September 13, 2024

DevOps Classroom notes 13/Sep/2024

**Infra Provisioning**

* The process of creating infrastructure by experessing our infra needs in a declarative way and then executing the template to create infra structure (Virtual)
* Declarative

I want a linux vm with 2 CPUs and 8 GB RAM

* Procedural

Go the vsphere

click ...

click ...

# you get the server

**Demo**

* In the demo lets understand an activity of creating a linux server and installing docker in it

September 14, 2024

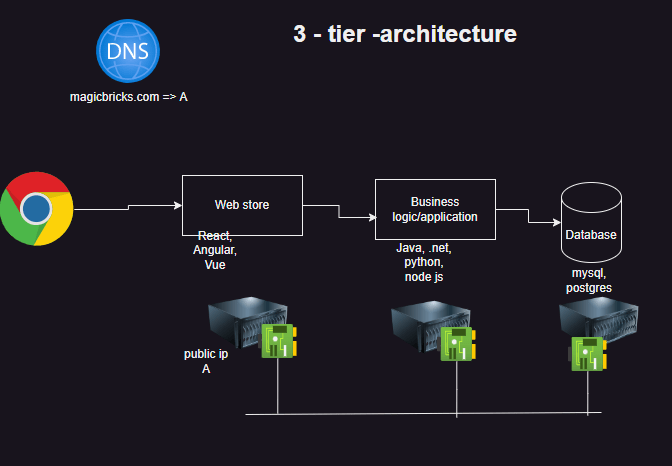
# DevOps Classroom notes 14/Sep/2024

## Infrastructure as Code w.r.t Infra Provisioning

* [Infrastructure as a code](https://directdevops.blog/2024/09/14/devops-classroom-notes-14-sep-2024/) ([IAC](https://directdevops.blog/2024/09/14/devops-classroom-notes-14-sep-2024/)) is a concept where you express your infrasturcture in a declarative way
* We need to express
  + What we want to create/manage (Resource)
  + Outputs
* Tools
  + AWS: Cloud formation
  + Azure: ARM Templates, Bicep
  + Terraform (open tofu): Almost any virtual environments
  + Pulumi

Cloud hosting

### Three tier architecture

* Lets look at a simple 3-tier architecture (Web sites)  
  
* The infrastructure to run the above app is
  + Network
  + Servers (Linux/Windows)
  + Public ip
  + DNS (optional)
* The application deployment will be possible when this infrastructure is ready
* Deploying application involves
  + running some commands on each server to install and configure applications
* Terraform is used for Infra creation

#### Other areas to consider

* Terraform will create infra which is virtual in nature.
* Virtual infra can be created
  + onpremises
    - Hypervisors (Vmware)
    - Private Cloud (Openstack)

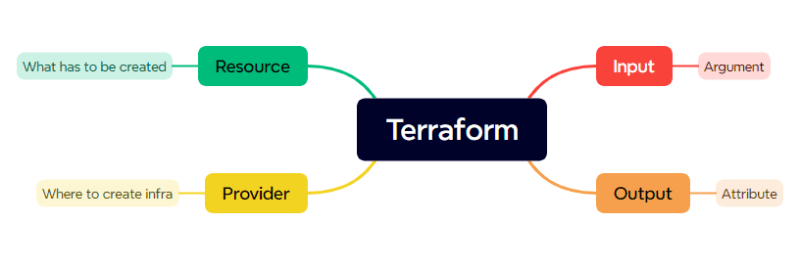
Cloud hosting

* + cloud
    - Azure
    - AWS
    - GCP
* We need to provide information to Terraform where to create infra and pass credentials
* Conceptually this is called as **Provider** in Terraform

#### Complications in Three Tier after running our application in production

* To address the increasing needs of your applications we have to do scaling
  + Vertical Scaling: Increasing the resources of the same server
  + Horizontal Scaling: Increasing number of servers
* For this every cloud offers auto scaling
  + AWS => auto scaling group
  + GCP => INstance Groups
  + Azure => Virtual machine scale sets

### Terraform components (So far)

* Provider
* Resource  
  

### How Terraform Works

* Terraform is developed in Google Go language
* Terraform is a single executable.
* Terraform can interact with specific providers (AWS, Azure, Vmware) to create/manage infrastructure [Refer Here](https://registry.terraform.io/browse/providers)
* Each Provider will have lots of resources which we can define to create infra.
* What terraform can do is (as of our understanding)
  + init (initialize => downloading providers)
  + apply (create/update infra)
  + destroy (delete infrawork)

