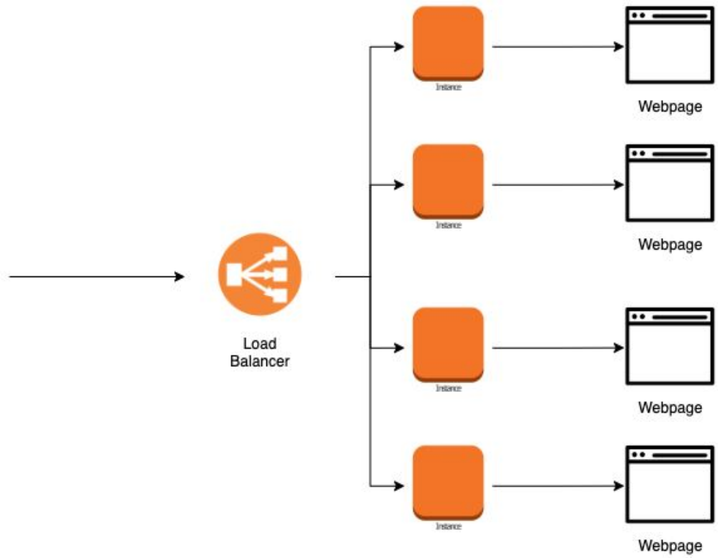
Create a Load Balanced Website

SWEN 514/614: Engineering Cloud Software Systems

Arjun Ravikumar, Tejal Shanbhag

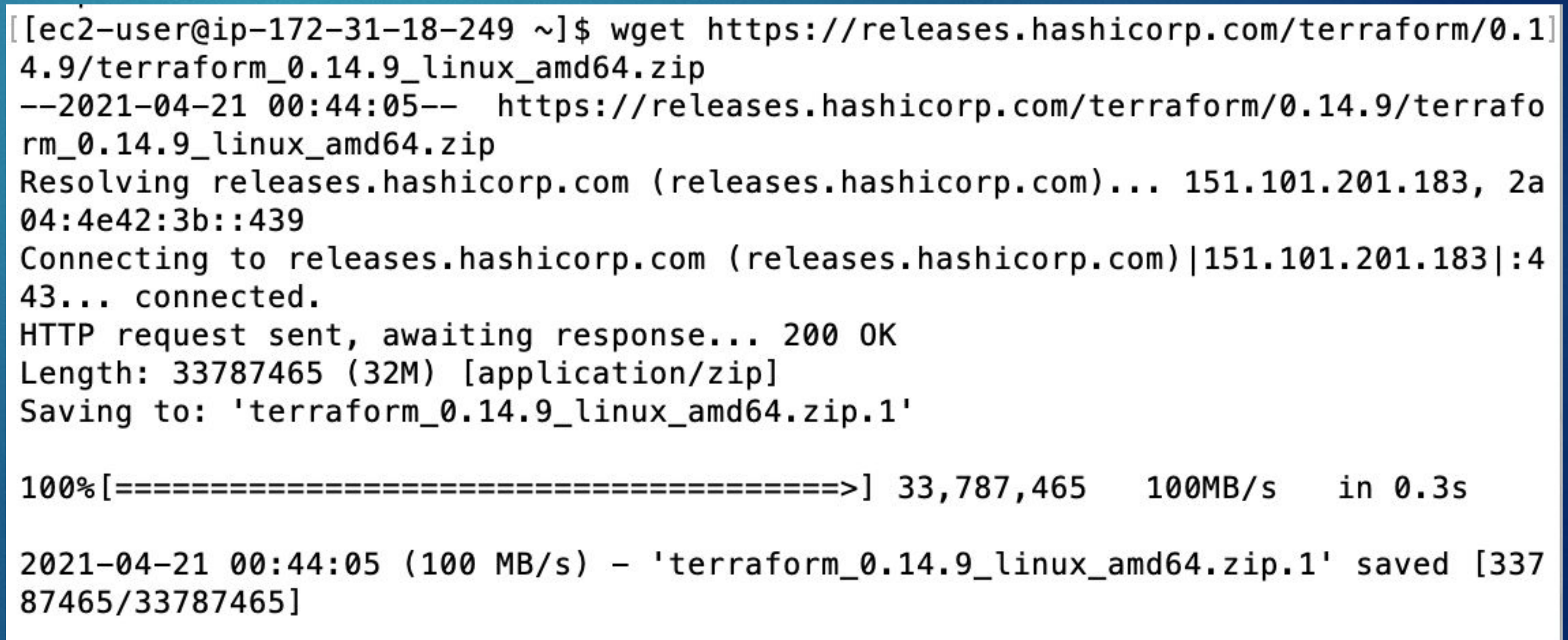
Overview

- In this activity, you will create a load balanced web page using terraform.
- The webpage will be a static webpage displaying which device you are connected to.
- The aim of activity to understand how fast you can automate the task of creating an infrastructure.



Creating an EC2 Instance

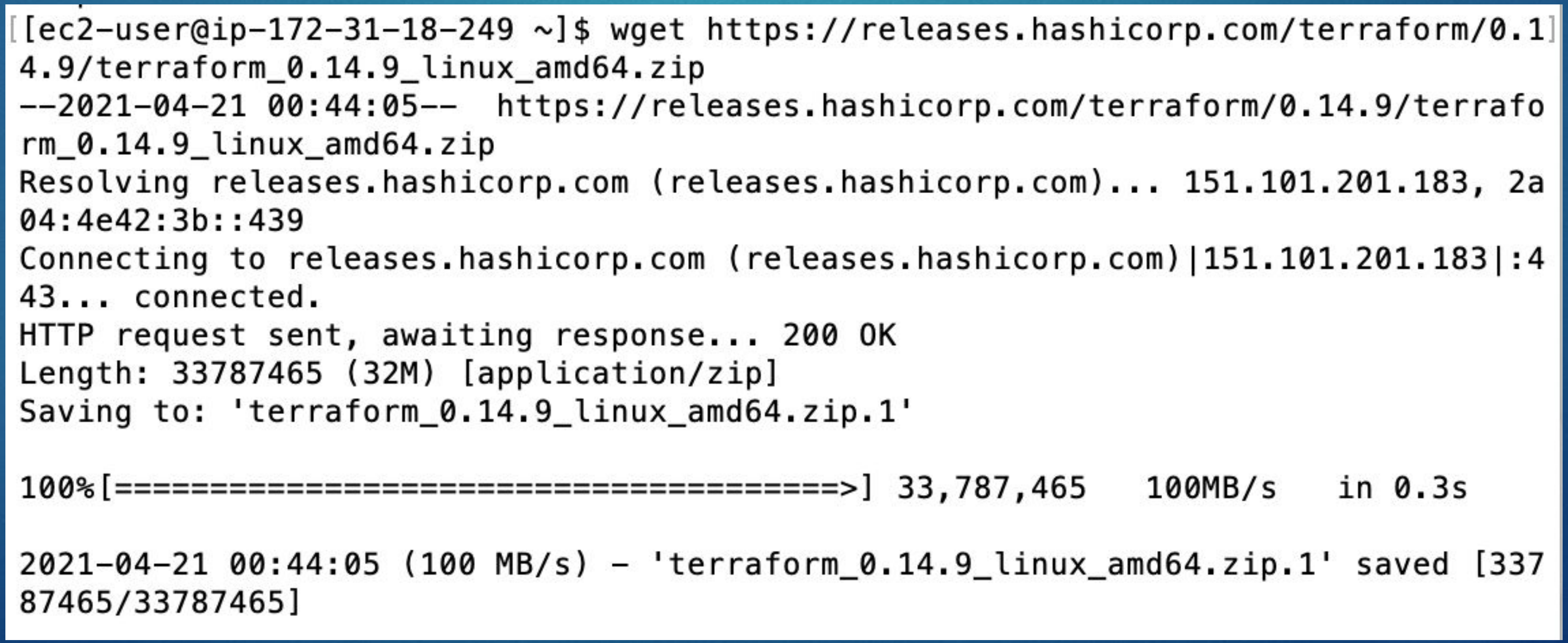
- Create a basic EC2 instance with all the default configurations.
- This is will be the master node through which we will create and destroy our infrastructure.
- After creating the EC2 instance SSH into the instance.
- Then run the command
  wget https://releases.hashicorp.com/terraform/0.14.9/terraform_0.14.9_linux_amd64.zip
- Which should give



```
[ec2-user@ip-172-31-18-249 ~]$ wget https://releases.hashicorp.com/terraform/0.1
4.9/terraform_0.14.9_linux_amd64.zip
--2021-04-21 00:44:05--  https://releases.hashicorp.com/terraform/0.14.9/terrafo
rm_0.14.9_linux_amd64.zip
Resolving releases.hashicorp.com (releases.hashicorp.com)... 151.101.201.183, 2a
04:4e42:3b::439
Connecting to releases.hashicorp.com (releases.hashicorp.com)|151.101.201.183|:4
43... connected.
HTTP request sent, awaiting response... 200 OK
Length: 33787465 (32M) [application/zip]
Saving to: 'terraform_0.14.9_linux_amd64.zip.1'

100%[======================================>] 33,787,465   100MB/s   in 0.3s

2021-04-21 00:44:05 (100 MB/s) - 'terraform_0.14.9_linux_amd64.zip.1' saved [337
87465/33787465]
```

