

Inbox (16,273)

Start Course -

Your VPCs | VP

Instances | EC

Start Course X

Terraform-case-St

Start Course -

vpc - Google L

Support Ticke

PNG to PDF -

+

←

→

↺

🔒

https://lms.intellipaat.com/start-course/

☆

📧

📄

☰

←

Back

Case Study – Terraform

Notes

SELF-PACED

LIVE CLASSES

STUDY MATERIALS

🔍

Hide

📄

Assignment 1 – Terraform

📄

Assignment 2 – Terraform

📄

Assignment 3 – Terraform

📄

Assignment 4 – Terraform

📄

Assignment 5 – Terraform

📄

Case Study – Terraform

ELK

▼

Nagios

▼


Docker II

▼

Projects

▼

DevOps Certification Training



CASE STUDY -CREATING AN ARCHITECTURE USING TERRAFORM ON AWS

You work as a DevOps Engineer in leading Software Company. You have been asked to build an infrastructure safely and efficiently.

The company Requirements:

1. Use AWS cloud Provider and the software to be installed is Apache2
2. Use Ubuntu AMI

The company wants the Architecture to have the following services:

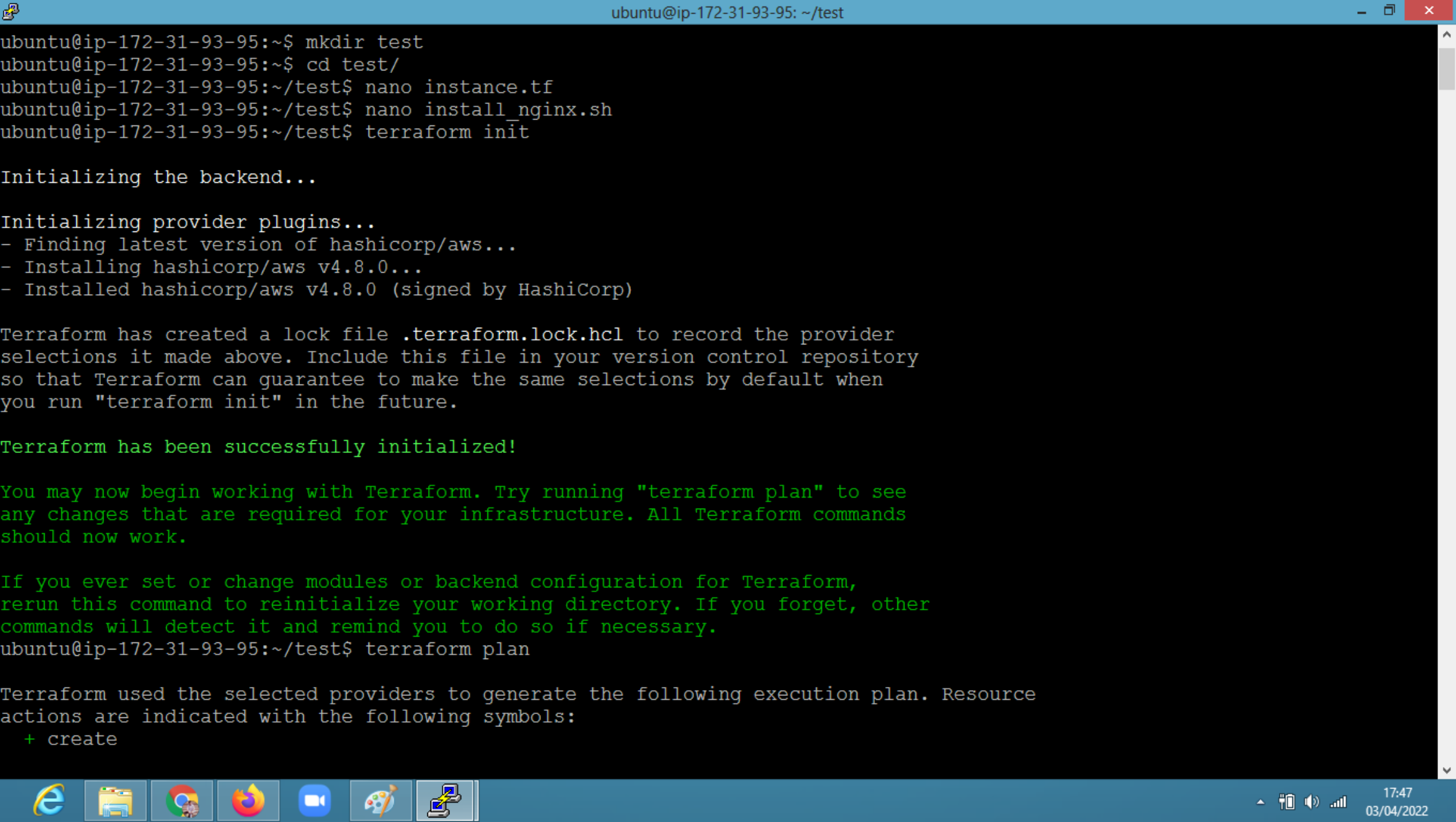
1. Create a template with a VPC, 2 subnets and 1 instance in each subnet
2. Attach Security groups, internet gateway and network interface to the instance