### Template

* In terraform we need to express infrastucture in a declartive fashion and what we create is called as template.
* Terraform takes a folder as input considering all .tf files a folder

terraform {

required\_providers {

aws = {

source = "hashicorp/aws"

version = "5.67.0"

}

}

}

provider "aws" {

# Configuration options

region = "ap-south-1"

}

resource "aws\_vpc" "base" {

cidr\_block = "10.100.0.0/16"

tags = {

Name = "from devops team"

}

}

### Share this:

* [Click to share on Twitter (Opens in new window)](https://directdevops.blog/2024/09/14/devops-classroom-notes-14-sep-2024/?share=twitter&nb=1)
* [Click to share on Facebook (Opens in new window)](https://directdevops.blog/2024/09/14/devops-classroom-notes-14-sep-2024/?share=facebook&nb=1)
* [Click to share on WhatsApp (Opens in new window)](https://directdevops.blog/2024/09/14/devops-classroom-notes-14-sep-2024/?share=jetpack-whatsapp&nb=1)

### Like this:

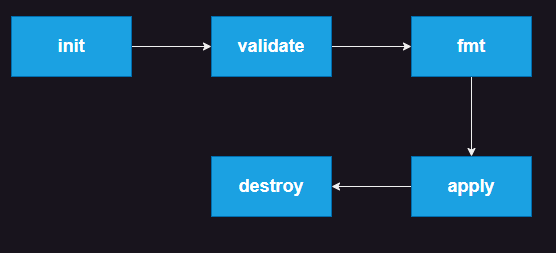
September 15, 2024

DevOps Classroom notes 15/Sep/2024

**Terraform installation**

* [Refer Here](https://developer.hashicorp.com/terraform/tutorials/aws-get-started/install-cli) for docs and classroom video for setup

**Terraform Workflow**

* Create a new folder for every infra
* we will be adding .tf files
* initialize [terraform](https://directdevops.blog/2024/09/15/devops-classroom-notes-15-sep-2024/)
* validate terraform
* format terraform
* apply
* destroy  
  

**Configuring Terraform AWS and AzureRm Provider (Authentication)**

* Ensure your system is configured as discussed in the class
* [Azure](https://directdevops.blog/2024/09/15/devops-classroom-notes-15-sep-2024/) example

terraform {

required\_providers {

azurerm = {

source = "hashicorp/azurerm"

version = "4.2.0"

}

}

}

provider "azurerm" {

# Configuration options

features { }

subscription\_id = "<your-subcription-id>"

}

resource "azurerm\_resource\_group" "test" {

name = "test"

location = "eastus"

}

* AWS Example

terraform {

required\_providers {

aws = {

source = "hashicorp/aws"

version = "5.67.0"

}

}

}

provider "aws" {

# Configuration options

}

resource "aws\_vpc" "base" {

cidr\_block = "10.0.0.0/16"

tags = {

Name = "aws-vpc"

}

}

**Share this:**

September 17, 2024

DevOps Classroom notes 17/Sep/2024

**Terraform Ways of Working**

* Ensure you know how to manually create resources
* List out all the resources and ensure you have creation order
* For each resource, list all the inputs you are passing
* For each resource you have created manually, find the equivelent [terraform](https://directdevops.blog/2024/09/17/devops-classroom-notes-17-sep-2024/) resource.

**Activity 1: Create a vpc and 4 subnets in AWS**

* Manual steps: For guided creation on console refer class room video.
* [Refer Here](https://github.com/asquarezone/TerraformZone/commit/663748901575a6eac83ec73f486930edd4818037) for changes done

**Resource Block**

* In terraform, the language which we use is called as HCL (Hashicorp configuration language) [Refer Here](https://developer.hashicorp.com/terraform/language/syntax/configuration)
* Resource syntax

resource "type\_of\_resource" "name" {

arg1 = value1

arg2 = value2

...

argn = valuen

}

* Values can be of different types [Refer Here](https://developer.hashicorp.com/terraform/language/expressions/types)
* Basic:
  + string
  + number
  + bool
* To use attributes the syntax is <resource-type>.<name>.<attribute>

**Activity 2: Create a virtual network in Azure**

* [Refer Here](https://learn.microsoft.com/en-us/azure/virtual-network/quick-create-portal) for virtual network quick start
* Manual steps: For guided creation on console refer class room video.
* We have done till resource group creation and to be continued in the next session
* [Refer Here](https://github.com/asquarezone/TerraformZone/commit/1403f3c2670a4b6441748deda7ac3a42ea6b4090) for the changes done so far.