Downloading Terraform in the EC2

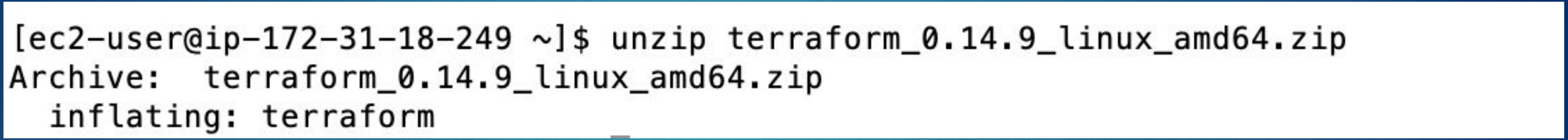
- Then run the command wget https://releases.hashicorp.com/terraform/0.14.9/terraform_0.14.9_linux_amd64.zip



```
[ec2-user@ip-172-31-18-249 ~]$ wget https://releases.hashicorp.com/terraform/0.1
4.9/terraform_0.14.9_linux_amd64.zip
--2021-04-21 00:44:05--  https://releases.hashicorp.com/terraform/0.14.9/terrafo
rm_0.14.9_linux_amd64.zip
Resolving releases.hashicorp.com (releases.hashicorp.com)... 151.101.201.183, 2a
04:4e42:3b::439
Connecting to releases.hashicorp.com (releases.hashicorp.com)|151.101.201.183|:4
43... connected.
HTTP request sent, awaiting response... 200 OK
Length: 33787465 (32M) [application/zip]
Saving to: 'terraform_0.14.9_linux_amd64.zip.1'

100%[=======================================>] 33,787,465   100MB/s    in 0.3s

2021-04-21 00:44:05 (100 MB/s) - 'terraform_0.14.9_linux_amd64.zip.1' saved [337
87465/33787465]
```

