1. **Terraform Installed:**
2. **AWS Credentials Configured:** You should have your AWS credentials set up using the AWS CLI (aws configure) or environment variables.

**Steps:**

1. **Run the Terraform destroy command:** This command will destroy all resources that were created by your previous Terraform deployment:

**terraform destroy**

1. **Create the Terraform Configuration File:** Create a file called main.tf inside the directory with the following content:

provider "aws" {

  region = "us-east-1"  # N. Virginia Region

}

# Create a VPC

resource "aws\_vpc" "main\_vpc" {

  cidr\_block = "10.0.0.0/16"

  tags = {

    Name = "main-vpc"

  }

}

# Create an Internet Gateway

resource "aws\_internet\_gateway" "main\_igw" {

  vpc\_id = aws\_vpc.main\_vpc.id

  tags = {

    Name = "main-igw"

  }

}

# Create a Subnet inside the VPC

resource "aws\_subnet" "main\_subnet" {

  vpc\_id     = aws\_vpc.main\_vpc.id

  cidr\_block = "10.0.1.0/24"

  map\_public\_ip\_on\_launch = true  # Enable public IP assignment

  tags = {

    Name = "main-subnet"

  }

}

# Create a Route Table

resource "aws\_route\_table" "main\_route\_table" {

  vpc\_id = aws\_vpc.main\_vpc.id

  route {

    cidr\_block = "0.0.0.0/0"

    gateway\_id = aws\_internet\_gateway.main\_igw.id

  }

  tags = {

    Name = "main-route-table"

  }

}

# Associate the Subnet with the Route Table

resource "aws\_route\_table\_association" "main\_rta" {

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

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

}

# Create a Security Group for the EC2 instance

resource "aws\_security\_group" "main\_sg" {

  vpc\_id = aws\_vpc.main\_vpc.id

  # Allow inbound HTTP and SSH traffic

  ingress {

    from\_port   = 80

    to\_port     = 80

    protocol    = "tcp"

    cidr\_blocks = ["0.0.0.0/0"]

  }

  ingress {

    from\_port   = 22

    to\_port     = 22

    protocol    = "tcp"

    cidr\_blocks = ["0.0.0.0/0"]

  }

  # Allow all outbound traffic

  egress {

    from\_port   = 0

    to\_port     = 0

    protocol    = "-1"

    cidr\_blocks = ["0.0.0.0/0"]

  }

  tags = {

    Name = "main-sg"

  }

}

# Create an EC2 instance in the VPC's Subnet and run Apache installation script

resource "aws\_instance" "new\_instance" {

  ami                    = "ami-0e86e20dae9224db8"

  instance\_type          = "t2.micro"

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

  vpc\_security\_group\_ids = [aws\_security\_group.main\_sg.id]

  associate\_public\_ip\_address = true  # Automatically assign a public IPv4 address

  user\_data = file("install\_apache.sh")  # Run the Apache installation script

  tags = {

    Name = "new-instance-with-apache"

  }

}

# Output the EC2 instance's public IP

output "instance\_public\_ip" {

  value = aws\_instance.new\_instance.public\_ip

}

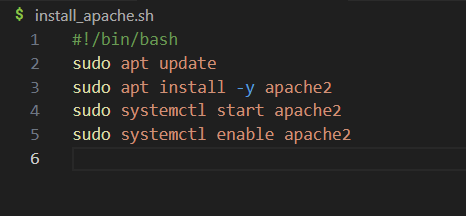
# Save the instance public IP to a file

resource "local\_file" "instance\_ip\_file" {

  content  = aws\_instance.new\_instance.public\_ip

  filename = "${path.module}/instance\_ip.txt"

}



1. **Initialize Terraform:**

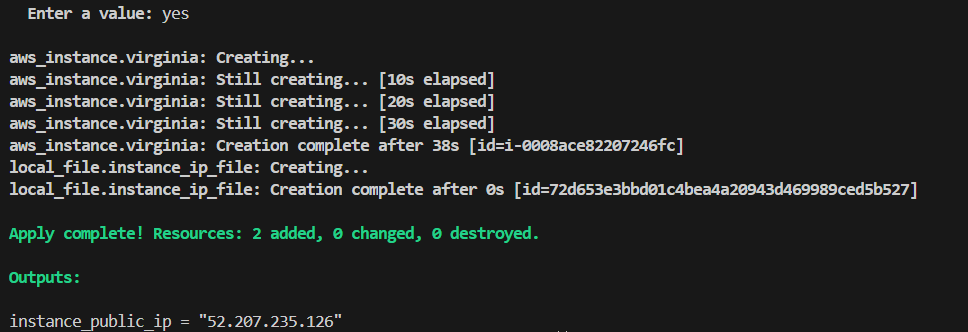
**terraform init**

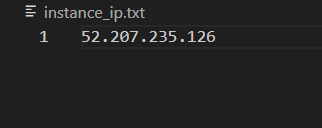
1. **Preview the Changes:** After initialization, you can preview the changes Terraform will make by running:

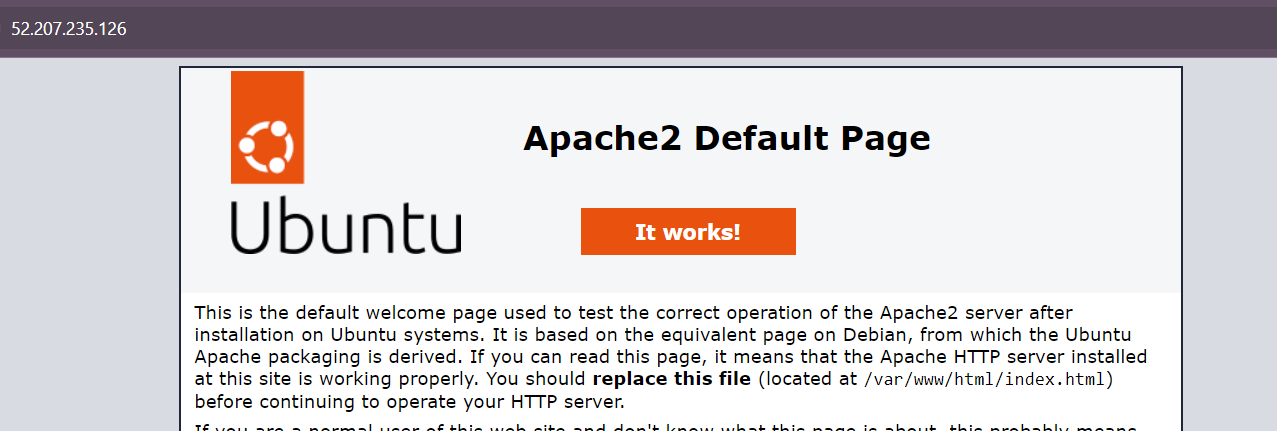
**terraform plan**

1. **Apply the Changes:** Apply the configuration to create the EC2 instance in the default VPC's subnet:

**terraform apply**

****

****

****

****