#Update Provider

#AWS as an provider

provider "aws" {

region = "us-east-1"

access\_key = "AKIATOGDQR4Y4YJSJ2UL"

secret\_key = "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"

}

# Creating VPC

resource "aws\_vpc" "demovpc" {

cidr\_block = "10.0.0.0/16"

instance\_tenancy = "default"

tags = {

Name = "Demo VPC"

}

}

Subnet.tf

# Creating 1st web subnet

resource "aws\_subnet" "public-subnet-1" {

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

cidr\_block = "10.0.0.0/24"

map\_public\_ip\_on\_launch = true

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

tags = {

Name = "Web Subnet 1"

}

}

# Creating 2nd web subnet

resource "aws\_subnet" "public-subnet-2" {

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

cidr\_block = "10.0.1.0/26"

map\_public\_ip\_on\_launch = true

availability\_zone = "us-east-1b"

tags = {

Name = "Web Subnet 2"

}

}

# Creating 1st application subnet

resource "aws\_subnet" "application-subnet-1" {

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

cidr\_block = "10.0.2.0/26"

map\_public\_ip\_on\_launch = false

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

tags = {

Name = "Application Subnet 1"

}

}

# Creating 2n lication subnet

resource "aws\_subnet" "application-subnet-2" {

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

cidr\_block = "10.0.3.0/28"

map\_public\_ip\_on\_launch = false

availability\_zone = "us-east-1b"

tags = {

Name = "Application Subnet 2"

}

}

# Create Database Private Subnet

resource "aws\_subnet" "database-subnet-1" {

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

cidr\_block = "10.0.4.0/24"

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

tags = {

Name = "Database Subnet 1"

}

}

# Create Database Private Subnet

resource "aws\_subnet" "database-subnet-2" {

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

cidr\_block = "10.0.5.0/28"

availability\_zone = "us-east-1b"

tags = {

Name = "Database Subnet 1"

}

}

Igw.tf script

# Creating Internet Gateway

resource "aws\_internet\_gateway" "demogateway" {

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

}

Route-table.tf

# Creating Route Table

resource "aws\_route\_table" "route" {

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

route {

cidr\_block = "0.0.0.0/0"

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

}

tags = {

Name = "Route to internet"

}

}

# Associating Route Table

resource "aws\_route\_table\_association" "rt1" {

subnet\_id = "${aws\_subnet.public-subnet-1.id}"

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

}

# Associating Route Table

resource "aws\_route\_table\_association" "rt2" {

subnet\_id = "${aws\_subnet.public-subnet-2.id}"

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

}

Data.sh

#!/bin/bash

sudo yum update -y

sudo yum install -y httpd.x86\_64

sudo systemctl start httpd.service

sudo systemctl enable httpd.service

sudo echo "Hello World from $(hostname -f)" > /var/www/html/index.html

sg.tf

# Creating Security Group

resource "aws\_security\_group" "demosg" {

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

# Inbound Rules

# HTTP access from anywhere

ingress {

from\_port = 80

to\_port = 80

protocol = "tcp"

cidr\_blocks = ["0.0.0.0/0"]

}

# HTTPS access from anywhere

ingress {

from\_port = 443

to\_port = 443

protocol = "tcp"

cidr\_blocks = ["0.0.0.0/0"]

}

# SSH access from anywhere

ingress {

from\_port = 22

to\_port = 22

protocol = "tcp"

cidr\_blocks = ["0.0.0.0/0"]

}

# Outbound Rules

# Internet access to anywhere

egress {

from\_port = 0

to\_port = 0

protocol = "-1"

cidr\_blocks = ["0.0.0.0/0"]

}

tags = {

Name = "Web SG"

}

}

Ec2.tf

# Creating 1st EC2 instance in Public Subnet

resource "aws\_instance" "demoinstance" {

ami = "ami-0b0dcb5067f052a63"

instance\_type = "t2.micro"

count = 1

key\_name = "iamclass"

vpc\_security\_group\_ids = ["${aws\_security\_group.demosg.id}"]

subnet\_id = "${aws\_subnet.public-subnet-1.id}"

associate\_public\_ip\_address = true

user\_data = "${file("data.sh")}"

tags = {

Name = "My Public Instance"

