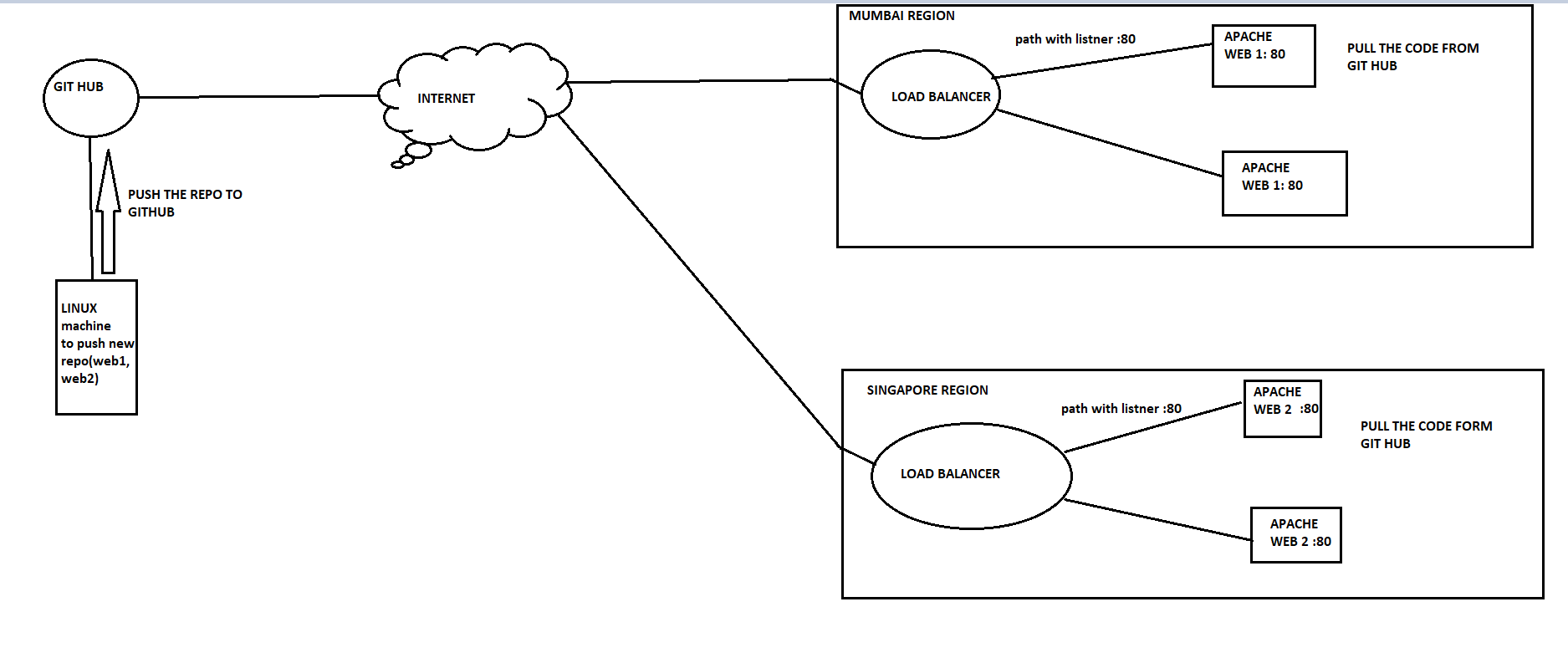
GIT configuration and hosting a web page under Load balancer on two diff region using Terraform

Architecture:



Terraform conf of creating an instance’s on two subnet under one vpc on different region (MUMBAI , SINGAPORE)

WORKFLOW:

Creating an instance where git is configured and having web page content.

And pushing to the git hub.

Now from the GITHUB Two instance’s on two region clone the repo which contains the webpage in /var/www/html

And connecting the instance’s under a LOAD BALANCER that evenly distribute the hit from the user’s.