Download Attachment

✓

18:42

03/04/2022



```
ubuntu@ip-172-31-93-95: ~/test
ubuntu@ip-172-31-93-95:~$ mkdir test
ubuntu@ip-172-31-93-95:~$ cd test/
ubuntu@ip-172-31-93-95:~/test$ nano instance.tf
ubuntu@ip-172-31-93-95:~/test$ nano install_nginx.sh
ubuntu@ip-172-31-93-95:~/test$ terraform init

Initializing the backend...

Initializing provider plugins...
- Finding latest version of hashicorp/aws...
- Installing hashicorp/aws v4.8.0...
- Installed hashicorp/aws v4.8.0 (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.
ubuntu@ip-172-31-93-95:~/test$ terraform plan

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



```
ubuntu@ip-172-31-93-95:~$ ls
```

```
test
```

```
ubuntu@ip-172-31-93-95:~$ cd test/
```

```
ubuntu@ip-172-31-93-95:~/test$ ls
```

```
install_nginx.sh  instance.tf  terraform.tfstate  terraform.tfstate.backup
```

```
ubuntu@ip-172-31-93-95:~/test$
```



```
provider "aws" {
  access_key = "AKIAU7PYSBVTHSC6XYNM"
  secret_key = "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"
  region     = "us-east-1"
}
```

```
resource "aws_instance" "web" {
  ami= "ami-04505e74c0741db8d"
  instance_type = "t2.micro"
  key_name= "hotfix"
  user_data      = "${file("install_nginx.sh")}"
  tags = {
    Name = "web"
  }
}

output "IP" {
  value = aws_instance.web.public_ip
}
```

ar/www/html/index.html

```
ubuntu@ip-172-31-93-95:~/test$ 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_instance.web will be created

```
+ resource "aws_instance" "web" {
  + ami                  = "ami-04505e74c0741db8d"
  + arn                  = (known after apply)
  + associate_public_ip_address = (known after apply)
  + availability_zone     = (known after apply)
  + cpu_core_count        = (known after apply)
  + cpu_threads_per_core  = (known after apply)
  + disable_api_termination = (known after apply)
  + ebs_optimized         = (known after apply)
  + get_password_data      = false
  + host_id               = (known after apply)
  + id                   = (known after apply)
  + instance_initiated_shutdown_behavior = (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              = "hotfix1"
  + monitoring            = (known after apply)
  + outpost_arn           = (known after apply)
  + password_data         = (known after apply)
  + placement_group       = (known after apply)
  + placement_partition_number = (known after apply)
  + primary_network_interface_id = (known after apply)
```



```
ubuntu@ip-172-31-93-95:~/test$ nano instance.tf
ubuntu@ip-172-31-93-95:~/test$ terraform apply
```

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_instance.web will be created

```
+ resource "aws_instance" "web" {
  + ami                  = "ami-04505e74c0741db8d"
  + arn                  = (known after apply)
  + associate_public_ip_address = (known after apply)
  + availability_zone     = (known after apply)
  + cpu_core_count        = (known after apply)
  + cpu_threads_per_core  = (known after apply)
  + disable_api_termination = (known after apply)
  + ebs_optimized         = (known after apply)
  + get_password_data      = false
  + host_id               = (known after apply)
  + id                   = (known after apply)
  + instance_initiated_shutdown_behavior = (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              = "hotfix"
  + monitoring            = (known after apply)
  + outpost_arn           = (known after apply)
  + password_data         = (known after apply)
  + placement_group       = (known after apply)
  + placement_partition_number = (known after apply)
```



```
+ iops                = (known after apply)
+ kms_key_id          = (known after apply)
+ tags                = (known after apply)
+ throughput          = (known after apply)
+ volume_id           = (known after apply)
+ volume_size         = (known after apply)
+ volume_type         = (known after apply)
}
}
```

Plan: 1 to add, 0 to change, 0 to destroy.

