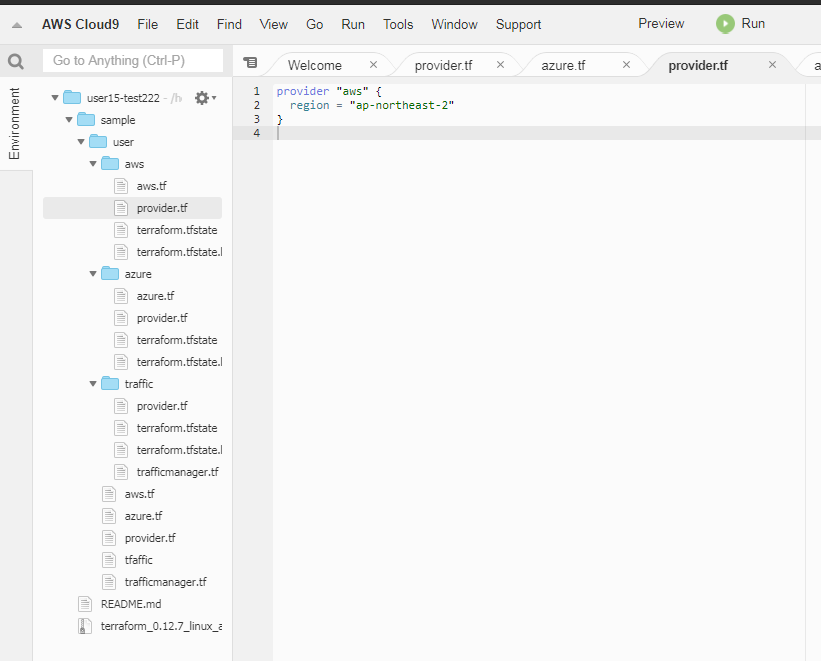
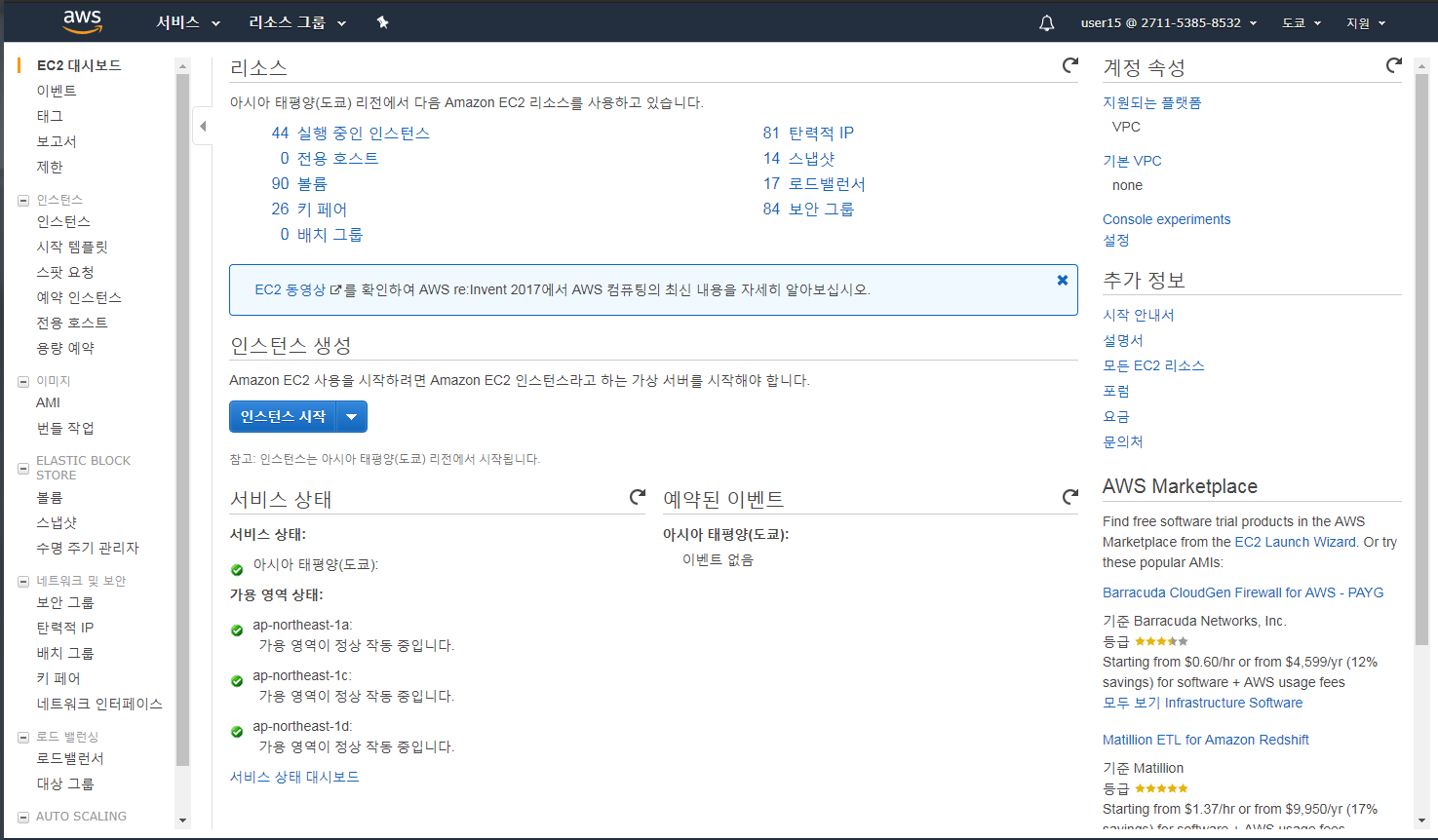
AWS의 provider.tf