TERRAFORM CONF:

provider "aws" {         #creating provider with alias name for mumbai

    alias = "mumbai"

    region = "ap-south-1"

}

provider "aws" {      #creating provider with alias name for singapore

    alias = "singapore"

    region = "ap-southeast-1"

}

resource "aws\_vpc" "mumbai\_vpc" { #creating vpc for webserver instances in mumbai

    tags = {

      Name= "Mumbai\_vpc"

    }

    provider = aws.mumbai

    cidr\_block = var.cidr\_block\_mumbai

    instance\_tenancy = "default"

}

resource "aws\_vpc" "mumbai\_server\_git" {  #creating vpc for instance contain webpage in mumbai

    tags = {

      Name= "mumbai\_servergit\_vpc"

    }

    cidr\_block = "10.0.0.0/16"

    instance\_tenancy = "default"

    provider = aws.mumbai

}

resource "aws\_vpc" "singapore\_vpc" { #creating vpc for webserver instances in singapore

    tags = {

      Name= "singapore\_vpc"

    }

    provider = aws.singapore

    cidr\_block = var.cidr\_block\_singapore

    instance\_tenancy = "default"

}

resource "aws\_subnet" "git\_server\_pub\_sub" {  #creating public subnet for instance contain webpage

    vpc\_id = aws\_vpc.mumbai\_server\_git.id

    provider = aws.mumbai

    cidr\_block = "10.0.1.0/24"

    map\_public\_ip\_on\_launch = true

    availability\_zone = "ap-south-1a"

    depends\_on = [ aws\_vpc.mumbai\_server\_git]

}

resource "aws\_subnet" "git\_pub\_sub1" {   #creating public subnet for instance contain websever in az1 mumbai

    vpc\_id = aws\_vpc.mumbai\_vpc.id

    provider = aws.mumbai

    cidr\_block = "172.16.2.0/24"

    map\_public\_ip\_on\_launch = true

    availability\_zone = "ap-south-1b"

    depends\_on = [ aws\_subnet.git\_server\_pub\_sub ]

}

resource "aws\_subnet" "git\_pub\_sub2" { #creating public subnet for instance contain webserver in az2 mumbai

    vpc\_id = aws\_vpc.mumbai\_vpc.id

    provider = aws.mumbai

    cidr\_block = "172.16.3.0/24"

    map\_public\_ip\_on\_launch = true

    availability\_zone = "ap-south-1a"

    depends\_on = [ aws\_subnet.git\_pub\_sub1 ]

}

resource "aws\_subnet" "git\_pub\_singapore\_sub1" { #creating public subnet for instance contain webserver in az1 singapore

    provider = aws.singapore

    vpc\_id = aws\_vpc.singapore\_vpc.id

    cidr\_block = "192.168.1.0/24"

    map\_public\_ip\_on\_launch = true

    availability\_zone = "ap-southeast-1a"

    depends\_on = [ aws\_vpc.singapore\_vpc ]

}

resource "aws\_subnet" "git\_pub\_singapore\_sub2" {  #creating public subnet for instance contain webserver in az2 singapore

    provider = aws.singapore

    vpc\_id = aws\_vpc.singapore\_vpc.id

    cidr\_block = "192.168.2.0/24"

    map\_public\_ip\_on\_launch = true

    availability\_zone = "ap-southeast-1b"

    depends\_on = [ aws\_subnet.git\_pub\_singapore\_sub1]

}

resource "aws\_internet\_gateway" "igw\_mumbai\_git" { #creating internet gateway for instances

    tags = {

      Name= "igw\_mumbai\_git"

    }

    vpc\_id = aws\_vpc.mumbai\_vpc.id

    provider = aws.mumbai

    depends\_on = [ aws\_subnet.git\_pub\_sub2 ]

}

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

    vpc\_id = aws\_vpc.mumbai\_server\_git.id

    provider = aws.mumbai

    depends\_on = [ aws\_subnet.git\_server\_pub\_sub ]

}

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

    tags = {

      Name= "igw\_singapore\_git"

    }

    vpc\_id = aws\_vpc.singapore\_vpc.id

    provider = aws.singapore

    depends\_on = [ aws\_subnet.git\_pub\_singapore\_sub2 ]

}

resource "aws\_route\_table" "pub\_git" {    #creating route table and adding the igw route to the RT

    vpc\_id = aws\_vpc.mumbai\_server\_git.id

    route {

        cidr\_block = "0.0.0.0/0"

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

    }

    provider = aws.mumbai

    depends\_on = [ aws\_internet\_gateway.igw\_git ]

}

resource "aws\_route\_table" "pub\_mum\_git\_rt" {

    vpc\_id = aws\_vpc.mumbai\_vpc.id

    route {

        cidr\_block = "0.0.0.0/0"

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

    }

    provider = aws.mumbai

    tags = {

      Name="pub\_mum\_git\_rt"