Changes to Outputs:

```
+ 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.web: Creating...

aws_instance.web: Still creating... [10s elapsed]

aws_instance.web: Still creating... [20s elapsed]

aws_instance.web: Still creating... [30s elapsed]

aws_instance.web: Creation complete after 32s [id=i-08fa1385b71658f22]

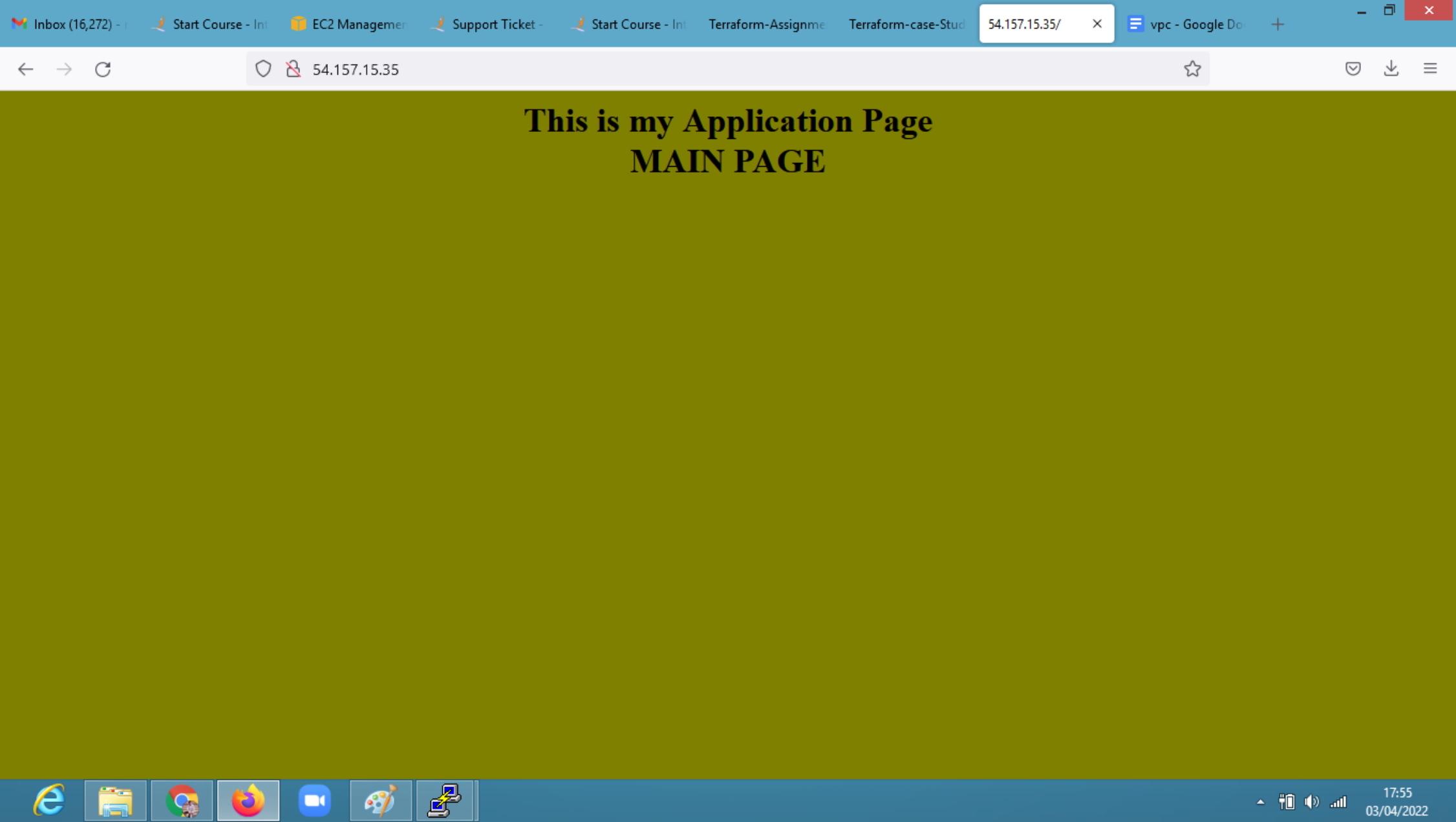
Apply complete! Resources: 1 added, 0 changed, 0 destroyed.

Outputs:

IP = "54.157.15.35"

ubuntu@ip-172-31-93-95:~/test\$





This is my Application Page

MAIN PAGE



