The ModuleDemo/Custommodules/customvpc/outputs.tf file is created with the following code:

output "vpc\_id" {

value = aws\_vpc.devVPC.id

}

output "aws\_internet\_gateway" {

value = aws\_internet\_gateway.igw.id

}

output "public\_subnet" {

value = aws\_subnet.public\_subnet.id

}

The ModuleDemo/Custommodules/customvpc/terraformVPC.tf file is created with the following code:

#Providers are a logical abstraction of an upstream API. They help to understand API interactions and exposing provider resources such AWS, Google,

provider "aws" {

region = var.apac\_region

}

# Query all available Availability Zone; we will use specific availability zone using index - The Availability Zones data source provides access to the list of AWS Availability Zones which can be accessed by an AWS account specific to region configured in the provider.

data "aws\_availability\_zones" "aws\_az" {}

# Provides a VPC resource - https://registry.terraform.io/providers/hashicorp/aws/latest/docs/resources/vpc

resource "aws\_vpc" "devVPC" {

cidr\_block = var.vpc\_cidr\_block

enable\_dns\_hostnames = true # default is false

tags = {

Name = "dev\_terraform\_vpc"

}

}

# Public Subnet - Provides an VPC subnet resource - https://registry.terraform.io/providers/hashicorp/aws/latest/docs/resources/subnet.

resource "aws\_subnet" "public\_subnet" {

cidr\_block = var.public\_cidr

vpc\_id = aws\_vpc.devVPC.id

map\_public\_ip\_on\_launch = true

availability\_zone = data.aws\_availability\_zones.aws\_az.names[1]

tags = {

Name = "dev\_terraform\_vpc\_public\_subnet"

}

}

# Private Subnet - Provides an VPC subnet resource - https://registry.terraform.io/providers/hashicorp/aws/latest/docs/resources/subnet.

resource "aws\_subnet" "private\_subnet" {

cidr\_block = var.private\_cidr

vpc\_id = aws\_vpc.devVPC.id

map\_public\_ip\_on\_launch = false

availability\_zone = data.aws\_availability\_zones.aws\_az.names[1]

tags = {

Name = "dev\_terraform\_vpc\_private\_subnet"

}

}

#To access EC2 instance inside a Virtual Private Cloud (VPC) we need an Internet Gateway and a routing table connecting the subnet to the Internet Gateway

# Creating Internet Gateway

# Provides a resource to create a VPC Internet Gateway - https://registry.terraform.io/providers/hashicorp/aws/latest/docs/resources/internet\_gateway.

resource "aws\_internet\_gateway" "igw" {

vpc\_id = aws\_vpc.devVPC.id

tags = {

Name = "dev\_terraform\_vpc\_igw"

}

}

# Provides a resource to create a VPC routing table - https://registry.terraform.io/providers/hashicorp/aws/latest/docs/resources/route\_table

resource "aws\_route\_table" "public\_route" {

vpc\_id = aws\_vpc.devVPC.id

route {

cidr\_block = var.cidr\_blocks

gateway\_id = aws\_internet\_gateway.igw.id

}

tags = {

Name = "dev\_terraform\_vpc\_public\_route"

}

}

# Provides a resource to create an association between a Public Route Table and a Public Subnet - https://registry.terraform.io/providers/hashicorp/aws/latest/docs/resources/route\_table\_association

resource "aws\_route\_table\_association" "public\_subnet\_association" {

route\_table\_id = aws\_route\_table.public\_route.id

subnet\_id = aws\_subnet.public\_subnet.id

depends\_on = [aws\_route\_table.public\_route, aws\_subnet.public\_subnet]

}

The ModuleDemo/Custommodules/customvpc/vars.tf file is created with the following code:

# Input Variables https://www.terraform.io/language/values/variables

variable "apac\_region" {

default = "ap-south-1"

}

variable "cidr\_blocks" {

}

#Network Mask - 255.255.255.0 Addresses Available - 256

variable "vpc\_cidr\_block" {

}

variable "public\_cidr" {

}

variable "private\_cidr" {

}

The ModuleDemo/autoscaling.tf file is created with the following code (remember to input your OwnerID for the AMI ID filter):

resource "aws\_launch\_configuration" "nginx\_launch\_config" {

image\_id = data.aws\_ami.packeramis.id

instance\_type = var.instance\_type

security\_groups = [aws\_security\_group.sg\_allow\_ssh\_http.id]

user\_data = data.template\_file.init.rendered

lifecycle {

create\_before\_destroy = true

}

}

resource "aws\_autoscaling\_group" "nginx\_autoscaling\_group" {

launch\_configuration = aws\_launch\_configuration.nginx\_launch\_config.id

vpc\_zone\_identifier = [module.customvpc.public\_subnet]

health\_check\_type = "ELB"

min\_size = 2

max\_size = 5

load\_balancers = [aws\_elb.nginx-elb.id]

tag {

key = "Name"

value = "dev\_terraform\_nginx\_instance\_asg"

propagate\_at\_launch = true

}

}

resource "aws\_autoscaling\_policy" "nginx\_cpu\_policy\_scaleup" {

name = "nginx\_cpu\_policy\_scaleup"

autoscaling\_group\_name = aws\_autoscaling\_group.nginx\_autoscaling\_group.name

adjustment\_type = "ChangeInCapacity"

scaling\_adjustment = 1

cooldown = "120"

