CSDO1010 Assignment 4

Paul Krznaric

# Repo:

https://github.com/PaulKrznaric/Yorku-devops-assignment4

# Proof: Terraform Apply:

Graphical user interface, text

Description automatically generated  
> terraform init

Initializing modules...

- compute in modules/compute

- vpc in modules/vpc

Initializing the backend...

Initializing provider plugins...

- Finding hashicorp/aws versions matching "~> 3.44.0"...

- Installing hashicorp/aws v3.44.0...

- Installed hashicorp/aws v3.44.0 (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.

> terraform validate

Success! The configuration is valid.

> terraform apply

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:

# module.compute.aws\_instance.docker will be created

+ resource "aws\_instance" "docker" {

+ ami = (sensitive)

+ arn = (known after apply)

+ associate\_public\_ip\_address = true

+ availability\_zone = (known after apply)

+ cpu\_core\_count = (known after apply)

+ cpu\_threads\_per\_core = (known after apply)

+ get\_password\_data = false

+ host\_id = (known after apply)

+ id = (known after apply)

+ instance\_initiated\_shutdown\_behavior = (known after apply)

+ instance\_state = (known after apply)

+ instance\_type = "t2.micro"

+ ipv6\_address\_count = (known after apply)

+ ipv6\_addresses = (known after apply)

+ key\_name = "docker"

+ outpost\_arn = (known after apply)

+ password\_data = (known after apply)

+ placement\_group = (known after apply)

+ primary\_network\_interface\_id = (known after apply)

+ private\_dns = (known after apply)

+ private\_ip = (known after apply)

+ public\_dns = (known after apply)

+ public\_ip = (known after apply)

+ secondary\_private\_ips = (known after apply)

+ security\_groups = (known after apply)

+ source\_dest\_check = true

+ subnet\_id = (known after apply)

+ tags = {

+ "Name" = "docker\_tf"

}

+ tags\_all = {

+ "Name" = "docker\_tf"

}

+ tenancy = (known after apply)

+ vpc\_security\_group\_ids = (known after apply)

+ capacity\_reservation\_specification {

+ capacity\_reservation\_preference = (known after apply)

+ capacity\_reservation\_target {

+ capacity\_reservation\_id = (known after apply)

}

}

+ ebs\_block\_device {

+ delete\_on\_termination = (known after apply)

+ device\_name = (known after apply)

+ encrypted = (known after apply)

+ iops = (known after apply)

+ kms\_key\_id = (known after apply)

+ snapshot\_id = (known after apply)

+ tags = (known after apply)

+ throughput = (known after apply)

+ volume\_id = (known after apply)

+ volume\_size = (known after apply)

+ volume\_type = (known after apply)

}

+ enclave\_options {

+ enabled = (known after apply)

}

+ ephemeral\_block\_device {

+ device\_name = (known after apply)

+ no\_device = (known after apply)

+ virtual\_name = (known after apply)

}

+ metadata\_options {

+ http\_endpoint = (known after apply)

+ http\_put\_response\_hop\_limit = (known after apply)

+ http\_tokens = (known after apply)

}

+ network\_interface {

+ delete\_on\_termination = (known after apply)

+ device\_index = (known after apply)

+ network\_interface\_id = (known after apply)

}

+ root\_block\_device {

+ delete\_on\_termination = (known after apply)

+ device\_name = (known after apply)

+ encrypted = (known after apply)

+ iops = (known after apply)

+ kms\_key\_id = (known after apply)

+ tags = (known after apply)

+ throughput = (known after apply)

+ volume\_id = (known after apply)

+ volume\_size = (known after apply)

+ volume\_type = (known after apply)

}

}

# module.compute.aws\_key\_pair.aws-key will be created

+ resource "aws\_key\_pair" "aws-key" {

+ arn = (known after apply)

+ fingerprint = (known after apply)

+ id = (known after apply)

+ key\_name = "docker"

+ key\_pair\_id = (known after apply)

