dependency selections to match a changed configuration, run:
  terraform init -upgrade

C:\Users\stani\Desktop\DevOpsAcademy\ScaleFocus-Academy-Homework\20.1.Terraform and Azure Active Directory\Task_2>terraform init -upgrade

Initializing the backend...

Initializing provider plugins...
- Finding hashicorp/azure versions matching "~>3.0.0"...
- Installing hashicorp/azure v3.0.0...
- Installed hashicorp/azure v3.0.0 (signed by HashiCorp)

Terraform has made some changes to the provider dependency selections recorded in the .terraform.lock.hcl file. Review these changes and commit them to your version control system if they represent changes you intended to make.

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.

C:\Users\stani\Desktop\DevOpsAcademy\ScaleFocus-Academy-Homework\20.1.Terraform and Azure Active Directory\Task_2>terraform plan

data.azurerm_subscription.current: Reading...
data.azurerm_subscription.current: Read complete after 8s [id=/subscriptions/c9b3d3c3-cde8-4091-9b09-c26f8cf08066]

No changes. Your infrastructure matches the configuration.

Terraform has compared your real infrastructure against your configuration and found no differences, so no changes are needed.
```

- When we provide the needed arguments, the result will show the subscription_id without any errors.

```
main.tf
terraform {
  required_providers {
    azure = {
      source = "hashicorp/azurerm"
      version = "~>3.35.0"
    }
  }
}

# Configure the Microsoft Azure Provider
provider "azurerm" {
  features {}
}

data "azurerm_subscription" "current" {}

resource "random_string" "random" {
  length = 8
  special = false
  lower = true
  upper = false
}

resource "azurerm_resource_group" "example" {
  name = "${random_string.random.result}-rg"
  location = "West Europe"
}

resource "azurerm_storage_account" "example" {
  name = "${random_string.random.result}sa"
  resource_group_name = azurerm_resource_group.example.name
  location = azurerm_resource_group.example.location
  account_tier = "Standard"
  account_replication_type = "GRS"

  blob_properties {
    restore_policy {
      days = 7
    }
  }

  tags = {
    environment = "staging"
  }
}
```

```
To update the locked dependency selections to match a changed configuration, run:
  terraform init -upgrade

C:\Users\stani\Desktop\DevOpsAcademy\ScaleFocus-Academy-Homework\20.1.Terraform and Azure Active Directory\Task_2>terraform init -upgrade

Initializing the backend...

Initializing provider plugins...
- Finding hashicorp/azurerm versions matching "~>3.35.0"...
- Finding latest version of hashicorp/random...
- Installing hashicorp/azurerm v3.35.0...
- Installed hashicorp/azurerm v3.35.0 (signed by HashiCorp)
- Installing hashicorp/random v3.4.3...
- Installed hashicorp/random v3.4.3 (signed by HashiCorp)

Terraform has made some changes to the provider dependency selections recorded in the .terraform.lock.hcl file. Review those changes and commit them to your version control system if they represent changes you intended to make.

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.

C:\Users\stani\Desktop\DevOpsAcademy\ScaleFocus-Academy-Homework\20.1.Terraform and Azure Active Directory\Task_2>terraform plan

Error: Unsupported block type

on main.tf line 35, in resource "azurerm_storage_account" "example":
35:   restore_policy {
   ~

Blocks of type "restore_policy" are not expected here.

C:\Users\stani\Desktop\DevOpsAcademy\ScaleFocus-Academy-Homework\20.1.Terraform and Azure Active Directory\Task_2>
```

- Now we will set the azurerm provider version to 3.35.0, and will add the following code to the configuration file.

When we do it, we will receive an error for the restore_policy. In order to fix this error we can go to the terraform registry and read the provided documentation.

- The field is not present in version 3.35.0 and this is causing our problem. The change for the restore_policy on the blob_properties is introduced in version 3.36.0 of the azurerm provider and we have set fixed version of 3.35.0 for the provider.



