**terraform-template-task**

***1) Create VPC***

**>>create a directory with name** terraform-x **and add a file** main.tf **then add the template**

provider "aws" {

  region = "us-east-2"

}

resource "aws\_vpc" "main" {

  cidr\_block = "192.168.0.0/24"

  tags = {

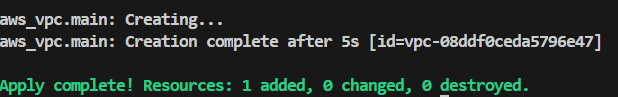
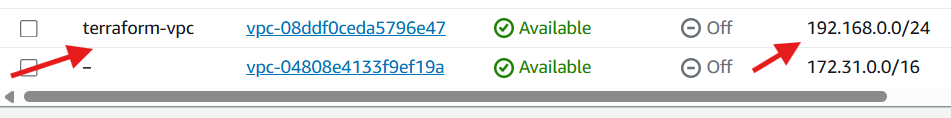
    Name = "terraform-vpc"

  }

}

**🡪 crl+s**

**#** terraform init > terraform plan > terraform apply **<check on console if its created>**

 ******

***2) Create Internet gateway***

**>>add this template to exixting** main.tf **file**

resource "aws\_internet\_gateway" "gw" {

  vpc\_id = aws\_vpc.main.id

tags = {

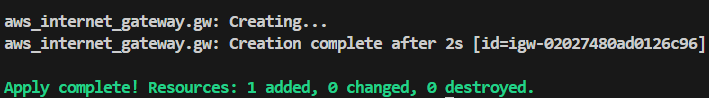
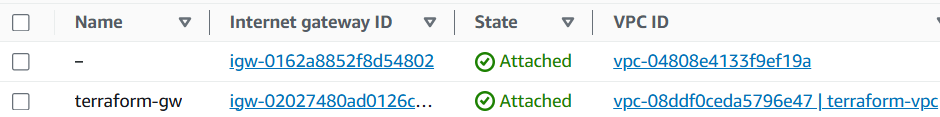
    Name = "terraform-gw"

  }

}

**🡪 crl+s**

**#** terraform init > terraform plan > terraform apply **<check on console if its created>**

** **

***3) Create Custom Route Table***

**>>add this template to exixting** main.tf **file**

resource "aws\_route\_table" "rt" {

  vpc\_id = aws\_vpc.main.id

  route {

    cidr\_block = "0.0.0.0/0"

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

  }

tags = {

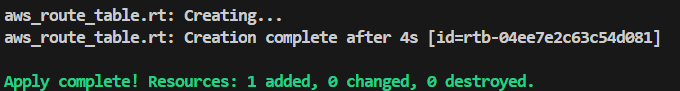
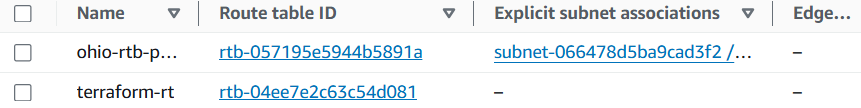
    Name = "terraform-rt"

  }

}

**🡪 crl+s**

**#** terraform init > terraform plan > terraform apply  **<check on console if its created>**

*** ***

***4) Create Subnet***

**>>add this template to exixting** main.tf **file**

resource "aws\_subnet" "subnet" {

  vpc\_id    = aws\_vpc.main.id

  cidr\_block = "192.168.0.0/28"

  availability\_zone = "us-east-2a"

  tags = {

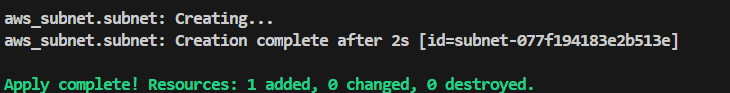
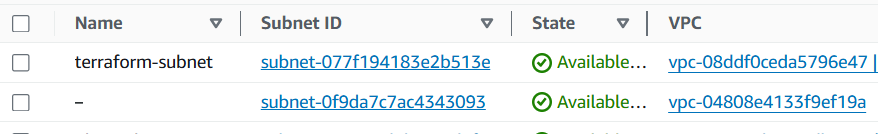
    Name = "terraform-subnet"