리전별

ap-northeast-2 (서울) – 알려주는데로…

aws-EC2로 확인. 아래 서비스 상태 부분에 어떤 리전인지? 1a, 1b, 1c 선택함. (두개 자유롭게, 앞에 2개 권장)



**AWS의 aws.tf**

resource "aws\_vpc" "awsvpc" {

cidr\_block = "197.0.0.0/16"

enable\_dns\_hostnames = true

enable\_dns\_support = true

instance\_tenancy = "default"

tags = {

Name = "user15vpc"

}

}

resource "aws\_internet\_gateway" "awsipg" {

vpc\_id = "${aws\_vpc.awsvpc.id}"

tags = {

Name = "user15igw"

}

}

resource "aws\_subnet" "public\_1a" {

vpc\_id = "${aws\_vpc.awsvpc.id}"

availability\_zone = "ap-northeast-2a"

cidr\_block = "197.0.1.0/24"

tags = {

Name = "user15subnet1"

}

}

resource "aws\_subnet" "public\_1b" {

vpc\_id = "${aws\_vpc.awsvpc.id}"

availability\_zone = "ap-northeast-2c"

cidr\_block = "197.0.2.0/24"

tags = {

Name = "user15subnet2"

}

}

resource "aws\_eip" "awseip1" {

vpc = false

tags = {

Name = "user15eip1"

}

}

resource "aws\_eip" "awseip2" {

vpc = false

tags = {

Name = "user15eip2"

}

}

resource "aws\_nat\_gateway" "natgate\_1a" {

allocation\_id = "${aws\_eip.awseip1.id}"

subnet\_id = "${aws\_subnet.public\_1a.id}"

tags = {

Name = "user15ngw"

}

}

resource "aws\_nat\_gateway" "natgate\_1b" {

allocation\_id = "${aws\_eip.awseip2.id}"

subnet\_id = "${aws\_subnet.public\_1b.id}"

tags = {

Name = "user15ngw"

}

}

resource "aws\_route\_table" "awsrtp" {

vpc\_id = "${aws\_vpc.awsvpc.id}"

route {

cidr\_block = "0.0.0.0/0"

gateway\_id = "${aws\_internet\_gateway.awsipg.id}"

