ASSIGNMENT NO. 01

01.

- **Create one IAM user and one IAM Group using Terraform.**
- **Make sure you will use variables for names of IAM users and Group.**
- **Note: - Below files are required.**
  - **main.tf**
  - **variables.tf**
  - **your_name_custom.tfvars**

SOLUTION:-

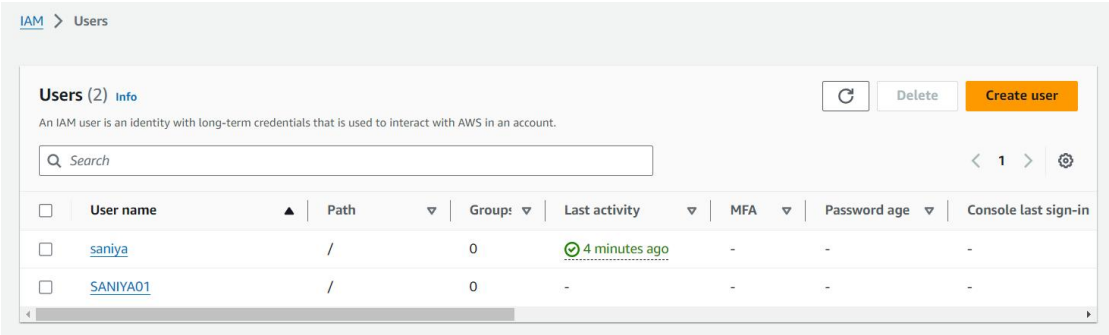
```
root@DESKTOP-VIDGD8F:Ques01# ll
total 0
drwxrwxrwx 1 saniya saniya 512 Dec 11 16:06 .
drwxrwxrwx 1 saniya saniya 512 Dec 11 14:55 ..
-rwxrwxrwx 1 saniya saniya 199 Dec 11 16:21 main.tf
-rwxrwxrwx 1 saniya saniya 171 Dec 11 16:04 provider.tf
-rwxrwxrwx 1 saniya saniya  36 Dec 11 16:21 saniya.tfvars
-rwxrwxrwx 1 saniya saniya  35 Dec 11 16:20 variables.tf
```

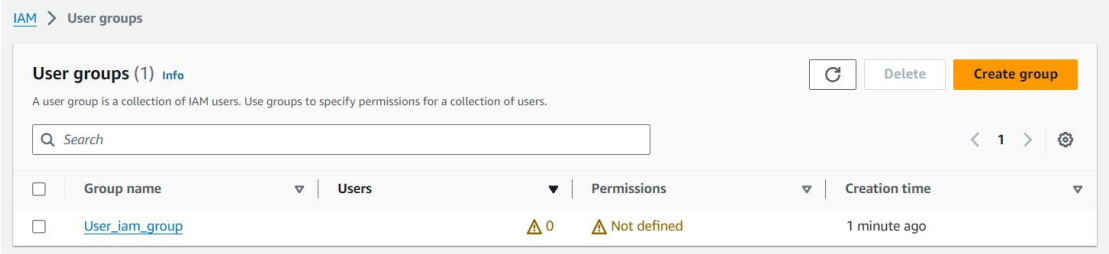
```
root@DESKTOP-VIDGD8F:Ques01# terraform init

Initializing the backend...

Initializing provider plugins...
- Finding hashicorp/aws versions matching "5.30.0"...
- Installing hashicorp/aws v5.30.0...
- Installed hashicorp/aws v5.30.0 (signed by HashiCorp)

Terraform has created a lock file .terraform.lock.hcl to record the provider
selections it made above. Include this file in your version control repository
so that Terraform can guarantee to make the same selections by default when
you run "terraform init" in the future.

Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.

If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
```

```
root@DESKTOP-VIDGD8F:Ques01# terraform apply -var-file=./saniya.tfvars

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols:
  + create

Terraform will perform the following actions:

  # aws_iam_group.developers will be created
  + resource "aws_iam_group" "developers" {
      + arn       = (known after apply)
      + id        = (known after apply)
      + name      = "User_iam_group"
      + path      = "/"
      + unique_id = (known after apply)
    }

  # aws_iam_user.my_user will be created
  + resource "aws_iam_user" "my_user" {
      + arn           = (known after apply)
      + force_destroy = false
      + id            = (known after apply)
      + name          = "SANIYA01"
      + path          = "/"
      + tags          = {
          + "tag-key" = "SANIYA"
        }
      + tags_all      = {
          + "tag-key" = "SANIYA"
        }
      + unique_id     = (known after apply)
    }

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