```
ubuntu@ip-172-31-93-95:~$ ls
```

```
test
```

```
ubuntu@ip-172-31-93-95:~$ cd test/
```

```
ubuntu@ip-172-31-93-95:~/test$ ls
```

```
instance.tf  terraform.tfstate  terraform.tfstate.backup
```

```
ubuntu@ip-172-31-93-95:~/test$ vi instance.tf
```



```
ubuntu@ip-172-31-93-95:~/test$ vi instance.tf
# defining provider
provider "aws" {
  access_key = "AKIAU7PYSBVTHSC6XYNM"
  secret_key = "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"
  region     = "us-east-1"
}

# creating vpc
resource "aws_vpc" "main" {
  cidr_block = "10.0.0.0/16"
  tags={
    Name = "demo_vpc_terraform"
  }
}

# creating a public subnet in vpc
resource "aws_subnet" "public_subnet" {
  vpc_id      = aws_vpc.main.id
  cidr_block  = "10.0.1.0/24"
  availability_zone = "us-east-1a"
  map_public_ip_on_launch = "1"
  tags = {
    Name = "public_tf_subnet"
  }
}

# creating a sg and associating it with the vpc
resource "aws_security_group" "securtiy_group" {
  vpc_id = aws_vpc.main.id

  ingress {
    from_port = 0
```



```
ingress {
  from_port      = 0
  to_port        = 0
  protocol       = "-1"
  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 = "tf-security_grp"
}

# creating a internet gateway and associating with vpc
resource "aws_internet_gateway" "gw" {
  vpc_id = aws_vpc.main.id

  tags = {
    Name = "tf-IGW"
  }
}

#route table for public subnet with IGW
resource "aws_route_table" "table_public" {
  vpc_id = "${aws_vpc.main.id}"
  route {
    cidr_block = "0.0.0.0/0"
  }
}
```



```
vpc_id = "${aws_vpc.main.id}"
route {
  cidr_block = "0.0.0.0/0"
  gateway_id = "${aws_internet_gateway.gw.id}"
}
tags = {
  Name = "rt_public"
}
}
# route table association public subnet

resource "aws_route_table_association" "association_rt_public" {
  subnet_id      = aws_subnet.public_subnet.id
  route_table_id = aws_route_table.table_public.id
}

# launch an instance
resource "aws_instance" "web" {
  ami           = "ami-04505e74c0741db8d"
  instance_type = "t2.micro"
  vpc_security_group_ids = ["${aws_security_group.securtiy_group.id}"]
  subnet_id = aws_subnet.public_subnet.id
  key_name = "hotfix"
  tags = {
    Name = "TF_public_instance"
  }
}