    }

    depends\_on = [ aws\_internet\_gateway.igw\_mumbai\_git ]

}

resource "aws\_route\_table" "pub\_rt\_singapore" {

    vpc\_id = aws\_vpc.singapore\_vpc.id

    route {

        cidr\_block = "0.0.0.0/0"

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

    }

    provider = aws.singapore

    tags = {

      Name= "pub\_sing\_git\_rt"

    }

    depends\_on = [ aws\_internet\_gateway.igw\_singapore\_git ]

}

resource "aws\_route\_table\_association" "server\_git\_asso" { #associating the public subnet in rt for instance contain webpage

    route\_table\_id = aws\_route\_table.pub\_git.id

    subnet\_id = aws\_subnet.git\_server\_pub\_sub.id

    depends\_on = [ aws\_route\_table.pub\_git ]

    provider = aws.mumbai

}

resource "aws\_route\_table\_association" "mum\_web1" {  #associating the 1nd subnet in rt mumbai

    route\_table\_id = aws\_route\_table.pub\_mum\_git\_rt.id

    subnet\_id = aws\_subnet.git\_pub\_sub1.id

    depends\_on = [ aws\_route\_table.pub\_mum\_git\_rt ]

    provider = aws.mumbai

}

resource "aws\_route\_table\_association" "mum\_web2" { #associating the 2nd subnet in rt mumbai

    route\_table\_id = aws\_route\_table.pub\_mum\_git\_rt.id

    subnet\_id = aws\_subnet.git\_pub\_sub2.id

    provider = aws.mumbai

    depends\_on = [ aws\_route\_table.pub\_mum\_git\_rt ]

}

resource "aws\_route\_table\_association" "sig\_web1" {  #associating the 1nd subnet in rt singapore

    route\_table\_id = aws\_route\_table.pub\_rt\_singapore.id

    subnet\_id = aws\_subnet.git\_pub\_singapore\_sub1.id

    provider = aws.singapore

    depends\_on = [ aws\_route\_table.pub\_rt\_singapore ]

}

resource "aws\_route\_table\_association" "sig\_web2" { #associating the 2nd subnet in rt singapore

    route\_table\_id = aws\_route\_table.pub\_rt\_singapore.id

    subnet\_id = aws\_subnet.git\_pub\_singapore\_sub2.id

    provider = aws.singapore

    depends\_on = [ aws\_route\_table.pub\_rt\_singapore ]

}

resource "aws\_security\_group" "server\_git" {    # Create a security group for instance contain webpage

    vpc\_id = aws\_vpc.mumbai\_server\_git.id

    name = "server\_git"

    tags = {

      Name= "server\_git"

    }

    depends\_on = [ aws\_route\_table\_association.server\_git\_asso ]

    provider = aws.mumbai

}

resource "aws\_security\_group" "web\_git" {    # Create a security group for mumbai region instance

    vpc\_id = aws\_vpc.mumbai\_vpc.id

    name = "web\_git"

    tags = {

      Name= "web\_git"

    }

    depends\_on = [ aws\_route\_table\_association.mum\_web2 ]

    provider = aws.mumbai

}

resource "aws\_security\_group" "web\_sig\_git" {    # Create a security group for singapore region instance

    vpc\_id = aws\_vpc.singapore\_vpc.id

    name = "web\_sig\_git"

    tags = {

      Name= "web\_sig\_git"

    }

    depends\_on = [ aws\_route\_table\_association.sig\_web2 ]

    provider = aws.singapore

}

resource "aws\_security\_group\_rule" "ingress\_ssh\_git\_server" {

    type = "ingress"

    from\_port = 22

    to\_port = 22

    protocol = "tcp"

    cidr\_blocks = ["0.0.0.0/0"]

    security\_group\_id = aws\_security\_group.server\_git.id

    provider = aws.mumbai

}

resource "aws\_security\_group\_rule" "ingress\_http\_git\_server" {

    type = "ingress"

    from\_port = 80

    to\_port = 80

    protocol = "tcp"

    cidr\_blocks = ["0.0.0.0/0"]

    security\_group\_id = aws\_security\_group.server\_git.id

    provider = aws.mumbai

}

resource "aws\_security\_group\_rule" "egress\_allow\_git\_sever" {

    type = "egress"

    protocol = "-1"