}

}

resource "aws\_route\_table\_association" "awsrtp1a" {

subnet\_id = "${aws\_subnet.public\_1a.id}"

route\_table\_id = "${aws\_route\_table.awsrtp.id}"

}

resource "aws\_route\_table\_association" "awsrtp1b" {

subnet\_id = "${aws\_subnet.public\_1b.id}"

route\_table\_id = "${aws\_route\_table.awsrtp.id}"

}

resource "aws\_default\_security\_group" "awssecurity" {

vpc\_id = "${aws\_vpc.awsvpc.id}"

ingress {

protocol = -1

self = true

from\_port = 0

to\_port = 0

}

egress {

from\_port = 0

to\_port = 0

protocol = "-1"

cidr\_blocks = ["0.0.0.0/0"]

}

tags = {

Name = "user15sg"

}

}

resource "aws\_default\_network\_acl" "awsnetworkacl" {

default\_network\_acl\_id = "${aws\_vpc.awsvpc.default\_network\_acl\_id}"

ingress {

protocol = -1

rule\_no = 100

action = "allow"

cidr\_block = "0.0.0.0/0"

from\_port = 0

to\_port = 0

}

egress {

protocol = -1

rule\_no = 100

action = "allow"

cidr\_block = "0.0.0.0/0"

from\_port = 0

to\_port = 0

}

subnet\_ids = [

"${aws\_subnet.public\_1a.id}",

"${aws\_subnet.public\_1b.id}",

]

}

variable "amazon\_linux" {

# Amazon Linux AMI 2017.03.1 (HVM), SSD Volume Type - ami-4af5022c

default = "ami-0be3e6f84d3b968cd" 🡪 리전에 맞는 ami 찾아오기

}

resource "aws\_security\_group" "webserverSecurutyGroup" {

name = "user15webserverSecurutyGroup"

description = "open ssh port for webserverSecurutyGroup"

vpc\_id = "${aws\_vpc.awsvpc.id}"

ingress {

from\_port = 22

to\_port = 22

protocol = "tcp"

cidr\_blocks = ["0.0.0.0/0"]

}

ingress {

from\_port = 80

to\_port = 80

protocol = "tcp"

cidr\_blocks = ["0.0.0.0/0"]

}

egress {

from\_port = 0

to\_port = 0

protocol = "-1"

cidr\_blocks = ["0.0.0.0/0"]

}

}

resource "aws\_instance" "web1" {

ami = "${var.amazon\_linux}"

availability\_zone = "ap-northeast-2a"

instance\_type = "t2.micro"

key\_name = "user15-key"

vpc\_security\_group\_ids = [

"${aws\_security\_group.webserverSecurutyGroup.id}",

"${aws\_default\_security\_group.awssecurity.id}",

]

subnet\_id = "${aws\_subnet.public\_1a.id}"

associate\_public\_ip\_address = true

tags = {

Name = "user15web1"

}

}

resource "aws\_instance" "web2" {

ami = "${var.amazon\_linux}"

availability\_zone = "ap-northeast-2c"

instance\_type = "t2.micro"

key\_name = "user15-key"

vpc\_security\_group\_ids = [

"${aws\_security\_group.webserverSecurutyGroup.id}",

"${aws\_default\_security\_group.awssecurity.id}",

]

subnet\_id = "${aws\_subnet.public\_1b.id}"

associate\_public\_ip\_address = true

tags = {

Name = "user15web2"

}

}

resource "aws\_alb" "frontend" {

name = "alb2user15"

internal = false

security\_groups = ["${aws\_security\_group.webserverSecurutyGroup.id}"]

subnets = [

"${aws\_subnet.public\_1a.id}",

"${aws\_subnet.public\_1b.id}"