+ public\_key = "ssh-rsa  paulkrznaric@Pauls-MBP.local"

+ tags\_all = (known after apply)

}

# module.vpc.aws\_internet\_gateway.tf\_igw will be created

+ resource "aws\_internet\_gateway" "tf\_igw" {

+ arn = (known after apply)

+ id = (known after apply)

+ owner\_id = (known after apply)

+ tags = {

+ "Name" = "Terraform-Gateway"

}

+ tags\_all = {

+ "Name" = "Terraform-Gateway"

}

+ vpc\_id = (known after apply)

}

# module.vpc.aws\_route\_table.tf\_public\_route will be created

+ resource "aws\_route\_table" "tf\_public\_route" {

+ arn = (known after apply)

+ id = (known after apply)

+ owner\_id = (known after apply)

+ propagating\_vgws = (known after apply)

+ route = [

+ {

+ carrier\_gateway\_id = ""

+ cidr\_block = "0.0.0.0/0"

+ destination\_prefix\_list\_id = ""

+ egress\_only\_gateway\_id = ""

+ gateway\_id = (known after apply)

+ instance\_id = ""

+ ipv6\_cidr\_block = ""

+ local\_gateway\_id = ""

+ nat\_gateway\_id = ""

+ network\_interface\_id = ""

+ transit\_gateway\_id = ""

+ vpc\_endpoint\_id = ""

+ vpc\_peering\_connection\_id = ""

},

]

+ tags = {

+ "Name" = "Terraform-Public-RouteTable"

}

+ tags\_all = {

+ "Name" = "Terraform-Public-RouteTable"

}

+ vpc\_id = (known after apply)

}

# module.vpc.aws\_route\_table\_association.tf\_public\_assoc will be created

+ resource "aws\_route\_table\_association" "tf\_public\_assoc" {

+ id = (known after apply)

+ route\_table\_id = (known after apply)

+ subnet\_id = (known after apply)

}

# module.vpc.aws\_security\_group.tf\_public\_sg will be created

+ resource "aws\_security\_group" "tf\_public\_sg" {

+ arn = (known after apply)

+ description = "Used for access to the public instances"

+ egress = [

+ {

+ cidr\_blocks = [

+ "0.0.0.0/0",

]

+ description = ""

+ from\_port = 0

+ ipv6\_cidr\_blocks = []

+ prefix\_list\_ids = []

+ protocol = "-1"

+ security\_groups = []

+ self = false

+ to\_port = 0

},

]

+ id = (known after apply)

+ ingress = [

+ {

+ cidr\_blocks = [

+ "0.0.0.0/0",

]

+ description = "Allow SSH traffic"

+ from\_port = 22

+ ipv6\_cidr\_blocks = []

+ prefix\_list\_ids = []

+ protocol = "tcp"

+ security\_groups = []

+ self = false

+ to\_port = 22

},

+ {

+ cidr\_blocks = [

+ "0.0.0.0/0",

]

+ description = "allow traffic from TCP/80"

+ from\_port = 80

+ ipv6\_cidr\_blocks = []

+ prefix\_list\_ids = []

+ protocol = "tcp"

+ security\_groups = []

+ self = false

+ to\_port = 80

},

]

+ name = "tf\_public\_sg"

+ name\_prefix = (known after apply)

+ owner\_id = (known after apply)

+ revoke\_rules\_on\_delete = false

+ tags = {

+ "Name" = "Terraform-SecurityGroup"

}

+ tags\_all = {

+ "Name" = "Terraform-SecurityGroup"

}

+ vpc\_id = (known after apply)

}

# module.vpc.aws\_subnet.tf\_public\_subnet will be created

+ resource "aws\_subnet" "tf\_public\_subnet" {

+ arn = (known after apply)

+ assign\_ipv6\_address\_on\_creation = false

+ availability\_zone = "us-east-1a"

+ availability\_zone\_id = (known after apply)

+ cidr\_block = "10.0.1.0/24"

+ id = (known after apply)