```
main.tf
terraform {
  required_providers {
    azure = {
      source = "hashicorp/azure"
      version = "~>3.36.0"
    }
  }
}

provider "azure" {
  features {}
}

data "azurerm_subscription" "current" {}

resource "random_string" "random" {
  length = 8
  special = false
  lower = true
  upper = false
}

resource "azurerm_resource_group" "example" {
  name = "${random_string.random.result}-rg"
  location = "West Europe"
}

resource "azurerm_storage_account" "example" {
  name = "${random_string.random.result}sa"
  resource_group_name = azurerm_resource_group.example.name
  location = azurerm_resource_group.example.location
  account_tier = "Standard"
  account_replication_type = "GRS"

  blob_properties {
    restore_policy {
      days = 7
    }

    delete_retention_policy {
      days = 8
    }

    versioning_enabled = true
    change_feed_enabled = true
  }

  tags = {
    environment = "staging"
  }
}
```

```

c:\Users\stani\Desktop\DevOpsAcademy\ScaleFocus-Academy-Homework\20.1.Terraform and Azure Active Directory\Task_2>terraform init -upgrade

Initializing the backend...

Initializing provider plugins...
- Finding hashicorp/azurerm versions matching "3.36.0"...
- Finding latest version of hashicorp/random...
- Using previously-installed hashicorp/random v3.4.3
- Installing hashicorp/azurerm v3.36.0...
- Installed hashicorp/azurerm v3.36.0 (signed by HashiCorp)

Terraform has made some changes to the provider dependency selections recorded
in the .terraform.lock.hcl file. Review those changes and commit them to your
version control system if they represent changes you intended to make.

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.

C:\Users\stani\Desktop\DevOpsAcademy\ScaleFocus-Academy-Homework\20.1.Terraform and Azure Active Directory\Task_2>terraform plan
data.azurerm_subscription.current: Reading...
data.azurerm_subscription.current: Read complete after 0s [id=/subscriptions/c983dec5-cde0-4991-9469-c26f8cf60856]

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols:
  + create

Terraform will perform the following actions:

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

# azurerm_storage_account.example will be created
+ resource "azurerm_storage_account" "example" {
  + access_tier = (known after apply)
  + account_kind = "StorageV2"
  + account_replication_type = "GRS"
  + account_tier = "Standard"
  + allow_nested_items_to_be_public = true
  + cross_tenant_replication_enabled = true
}
```

- Now when we changed the version to 3.36.0 and we also need to add delete_retention_policy under the restore_policy block.

```
+ cross_tenant_replication_enabled = true
+ default_to_oauth_authentication = false
+ enable_https_traffic_only       = true
+ id                             = (known after apply)
+ infrastructure_encryption_enabled = false
+ is_hns_enabled                  = false
+ large_file_share_enabled        = (known after apply)
+ location                        = "westeurope"
+ min_tls_version                  = "TLS1_2"
+ name                           = (known after apply)
+ nfsv3_enabled                   = false
+ primary_access_key               = (sensitive value)
+ primary_blob_connection_string   = (sensitive value)
+ primary_blob_endpoint            = (known after apply)
+ primary_blob_host                = (known after apply)
+ primary_connection_string        = (sensitive value)
+ primary_dfs_endpoint             = (known after apply)
+ primary_dfs_host                 = (known after apply)
+ primary_file_endpoint            = (known after apply)
+ primary_file_host                = (known after apply)
+ primary_location                 = (known after apply)
+ primary_queue_endpoint           = (known after apply)
+ primary_queue_host               = (known after apply)
+ primary_table_endpoint           = (known after apply)
+ primary_table_host               = (known after apply)
+ primary_web_endpoint             = (known after apply)
+ primary_web_host                 = (known after apply)
+ public_network_access_enabled    = true
+ queue_encryption_key_type        = "Service"
+ resource_group_name              = (known after apply)
+ secondary_access_key              = (sensitive value)
+ secondary_blob_connection_string = (sensitive value)
+ secondary_blob_endpoint           = (known after apply)
+ secondary_blob_host              = (known after apply)
+ secondary_connection_string       = (sensitive value)
+ secondary_dfs_endpoint            = (known after apply)
+ secondary_dfs_host               = (known after apply)
+ secondary_file_endpoint           = (known after apply)
+ secondary_file_host              = (known after apply)
+ secondary_location               = (known after apply)
+ secondary_queue_endpoint          = (known after apply)
+ secondary_queue_host              = (known after apply)
+ secondary_table_endpoint          = (known after apply)
+ secondary_table_host              = (known after apply)
+ secondary_web_endpoint            = (known after apply)
+ secondary_web_host               = (known after apply)
+ sftp_enabled                     = false
+ shared_access_key_enabled        = true
+ table_encryption_key_type         = "Service"
+ tags                             = {
+   + "environment" = "staging"
}
```

- Part 2 of the output.