```
root@DESKTOP-VIDGD8F:Ques01# terraform apply -var-file=./saniya.tfvars -auto-approve

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols:
  + create

Terraform will perform the following actions:

  # aws_iam_group.developers will be created
  + resource "aws_iam_group" "developers" {
      + arn       = (known after apply)
      + id        = (known after apply)
      + name      = "User_iam_group"
      + path      = "/"
      + unique_id = (known after apply)
    }

  # aws_iam_user.my_user will be created
  + resource "aws_iam_user" "my_user" {
      + arn           = (known after apply)
      + force_destroy = false
      + id            = (known after apply)
      + name          = "SANIYA01"
      + path          = "/"
      + tags          = {
          + "tag-key" = "SANIYA"
        }
      + tags_all      = {
          + "tag-key" = "SANIYA"
        }
      + unique_id     = (known after apply)
    }

Plan: 2 to add, 0 to change, 0 to destroy.
aws_iam_group.developers: Creating...
aws_iam_user.my_user: Creating...
aws_iam_user.my_user: Creation complete after 2s [id=SANIYA01]
aws_iam_group.developers: Creation complete after 2s [id=User_iam_group]

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





02.

- **Create one EC2 Instance and Elastic IP using Terraform**
- **Map elastic IP with EC2 instance.**