    cidr\_blocks = ["0.0.0.0/0"]

    from\_port = 0

    to\_port = 0

    security\_group\_id = aws\_security\_group.server\_git.id

    provider = aws.mumbai

}

resource "aws\_security\_group\_rule" "ingress\_ssh\_web\_mumbai" {

    type = "ingress"

    from\_port = 22

    to\_port = 22

    protocol = "tcp"

    cidr\_blocks = ["0.0.0.0/0"]

    security\_group\_id = aws\_security\_group.web\_git.id

    provider = aws.mumbai

}

resource "aws\_security\_group\_rule" "ingress\_http\_web\_mumbai" {

    type = "ingress"

    from\_port = 80

    to\_port = 80

    protocol = "tcp"

    cidr\_blocks = ["0.0.0.0/0"]

    security\_group\_id = aws\_security\_group.web\_git.id

    provider = aws.mumbai

}

resource "aws\_security\_group\_rule" "egress\_allow\_web\_mumbai" {

    type = "egress"

    protocol = "-1"

    cidr\_blocks = ["0.0.0.0/0"]

    from\_port = 0

    to\_port = 0

    security\_group\_id = aws\_security\_group.web\_git.id

    provider = aws.mumbai

}

resource "aws\_security\_group\_rule" "ingress\_ssh\_web\_sing" {

    type = "ingress"

    from\_port = 22

    to\_port = 22

    protocol = "tcp"

    cidr\_blocks = ["0.0.0.0/0"]

    security\_group\_id = aws\_security\_group.web\_sig\_git.id

    provider = aws.singapore

}

resource "aws\_security\_group\_rule" "ingress\_http\_web\_sing" {

    type = "ingress"

    from\_port = 80

    to\_port = 80

    protocol = "tcp"

    cidr\_blocks = ["0.0.0.0/0"]

    security\_group\_id = aws\_security\_group.web\_sig\_git.id

    provider = aws.singapore

}

resource "aws\_security\_group\_rule" "egress\_allow\_web\_sing" {

    type = "egress"

    protocol = "-1"

    cidr\_blocks = ["0.0.0.0/0"]

    from\_port = 0

    to\_port = 0

    security\_group\_id = aws\_security\_group.web\_sig\_git.id

    provider = aws.singapore

}

resource "aws\_instance" "git\_server" {  #instance to push the webpage to github

    tags = {

      Name= "git\_server"

    }

    provider = aws.mumbai

    ami = var.aws\_ami

    instance\_type = var.aws\_instance

    key\_name = "appkey"

    subnet\_id = aws\_subnet.git\_server\_pub\_sub.id

    vpc\_security\_group\_ids = [aws\_security\_group.server\_git.id]

    depends\_on = [ aws\_security\_group.server\_git ]

    connection {

      type = "ssh"

      user = "ec2-user"

      private\_key = file("E:/appkey.pem")

      host = self.public\_ip

    }

    provisioner "remote-exec" {

        inline = [

            "echo 'machine connected'",

            "sudo yum install git -y",

            "sudo git config --global user.name '<user-name>' ",

            "sudo git config --global user.mail ‘<mail>’"

         ]

    }

}

resource "aws\_instance" "web\_1\_lb" {   #creatin instance containing web1 in az1

    provider = aws.mumbai

    tags = {

      Name= "web\_1\_lb\_az1"

    }

      ami= var.aws\_ami

      instance\_type= var.aws\_instance

      key\_name = "appkey"

      subnet\_id = aws\_subnet.git\_pub\_sub1.id

      vpc\_security\_group\_ids = [aws\_security\_group.web\_git.id]

      depends\_on = [ aws\_security\_group.web\_git ]

      connection {

        type = "ssh"

        user = "ec2-user"

        private\_key = file("E:/appkey.pem")

        host = self.public\_ip

      }

      provisioner "remote-exec" {  #by using provisioner installing req package of git and httpd

        inline = [

            "echo 'machine connected web1'",

            "sudo yum install httpd git -y",

            "sudo systemctl start httpd",

            "sudo systemctl enable httpd",

            "sudo chown -R ec2-user:ec2-user /var/www/html",

            "sudo git config --global user.name '<user-name>' ",

            "sudo git config --global user.mail <user-mail>'"

         ]

}

}

resource "aws\_instance" "web\_1\_az2" {  #creatin instance containing web1 in az 2

    provider = aws.mumbai

    tags = {

      Name= "web\_1\_lb\_az2"