}

resource "aws\_autoscaling\_policy" "nginx\_cpu\_policy\_scaledown" {

name = "nginx\_cpu\_policy\_scaledown"

autoscaling\_group\_name = aws\_autoscaling\_group.nginx\_autoscaling\_group.name

adjustment\_type = "ChangeInCapacity"

scaling\_adjustment = -1

cooldown = "120"

}

#Get latest AMI ID based on Filter - Here AMI created using Packer

data "aws\_ami" "packeramis" {

owners = ["1xxxxxxxxxxx"] #change the owner ID as per your account

most\_recent = true

filter {

name = "name"

values = ["packer-nginx\*"]

}

}

#The template\_file data source usually loaded from an external file.

data "template\_file" "init" {

template = file("${path.module}/userdata.tpl")

}

The ModuleDemo/elb.tf file is created with the following code:

# Elastic Load Balancer resource, also known as a Classic Load Balancer - https://registry.terraform.io/providers/hashicorp/aws/latest/docs/resources/elb

resource "aws\_elb" "nginx-elb" {

name = "nginx-elb"

subnets = [module.customvpc.public\_subnet]

security\_groups = [aws\_security\_group.sg\_allow\_ssh\_http.id]

listener {

instance\_port = 80

instance\_protocol = "http"

lb\_port = 80

lb\_protocol = "http"

}

health\_check {

healthy\_threshold = 3

unhealthy\_threshold = 3

timeout = 3

target = "HTTP:80/"

interval = 30

}

tags = {

Name = "nginx\_elb"

}

}

The ModuleDemo/devVPC.tf file is created with the following code:

provider "aws" {

region = var.apac\_region

}

module "customvpc" {

source = "./Custommodules/customvpc"

region = var.apac\_region

cidr\_blocks = "0.0.0.0/0"

vpc\_cidr\_block = "10.0.1.0/24"

public\_cidr = "10.0.1.0/28"

private\_cidr = "10.0.1.16/28"

}

In this code, note the source = "./Custommodules/customvpc". We are providing the path of the module here.

The ModuleDemo/securitygroups.tf file is created with the following code:

# Provides a security group resource - https://registry.terraform.io/providers/hashicorp/aws/latest/docs/resources/security\_group

resource "aws\_security\_group" "sg\_allow\_ssh\_http" {

vpc\_id = module.customvpc.vpc\_id

name = "dev\_terraform\_vpc\_allow\_ssh\_http"

tags = {

Name = "dev\_terraform\_sg\_allow\_ssh\_http"

}

}

# Ingress Security Port 22 (Inbound) - Provides a security group rule resource (https://registry.terraform.io/providers/hashicorp/aws/latest/docs/resources/security\_group\_rule)

resource "aws\_security\_group\_rule" "ssh\_ingress\_access" {

from\_port = 22

protocol = "tcp"

security\_group\_id = aws\_security\_group.sg\_allow\_ssh\_http.id

to\_port = 22

type = "ingress"

cidr\_blocks = [var.cidr\_blocks]

}

# Ingress Security Port 80 (Inbound)

resource "aws\_security\_group\_rule" "http\_ingress\_access" {

from\_port = 80

protocol = "tcp"

security\_group\_id = aws\_security\_group.sg\_allow\_ssh\_http.id

to\_port = 80

type = "ingress"

cidr\_blocks = [var.cidr\_blocks]

}

# Egress Security (Outbound)

resource "aws\_security\_group\_rule" "egress\_access" {

from\_port = 0

protocol = "-1"

security\_group\_id = aws\_security\_group.sg\_allow\_ssh\_http.id

to\_port = 0

type = "egress"

cidr\_blocks = [var.cidr\_blocks]

}

The ModuleDemo/userdata.tpl file is created with the following code:

#!/bin/bash

sudo service nginx start

hostname >> /usr/share/nginx/html/index.html

The ModuleDemo/vars.tf file is created with the following code:

# Input Variables https://www.terraform.io/language/values/variables

variable "apac\_region" {

default = "ap-south-1"

}

variable "cidr\_blocks" {

default = "0.0.0.0/0"

}

variable "instance\_type" {

default = "t2.small"

}

The ModuleDemo/firstawsami-packer.json file is created with the following code:

{

"variables": {

"aws\_access\_key": "",

"aws\_secret\_key": ""

},

"builders": [

{

"type": "amazon-ebs",

"access\_key": "{{user `aws\_access\_key`}}",

"secret\_key": "{{user `aws\_secret\_key`}}",

"region": "ap-south-1",

"source\_ami": "ami-0dafa01c8100180f8",

"instance\_type": "t2.small",

"ssh\_username": "ec2-user",

"ami\_name": "packer-nginx-ami-{{timestamp}}"

}

],

"provisioners": [

{

"type": "shell",

"inline": [

"sudo yum update –y",

"sudo amazon-linux-extras install -y nginx1.12"

]

}

]

}

Execute packer build -var "aws\_access\_key=XXXXXXXXXXXXXXXXXXXX" -var "aws\_secret\_key=XXXXXXXXXXXXXXXXXXXX" firstawsami-packer.json

\*\* Let’s push code into the GitHub repository and use it in a Terraform file using GitHub URL:

We need to make changes in the devVPC.tf file - source = "github.com/terraform-home/custommoduledemo/Custommodules//customvpc":

provider "aws" {

region = var.apac\_region

}

module "customvpc" {

source = "github.com/terraform-home/custommoduledemo/Custommodules//customvpc"

region = var.apac\_region

cidr\_blocks = "0.0.0.0/0"

vpc\_cidr\_block = "10.0.1.0/24"

public\_cidr = "10.0.1.0/28"

private\_cidr = "10.0.1.16/28"

}