]

lifecycle { create\_before\_destroy = true }

}

resource "aws\_alb\_target\_group" "frontendalb" {

name = "frontendtargetgroupuser15"

port = 80

protocol = "HTTP"

vpc\_id = "${aws\_vpc.awsvpc.id}"

health\_check {

interval = 30

path = "/"

healthy\_threshold = 3

unhealthy\_threshold = 3

}

}

resource "aws\_alb\_target\_group\_attachment" "frontend1" {

target\_group\_arn = "${aws\_alb\_target\_group.frontendalb.arn}"

target\_id = "${aws\_instance.web1.id}"

port = 80

}

resource "aws\_alb\_target\_group\_attachment" "frontend2" {

target\_group\_arn = "${aws\_alb\_target\_group.frontendalb.arn}"

target\_id = "${aws\_instance.web2.id}"

port = 80

}

resource "aws\_alb\_listener" "http" {

load\_balancer\_arn = "${aws\_alb.frontend.arn}"

port = "80"

protocol = "HTTP"

default\_action {

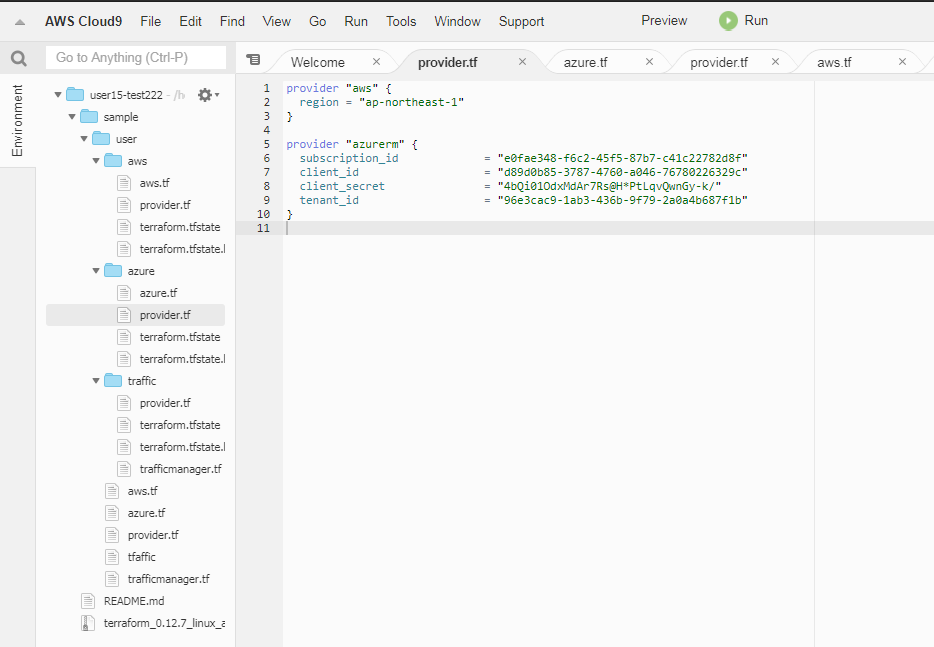
target\_group\_arn = "${aws\_alb\_target\_group.frontendalb.arn}"

type = "forward"