```
+ secondary_queue_endpoint = (known after apply)
+ secondary_table_endpoint = (known after apply)
+ secondary_table_host     = (known after apply)
+ secondary_web_endpoint   = (known after apply)
+ secondary_web_host       = (known after apply)
+ sftp_enabled             = false
+ shared_access_key_enabled = true
+ table_encryption_key_type = "Service"
+ tags                     = {
+   "environment" = "staging"
+ }
}

+ blob_properties {
+   change_feed_enabled      = true
+   default_service_version = (known after apply)
+   last_access_time_enabled = false
+   versioning_enabled       = true

+   delete_retention_policy {
+     days = 8
+   }

+   restore_policy {
+     days = 7
+   }
+ }
}

# random_string.random will be created
+ resource "random_string" "random" {
+   id          = (known after apply)
+   length      = 8
+   lower       = true
+   min_lower   = 0
+   min_numeric = 0
+   min_special = 0
+   min_upper   = 0
+   number      = true
+   numeric     = true
+   result      = (known after apply)
+   special     = false
+   upper       = false
+ }

Plan: 3 to add, 0 to change, 0 to destroy.
```

```
Note: You didn't use the -out option to save this plan, so Terraform can't guarantee to take exactly
these actions if you run "terraform apply" now.

C:\Users\stani\Desktop\DevOpsAcademy\ScaleFocus-Academy-Homework\20.1.Terraform and Azure Active Direc
tory\Task_2>
```

- How many resources have you defined in your code and how many resources does the plan output show? Are they the same and why?
 - The code defines 3 resources: a resource group, a storage account, and a random string. The plan output shows that 3 resources will be added, which matches the number of resources defined in the code.
- What is the location of your resource group and what is the location of the storage account?
 - The location of the resource group and storage account is "westeurope", as specified in the code.



- After Deploying the code:
 - How many resources do you have on your subscription? (To list all resources, type “All resources” in the search bar on the top in Azure Portal).

Microsoft Azure

Search resources, services, and docs (G+/I)

Home >

All resources

Default Directory (StanislavNikolovNew1@outlook.onmicrosoft.com)

+ Create Manage view Refresh Export to CSV Open query Assign tags Delete

Filter for any field... Subscription equals all Resource group equals all Type equals all Location equals all Add filter

0 Unsecure resources 0 Recommendations

Name	Type	Resource group	Location	Subscription
csb100320028852c669	Storage account	cloud-shell-storage-westeurope	West Europe	Azure Pass - Sponsorship
gvk3m4sa	Storage account	gvk3m4-rg	West Europe	Azure Pass - Sponsorship
NetworkWatcher_westeurope	Network Watcher	NetworkWatcherRG	West Europe	Azure Pass - Sponsorship

gvk3m4sa

Storage account

Search

Upload Open in Explorer Delete Move Refresh Open in mobile CLI / PS Feedback

JSON View

Essentials

Resource group (move) : gvk3m4-rg Performance : Standard

Location : West Europe Replication : Geo-redundant storage (GRS)

Primary/Secondary Location : Primary: West Europe, Secondary: North Europe Account kind : StorageV2 (general purpose v2)

Subscription (move) : Azure Pass - Sponsorship Provisioning state : Succeeded

Subscription ID : c983dec5-cde0-4991-9469-c26f8cf60056 Created : 10.04.2023 r, 23:29:46 u.

Disk state : Primary: Available, Secondary: Available

Tags (edit) : environment : staging

Properties Monitoring Capabilities (7) Recommendations (0) Tutorials Tools + SDKs

Blob service

Hierarchical namespace Disabled

Default access tier Hot

Blob public access Enabled

Blob soft delete Enabled (8 days)