  }

}

**🡪 crl+s**

**#** terraform init > terraform plan > terraform apply  **<check on console if its created>**

** ****

***5) Associate subnet with Route Table***

**>>add this template to exixting** main.tf **file**

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

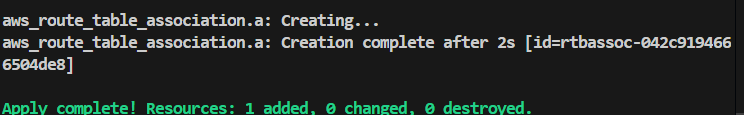
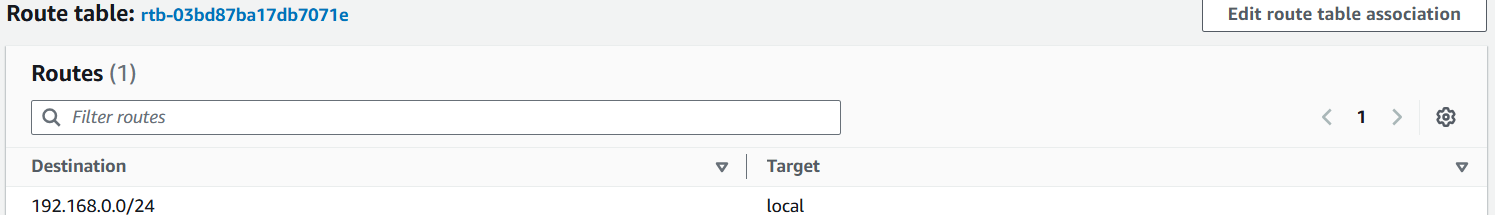
  subnet\_id      = aws\_subnet.subnet.id

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

}

**🡪 crl+s**

**#** terraform init > terraform plan > terraform apply  **<check on console if its created>**

***6) Create Security Group to allow port 22,80,443***

**>>add this template to exixting** main.tf **file**

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

vpc\_id = aws\_vpc.main.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"]

}

ingress {

from\_port = 443

to\_port = 443

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"]

}

tags = {

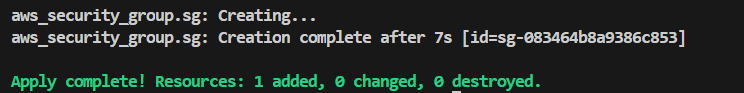
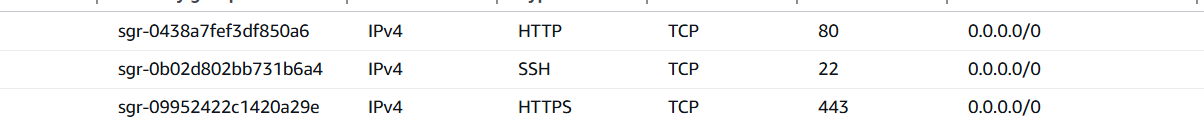
Name = "main-sg"

}

}

**🡪 crl+s**

**#** terraform init > terraform plan > terraform apply  **<check on console if its created>**

*** ***

***7) Create a network interface with an ip in the subnet that was created in step 4***

**>>add this template to exixting** main.tf **file**

resource "aws\_network\_interface" "eni" {

  subnet\_id       = aws\_subnet.subnet.id

  private\_ips     = ["192.168.0.12"]

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

tags = {

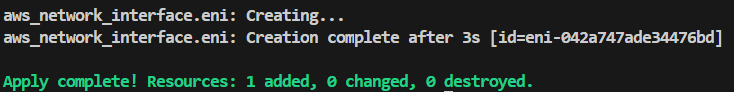
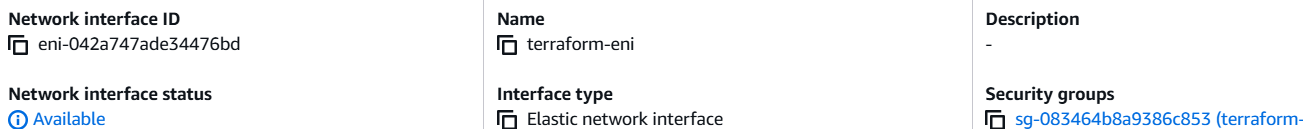
    Name = "terraform-eni"

  }

}

**🡪 crl+s**

**#** terraform init > terraform plan > terraform apply  **<check on console if its created>**

 ******

***8) Assign an elastic IP to the network interface created in step 7***

**>>add this template to exixting** main.tf **file**

resource "aws\_eip" "eip" {

  vpc               = true

  network\_interface = aws\_network\_interface.eni.id

  tags = {

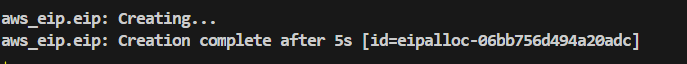
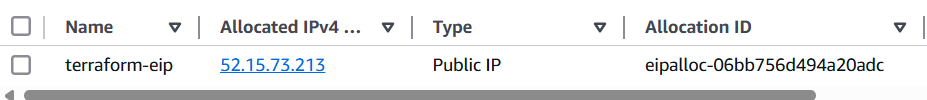
    Name = "terraform-eip"