    }

      ami= var.aws\_ami

      instance\_type= var.aws\_instance

      key\_name = "appkey"

      subnet\_id = aws\_subnet.git\_pub\_sub2.id

      vpc\_security\_group\_ids = [aws\_security\_group.web\_git.id]

      depends\_on = [ aws\_instance.web\_1\_lb ]

      connection {

        type = "ssh"

        user = "ec2-user"

        private\_key = file("E:/appkey.pem")

        host = self.public\_ip

      }

      provisioner "remote-exec" {  #by using provisioner installing req package of git and httpd

        inline = [

            "echo 'machine connected web1-az2'",

            "sudo yum install httpd git -y",

            "sudo systemctl start httpd",

            "sudo systemctl enable httpd",

            "sudo chown -R ec2-user:ec2-user /var/www/html",

            "sudo git config --global user.name '<user-name>'",

            "sudo git config --global user.mail ‘<user-mail>com'"

         ]

}

}

resource "aws\_instance" "web\_2\_az1" { #creatin instance containing web2 in az 1

    provider = aws.singapore

    tags = {

      Name= "web\_2\_lb\_az1"

    }

      ami= var.aws\_ami\_si

      instance\_type= var.aws\_instance

      key\_name = "lab"

      subnet\_id = aws\_subnet.git\_pub\_singapore\_sub1.id

      vpc\_security\_group\_ids = [aws\_security\_group.web\_sig\_git.id]

      depends\_on = [ aws\_security\_group.web\_sig\_git]

      connection {

        type = "ssh"

        user = "ec2-user"

        private\_key = file("E:/lab.pem")

        host = self.public\_ip

      }

      provisioner "remote-exec" {  #by using provisioner installing req package of git and httpd

        inline = [

            "echo 'machine connected web2'",

            "sudo yum install httpd git -y",

            "sudo systemctl start httpd",

            "sudo systemctl enable httpd",

            "sudo chown -R ec2-user:ec2-user /var/www/html",

            "sudo git config --global user.name '<user-name>' ",

            "sudo git config --global user.mail '<user-mail>com'"

         ]

}

}

resource "aws\_instance" "web\_2\_az2" { #creatin instance containing web2

    provider = aws.singapore

    tags = {

      Name= "web\_2\_lb\_az2"

    }

      ami= var.aws\_ami\_si

      instance\_type= var.aws\_instance

      key\_name = "lab"

      subnet\_id = aws\_subnet.git\_pub\_singapore\_sub2.id

      vpc\_security\_group\_ids = [aws\_security\_group.web\_sig\_git.id]

      depends\_on = [ aws\_instance.web\_2\_az1]

      connection {

        type = "ssh"

        user = "ec2-user"

        private\_key = file("E:/lab.pem")

        host = self.public\_ip

      }

      provisioner "remote-exec" {   #by using provisioner installing req package of git and httpd

        inline = [

            "echo 'machine connected web2'",

            "sudo yum install httpd git -y",

            "sudo systemctl start httpd",

            "sudo systemctl enable httpd",

            "sudo chown -R ec2-user:ec2-user /var/www/html",

            "sudo git config --global user.name '<user-name>' ",

            "sudo git config --global user.mail ‘<user-mail>"

         ]

}

}

resource "aws\_security\_group" "sgw\_lb\_sing" {    # Create a security group for lb singapore

    vpc\_id = aws\_vpc.singapore\_vpc.id

    name = "sgw\_lb\_sing"

    tags = {

      Name= "sgw\_lb\_sing"

    }

    depends\_on = [ aws\_instance.web\_2\_az2 ]

    provider = aws.singapore

}

resource "aws\_security\_group\_rule" "ingress\_http\_lb\_server" {

    type = "ingress"

    from\_port = 80

    to\_port = 80

    protocol = "tcp"

    cidr\_blocks = ["0.0.0.0/0"]

    security\_group\_id = aws\_security\_group.sgw\_lb\_sing.id

    provider = aws.singapore

}

resource "aws\_security\_group\_rule" "egress\_allow\_lb\_sever" {

    type = "egress"

    protocol = "-1"

    cidr\_blocks = ["0.0.0.0/0"]

    from\_port = 0

    to\_port = 0

    security\_group\_id = aws\_security\_group.sgw\_lb\_sing.id

    provider = aws.singapore

}

resource "aws\_security\_group" "lb\_mumbai" {    # Create a security group for lb mumbai

    vpc\_id = aws\_vpc.mumbai\_vpc.id

    name = "sgw\_lb\_mumbai"

    tags = {

      Name= "sgw\_lb\_mumbai"