Container soft delete Disabled

Versioning Enabled

Change feed Enabled

NFS v3 Disabled

Allow cross-tenant replication Enabled

File service

Large file share Disabled

Active Directory Not configured

Default share-level permissions Disabled

Soft delete Enabled (7 days)

Share capacity 5 TiB

Queue service

Security

Require secure transfer for REST API operations Enabled

Storage account key access Enabled

Minimum TLS version Version 1.2

Infrastructure encryption Disabled

Networking

Allow access from All networks

Number of private endpoint connections 0

Network routing Microsoft network routing

Access for trusted Microsoft services Yes

Endpoint type Standard



I have 3 resources. 2 storage accounts(one for the cloud shell and one created now with terraform)

- Are the number of resources shown in the All resources portal window the same with the ones from your plan?

The one storage account, that we created is shown in here, so I think this is the name, that's assigned in the variable.

- Give short explanation about the resources that are not shown?
The only resource, that I can't see is random_string with name random.

Task 3: Using variables and outputs.

```
variable "my_name" {
  type = string
  description = "First name of the student"
}
resource "azurerm_resource_group" "example2" {
  name     = "${var.my_name}-${random_string.random.result}-rg"
  location = var.location
}

variable "location" {
  type = string
  description = "The location where all resources will be placed."
  default = "West Europe"
}
```

```
C:\Users\stani\Desktop\DevOpsAcademy\ScaleFocus-Academy-Homework\20.1.Terraform and Azure Active Directory
var.my_name
First name of the student

Enter a value: Stan

random_string.random: Refreshing state... [id=gvgkg3m0]
data.azurem_subscription.current: Reading... [id=/subscriptions/c983dec5-cde8-4991-9469-c26f8c6885]
azurerm_resource_group.example: Refreshing state... [id=/subscriptions/c983dec5-cde8-4991-9469-c26f8c6885]
data.azurem_subscription.current: Read complete after 6s [id=/subscriptions/c983dec5-cde8-4991-9469-c26f8c6885]
azurerm_storage_account.example: Refreshing state... [id=/subscriptions/c983dec5-cde8-4991-9469-c26f8c6885]
microsoft.Storage/storageAccounts/gvgkg3m0]

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols:
  ~ create

Terraform will perform the following actions:

# azurerm_resource_group.example2 will be created
+ resource "azurerm_resource_group" "example2" {
+   id           = (known after apply)
+   location     = "westeurope"
+   name         = "Stan-gvgkg3m0-rg"
}

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

Note: You didn't use the -out option to save this plan, so Terraform can't guarantee to take exactly these actions if you run "terraform apply" now.

C:\Users\stani\Desktop\DevOpsAcademy\ScaleFocus-Academy-Homework\20.1.Terraform and Azure Active Directory
```

- How many variables do we have defined, and which are they?
 - We have defined 2 variables, “my_name” variable and “location” variable, which has a default value of “West Europe”
- Why did terraform asked us to input a value only for the my_name variable?
 - Terraform asked for input only for the my_name variable because it was not defined with a default value in the variables.tf file, while the location variable was defined with a default value of "West Europe".
 - When executing terraform plan, Terraform checks for any input variables that are not defined with a default value or specified in a variable file and prompts the user to input their values. In this case, since my_name did not have a default value, Terraform prompted for input.