**Share this:**

September 19, 2024

# DevOps Classroom notes 19/Sep/2024

## Terraform contd

### Activity 2: Create a virtual network in [Azure](https://directdevops.blog/2024/09/19/devops-classroom-notes-19-sep-2024/) (contd)

* we have already created a resource group
* For Manual steps watch classroom video
* [Refer Here](https://github.com/asquarezone/TerraformZone/commit/7f54812c0d11c105d8505db78801f43ba6b951b5) for the changes done

### Dependencies

* When a resource in a terraform uses attributes of other resource then an implicit dependency is created i.e the current resource will not be created until the other resource is created
* Dependencies can be made explicit by adding a meta argument depends\_on [Refer Here](https://developer.hashicorp.com/terraform/language/meta-arguments/depends_on)

#### idempotance

* Terraform execution is idempotent as execution irrespective of number of times will give you the same result (i.e desired state)

#### Terraform console

* [Refer Here](https://developer.hashicorp.com/terraform/cli/commands/console)

### Reasons for writing Template

* We would want to reuse the terraform templates across various environments and for every environment there will be changes
* Changes can be in
  + names
  + values
  + count of items
* Try to make terraform template generic not specific
* Specific example

def multiply():

return 10 \* 20

* Generic

# required arguments

def multiply(a, b):

return a \* b

# default arguments

def multiply(a=10, b=20):

return a \* b

### Activity 1.1: Lets change the vpc template

* Lets ask the user the vpc cidr range and subnet cidr ranges
* Lets ask the user for the Name of vpc and subnets
* Exercise: Try implementing variables

### Activity 2.1: Lets change the virtual network template

* Lets ask the user the vnet cidr range and subnet cidr ranges
* Lets ask the user for the Name of virtual network and subnets  
  [Refer Here](https://github.com/asquarezone/TerraformZone/commit/3f95af14f2472ad48864226beb38061105535b8a) for changes

### Variable

* [Refer Here](https://developer.hashicorp.com/terraform/language/values/variables) for official docs
* Terraform provides variables where user can set values while applying
* Variable definition

variable "<variable\_name>" {

type = string | number | bool | map | object | list

description =

default =

}

* Lets look at example

# definition

variable "vpc\_cidr" {

type = string

default = "10.0.0.0/16"

description = "vpc cidr"

}

resource "aws\_vpc" "base" {

# usage

cidr\_block = var.vpc\_cidr

}

* Good practice is to define all variables in a file called as variables.tf or inputs.tf
* Variables can be passed at runtime during apply by adding -var to the apply

terraform apply -var vpc\_cidr='192.168.0.0/16'

### Share this:

September 20, 2024

# DevOps Classroom notes 20/Sep/2024

## loops in Terraform

### Count meta argument

* [Refer Here](https://developer.hashicorp.com/terraform/language/meta-arguments/count) for official docs

#### Activity 2.2: use count to create multiple subnets in a loop

* [Refer Here](https://github.com/asquarezone/TerraformZone/commit/7cf22f0f7d1ded5b3626065db9d13b689d076773) for changes.

#### Activity 1.2: Lets change the vpc template and

* Lets ask the user the vpc cidr range and subnet cidr ranges
* Lets ask the user for the Name of vpc and subnets
* Also implement count to create 4 subnets
* inputs to vpc
  + cidr block
  + tags
* inputs to subnet
  + cidr block
  + availability zone
  + tags
* [Refer Here](https://github.com/asquarezone/TerraformZone/commit/567b2f9935434388de15303b0834be5b4f240679) for the changes done

#### Activity 2.3 use complex variables in azure

* [Refer Here](https://github.com/asquarezone/TerraformZone/commit/dfda3608179ccd50fd1856c867d2085905ec1640) for changes done to include complex variables.

### Passing variables during apply

* we have already looked at option terraform apply -var <var-name>='value'
* Terraform also gives us an option to pass multiple variables at one shot in variables file
* This variables file will have extension of .tfvars and to pass it during apply terraform apply -var-file='dev.tfvars'

#### Activity 1.3 use variables file for aws template

* we have created a file called dev.tfvars which represents values for dev environment
* Command used to create infra

terraform apply -var-file='dev.tfvars'

* [Refer Here](https://github.com/asquarezone/TerraformZone/commit/b51d53a9c41e95c8eacf9bf345cb393dd9cbe856)
* Exercise: DO the same for azure

## Terraform functions

* [Refer Here](https://developer.hashicorp.com/terraform/language/functions) for the official docs of terraform functions

#### Activity 1.4: Use function to get how many subnets to be created

* [Refer Here](https://github.com/asquarezone/TerraformZone/commit/96b93d68283c465047cd16cb6fe715b019f10de5)
* Apply the same for azure [Refer Here](https://github.com/asquarezone/TerraformZone/commit/5492bf6d5e30b440526f996d8175ee37a4feda4e)