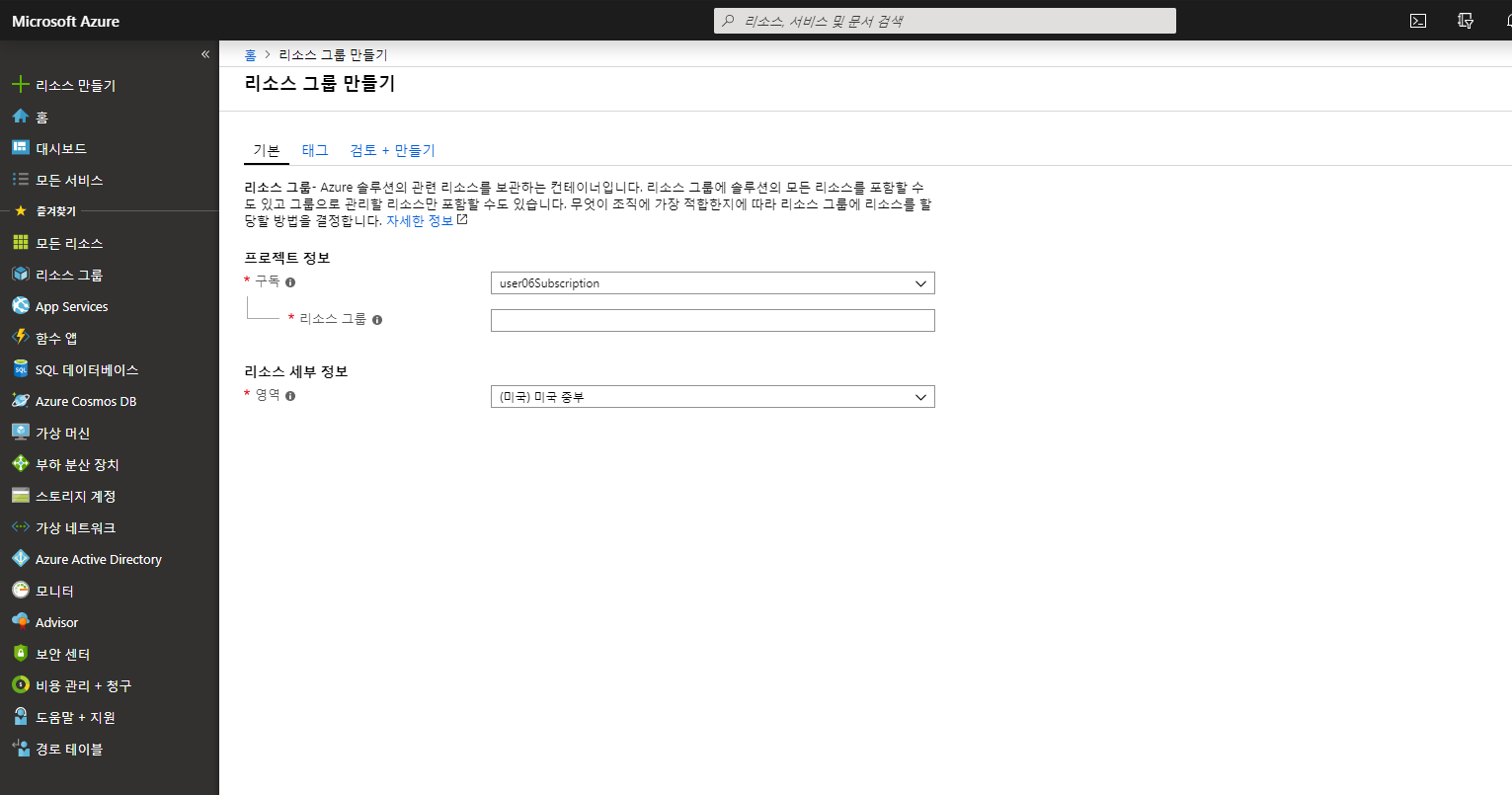
}

}

Azure의 provider.tf



Aws의 리전은 없어도 됨.



요기서 리전 확인

**Azure의 azure.tf**

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

name = "user15test"

location = "koreacentral"

}

variable "application\_port" {

description = "The port that you want to expose to the external load balancer"

default = 80

}

resource "azurerm\_network\_security\_group" "secGroup" {

name = "secGroupuser15test"

location = "koreacentral"

resource\_group\_name ="${azurerm\_resource\_group.rg.name}"

security\_rule {

name ="SSH"

priority = "1001"

direction = "Inbound"

access = "Allow"

protocol = "Tcp"

source\_port\_range = "\*"

destination\_port\_range = "22"

source\_address\_prefix = "\*"

destination\_address\_prefix = "\*"

}

security\_rule {

name = "HTTP"

priority = "2001"

direction = "Inbound"

access = "Allow"

protocol = "Tcp"

source\_port\_range = "\*"

destination\_port\_range = "80"

source\_address\_prefix = "\*"

destination\_address\_prefix = "\*"

}

}

resource "azurerm\_virtual\_network" "vnetwork" {

name = "user15test"

address\_space = ["97.0.0.0/16"]

location = "koreacentral"

resource\_group\_name = "${azurerm\_resource\_group.rg.name}"