```
input.tfvars
my_name = "stan"

variables.tf
variable "my_name" {
  type = string
  description = "First name of the student"
}
resource "azurerm_resource_group" "example2" {
  name     = "${var.my_name}-${random_string.random.result}-rg"
  location = var.location
}
variable "location" {
  type = string
  description = "The location where all resources will be placed."
  default = "West Europe"
}
```

```
C:\Users\stan\Desktop\DevOpsAcademy\ScaleFocus-Academy-Homework\20.1.Terraform and Azure Active Directory\Task 2>terraform plan -var-file=input.tfvars
random_string.random: Refreshing state... [id=gvkmg3m4]
data.azurem_subscription.current: Reading... [id=/subscriptions/c983dec5-cde0-4991-9469-c26f8cf68856/resourceGroups/gvkmg3m4-rg]
data.azurem_subscription.current: Read complete after 0s [id=/subscriptions/c983dec5-cde0-4991-9469-c26f8cf68856]
azurerm_resource_group.example: Refreshing state... [id=/subscriptions/c983dec5-cde0-4991-9469-c26f8cf68856/resourceGroups/gvkmg3m4-rg/providers/Microsoft.Storage/storageAccounts/gvkmg3m4sa]

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols:
  create

Terraform will perform the following actions:

# azurerm_resource_group.example2 will be created
+ resource "azurerm_resource_group" "example2" {
+   id       = (known after apply)
+   location = "westeurope"
+   name     = "stan-gvkmg3m4-rg"
}

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

Note: You didn't use the -out option to save this plan, so Terraform can't guarantee to take exactly
```

- Creating a file for the input variables(input.tfvars), also executing the terraform plan with variable file.

```
main.tf
terraform {
  required_providers {
    azurerm = {
      source = "hashicorp/azurerm"
      version = "~>3.36.0"
    }
  }
}
provider "azurerm" {
  features {}
}
locals {
  resource_prefix = "${var.my_name}${random_string.random.result}"
}
data "azurerm_subscription" "current" {}

resource "random_string" "random" {
  length      = 8
  special     = false
  lower       = true
  upper       = false
}

resource "azurerm_resource_group" "example" {
  name     = "${random_string.random.result}-rg"
  location = var.location
}

resource "azurerm_storage_account" "example" {
  name                = "${random_string.random.result}sa"
  resource_group_name = azurerm_resource_group.example.name
  location             = var.location
  account_tier         = "Standard"
  account_replication_type = "GRS"

  blob_properties {
    restore_policy {
      days = 7
    }

    delete_retention_policy {
      days = 8
    }
  }

  versioning_enabled = true
  change_feed_enabled = true
}

tags = {
  environment = "staging"
}
```

```
C:\Users\stan\Desktop\DevOpsAcademy\ScaleFocus-Academy-Homework\20.1.Terraform and Azure Active Directory\Task 2>terraform plan -var-file=input.tfvars
random_string.random: Refreshing state... [id=gvkmg3m4]
data.azurem_subscription.current: Reading... [id=/subscriptions/c983dec5-cde0-4991-9469-c26f8cf68856/resourceGroups/gvkmg3m4-rg]
data.azurem_subscription.current: Read complete after 0s [id=/subscriptions/c983dec5-cde0-4991-9469-c26f8cf68856]
azurerm_resource_group.example: Refreshing state... [id=/subscriptions/c983dec5-cde0-4991-9469-c26f8cf68856/resourceGroups/gvkmg3m4-rg/providers/Microsoft.Storage/storageAccounts/gvkmg3m4sa]

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols:
  create

Terraform will perform the following actions:

# azurerm_resource_group.example2 will be created
+ resource "azurerm_resource_group" "example2" {
+   id       = (known after apply)
+   location = "westeurope"
+   name     = "stan-gvkmg3m4-rg"
}

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

Note: You didn't use the -out option to save this plan, so Terraform can't guarantee to take exactly these actions if you run "terraform apply" now.