### Share this:

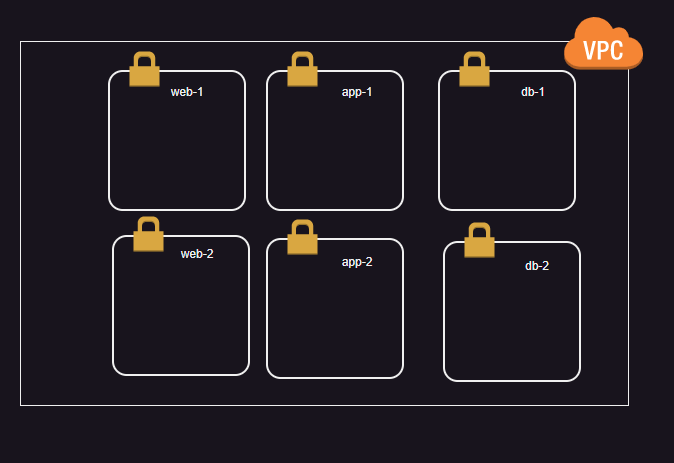
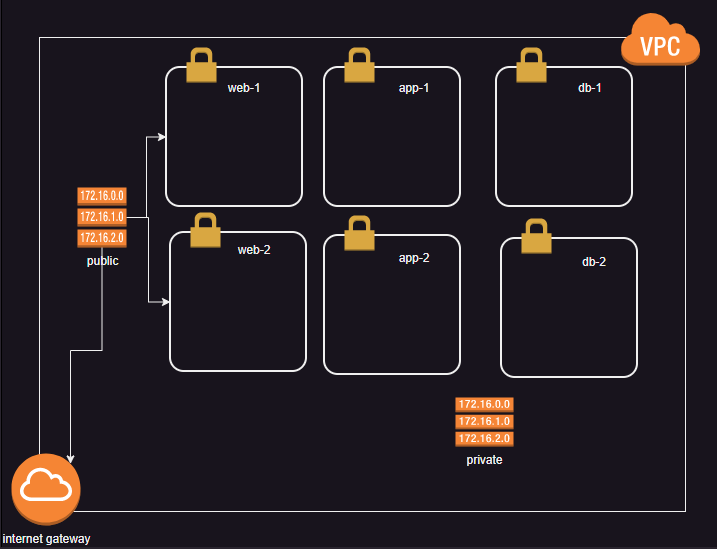
September 21, 2024

DevOps Classroom notes 21/Sep/2024

**AWS Ntier Architecture**

* Ntier is very popular architecture for deploying web [applications](https://directdevops.blog/2024/09/21/devops-classroom-notes-21-sep-2024/)
  + web-tier: Webservers will be hosted which serve webpages and they will be accessed directly/indirectly from internet
  + app/business tier: This tier has applications which run the business logic
  + data tier: we store the data in the [database](https://directdevops.blog/2024/09/21/devops-classroom-notes-21-sep-2024/)

**Phase 1: Create basic network and subnets**

* Goal  
  
* We have two categories of subnets of here
  + public subnets (web-1, web-2)
  + private subnets (others)
* [Refer Here](https://github.com/asquarezone/TerraformZone/commit/131917ebfa337f7adac06a6ab2322605146d24cb) for changes done to create vpc and subnets
* Now lets add an internet gateway [Refer Here](https://github.com/asquarezone/TerraformZone/commit/d14655c53f1bc5fad346d8f9fbcaa19e2be6ed27) for changes
* Now lets create a public route table and private route table and associate them to subnets [Refer Here](https://github.com/asquarezone/TerraformZone/commit/500299f164261b648a411bb3ede0bf78b4216018) for the changes
* Now lets add route for public route table to forward to internet gateway [Refer Here](https://github.com/asquarezone/TerraformZone/commit/eefa9e7c7e7b115d5e4a908c35856a99c29c494a)  
  
* Now we need to create security groups
  + web
    - allow 80 and 22 port from any where
  + app
    - allow 8000 port within vpc
  + db
    - allow 3306 port within vpc
* [Refer Here](https://github.com/asquarezone/TerraformZone/commit/acd1434cb42e087981ace7e51bd78d4d70487248) for the changes
* Now lets try to create a ec2 instance in web
  + ami id
  + key pair
  + security group
  + subnet
  + public ip
  + instance type
* [Refer Here](https://github.com/asquarezone/TerraformZone/commit/e3324232c526ba85d97f11006c5ace58dec15320) for creating an ec2 instance and outputs