Installing Terraform in the EC2

- Run the command unzip terraform_0.14.9_linux_amd64.zip

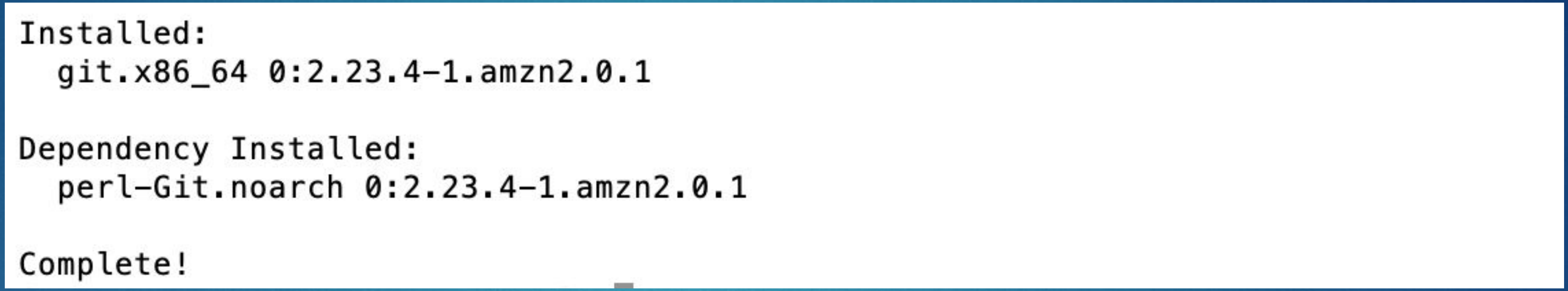


```
[ec2-user@ip-172-31-18-249 ~]$ unzip terraform_0.14.9_linux_amd64.zip
Archive:  terraform_0.14.9_linux_amd64.zip
  inflating: terraform
```

- and then run sudo mv terraform /usr/local/bin

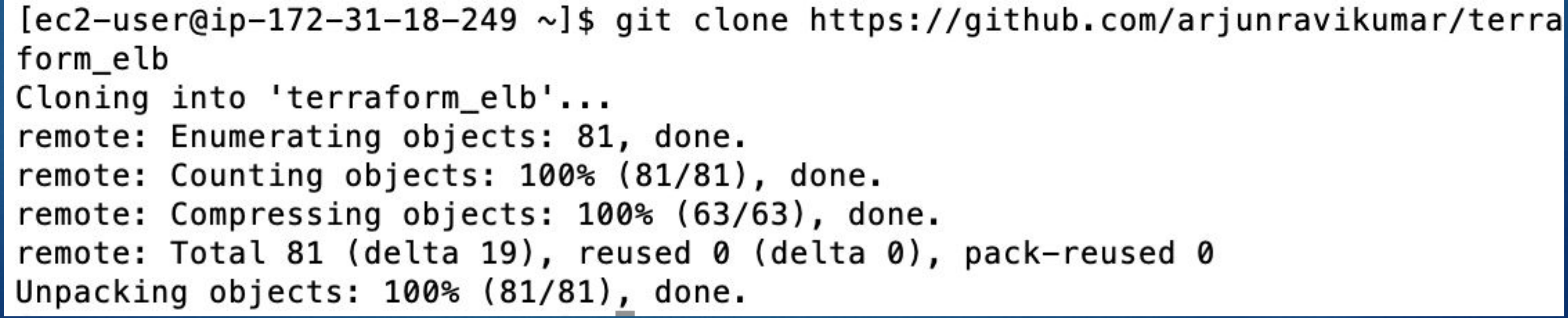
Installing Git in the EC2

- Run the command sudo yum install git -y



```
Installed:
  git.x86_64 0:2.23.4-1.amzn2.0.1

Dependency Installed:
  perl-Git.noarch 0:2.23.4-1.amzn2.0.1

Complete!
```