+ ipv6\_cidr\_block\_association\_id = (known after apply)

+ map\_public\_ip\_on\_launch = false

+ owner\_id = (known after apply)

+ tags = {

+ "Name" = "Terraform-Subnet"

}

+ tags\_all = {

+ "Name" = "Terraform-Subnet"

}

+ vpc\_id = (known after apply)

}

# module.vpc.aws\_vpc.tf\_vpc will be created

+ resource "aws\_vpc" "tf\_vpc" {

+ arn = (known after apply)

+ assign\_generated\_ipv6\_cidr\_block = false

+ cidr\_block = "10.0.0.0/16"

+ default\_network\_acl\_id = (known after apply)

+ default\_route\_table\_id = (known after apply)

+ default\_security\_group\_id = (known after apply)

+ dhcp\_options\_id = (known after apply)

+ enable\_classiclink = (known after apply)

+ enable\_classiclink\_dns\_support = (known after apply)

+ enable\_dns\_hostnames = true

+ enable\_dns\_support = true

+ id = (known after apply)

+ instance\_tenancy = "default"

+ ipv6\_association\_id = (known after apply)

+ ipv6\_cidr\_block = (known after apply)

+ main\_route\_table\_id = (known after apply)

+ owner\_id = (known after apply)

+ tags = {

+ "Name" = "Terraform-VPC"

}

+ tags\_all = {

+ "Name" = "Terraform-VPC"

}

}

Plan: 8 to add, 0 to change, 0 to destroy.

Changes to Outputs:

+ Sever-Public-IP = (known after apply)

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

module.vpc.aws\_vpc.tf\_vpc: Creating...

module.compute.aws\_key\_pair.aws-key: Creating...

module.compute.aws\_key\_pair.aws-key: Creation complete after 1s [id=docker]

module.vpc.aws\_vpc.tf\_vpc: Still creating... [10s elapsed]

module.vpc.aws\_vpc.tf\_vpc: Creation complete after 14s [id=vpc-07c96ca6a731cf194]

module.vpc.aws\_internet\_gateway.tf\_igw: Creating...

module.vpc.aws\_subnet.tf\_public\_subnet: Creating...

module.vpc.aws\_security\_group.tf\_public\_sg: Creating...

module.vpc.aws\_subnet.tf\_public\_subnet: Creation complete after 1s [id=subnet-09e42e0326ec31bb5]

module.vpc.aws\_internet\_gateway.tf\_igw: Creation complete after 1s [id=igw-0c0d04f5cddc40441]

module.vpc.aws\_route\_table.tf\_public\_route: Creating...

module.vpc.aws\_route\_table.tf\_public\_route: Creation complete after 1s [id=rtb-092ba09e20ae6ea94]

module.vpc.aws\_route\_table\_association.tf\_public\_assoc: Creating...

module.vpc.aws\_route\_table\_association.tf\_public\_assoc: Creation complete after 0s [id=rtbassoc-0706c3507feef0b5d]

module.vpc.aws\_security\_group.tf\_public\_sg: Creation complete after 3s [id=sg-02542384ac1406fe4]

module.compute.aws\_instance.docker: Creating...

module.compute.aws\_instance.docker: Still creating... [10s elapsed]

module.compute.aws\_instance.docker: Still creating... [20s elapsed]

module.compute.aws\_instance.docker: Still creating... [30s elapsed]

module.compute.aws\_instance.docker: Still creating... [40s elapsed]

module.compute.aws\_instance.docker: Creation complete after 43s [id=i-07c7a99d86a2ce1bd]

Apply complete! Resources: 8 added, 0 changed, 0 destroyed.

Outputs:

Sever-Public-IP = "3.95.226.143"

Graphical user interface, application, website

Description automatically generated

## SSH:

Text

Description automatically generated

## Docker-Compose Version Graphical user interface, text Description automatically generated

## Docker-compose up -d

Text

Description automatically generated

## Ghost Blog

Graphical user interface, website

Description automatically generated