**Conditional resource creation**

* We have already used the count meta argument, in the count if we pass zero then resource will not be created. [Refer Here](https://developer.hashicorp.com/terraform/language/expressions/conditionals)

resource "aws\_internet\_gateway" "ntier" {

# conditional creation

count = length(var.public\_subnets) > 0 ? 1 : 0

vpc\_id = aws\_vpc.ntier.id

tags = {

Name = "ntier-igw"

}

}

**Locals**

* [Refer Here](https://developer.hashicorp.com/terraform/language/values/locals) for official docs

**Outputs**

* [Refer Here](https://developer.hashicorp.com/terraform/language/values/outputs) for terraform outputs

**Terraform block**

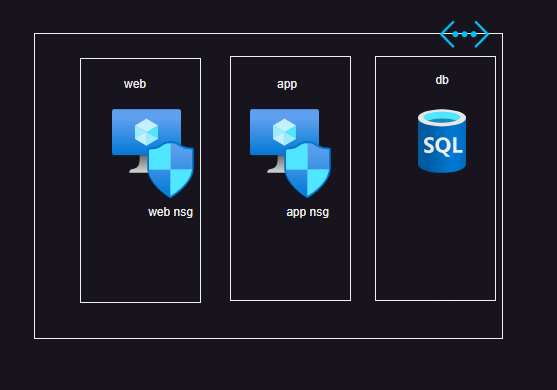
* [Refer Here](https://developer.hashicorp.com/terraform/language/terraform) for official docs of terraform block
* From terraform block we can restrict
  + provider versions
  + which terraform version is required to run the template
* Versioning constraints [Refer Here](https://developer.hashicorp.com/terraform/language/expressions/version-constraints)
* [Refer Here](https://github.com/asquarezone/TerraformZone/commit/290a2703107fade403388d84901d2022ccdfe37e) for changes in providers to restrict terraform and providers version

**Share this:**

September 22, 2024

DevOps Classroom notes 22/Sep/2024

**Azure Ntier Architecture**

* Overview  
  
* Create network and subnets [Refer Here](https://github.com/asquarezone/TerraformZone/commit/622e7d796bdcd98ebb9254bea3b5187e21ad1c31)
* Create a webnsg where
  + 80,443 and 22 port are open from anywhere
* [Refer Here](https://github.com/asquarezone/TerraformZone/commit/0d45dd920367f5b0426ea80993836e2004b1c1c1) for the changes done to add web nsg
* Now add app nsg which opens 8000 port within network [Refer Here](https://github.com/asquarezone/TerraformZone/commit/bb1d46129b2fdb06bfcd00b805d12d9d658d86a7) for changes done
* create a public ip address [Refer Here](https://github.com/asquarezone/TerraformZone/commit/69554dedba83be37a10aa04806b551aa7bf846d7) for the changes
* create a network interface in web subnet with public ip and web nsg attached [Refer Here](https://github.com/asquarezone/TerraformZone/commit/f7b5e2e7b1ed1169078d1af50d5355acfc985362) for changes
* now attach network interface to the vm
* while creating the vm specify
  + size
  + os image
* [Refer Here](https://learn.microsoft.com/en-us/azure/virtual-machines/linux/quick-create-terraform?tabs=azure-cli) for official docs of microsoft azure to create linux vm from terraform
* [Refer Here](https://github.com/asquarezone/TerraformZone/commit/dfa0b1615331d3fe469e1254846e17dbb9793078)

**Datasource**

* We can use Datasource in terraform to fetch the information about resource
* [Refer Here](https://developer.hashicorp.com/terraform/language/data-sources) for official docs
* View the classroom recording for examples

terraform {

required\_providers {

azurerm = {

source = "hashicorp/azurerm"

version = "4.3.0"

}

aws = {

source = "hashicorp/aws"

version = "5.68.0"

}

}

}

provider "azurerm" {

features {}

}

provider "aws" {

}

# lets fetch default vpc id of mumbai region

data "aws\_vpc" "default" {

default = true

}

output "default-vpc-id" {

value = data.aws\_vpc.default.id

}

data "azurerm\_network\_security\_group" "web" {

name = "web"

resource\_group\_name = "ntier"

}

output "nsg\_id" {

value = data.azurerm\_network\_security\_group.web.id

}

# get the resource group name and location and create a new network

data "azurerm\_resource\_group" "base" {

name = "ntier1"