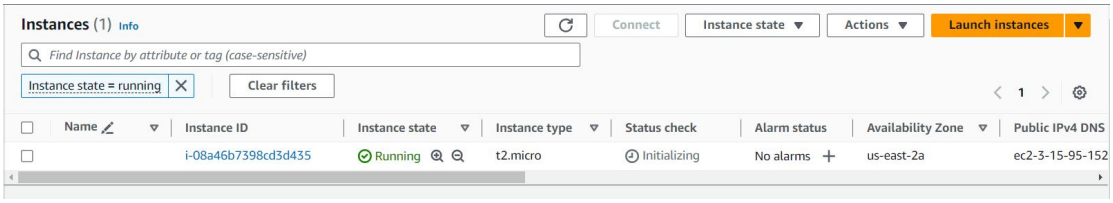
SOLUTION:-

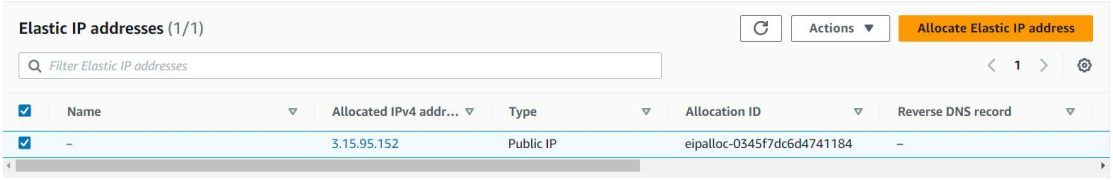
```
root@DESKTOP-VIDGD8F:Ques02# ll
total 0
drwxrwxrwx 1 saniya saniya 512 Dec 11 16:39 .
drwxrwxrwx 1 saniya saniya 512 Dec 11 14:55 ..
-rwxrwxrwx 1 saniya saniya 343 Dec 11 16:49 main.tf
-rwxrwxrwx 1 saniya saniya 171 Dec 11 16:39 provider.tf
-rwxrwxrwx 1 saniya saniya  91 Dec 11 16:52 saniya.tfvars
-rwxrwxrwx 1 saniya saniya 174 Dec 11 16:48 variable.tf
```

```
root@DESKTOP-VIDGD8F:Ques02# terraform init

Initializing the backend...

Initializing provider plugins...
- Finding hashicorp/aws versions matching "5.30.0"...
- Installing hashicorp/aws v5.30.0...
- Installed hashicorp/aws v5.30.0 (signed by HashiCorp)

Terraform has created a lock file .terraform.lock.hcl to record the provider
selections it made above. Include this file in your version control repository
so that Terraform can guarantee to make the same selections by default when
you run "terraform init" in the future.

Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.

If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
root@DESKTOP-VIDGD8F:Ques02#
```

```
root@DESKTOP-VIDGD8F:Ques02# terraform plan -var-file=./saniya.tfvars

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following
symbols:
  + create

Terraform will perform the following actions:

  # aws_eip.lb will be created
  + resource "aws_eip" "lb" {
      + allocation_id        = (known after apply)
      + association_id       = (known after apply)
      + carrier_ip           = (known after apply)
      + customer_owned_ip    = (known after apply)
      + domain               = "vpc"
      + id                   = (known after apply)
      + instance             = (known after apply)
      + network_border_group = (known after apply)
      + network_interface    = (known after apply)
      + private_dns          = (known after apply)
      + private_ip           = (known after apply)
      + public_dns           = (known after apply)
      + public_ip            = (known after apply)
      + public_ipv4_pool     = (known after apply)
      + tags_all             = (known after apply)
      + vpc                  = (known after apply)
    }
```

```
root@DESKTOP-VIDGD8F:Ques02# terraform apply -var-file=./saniya.tfvars -auto-approve
aws_key_pair.aws_3: Refreshing state... [id=AWS]

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the followi
symbols:
  + create

Terraform will perform the following actions:

  # aws_eip.lb will be created
  + resource "aws_eip" "lb" {
      + allocation_id        = (known after apply)
      + association_id       = (known after apply)
      + carrier_ip           = (known after apply)
      + customer_owned_ip    = (known after apply)
      + domain               = "vpc"
      + id                   = (known after apply)
      + instance             = (known after apply)
      + network_border_group = (known after apply)
      + network_interface    = (known after apply)
      + private_dns          = (known after apply)
      + private_ip           = (known after apply)
      + public_dns           = (known after apply)
      + public_ip            = (known after apply)
      + public_ipv4_pool     = (known after apply)
      + tags_all             = (known after apply)
      + vpc                  = (known after apply)
    }

  # aws_instance.AWS_INSTANCE will be created
  + resource "aws_instance" "AWS_INSTANCE" {
      + ami                                  = "ami-06d4b7182ac3480fa"
      + arn                                  = (known after apply)
```



03.

- **Create AWS VPC with Terraform.**
- **Please follow the given link for more on AWS VPC creation.**
    1. **Create a VPC.**
    2. **Create 2 Public Subnet & Create 2 Private Subnet.**
    3. **Create IGW (Internet Gateway) & Attach to the VPC.**
    4. **Create Public and Private Route Table.**
    5. **Add IGW in Public Route table (0.0.0.0/0).**
    6. **Add Public Subnet (1a & 1b) in Route table.**
    7. **Create a NAT Gateway in Public Subnet.**
    8. **Add NAT GW into the Private Route Table.**
    9. **Add Private Subnet in Private Route Table.**

SOLUTION:-

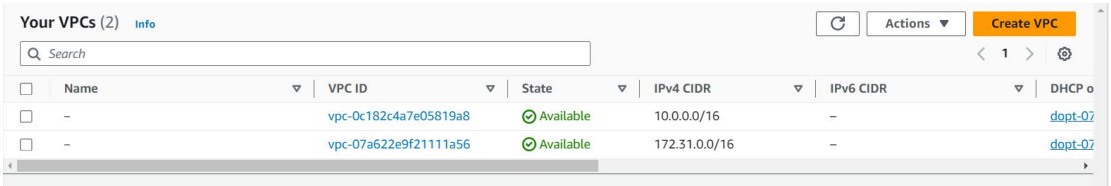
```
root@DESKTOP-VIDGD8F:Ques03# ll
total 0
drwxrwxrwx 1 saniya saniya 512 Dec 11 17:21 .
drwxrwxrwx 1 saniya saniya 512 Dec 11 14:55 ..
-rwxrwxrwx 1 saniya saniya   0 Dec 11 17:20 main.tf
-rwxrwxrwx 1 saniya saniya   0 Dec 11 17:20 provider.tf
-rwxrwxrwx 1 saniya saniya   0 Dec 11 17:21 saniya.tfvars
-rwxrwxrwx 1 saniya saniya   0 Dec 11 17:20 variable.tf
root@DESKTOP-VIDGD8F:Ques03#
```