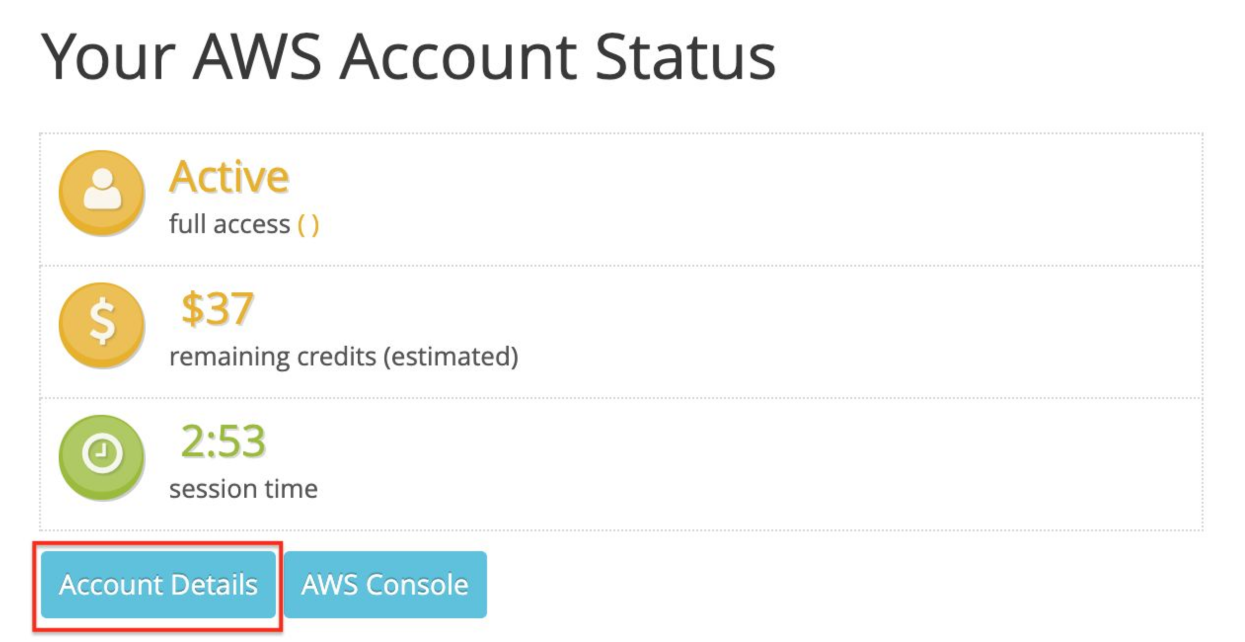
- and then run git clone https://github.com/arjunravikumar/terraform_elb to copy all the code from the git location



```
[ec2-user@ip-172-31-18-249 ~]$ git clone https://github.com/arjunravikumar/terra
form_elb
Cloning into 'terraform_elb'...
remote: Enumerating objects: 81, done.
remote: Counting objects: 100% (81/81), done.
remote: Compressing objects: 100% (63/63), done.
remote: Total 81 (delta 19), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (81/81), done.
```

Copy the AWS credentials into the EC2

- Create a new directory by doing a mkdir .aws
- Then create the file credentials inside the directory by doing vim .aws/credentials
- Navigate to your account status and then click Account details



Your AWS Account Status

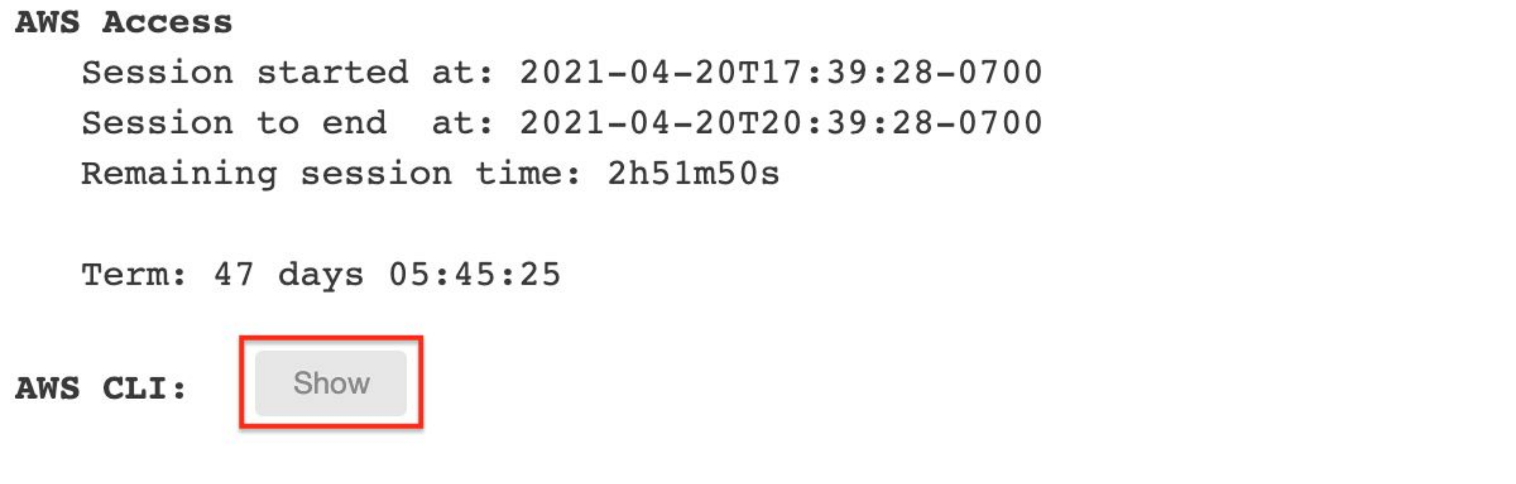
Active
full access ( )