}

resource "azurerm\_virtual\_network" "new" {

name = "new"

resource\_group\_name = data.azurerm\_resource\_group.base.name

location = data.azurerm\_resource\_group.base.location

address\_space = ["10.0.0.0/16"]

}

**Share this:**

September 24, 2024

DevOps Classroom notes 24/Sep/2024

**Terraform state management**

* [Terraform](https://directdevops.blog/2024/09/24/devops-classroom-notes-24-sep-2024/) always ensures desired state is equal to actual state during apply
* Terraform execution
  + [terraform](https://directdevops.blog/2024/09/24/devops-classroom-notes-24-sep-2024/) gets the current state from provider
  + compares that with desired state, difference becomes plan
  + executes the plan to create resources and stores the creation details in the state file
* **Note**: Never manually edit/change state files
* If you want to force recreation of a resource
  + taint: marking a resource as taint will force replacement during next apply
* If you want to stop managing a resource from terraform and still want the resource to exist
  + delete from state
  + delete from template

**Terraform Importing Resources**

* [Refer Here](https://developer.hashicorp.com/terraform/language/import) for official docs
* Watch classroom recording for example

**Share this:**

September 25, 2024

DevOps Classroom notes 25/Sep/2024

**Multiuser Executions in Terraform**

* [Terraform](https://directdevops.blog/2024/09/25/devops-classroom-notes-25-sep-2024/) is storing the state in local folders
* Consider the scenario where a [terraform](https://directdevops.blog/2024/09/25/devops-classroom-notes-25-sep-2024/) template has to be executed by any team member to create a dev environment
* The default location is local folder, the state has to be in common location which is accesible to all team members
* Locking is required for giving access to only one a time
* The above discussed features can be done in terraform with the help of backends

**Terraform Backend**

* [Refer Here](https://developer.hashicorp.com/terraform/language/state/backends) for official docs
* As discussed in the class have a look at following backend configuration to use s3 backend

terraform {

required\_providers {

aws = {

source = "hashicorp/aws"

version = "5.68.0"

}

}

backend "s3" {

bucket = "terraformbackendforlt"

key = "demo/terraform.tfstate"

region = "ap-south-1"

dynamodb\_table = "myterraformstate"

}

}

provider "aws" {

# Configuration options

}

* Findout how to use azurerm backend, as we have tried in the class the following providers section was used

terraform {

required\_providers {

azurerm = {

source = "hashicorp/azurerm"

version = "4.3.0"

}

}

backend "azurerm" {

resource\_group\_name = "terraformstate"

storage\_account\_name = "lttfstaterepo"

container\_name = "state"

key = "demo/terraform.tfstate"

}

}

provider "azurerm" {

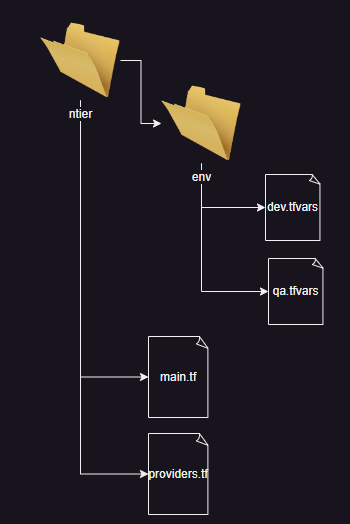
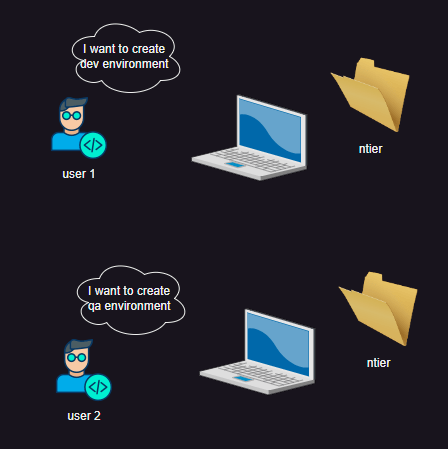
# Configuration options

features {

}

}

**Handling multiple environments**

* Consider the following template  
  
* In this template we have two tfvars with different variable values
* The problem with backend configuration is it is designed to manage only one environment  
  
* To solve the above mentioned problem we will be workspaces with backends

**Share this:**

September 26, 2024

DevOps Classroom notes 26/Sep/2024

**Terraform Workspace**

* [Refer Here](https://developer.hashicorp.com/terraform/language/state/workspaces) for official docs
* [Refer Here](https://github.com/asquarezone/TerraformZone/commit/bb6ea74ea1435dba333d9aa511c0481db4ec3eec) for the changes done