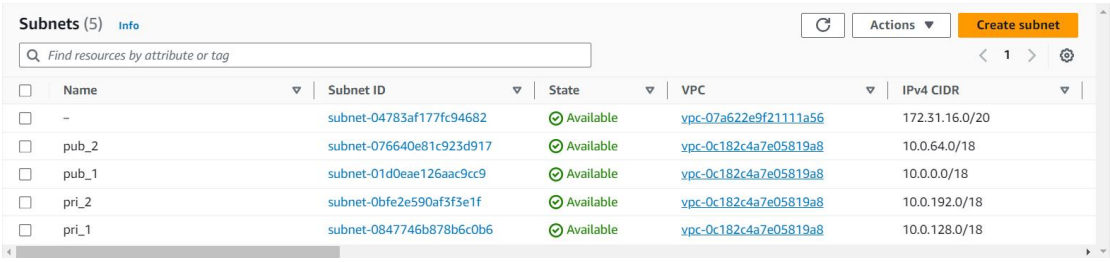
```
root@DESKTOP-VIDGD8F:Ques03# terraform init

Initializing the backend...

Initializing provider plugins...
- Finding hashicorp/aws versions matching "5.30.0"...
- Installing hashicorp/aws v5.30.0...
- Installed hashicorp/aws v5.30.0 (signed by HashiCorp)

Terraform has created a lock file .terraform.lock.hcl to record the provider
selections it made above. Include this file in your version control repository
so that Terraform can guarantee to make the same selections by default when
you run "terraform init" in the future.

Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.

If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
```

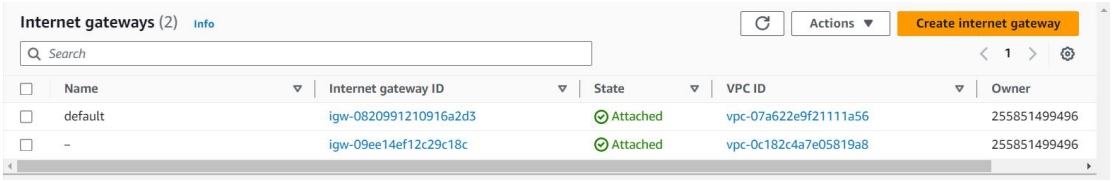
```
root@DESKTOP-VIDGD8F:Ques03# terraform plan

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following
symbols:
  + create

Terraform will perform the following actions:

  # aws_eip.aws_eip will be created
  + resource "aws_eip" "aws_eip" {
      + allocation_id        = (known after apply)
      + association_id       = (known after apply)
      + carrier_ip           = (known after apply)
      + customer_owned_ip    = (known after apply)
      + domain               = "vpc"
      + id                   = (known after apply)
      + instance             = (known after apply)
      + network_border_group = (known after apply)
      + network_interface    = (known after apply)
      + private_dns          = (known after apply)
      + private_ip           = (known after apply)
      + public_dns           = (known after apply)
      + public_ip            = (known after apply)
      + public_ipv4_pool     = (known after apply)
      + tags_all             = (known after apply)
      + vpc                  = (known after apply)
    }

  # aws_internet_gateway.AWS_igw will be created
  + resource "aws_internet_gateway" "AWS_igw" {
      + arn      = (known after apply)
      + id       = (known after apply)
      + owner_id = (known after apply)
```