  }

}

**🡪 crl+s**

**#** terraform init > terraform plan > terraform apply  **<check on console if its created>**

***Note:***

***1) Create single main.tf which will be created the above resources and do not hardcode the id's.***

***9) Create Ubuntu server and install/enable apache2***

**>>add this template to exixting** main.tf **file**

resource "aws\_instance" "web" {

  ami           = "ami-036841078a4b68e14"

  instance\_type = "t2.micro"

  network\_interface {

    network\_interface\_id = aws\_network\_interface.eni.id

    device\_index         = 0

  }

  user\_data = <<-EOF

              #!/bin/bash

              sudo apt-get update -y

              sudo apt-get install -y apache2

              sudo systemctl start apache2

              sudo systemctl enable apache2

              EOF

  tags = {

    Name = "terraform-server"

  }

}

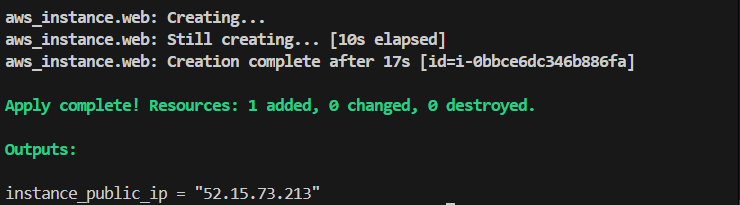
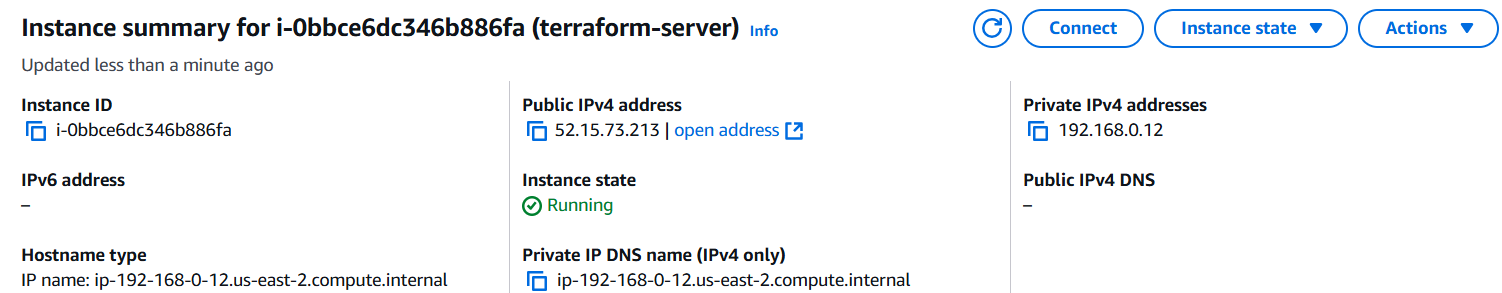
output "instance\_public\_ip" {