**Executing Script post EC2 creation in AWS**

* [Refer Here](https://github.com/asquarezone/TerraformZone/commit/8e1cbb24b6e95b6b8c40f50149b5bdfef7af8ced) for the changes
* Exercise: Executing Script post vm creation in Azure

**Share this:**

* [Click to share on Twitter (Opens in new window)](https://directdevops.blog/2024/09/26/devops-classroom-notes-26-sep-2024/?share=twitter&nb=1)
* [Click to share on Facebook (Opens in new window)](https://directdevops.blog/2024/09/26/devops-classroom-notes-26-sep-2024/?share=facebook&nb=1)
* [Click to share on WhatsApp (Opens in new window)](https://directdevops.blog/2024/09/26/devops-classroom-notes-26-sep-2024/?share=jetpack-whatsapp&nb=1)

September 27, 2024

DevOps Classroom notes 27/Sep/2024

**Terraform Best Practices**

* [Refer Here](https://cloud.google.com/docs/terraform/best-practices/general-style-structure) for best practices

**How enterprises use Terraform**

* We write templates
* All the resusable templates are converted into modules and store in a registry
* Registry Options
  + Terraform Cloud
  + Jfrog
  + Git

Cloud hosting

* Structure

template (folder)

|

---- providers.tf

---- main.tf # calling modules by passing values

* Security Scanning terrascan

**Terraform Registry (Public)**

* Lets create a vpc, security group and an ec2 instance by using reusable community modules
* [Refer Here](https://registry.terraform.io/browse/modules?provider=aws) for aws community modules
* [Refer Here](https://github.com/asquarezone/TerraformZone/commit/dd7460b859298179c72fe7f620fe36675498f675) for changes done to use community modules

**Terrraform Provisioners**

* [Refer Here](https://developer.hashicorp.com/terraform/language/resources/provisioners/syntax) for official docs
* Types
  + file
  + local-exec
  + remote-exec
* Connection block: in this block we will pass credentials
* How to use provisioner in terraform to install nginx into ec2 instance.
* [Refer Here](https://github.com/asquarezone/TerraformZone/commit/4c0b34f50f2d80d8a406bc8c77fe91d3b13fe1be) for changes

**Share this:**

September 28, 2024

DevOps Classroom notes 28/Sep/2024

**Terraform modules**

* Module is a reusable [Terraform](https://directdevops.blog/2024/09/28/devops-classroom-notes-28-sep-2024/) template
* Every [terraform](https://directdevops.blog/2024/09/28/devops-classroom-notes-28-sep-2024/) template is a root module which calls the child modules
* child modules are resuable terraform configuration
* Child Modules can be present
  + in local system
  + in git repo
  + in Terraform cloud
  + in jfrog

**Activity – 1: Create a ntier architecture in aws**

* Lets convert the work done earlier into modules
* to create ntier architecture, we need
  + vpc
  + security groups
  + ec2/autoscaling groups
  + load balancers
* We have started creating modules inside the root module

ntier

modules

vpc

main.tf

variables.tf

outputs.tf

providers.tf

main.tf

* [Refer Here](https://github.com/asquarezone/TerraformZone/commit/4e4acea5f68e2828895b9f55c16723d3a850e41b) for changes
* Using modules like this is done on servers where all the terraform modules are present on a network file share.
* Lets have our modules hosted seperately on Git
* [Refer Here](https://github.com/asquarezone/LTTerraformModules/tree/main/aws/vpc) for the module
* for referring module from git [Refer Here](https://developer.hashicorp.com/terraform/language/modules/sources#generic-git-repository) for git and [Refer Here](https://developer.hashicorp.com/terraform/language/modules/sources#modules-in-package-sub-directories) for subdirectories
* [Refer Here](https://github.com/asquarezone/LTTerraformModules/tree/main/aws/securitygroup) for the security group module
* [Refer Here](https://github.com/asquarezone/TerraformZone/commit/d6b2cf00c9cf69a8abaac985d3736784f18861bc) for using the security group module from git

**Activity – 2: Create a ntier architecture in azure**

* Lets convert the work done earlier into modules
* to create ntier architecture, we need
  + virtual network
  + network security groups
  + vm/vmss
  + load balancers

**Terrascan**

* This scans terraform for security vulnerabilities and generates the report
* [Refer Here](https://runterrascan.io/)

**Terraform cloud**

* [Refer Here](https://app.terraform.io/public/signup/account) for creating account
* [Refer Here](https://github.com/asquarezone/TerraformZone/commit/064726b4c3ccf813c5cff089282578533f46785d) for terrform cloud usage