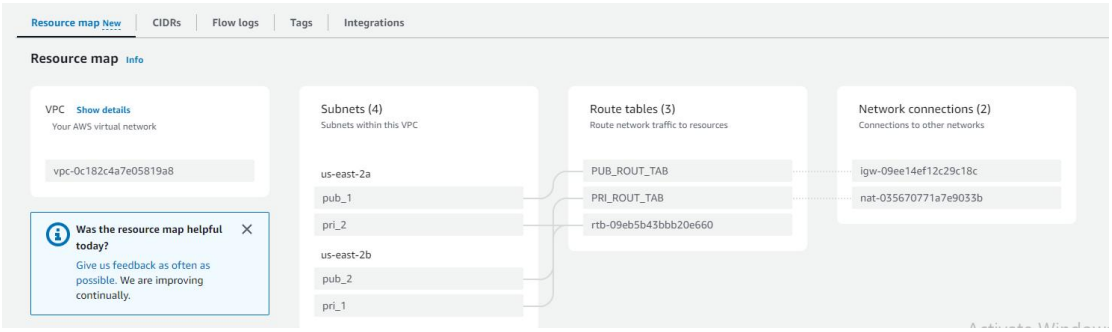


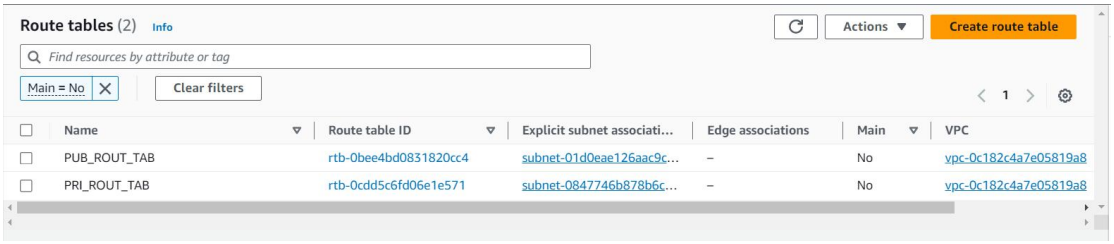
```
root@DESKTOP-VIDGD8F:Ques03# terraform apply
aws_eip.aws_eip: Refreshing state... [id=eipalloc-0f87ba545c864707a]
aws_vpc.data: Refreshing state... [id=vpc-0c182c4a7e05819a8]
aws_route_table.pub_rot_tab: Refreshing state... [id=rtb-0bee4bd0831820cc4]
aws_subnet.private01: Refreshing state... [id=subnet-0847746b878b6c0b6]
aws_internet_gateway.AWS_igw: Refreshing state... [id=igw-09ee14ef12c29c18c]
aws_route_table.pri_rot_tab: Refreshing state... [id=rtb-0cdd5c6fd06e1e571]
aws_subnet.private02: Refreshing state... [id=subnet-0bfe2e590af3f3e1f]
aws_subnet.public01: Refreshing state... [id=subnet-01d0eae126aac9cc9]
aws_nat_gateway.aws_nat: Refreshing state... [id=nat-035670771a7e9033b]
aws_route_table_association.pri_rot_aso1: Refreshing state... [id=rtbassoc-01c07e43a4c12dc74]
aws_route_table_association.pub_rot_aso2: Refreshing state... [id=rtbassoc-0ff99539e28192cc6]
aws_route.pub_route: Refreshing state... [id=r-rtb-0bee4bd0831820cc41080289494]
aws_route.pri_route: Refreshing state... [id=r-rtb-0cdd5c6fd06e1e5711080289494]

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following
symbols:
  + create
  ~ update in-place

Terraform will perform the following actions:

  # aws_route.pri_route will be updated in-place
  ~ resource "aws_route" "pri_route" {
      + gateway_id             = "nat-035670771a7e9033b"
        id                     = "r-rtb-0cdd5c6fd06e1e5711080289494"
      - nat_gateway_id         = "nat-035670771a7e9033b" -> null
        # (4 unchanged attributes hidden)
    }

  # aws_subnet.public02 will be created
  + resource "aws_subnet" "public02" {
      + arn                                            = (known after apply)
```