}

resource "azurerm\_subnet" "mysubnet" {

name = "MySubnetuuser15test"

resource\_group\_name = "${azurerm\_resource\_group.rg.name}"

virtual\_network\_name = "${azurerm\_virtual\_network.vnetwork.name}"

network\_security\_group\_id = "${azurerm\_network\_security\_group.secGroup.id}"

address\_prefix = "97.0.1.0/24"

}

resource "azurerm\_public\_ip" "publicdomainip" {

name = "domainipuser15final"

location = "koreacentral"

resource\_group\_name = "${azurerm\_resource\_group.rg.name}"

allocation\_method = "Static"

domain\_name\_label = "user15testskcncazureip"

}

resource "azurerm\_public\_ip" "rdpip" {

name = "rdpipuser15test${count.index}"

location = "koreacentral"

resource\_group\_name = "${azurerm\_resource\_group.rg.name}"

allocation\_method = "Dynamic"

count = 2

}

resource "azurerm\_lb" "lb" {

resource\_group\_name = "${azurerm\_resource\_group.rg.name}"

name = "user15loadbaltest"

location = "koreacentral"

frontend\_ip\_configuration {

name = "LoadBalancerFrontEnd"

public\_ip\_address\_id = "${azurerm\_public\_ip.publicdomainip.id}"

}

}

resource "azurerm\_lb\_backend\_address\_pool" "bp" {

resource\_group\_name = "${azurerm\_resource\_group.rg.name}"

loadbalancer\_id = "${azurerm\_lb.lb.id}"

name = "BackendPooluser15final"

}

resource "azurerm\_network\_interface\_backend\_address\_pool\_association" "bpAS" {

count = 2

network\_interface\_id = "${element(azurerm\_network\_interface.nic.\*.id, count.index)}"

ip\_configuration\_name = "ipconfig${count.index}"

backend\_address\_pool\_id = "${azurerm\_lb\_backend\_address\_pool.bp.id}"

}

resource "azurerm\_lb\_probe" "lb\_probe" {

resource\_group\_name = "${azurerm\_resource\_group.rg.name}"

loadbalancer\_id = "${azurerm\_lb.lb.id}"

name = "user15tcpProbetest"

protocol = "tcp"

port = 80

interval\_in\_seconds = 5

number\_of\_probes = 2

}

resource "azurerm\_lb\_rule" "lb\_rule" {

resource\_group\_name = "${azurerm\_resource\_group.rg.name}"

loadbalancer\_id = "${azurerm\_lb.lb.id}"

name = "user15LBRuletest"

protocol = "tcp"

frontend\_port = "${var.application\_port}"

backend\_port = "${var.application\_port}"

frontend\_ip\_configuration\_name = "LoadBalancerFrontEnd"

enable\_floating\_ip = false

backend\_address\_pool\_id = "${azurerm\_lb\_backend\_address\_pool.bp.id}"

idle\_timeout\_in\_minutes = 5

probe\_id = "${azurerm\_lb\_probe.lb\_probe.id}"

depends\_on = ["azurerm\_lb\_probe.lb\_probe"]

}

resource "azurerm\_network\_interface" "nic" {

name = "user15testnic${count.index}"

location = "koreacentral"

resource\_group\_name = "${azurerm\_resource\_group.rg.name}"

network\_security\_group\_id = "${azurerm\_network\_security\_group.secGroup.id}"

count = 2

ip\_configuration {

name = "ipconfig${count.index}"

subnet\_id = "${azurerm\_subnet.mysubnet.id}"

private\_ip\_address\_allocation = "Dynamic"

public\_ip\_address\_id = "${length(azurerm\_public\_ip.rdpip.\*.id) > 0 ? element(concat(azurerm\_public\_ip.rdpip.\*.id, list("")), count.index) : ""}"