output "IP" {
  value = aws_instance.web.public_ip
}
```



```
ubuntu@ip-172-31-93-95:~/test$ vi instance.tf
ubuntu@ip-172-31-93-95:~/test$ 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_instance.web will be created

```
+ resource "aws_instance" "web" {
  + ami                  = "ami-0892d3c7ee96c0bf7"
  + arn                  = (known after apply)
  + associate_public_ip_address = (known after apply)
  + availability_zone     = (known after apply)
  + cpu_core_count        = (known after apply)
  + cpu_threads_per_core   = (known after apply)
  + disable_api_termination = (known after apply)
  + ebs_optimized          = (known after apply)
  + get_password_data      = false
  + host_id                = (known after apply)
  + id                    = (known after apply)
  + instance_initiated_shutdown_behavior = (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               = "hotfix"
  + monitoring              = (known after apply)
  + outpost_arn             = (known after apply)
  + password_data          = (known after apply)
  + placement_group         = (known after apply)
  + placement_partition_number = (known after apply)
```



```
ubuntu@ip-172-31-93-95:~/test$ vi instance.tf
ubuntu@ip-172-31-93-95:~/test$ terraform apply
aws_vpc.main: Refreshing state... [id=vpc-0c8e583d6dd73daef]
aws_security_group.securtiy_group: Refreshing state... [id=sg-03fbbcf649a8773ad]
aws_internet_gateway.gw: Refreshing state... [id=igw-089cd4b0d7c86325a]
aws_subnet.public_subnet: Refreshing state... [id=subnet-0fa2149230acf222c]
aws_route_table.table_public: Refreshing state... [id=rtb-09bc70de9499ee73f]
aws_route_table_association.association_rt_public: Refreshing state... [id=rtbassoc-02c14c0a823117735]
```

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_instance.web will be created
+ resource "aws_instance" "web" {
  + ami                  = "ami-04505e74c0741db8d"
  + arn                  = (known after apply)
  + associate_public_ip_address = (known after apply)
  + availability_zone     = (known after apply)
  + cpu_core_count        = (known after apply)
  + cpu_threads_per_core  = (known after apply)
  + disable_api_termination = (known after apply)
  + ebs_optimized         = (known after apply)
  + get_password_data      = false
  + host_id               = (known after apply)
  + id                    = (known after apply)
  + instance_initiated_shutdown_behavior = (known after apply)
  + instance_state        = (known after apply)
  + instance_type         = "t2.micro"
  + ipv6_address_count     = (known after apply)
  + ipv6_addresses        = (known after apply)
```



```
+ kms_key_id      = (known after apply)
+ tags            = (known after apply)
+ throughput      = (known after apply)
+ volume_id       = (known after apply)
+ volume_size     = (known after apply)
+ volume_type     = (known after apply)
}
}
```

Plan: 1 to add, 0 to change, 0 to destroy.

Changes to Outputs:

```
+ 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.web: Creating...

aws_instance.web: Still creating... [10s elapsed]

aws_instance.web: Still creating... [20s elapsed]

aws_instance.web: Still creating... [30s elapsed]

aws_instance.web: Still creating... [40s elapsed]

aws_instance.web: Creation complete after 42s [id=i-0862948a839e553b2]


Apply complete! Resources: 1 added, 0 changed, 0 destroyed.

Outputs:

IP = "18.209.29.21"

ubuntu@ip-172-31-93-95:~/test\$



 **New VPC Experience**
Tell us what you think

- VPC Dashboard
- EC2 Global View New
- Filter by VPC:
- VIRTUAL PRIVATE CLOUD**
 - Your VPCs**
 - Subnets
 - Route Tables
 - Internet Gateways
 - Egress Only Internet Gateways
 - Carrier Gateways
 - DHCP Options Sets
 - Elastic IPs
 - Managed Prefix Lists
 - Endpoints New

Your VPCs (1/3) Info



🔍 *Filter VPCs*

<input type="checkbox"/>	-	vpc-07422ea0339907ce5	Available	172.31.0.0/16	-
<input type="checkbox"/>	3-tier architecture	vpc-03b703bd043536472	Available	10.0.0.0/16	-
<input checked="" type="checkbox"/>	demo_vpc_terraform	vpc-0c8e583d6dd73daef	Available	10.0.0.0/16	-

vpc-0c8e583d6dd73daef / demo_vpc_terraform

Details CIDRs Flow logs Tags

Details

VPC ID	State	DNS hostnames	DNS resolution
 vpc-0c8e583d6dd73daef	 Available	Disabled	Enabled
Tenancy	DHCP options set	Main route table	Main network ACL
Default	dopt-0ea3b22fdafb8e6e7	rtb-0753ab29e55edd3a9	acl-0aef684528c64ed51
Default VPC	IPv4 CIDR	IPv6 pool	IPv6 CIDR (Network border group)

AWS Management Console

Search for services, features, blogs, docs, and more [Alt+S]