    }

    depends\_on = [ aws\_instance.web\_1\_az2 ]

    provider = aws.mumbai

}

resource "aws\_security\_group\_rule" "ingress\_http\_lb\_mumbai" {

    type = "ingress"

    from\_port = 80

    to\_port = 80

    protocol = "tcp"

    cidr\_blocks = ["0.0.0.0/0"]

    security\_group\_id = aws\_security\_group.lb\_mumbai.id

    provider = aws.mumbai

}

resource "aws\_security\_group\_rule" "egress\_allow\_lb\_mumb" {

    type = "egress"

    protocol = "-1"

    cidr\_blocks = ["0.0.0.0/0"]

    from\_port = 0

    to\_port = 0

    security\_group\_id = aws\_security\_group.lb\_mumbai.id

    provider = aws.mumbai

}

resource "aws\_elb" "mumbai\_lb" {  #creating a Load balancer on mumbai region

    name = "mumbai-lb"

    provider = aws.mumbai

    internal = false

    security\_groups = [aws\_security\_group.lb\_mumbai.id]

    subnets = [aws\_subnet.git\_pub\_sub1.id, aws\_subnet.git\_pub\_sub2.id]

    cross\_zone\_load\_balancing = true

    connection\_draining = true

    connection\_draining\_timeout = 300

    listener {

      instance\_port = 80

      instance\_protocol = "http"

      lb\_port = 80

      lb\_protocol = "http"

    }

    health\_check {

      target = "HTTP:80/web1/web1.html/"

      interval = 5

      timeout = 2

      healthy\_threshold = 2

      unhealthy\_threshold = 2

    }

}

resource "aws\_elb\_attachment" "web1\_az1" { #attach the instance to the lb

    provider = aws.mumbai

    elb = aws\_elb.mumbai\_lb.id

    instance= aws\_instance.web\_1\_lb.id

}

resource "aws\_elb\_attachment" "web1\_az2" { #attach the instance to the lb

    provider = aws.mumbai

    elb = aws\_elb.mumbai\_lb.id

    instance = aws\_instance.web\_1\_az2.id

}

resource "aws\_elb" "sing\_lb" {  #creating a Load balancer on singapore region

    provider = aws.singapore

    name = "lb-singapore"

    internal = false

    security\_groups = [ aws\_security\_group.sgw\_lb\_sing.id ]

    subnets = [ aws\_subnet.git\_pub\_singapore\_sub1.id, aws\_subnet.git\_pub\_singapore\_sub2.id ]

    cross\_zone\_load\_balancing = true

    connection\_draining = true

    connection\_draining\_timeout = 300

    listener {

      instance\_port = 80

      instance\_protocol = "http"

      lb\_port = 80

      lb\_protocol = "http"

    }

    health\_check {

      target = "HTTP:80/web2/web2.html/"

      interval = 5

      timeout = 2

      healthy\_threshold = 2

      unhealthy\_threshold = 2

    }

}

resource "aws\_elb\_attachment" "web2\_az1\_lb" { #attach the instance to the lb

    provider = aws.singapore

    elb = aws\_elb.sing\_lb.id

    instance = aws\_instance.web\_2\_az1.id

}

resource "aws\_elb\_attachment" "web2\_az2\_lb" { #attach the instance to the lb

    provider = aws.singapore

    elb = aws\_elb.sing\_lb.id

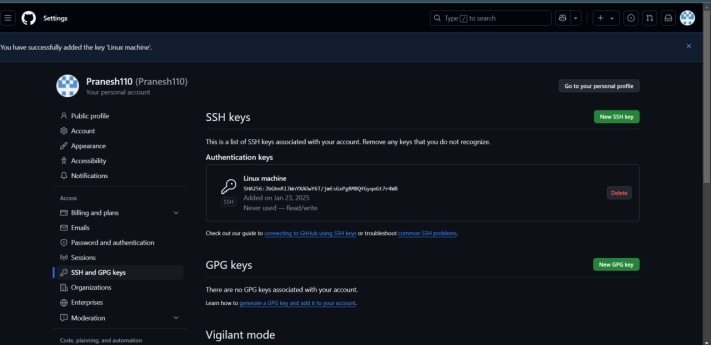
    instance = aws\_instance.web\_2\_az2.id