}

}

Database-sg.tf

# Create Database Security Group

resource "aws\_security\_group" "database-sg" {

name = "Database SG"

description = "Allow inbound traffic from application layer"

vpc\_id = aws\_vpc.demovpc.id

ingress {

description = "Allow traffic from application layer"

from\_port = 3306

to\_port = 3306

protocol = "tcp"

security\_groups = [aws\_security\_group.demosg.id]

}

egress {

from\_port = 32768

to\_port = 65535

protocol = "tcp"

cidr\_blocks = ["0.0.0.0/0"]

}

tags = {

Name = "Database SG"

}

}

# Creating 1st EC2 instance in Public Subnet

resource "aws\_instance" "demoinstance" {

ami = "ami-0b0dcb5067f052a63"

instance\_type = "t2.micro"

count = 1

key\_name = "iamclass"

vpc\_security\_group\_ids = ["${aws\_security\_group.demosg.id}"]

subnet\_id = "${aws\_subnet.public-subnet-1.id}"

associate\_public\_ip\_address = true

user\_data = "${file("data.sh")}"

tags = {

Name = "My Public Instance"

}

}

# Creating 1st EC2 instance in Public Subnet

resource "aws\_instance" "demoinstance1" {

ami = "ami-0b0dcb5067f052a63"

instance\_type = "t2.micro"

count = 1

key\_name = "iamclass"

vpc\_security\_group\_ids = ["${aws\_security\_group.demosg.id}"]

subnet\_id = "${aws\_subnet.public-subnet-1.id}"

associate\_public\_ip\_address = true

user\_data = "${file("data.sh")}"

tags = {

Name = "My Public Instance"

}

}

Alb.tf

# Creating External LoadBalancer

resource "aws\_lb" "external-alb" {

name = "External-LB"

internal = false

load\_balancer\_type = "application"

security\_groups = [aws\_security\_group.demosg.id]

subnets = [aws\_subnet.public-subnet-1.id, aws\_subnet.public-subnet-2.id]

}

resource "aws\_lb\_target\_group" "target-elb" {

name = "ALB-TG"

port = 80

protocol = "HTTP"

vpc\_id = aws\_vpc.demovpc.id

}

resource "aws\_lb\_target\_group\_attachment" "attachment" {

target\_group\_arn = aws\_lb\_target\_group.target-elb.arn

target\_id = aws\_instance.demoinstance.id

port = 80

depends\_on = [aws\_instance.demoinstance,]

}

resource "aws\_lb\_target\_group\_attachment" "attachment1" {

target\_group\_arn = aws\_lb\_target\_group.target-elb.arn

target\_id = aws\_instance.demoinstance1.id

port = 80

depends\_on = [aws\_instance.demoinstance1,]

}

resource "aws\_lb\_listener" "external-elb" {

load\_balancer\_arn = aws\_lb.external-alb.arn

port = 80

protocol = "HTTP"

default\_action {

type = "forward"

target\_group\_arn = aws\_lb\_target\_group.target-elb.arn

}

}

Rds.tf

# Creating RDS Instance

resource "aws\_db\_subnet\_group" "default" {

name = "main"

subnet\_ids = [aws\_subnet.database-subnet-1.id, aws\_subnet.database-subnet-2.id]

tags = {

Name = "My DB subnet group"

}

}

resource "aws\_db\_instance" "default" {

allocated\_storage = 10

db\_subnet\_group\_name = aws\_db\_subnet\_group.default.id

engine = "mysql"

engine\_version = "5.7"

instance\_class = "db.t2.micro"

multi\_az = true

db\_name = "mydb"

username = "admin"

password = "admin123"

skip\_final\_snapshot = true

vpc\_security\_group\_ids = [aws\_security\_group.database-sg.id] }

outputs.tf

# Getting the DNS of load balancer

output "lb\_dns\_name" {

description = "The DNS name of the load balancer"

value = "${aws\_lb.external-alb.dns\_name}"

}