N. Virginia | aarambh-user1 @ 3425-0804-7718

New EC2 Experience

EC2 Dashboard
EC2 Global View
Events
Tags
Limits

Instances

Instances New

Instance Types
Launch Templates
Spot Requests
Savings Plans
Reserved Instances New
Dedicated Hosts
Scheduled Instances
Capacity Reservations

Images

Instances (1/5) Info

Search

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone
web	i-08fa1385b71658f22	Terminated	t2.micro	-	No alarms +	us-east-1d
Docker	i-063a1a348b7acabae	Stopped	t2.micro	-	No alarms +	us-east-1a
<input checked="" type="checkbox"/> TF_public_inst...	i-0862948a839e553b2	✔ Running	t2.micro	🕒 Initializing	No alarms +	us-east-1a
Terraform-use	i-095e77b80bc648464	✔ Running	t2.micro	✔ 2/2 checks passed	No alarms +	us-east-1c
jenkins_instance	i-0464028f598ff7372	Terminated	t2.micro	-	No alarms +	us-east-1c

Instance: i-0862948a839e553b2 (TF_public_instance)

Select an instance above

- Details
- Security
- Networking
- Storage
- Status checks
- Monitoring
- Tags

Instance summary Info

Instance ID

i-0862948a839e553b2 (TF_public_instance)

IPv6 address

Private IPv4 addresses

10.0.1.235

Public IPv4 DNS

Public IPv4 address copied

18.209.29.21 | open address

Instance state

✔ Running

Feedback English (US)

© 2022, Amazon Internet Services Private Ltd. or its affiliates. Privacy Terms Cookie preferences

18:54 03/04/2022

The screenshot displays the AWS Management Console's EC2 Instances page. The left sidebar shows navigation options like EC2 Dashboard, Events, Tags, Limits, and Instances. The main area lists several instances, with 'TF_public_inst...' selected. Below the list, the details for instance 'i-0862948a839e553b2' are shown, including its ID, state (Running), type (t2.micro), and IP addresses. A tooltip indicates that the public IPv4 address has been copied.

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone
web	i-08fa1385b71658f22	Terminated	t2.micro	-	No alarms	us-east-1d
Docker	i-063a1a348b7acabae	Stopped	t2.micro	-	No alarms	us-east-1a
TF_public_inst...	i-0862948a839e553b2	Running	t2.micro	Initializing	No alarms	us-east-1a
Terraform-use	i-095e77b80bc648464	Running	t2.micro	2/2 checks passed	No alarms	us-east-1c
jenkins_instance	i-0464028f598ff7372	Terminated	t2.micro	-	No alarms	us-east-1c

Instance: i-0862948a839e553b2 (TF_public_instance)

Select an instance above

- Details** | Security | Networking | Storage | Status checks | Monitoring | Tags

Instance summary Info

Instance ID

i-0862948a839e553b2 (TF_public_instance)

IPv6 address

-

Public IPv4 address copied

18.209.29.21 | open address

Instance state

Running

Private IPv4 addresses

10.0.1.235

Public IPv4 DNS

-

The screenshot displays the AWS Management Console's EC2 Instances page. The left sidebar shows navigation options like "EC2 Dashboard", "Events", "Tags", "Limits", and "Instances". The main area lists several instances, with "TF_public_inst..." selected. Below the list, the details for instance "i-0862948a839e553b2" are shown, including its ID, state (Running), type (t2.micro), and IP addresses. A tooltip indicates that the public IPv4 address has been copied.

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone
web	i-08fa1385b71658f22	Terminated	t2.micro	-	No alarms	us-east-1d
Docker	i-063a1a348b7acabae	Stopped	t2.micro	-	No alarms	us-east-1a
TF_public_inst...	i-0862948a839e553b2	Running	t2.micro	Initializing	No alarms	us-east-1a
Terraform-use	i-095e77b80bc648464	Running	t2.micro	2/2 checks passed	No alarms	us-east-1c
jenkins_instance	i-0464028f598ff7372	Terminated	t2.micro	-	No alarms	us-east-1c

Instance: i-0862948a839e553b2 (TF_public_instance)

Select an instance above

Details	Security	Networking	Storage	Status checks	Monitoring	Tags
Instance summary Info						
Instance ID i-0862948a839e553b2 (TF_public_instance)		Public IPv4 address 18.209.29.21 open address			Private IPv4 addresses 10.0.1.235	
IPv6 address -		Instance state Running			Public IPv4 DNS -	

Inbox (16,274)

Start Course -

VPC ManagX

Instances | EC

Start Course -

Terraform-case-S

Start Course -

vpc - Google

Support Ticke

PNG to PDF -

+

←

→

↺

🔒

https://us-east-1.console.aws.amazon.com/vpc/home?region=us-east-1#securityGroups:

☆

📧

📄

☰

aws

Services

🔍

Search for services, features, blogs, docs, and more

[Alt+S]

📺

🔔

?