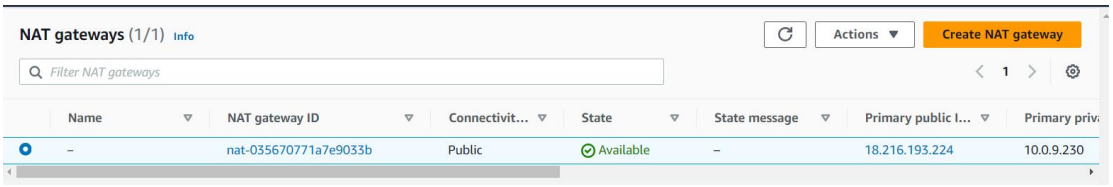



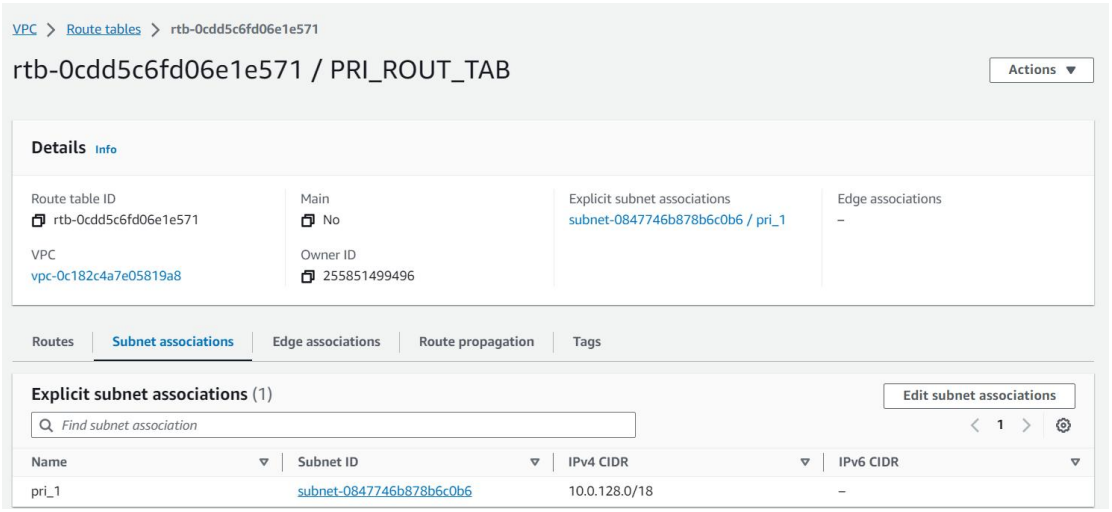


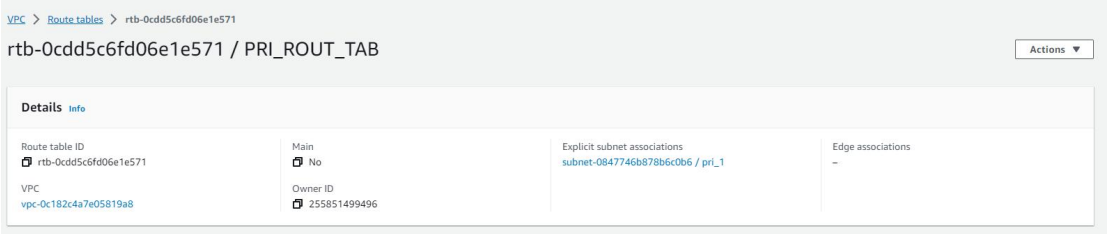

Internet gateways (2) Info
Actions
Create internet gateway
Search
Name
Internet gateway ID
State
VPC ID
Owner
default
igw-0820991210916a2d3
Attached
vpc-07a622e9f21111a56
255851499496
–
igw-09ee14ef12c29c18c
Attached
vpc-0c182c4a7e05819a8
255851499496


Resource map New
CIDRs
Flow logs
Tags
Integrations
Resource map Info
VPC Show details
Your AWS virtual network
vpc-0c182c4a7e05819a8
Was the resource map helpful today?
Give us feedback as often as possible. We are improving continually.
Subnets (4)
Subnets within this VPC
us-east-2a
pub_1
pri_2
us-east-2b
pub_2
pri_1
Route tables (3)
Route network traffic to resources
PUB_ROUT_TAB
PRI_ROUT_TAB
rtb-09eb5b43bbb20e660
Network connections (2)
Connections to other networks
igw-09ee14ef12c29c18c
nat-035670771a7e9033b


Route tables (2) Info
Actions
Create route table
Find resources by attribute or tag
Main = No
Clear filters
Name
Route table ID
Explicit subnet associati...
Edge associations
Main
VPC
PUB_ROUT_TAB
rtb-0bee4bd0831820cc4
subnet-01d0eae126aac9c...
–
No
vpc-0c182c4a7e05819a8
PRI_ROUT_TAB
rtb-0cdd5c6fd06e1e571
subnet-0847746b878b6c...
–
No
vpc-0c182c4a7e05819a8


NAT gateways (1/1) Info
Actions
Create NAT gateway
Filter NAT gateways
Name
NAT gateway ID
Connectivit...
State
State message
Primary public I...
Primary priv
–
nat-035670771a7e9033b
Public
Available
–
18.216.193.224
10.0.9.230


