Tasks To Be Performed:

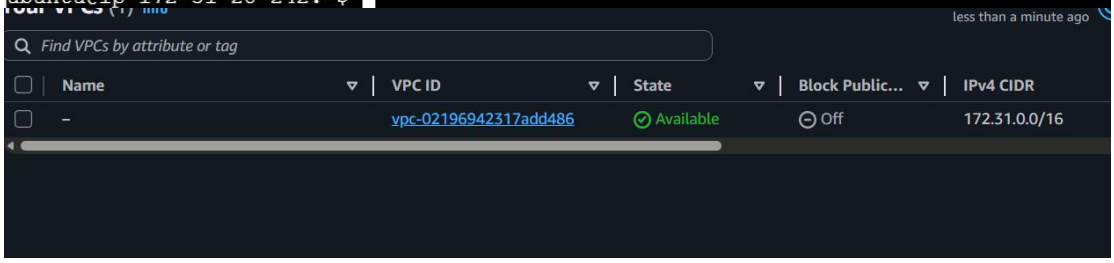
1. Destroy the previous deployments
2. Create a script to install Apache2
3. Run this script on a newly created EC2 instance
4. Print the IP address of the instance in a file on the local once deployed

---

1. Destroy the previous deployments

```
ubuntu@ip-172-31-28-242:~$ terraform destroy
aws_instance.task-4: Refreshing state... [id=i-01a1fa97a666c74d9]
aws_vpc.task-4-vpc: Refreshing state... [id=vpc-04e42c0416c9e4026]
aws_internet_gateway.task-4-igw: Refreshing state... [id=igw-0d5110650d3ea3c20]
aws_subnet.task-4-subnet: Refreshing state... [id=subnet-05cac536d8614db94]
aws_route_table.task-4-rt: Refreshing state... [id=rtb-0ae3000091662bbae]
aws_route_table_association.rt: Refreshing state... [id=rtbassoc-03e91d75df1d72ce3]

Terraform used the selected providers to generate the following execution plan. Resou
  - destroy

Terraform will perform the following actions:

  # aws_instance.task-4 will be destroyed
  - resource "aws_instance" "task-4" {
      - ami                                  = "ami-0cae6d6fe6048ca2c" -> null
      - arn                                  = "arn:aws:ec2:us-east-1:251985476962:i
      - associate_public_ip_address          = true -> null
      - availability_zone                    = "us-east-1b" -> null
      - disable_api_stop                     = false -> null
      - disable_api_termination              = false -> null
      - ebs_optimized                        = false -> null
      - force_destroy                        = false -> null
      - get_password_data                    = false -> null
      - hibernation                          = false -> null
      - id                                   = "i-01a1fa97a666c74d9" -> null
      - instance_initiated_shutdown_behavior = "stop" -> null
      - instance_state                       = "running" -> null
      - instance_type                        = "t2.micro" -> null
      - ipv6_address_count                   = 0 -> null
      - ipv6_addresses                       = [] -> null
      - key_name                             = "terra-kp" -> null
      - monitoring                           = false -> null
```

```
Plan: 0 to add, 0 to change, 6 to destroy.

Do you really want to destroy all resources?
  Terraform will destroy all your managed infrastructure, as shown above.
  There is no undo. Only 'yes' will be accepted to confirm.

  Enter a value: yes

aws_instance.task-4: Destroying... [id=i-01a1fa97a666c74d9]
aws_route_table_association.rt: Destroying... [id=rtbassoc-03e91d75df1d72ce3]
aws_route_table_association.rt: Destruction complete after 1s
aws_subnet.task-4-subnet: Destroying... [id=subnet-05cac536d8614db94]
aws_route_table.task-4-rt: Destroying... [id=rtb-0ae3000091662bbae]
aws_subnet.task-4-subnet: Destruction complete after 0s
aws_route_table.task-4-rt: Destruction complete after 0s
aws_internet_gateway.task-4-igw: Destroying... [id=igw-0d5110650d3ea3c20]
aws_internet_gateway.task-4-igw: Destruction complete after 1s
aws_vpc.task-4-vpc: Destroying... [id=vpc-04e42c0416c9e4026]
aws_vpc.task-4-vpc: Destruction complete after 0s
aws_instance.task-4: Still destroying... [id=i-01a1fa97a666c74d9, 00m10s elapsed]
aws_instance.task-4: Still destroying... [id=i-01a1fa97a666c74d9, 00m20s elapsed]
aws_instance.task-4: Destruction complete after 30s

Destroy complete! Resources: 6 destroyed.
ubuntu@ip-172-31-28-242:~$
```



2. Create a script to install Apache2

```
provider "aws"{
   region = "us-east-1"
   access_key = "AKIATVK4BHVRIRE5B5OU"
   secret_key = "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
}
resource "aws_instance" "task-1"{
    ami = "ami-0cae6d6fe6048ca2c"
    instance_type = "t2.micro"
    key_name = "terra-kp"
    user_data = <<-EOF
    #!/bin/bash
    sudo su
    apt-get update
    apt install apache2 -y
    EOF
    tags = {
      Name = "work-1"
    }
}
output "instance_ip" {
   value = aws_instance.task-1.public_ip
}
resource "local_file" "instance_ip_file"{
  content = aws_instance.task-1.public_ip
  filename = "${path.module}/instance_ip.txt"
}
```