}

In the webpage containg instance :

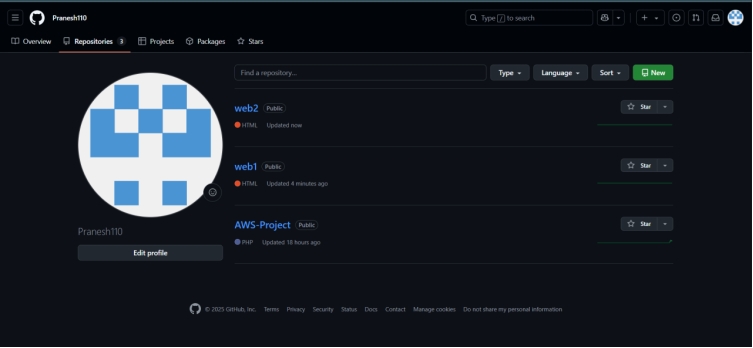
Generate and share the ssh key

1. ssh-keygen -t rsa -b 4096 -C “comment”
2. eval “$(ssh-agent -s)” – adding ssh-agent in bg
3. ssh-add <private key path> -- to add the priv key to ssh-agent to avoid key auth everytime.
4. Copy the public key add to github account in SSH-GPG



Create a repo :

1. make directory—mkdir name
2. cd to the dir
3. to initialize .git file for the repo—git init
4. ADD the webpage file to the repo
5. git add . – to stage the change
6. git commit – to commit the change
7. Create a repo in git hub and copy the ssh
8. Remote add – git remote add origin ssh-url
9. Push the repo to git hub—git push -u origin main/master

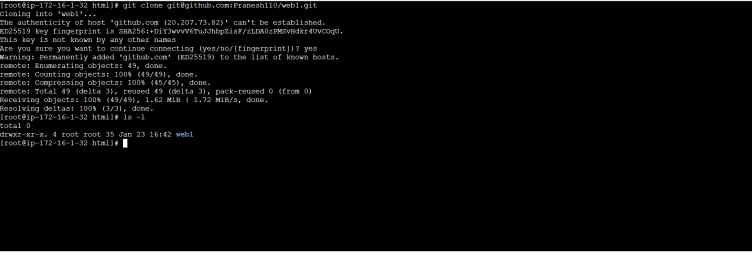


AS other instance’s already installed git and httpd by the terraform cong itself,

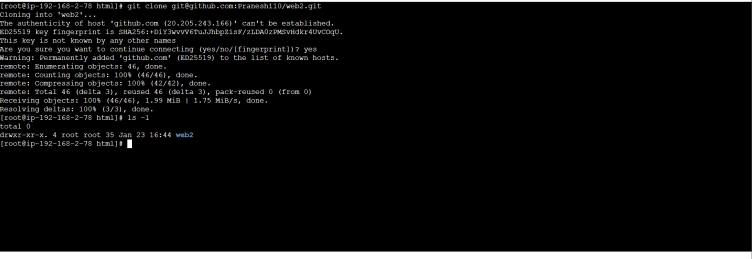
**NOW generate the SSH-KEY from the other instances and copy to git hub.**