VPC > Route tables > rtb-0cdd5c6fd06e1e571
rtb-0cdd5c6fd06e1e571 / PRI_ROUT_TAB
Actions
Details Info
Route table ID
rtb-0cdd5c6fd06e1e571
Main
No
Explicit subnet associations
subnet-0847746b878b6c0b6 / pri_1
Edge associations
–
VPC
vpc-0c182c4a7e05819a8
Owner ID
255851499496
Routes
Subnet associations
Edge associations
Route propagation
Tags
Explicit subnet associations (1)
Edit subnet associations
Find subnet association
Name
Subnet ID
IPv4 CIDR
IPv6 CIDR
pri_1
subnet-0847746b878b6c0b6
10.0.128.0/18
–



04.

● Create EC2 instance one of the public Subnets of VPC that you have created

● Validate your Connection using SSH.

SOLUTION:-

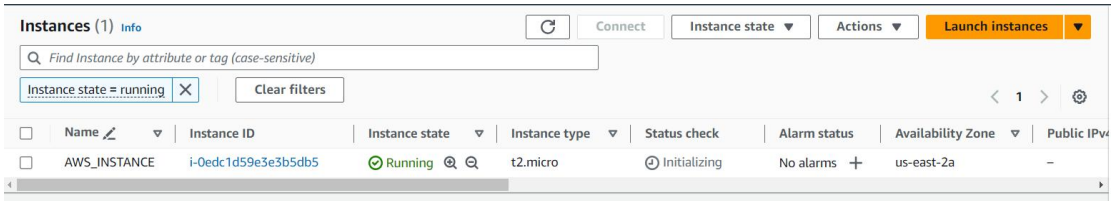
```
root@DESKTOP-VIDGD8F:Ques04# ll
total 0
drwxrwxrwx 1 saniya saniya 512 Dec 11 18:46 .
drwxrwxrwx 1 saniya saniya 512 Dec 11 14:55 ..
-rwxrwxrwx 1 saniya saniya   0 Dec 11 18:46 main.tf
-rwxrwxrwx 1 saniya saniya   0 Dec 11 18:46 provider.tf
-rwxrwxrwx 1 saniya saniya   0 Dec 11 18:46 saniya.tfvars
-rwxrwxrwx 1 saniya saniya   0 Dec 11 18:46 variable.tf
```