}

}

resource "azurerm\_availability\_set" "avset" {

name = "avsetuser15test"

location = "koreacentral"

resource\_group\_name = "${azurerm\_resource\_group.rg.name}"

platform\_fault\_domain\_count = 2

platform\_update\_domain\_count = 2

managed = true

}

resource "azurerm\_virtual\_machine" "vm" {

name = "user15vmtest${count.index}"

location = "koreacentral"

resource\_group\_name = "${azurerm\_resource\_group.rg.name}"

availability\_set\_id = "${azurerm\_availability\_set.avset.id}"

network\_interface\_ids = ["${element(azurerm\_network\_interface.nic.\*.id, count.index)}"]

vm\_size = "Standard\_D1\_v2"

count = 2

storage\_image\_reference {

publisher = "RedHat"

offer = "RHEL"

sku = "7.4"

version = "latest"

}

storage\_os\_disk {

name = "user15testosdisk${count.index}"

caching = "ReadWrite"

create\_option = "FromImage"

managed\_disk\_type = "Standard\_LRS"

}

os\_profile {

computer\_name = "user15vmtest${count.index}"

admin\_username = "azureuser"

admin\_password= "Passw0rd"

}

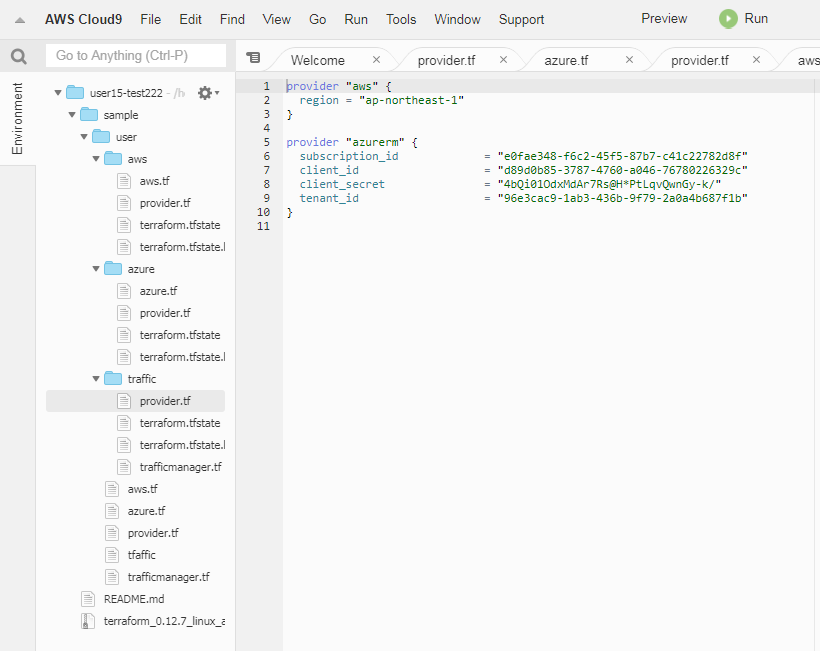
os\_profile\_linux\_config {

disable\_password\_authentication = false

}

}

**Trafficmanager 의 provider.tf**



Azure만 있으면 됨.

**Trafficmanager.tf**

resource "random\_id" "server" {

keepers = {

azi\_id = 1

}

byte\_length = 8

}

resource "azurerm\_traffic\_manager\_profile" "interprovider" {

name = "interprovider-trafficmanager"

resource\_group\_name = "user15test"

traffic\_routing\_method = "Weighted"

dns\_config {

relative\_name = "${random\_id.server.hex}"

ttl = 30

}

monitor\_config {

protocol = "http"

port = 80

path = "/"

}

}

resource "azurerm\_traffic\_manager\_endpoint" "azureLB" {

name = "first"

resource\_group\_name = "user15test"

profile\_name = "${azurerm\_traffic\_manager\_profile.interprovider.name}"

target = "user15testskcncazureip.koreacentral.cloudapp.azure.com"

type = "externalEndpoints"

weight = 1

}

resource "azurerm\_traffic\_manager\_endpoint" "awsLB" {

name = "second"

resource\_group\_name = "user15test"

profile\_name = "${azurerm\_traffic\_manager\_profile.interprovider.name}"

target = "alb2user15-770689960.ap-northeast-2.elb.amazonaws.com"

type = "externalEndpoints"

weight = 2

}