3. Run this script on a newly created EC2 instance

```
ubuntu@ip-172-31-28-242:~$ terraform apply

Terraform used the selected providers to generate the following execution plan. Resource action
  + create

Terraform will perform the following actions:

  # aws_instance.task-1 will be created
  + resource "aws_instance" "task-1" {
      + ami                                  = "ami-0cae6d6fe6048ca2c"
      + arn                                  = (known after apply)
      + associate_public_ip_address          = (known after apply)
      + availability_zone                    = (known after apply)
      + disable_api_stop                     = (known after apply)
      + disable_api_termination              = (known after apply)
      + ebs_optimized                        = (known after apply)
      + enable_primary_ipv6                  = (known after apply)
      + force_destroy                        = false
      + get_password_data                    = false
      + host_id                              = (known after apply)
      + host_resource_group_arn              = (known after apply)
      + iam_instance_profile                 = (known after apply)
      + id                                   = (known after apply)
      + instance_initiated_shutdown_behavior = (known after apply)
      + instance_lifecycle                   = (known after apply)
      + instance_state                       = (known after apply)
      + instance_type                        = "t2.micro"
      + ipv6_address_count                   = (known after apply)
      + ipv6_addresses                       = (known after apply)
      + key_name                             = "terra-kp"
      + monitoring                           = (known after apply)
      + outpost_arn                          = (known after apply)
```

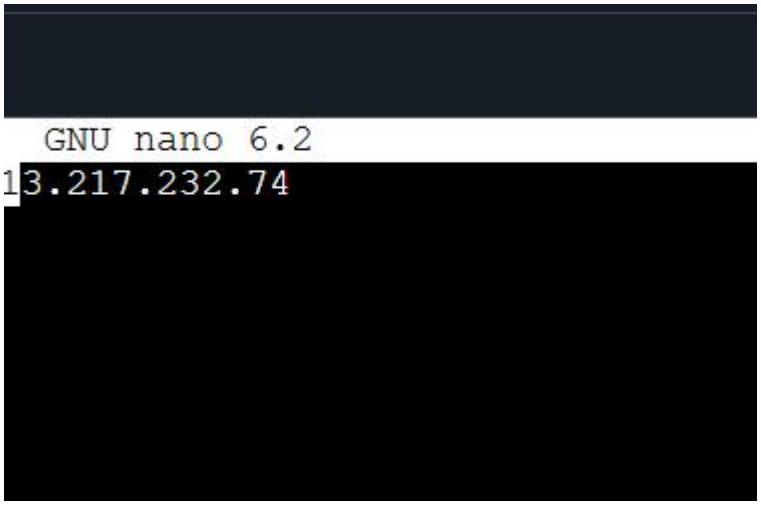
```
      + primary_network_interface (known after apply)

      + private_dns_name_options (known after apply)

      + root_block_device (known after apply)
    }

  # local_file.instance_ip_file will be created
  + resource "local_file" "instance_ip_file" {
      + content              = (known after apply)
      + content_base64sha256 = (known after apply)
      + content_base64sha512 = (known after apply)
      + content_md5          = (known after apply)
      + content_sha1         = (known after apply)
      + content_sha256       = (known after apply)
      + content_sha512       = (known after apply)
      + directory_permission = "0777"
      + file_permission      = "0777"
      + filename             = "./instance_ip.txt"
      + id                   = (known after apply)
    }

Plan: 2 to add, 0 to change, 0 to destroy.

Changes to Outputs:
  + instance_ip = (known after apply)


Note: You didn't use the -out option to save this plan, so Terraform can't guarantee to
ubuntu@ip-172-31-28-242:~$
```

```
ubuntu@ip-172-31-28-242:~$ terraform plan

Terraform used the selected providers to generate the following execution plan. Resource actions are i
  + create

Terraform will perform the following actions:

  # aws_instance.task-1 will be created
  + resource "aws_instance" "task-1" {
      + ami                                  = "ami-0cae6d6fe6048ca2c"
      + arn                                  = (known after apply)
      + associate_public_ip_address          = (known after apply)
      + availability_zone                    = (known after apply)
      + disable_api_stop                     = (known after apply)
      + disable_api_termination              = (known after apply)
      + ebs_optimized                        = (known after apply)
      + enable_primary_ipv6                  = (known after apply)
      + force_destroy                        = false
      + get_password_data                    = false
      + host_id                              = (known after apply)
      + host_resource_group_arn              = (known after apply)
      + iam_instance_profile                 = (known after apply)
      + id                                   = (known after apply)
      + instance_initiated_shutdown_behavior = (known after apply)
      + instance_lifecycle                   = (known after apply)
      + instance_state                       = (known after apply)
      + instance_type                        = "t2.micro"
      + ipv6_address_count                   = (known after apply)
      + ipv6_addresses                       = (known after apply)
      + key_name                             = "terra-kp"
      + monitoring                           = (known after apply)
      + outpost_arn                          = (known after apply)
      + password_data                        = (known after apply)
```

```
Plan: 2 to add, 0 to change, 0 to destroy.

Changes to Outputs:
  + instance_ip = (known after apply)

Do you want to perform these actions?
  Terraform will perform the actions described above.
  Only 'yes' will be accepted to approve.

  Enter a value: yes

aws_instance.task-1: Creating...
aws_instance.task-1: Still creating... [00m10s elapsed]
aws_instance.task-1: Creation complete after 13s [id=i-0af8e0c269572ed2d]
local_file.instance_ip_file: Creating...
local_file.instance_ip_file: Creation complete after 0s [id=3bfb78ca328c99920dc2c1199374ebc5deacee9c]

Apply complete! Resources: 2 added, 0 changed, 0 destroyed.

Outputs:

instance_ip = "13.217.232.74"
ubuntu@ip-172-31-28-242:~$
```

4. Print the IP address of the instance in a file on the local once deployed

```
Apply complete! Resources: 2 added, 0 changed, 0 destroyed.

Outputs:

instance_ip = "13.217.232.74"
```


GNU nano 6.2
13.217.232.74