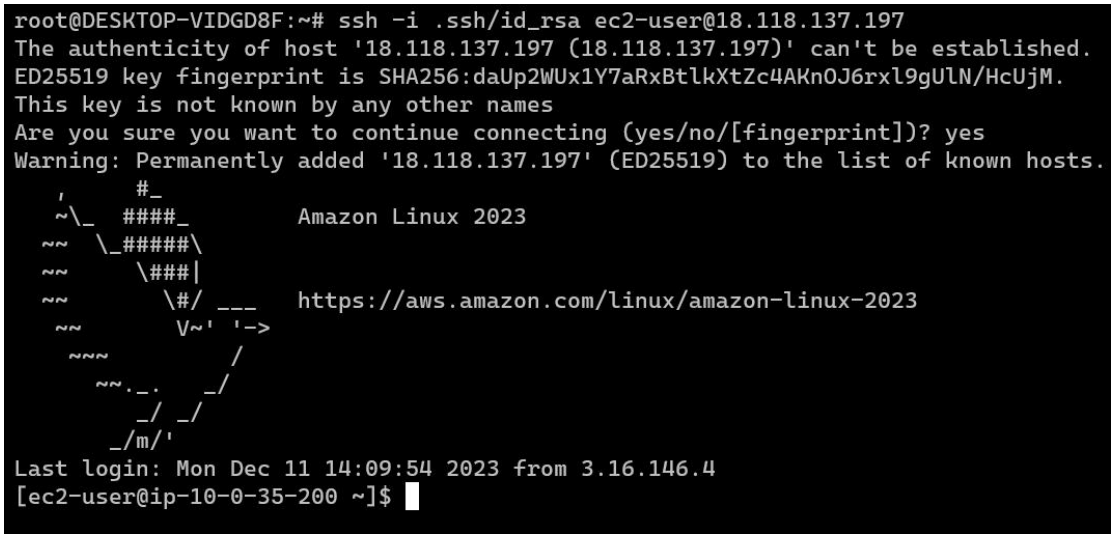
```
root@DESKTOP-VIDGD8F:Ques04# terraform init

Initializing the backend...

Initializing provider plugins...
- Finding hashicorp/aws versions matching "5.30.0"...
- Installing hashicorp/aws v5.30.0...
- Installed hashicorp/aws v5.30.0 (signed by HashiCorp)

Terraform has created a lock file .terraform.lock.hcl to record the provider
selections it made above. Include this file in your version control repository
so that Terraform can guarantee to make the same selections by default when
you run "terraform init" in the future.

Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.

If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
```

```
root@DESKTOP-VIDGD8F:Ques04# terraform plan -var-file=./saniya.tfvars

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following
symbols:
  + create

Terraform will perform the following actions:

  # aws_eip.aws_eip will be created
  + resource "aws_eip" "aws_eip" {
      + allocation_id        = (known after apply)
      + association_id       = (known after apply)
      + carrier_ip           = (known after apply)
      + customer_owned_ip    = (known after apply)
      + domain               = "vpc"
      + id                   = (known after apply)
      + instance             = (known after apply)
      + network_border_group = (known after apply)
      + network_interface    = (known after apply)
      + private_dns          = (known after apply)
      + private_ip           = (known after apply)
      + public_dns           = (known after apply)
      + public_ip            = (known after apply)
      + public_ipv4_pool     = (known after apply)
      + tags_all             = (known after apply)
      + vpc                  = (known after apply)
    }

  # aws_instance.WEB_INSTANCE will be created
  + resource "aws_instance" "WEB_INSTANCE" {
      + ami                                  = "ami-06d4b7182ac3480fa"
      + arn                                  = (known after apply)
      + associate_public_ip_address          = true
```



```
root@DESKTOP-VIDGD8F:Ques04# terraform apply -var-file=./saniya.tfvars
aws_key_pair.aws_3: Refreshing state... [id=AWS_KEY]
aws_eip.aws_eip: Refreshing state... [id=eipalloc-0635a1beb6daf20ec]
aws_vpc.data: Refreshing state... [id=vpc-08c4c9c90f37f5a97]
aws_route_table.pri_rot_tab: Refreshing state... [id=rtb-0052ebd5fa82f292f]
aws_internet_gateway.AWS_igw: Refreshing state... [id=igw-092acf62d47028b81]
aws_route_table.pub_rot_tab: Refreshing state... [id=rtb-0ec6e7131062ffc8b]
aws_subnet.public01: Refreshing state... [id=subnet-0c8611c9ad4260202]
aws_subnet.public02: Refreshing state... [id=subnet-03ff7205a9169fd28]
aws_subnet.private01: Refreshing state... [id=subnet-040bee3bb54aa4f4d]
aws_subnet.private02: Refreshing state... [id=subnet-0e51048003e7ff3f1]
aws_security_group.aws_sg: Refreshing state... [id=sg-0eeb5ed343700a1a3]
aws_nat_gateway.aws_nat: Refreshing state... [id=nat-0082c401b369cf1f1]
aws_route_table_association.pri_rot_aso1: Refreshing state... [id=rtbassoc-0ef2d8895b75ef8ea]
aws_route_table_association.pub_rot_aso2: Refreshing state... [id=rtbassoc-084e758d449182e66]
aws_route.pub_route: Refreshing state... [id=r-rtb-0ec6e7131062ffc8b1080289494]
aws_route.pri_route: Refreshing state... [id=r-rtb-0052ebd5fa82f292f1080289494]
aws_instance.AWS_INSTANCE: Refreshing state... [id=i-0b5abbacd1521b7f5]

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following
symbols:
  ~ update in-place
-/+ destroy and then create replacement

Terraform will perform the following actions:

  # aws_instance.AWS_INSTANCE must be replaced
-/+ resource "aws_instance" "AWS_INSTANCE" {
      ~ arn                                  = "arn:aws:ec2:us-east-2:255851499496:instance/i-0b5abbacd1521b7f5" -> (known after apply
)
      ~ availability_zone                    = "us-east-2a" -> (known after apply)
      ~ cpu_core_count                       = 1 -> (known after apply)
      ~ cpu_threads_per_core                 = 1 -> (known after apply)
```



GIT HUB LINK TO ASSIGNMENT 01
https://github.com/Saniya2822/TERRAFORM-.git