C:\Users\stan\Desktop\DevOpsAcademy\ScaleFocus-Academy-Homework\20.1.Terraform and Azure Active Directory\Task 2>
```

- Editing main.tf by adding locals{ } and changing location to the variable location.



```
main.tf
terraform {
  required_providers {
    azure = {
      source = "hashicorp/azure"
      version = "~>3.36.0"
    }
  }
}

provider "azure" {
  features {}
}

locals {
  resource_prefix = "${var.my_name}${random_string.random.result}"
}

data "azurerm_subscription" "current" {}

resource "random_string" "random" {
  length = 8
  special = false
  lower = true
  upper = false
}

resource "azurerm_resource_group" "example" {
  name     = "${random_string.random.result}-rg"
  location = var.location
}

input.tfvars
my_name = "stan"

output.tf
output "resource_group_name" {
  value = azurerm_resource_group.example.name
  description = "The name of the resource group we deployed"
}

output "storage_account_name" {
  value = azurerm_storage_account.example.name
  description = "The name of the storage account we deployed"
}
```

- Executing the terraform plan with the input variable file switch.

When you execute terraform plan it will give you information about the resources and parameters that are being created with “+”, destroyed and recreated with “-/+”, the ones destroyed with “-” and the ones that will be modified with “~”.

- In Terraform, "force replacement" occurs when a resource needs to be destroyed and recreated in order to apply changes to its configuration. This happens when a change is made to a property that is not updatable, or when a change is made to a property that is part of the resource's identity (such as its name).
- When Terraform detects a change that requires force replacement, it will mark the resource as "-/+" in the plan output, indicating that it will be destroyed and recreated. This can have implications for the availability of the resource during the update process, so it's important to understand why a resource is being replaced and what the impact will be.
- In general, it's a good idea to avoid making changes to properties that require force replacement unless it's absolutely necessary. If a resource must be replaced, Terraform will first attempt to gracefully destroy it and create a new one in its place, but this can still result in downtime or data loss depending on the resource and the change being made.



```
C:\Users\stani\Desktop\DevOpsAcademy\ScaleFocus-Academy-Homework\20.1.Terraform and Azure Active Directory\Task_2>terraform apply
var.my_name
  First name of the student

Enter a value: stan

random_string.random: Refreshing state... [id=gvkgm3m4]
azurerm_resource_group.example: Refreshing state... [id=/subscriptions/c983dec5-cde0-4991-9469-c26f8cf60056/resourceGroups/gvkgm3m4-rg]
data.azurerm_subscription.current: Reading...
azurerm_storage_account.example: Refreshing state... [id=/subscriptions/c983dec5-cde0-4991-9469-c26f8cf60056/resourceGroups/gvkgm3m4-rg/providers/Microsoft.Storage/storageAccounts/gvkgm3m4sa]
data.azurerm_subscription.current: Read complete after 0s [id=/subscriptions/c983dec5-cde0-4991-9469-c26f8cf60056]

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols:
  + create

Terraform will perform the following actions:

# azurerm_resource_group.example2 will be created
+ resource "azurerm_resource_group" "example2" {
  + id          = (known after apply)
  + location    = "westeurope"
  + name        = "stan-gvkgm3m4-rg"
}

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

Changes to Outputs:
  + resource_group_name = "gvkgm3m4-rg"
  + storage_account_name = "gvkgm3m4sa"

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.example2: Creating...
azurerm_resource_group.example2: Creation complete after 1s [id=/subscriptions/c983dec5-cde0-4991-9469-c26f8cf60056/resourceGroups/stan-gvkgm3m4-rg]

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

Outputs:
resource_group_name = "gvkgm3m4-rg"
storage_account_name = "gvkgm3m4sa"
```

- The outputs of the terraform apply.