October 2, 2024

DevOps Classroom notes 29/Sep/2024

**For each Block**

* [Refer Here](https://developer.hashicorp.com/terraform/language/meta-arguments/for_each) for foreach docs

variable "subnets" {

type = map(object({

cidr\_block = string

az = string

}))

default = {

"web" = {

cidr\_block = "10.0.0.0/24"

az = "ap-south-1a"

},

"app" = {

cidr\_block = "10.0.1.0/24"

az = "ap-south-1b"

}

}

}

resource "aws\_vpc" "base" {

cidr\_block = "10.0.0.0/16"

tags = {

Name = "base"

}

}

resource "aws\_subnet" "subnets" {

for\_each = var.subnets

cidr\_block = each.value.cidr\_block

availability\_zone = each.value.az

tags = {

Name = each.key

}

vpc\_id = aws\_vpc.base.id

}

**Dynamic Block**

* [Refer Here](https://developer.hashicorp.com/terraform/language/expressions/dynamic-blocks) for official docs
* Solution

variable "subnets" {

type = map(string)

default = {

"web" = "10.0.0.0/24"

"app" = "10.0.1.0/24"

"db" = "10.0.2.0/24"

}

}

terraform {

required\_providers {

azurerm = {

source = "hashicorp/azurerm"

version = "4.3.0"

}

}

}

provider "azurerm" {

# Configuration options

features {

}

}

resource "azurerm\_resource\_group" "rg" {

name = "rg-terraform"

location = "eastus"

}

resource "azurerm\_virtual\_network" "vnet" {

name = "vnet-terraform"

address\_space = ["10.0.0.0/16"]

location = azurerm\_resource\_group.rg.location

resource\_group\_name = azurerm\_resource\_group.rg.name

dynamic "subnet" {

for\_each = var.subnets

content {

name = subnet.key

address\_prefixes = [subnet.value]

}

}

}

**Creating Golden Images using Packer**

* [Refer Here](https://www.packer.io/) for packer

**Complete CI/CD Pipeline**

**CI/CD services**

**Share this:**

* [Click to share on Twitter (Opens in new window)](https://directdevops.blog/2024/10/02/devops-classroom-notes-02-oct-2024-2/?share=twitter&nb=1)
* [Click to share on Facebook (Opens in new window)](https://directdevops.blog/2024/10/02/devops-classroom-notes-02-oct-2024-2/?share=facebook&nb=1)
* [Click to share on WhatsApp (Opens in new window)](https://directdevops.blog/2024/10/02/devops-classroom-notes-02-oct-2024-2/?share=jetpack-whatsapp&nb=1)

**Like this:**

October 2, 2024

DevOps Classroom notes 29/Sep/2024

**For each Block**

* [Refer Here](https://developer.hashicorp.com/terraform/language/meta-arguments/for_each) for foreach docs

variable "subnets" {

type = map(object({

cidr\_block = string

az = string

}))

default = {

"web" = {

cidr\_block = "10.0.0.0/24"

az = "ap-south-1a"

},

"app" = {

cidr\_block = "10.0.1.0/24"

az = "ap-south-1b"

}

}

}

resource "aws\_vpc" "base" {

cidr\_block = "10.0.0.0/16"

tags = {

Name = "base"

}

}

resource "aws\_subnet" "subnets" {

for\_each = var.subnets

cidr\_block = each.value.cidr\_block

availability\_zone = each.value.az

tags = {

Name = each.key

}

vpc\_id = aws\_vpc.base.id

}

**Dynamic Block**

* [Refer Here](https://developer.hashicorp.com/terraform/language/expressions/dynamic-blocks) for official docs
* Solution

variable "subnets" {

type = map(string)

default = {

"web" = "10.0.0.0/24"

"app" = "10.0.1.0/24"

"db" = "10.0.2.0/24"

}

}

terraform {

required\_providers {

azurerm = {

source = "hashicorp/azurerm"

version = "4.3.0"

}

}

}

provider "azurerm" {

# Configuration options

features {

}

}

resource "azurerm\_resource\_group" "rg" {

name = "rg-terraform"

location = "eastus"

}

resource "azurerm\_virtual\_network" "vnet" {

name = "vnet-terraform"

address\_space = ["10.0.0.0/16"]

location = azurerm\_resource\_group.rg.location

resource\_group\_name = azurerm\_resource\_group.rg.name

dynamic "subnet" {

for\_each = var.subnets

content {

name = subnet.key

address\_prefixes = [subnet.value]

}

}

}

**Creating Golden Images using Packer**

* [Refer Here](https://www.packer.io/) for packer

**Complete CI/CD Pipeline**

**CI/CD services**

**Share this:**