  value = aws\_instance.web.public\_ip

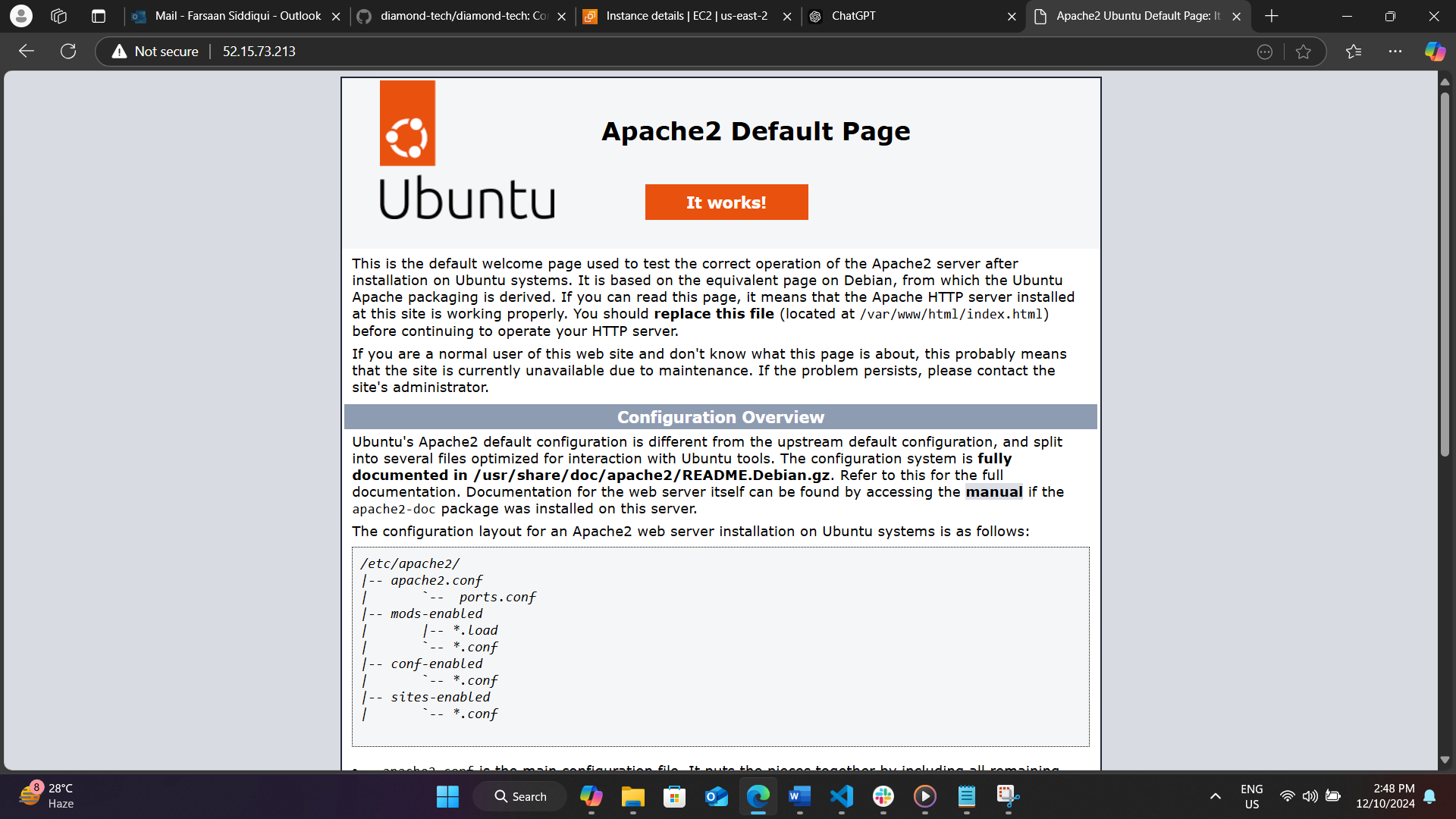
}

**🡪 crl+s**

**#** terraform init > terraform plan > terraform apply  **<check on console if its created>**

*** ***

**<checking if the apache is installed and running by using public ip** [*http://52.15.73.213:80*](http://52.15.73.213:80)***>***

******

**Configure s3 as backend and dynamo db locking for multi user execution.**

**🡪creating a s3 bucket and dynamo db**

**>>add this template to exixting** main.tf **file**

resource "aws\_s3\_bucket" "terraform\_state" {

  bucket = "terraform-task-bucket"

  acl    = "private"

}

resource "aws\_dynamodb\_table" "terraform\_state\_lock" {

  name         = "terraform-task-dynamodb"

  hash\_key     = "LockID"

  read\_capacity  = 20

  write\_capacity = 20

  attribute {

    name = "LockID"

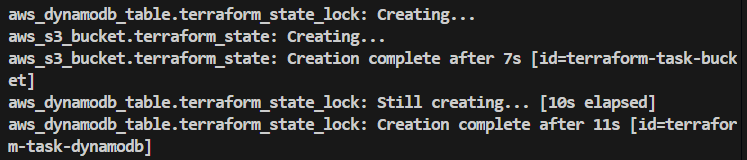
    type = "S"

  }

}

**🡪 crl+s**

**#** terraform init > terraform plan > terraform apply

****

**🡪configuring s3 as backend and dynamodb locking**

**>>add this template to exixting** main.tf **file**

terraform {

  backend "s3" {

    bucket         = "terraform-task-bucket"

    key            = "terraform.tfstate"

    region         = "us-east-2"

    dynamodb\_table = "terraform-task-dynamodb"

    encrypt        = true

  }

}

**🡪 crl+s**

**#** terraform init > terraform plan > terraform apply