N. Virginia ▾

aarambh-user1 @ 3425-0804-7718 ▾

New VPC Experience

Tell us what you think

Managed Prefix Lists

Endpoints New

Endpoint Services

NAT Gateways

Peering Connections

▼ SECURITY

Network ACLs

Security Groups

▼ NETWORK ANALYSIS

Reachability Analyzer

Network Access Analyzer

▼ DNS FIREWALL

Rule Groups New

Domain Lists New

▼ NETWORK FIREWALL

Firewalls

Firewall Policies

Security Groups (1/18) Info

🔄

Actions ▾

Export security groups to CSV ▾

Create security group

🔍 Filter security groups

< 1 > ⚙️

<input type="checkbox"/>	Name ▾	Security group ID ▾	Security group name ▾	VPC ID ▾	Description ▾	Owner
<input checked="" type="checkbox"/>	tf-security_grp	sg-03fbbcf649a8773ad	terraform-202204031...	vpc-0c8e583d6dd73daef	Managed by Terraform	342508047718

Details

Inbound rules

Outbound rules

Tags

Details

Security group name

📄 terraform-20220403130513680000000001

Security group ID

📄 sg-03fbbcf649a8773ad

Description

📄 Managed by Terraform

VPC ID

📄 vpc-0c8e583d6dd73daef

Owner

📄 342508047718

Inbound rules count

1 Permission entry

Outbound rules count

1 Permission entry

Feedback

English (US) ▾

© 2022, Amazon Internet Services Private Ltd. or its affiliates.

Privacy

Terms

Cookie preferences

e

📁

🌐

🦊

💬

🎨

🔌

18:57

03/04/2022

Inbox (16,274)

Start Course -

VPC Manager

Network int X

Start Course -

Terraform-case-S

Start Course -

vpc - Google L

Support Ticke

PNG to PDF -

+

←

→

↺

🔒

https://us-east-1.console.aws.amazon.com/ec2/v2/home?region=us-east-1#NIC:

☆

📧

📄

☰

aws

Services

Search for services, features, blogs, docs, and more

[Alt+S]

📺

🔔

?

N. Virginia ▾

aarambh-user1 @ 3425-0804-7718 ▾

Scheduled Instances

Capacity Reservations

▼ Images

AMIs New

AMI Catalog

▼ Elastic Block Store

Volumes New

Snapshots New

Lifecycle Manager New

▼ Network & Security

Security Groups

Elastic IPs

Placement Groups

Key Pairs

Network Interfaces

▼ Load Balancing

Load Balancers

Network interfaces (1/10) Info

🔄

Actions ▾

Create network interface

🔍 Filter network interfaces

< 1 > ⚙️

☐	Name ▾	Network interface ID ▾	Subnet ID ▾	VPC ID ▾	Availability Zone ▾
☑	Terraform-use	eni-070973f42a77d819f	subnet-0224f7014f970ee00 🔗	vpc-07422ea0339907ce5 🔗	us-east-1c

< ▬ >

Details

Flow logs

Tags

Details

▼ Network interface details

Network interface ID	Name	Description
📄 eni-070973f42a77d819f	📄 Terraform-use	-
Network interface status	Interface type	Security groups
🟢 In-use	📄 Elastic network interface	📄 sg-00c36799c2c7a977d (launch-wizard-7)
VPC ID	Subnet ID	Availability Zone
vpc-07422ea0339907ce5 🔗	subnet-0224f7014f970ee00 🔗	📄 us-east-1c

Feedback

English (US) ▾

© 2022, Amazon Internet Services Private Ltd. or its affiliates.

Privacy

Terms

Cookie preferences

🌐

📁

🌐

🦊

🗣️

🎨

⚡

🔼

📶

🔊

📶

19:00

03/04/2022