**clone the web1 repo to Mumbai region instance and web2 repo to Singapore region instances.**

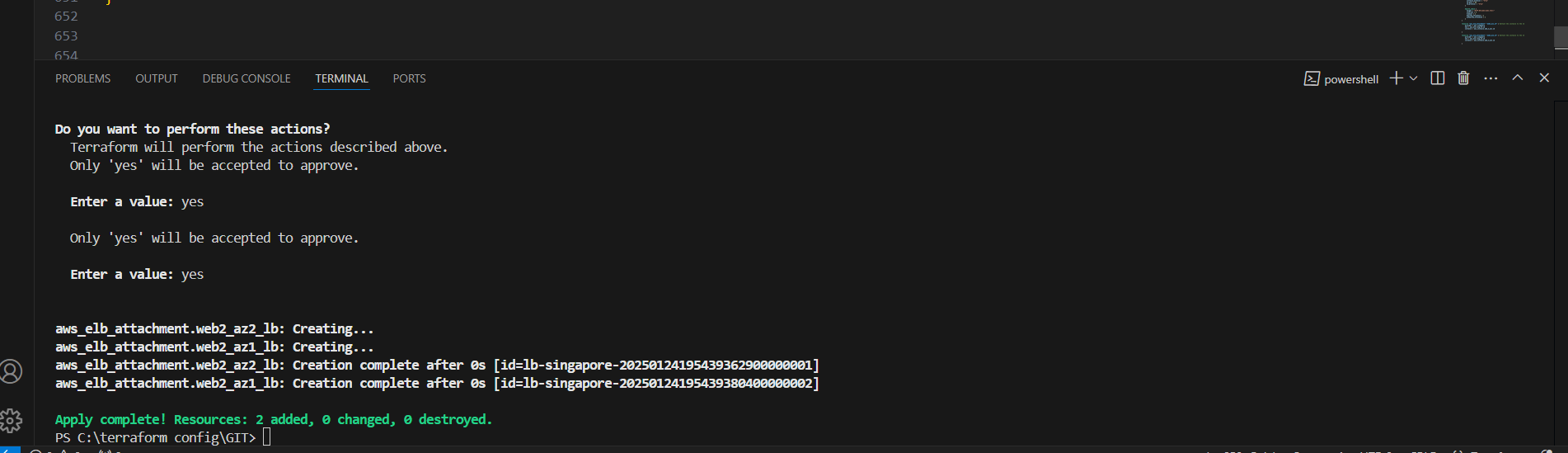
**MUMBAI:**



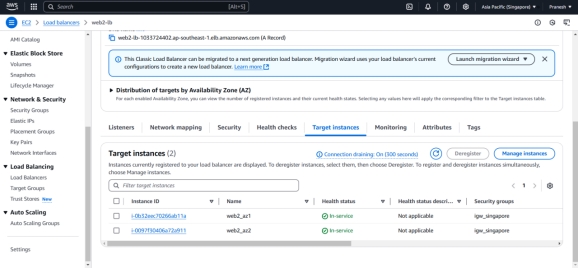
**SINGAPORE:**

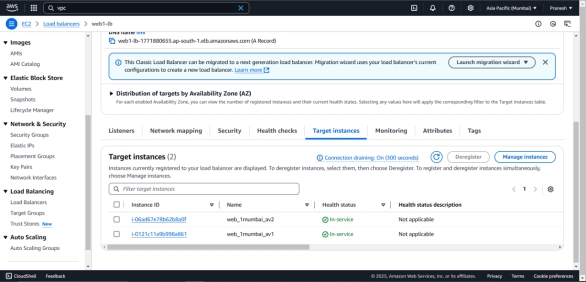


AFTER DOING THIS CONFG THE LOAD BALANCER IN TERRAFORM my creating sgw for lb and adding elb resources and attaching to the instances in diff region.



Check the instance are healthy after attached:

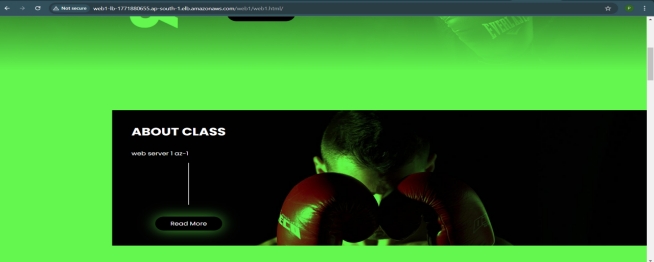




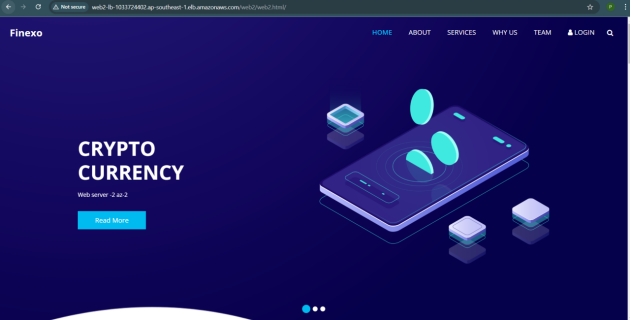
Ping the Load balancer DNS of lb on both region :

MUMBAI region LB:





SINGAPORE REGION LB:





YOU CAN SEE HIT ARE EVENLY DISTRIBUTE!!