$37
remaining credits (estimated)

2:53
session time

Copy the AWS credentials into the EC2

- Then click show

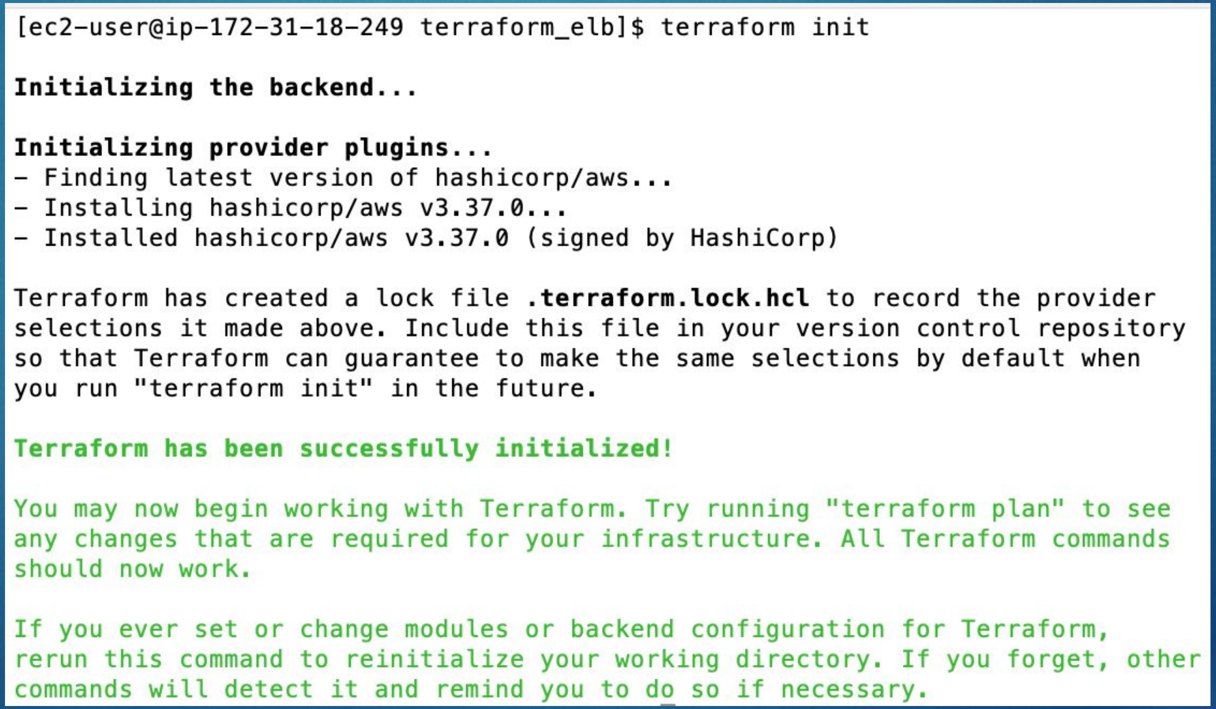


```
AWS Access
    Session started at: 2021-04-20T17:39:28-0700
    Session to end  at: 2021-04-20T20:39:28-0700
    Remaining session time: 2h51m50s

    Term: 47 days 05:45:25

AWS CLI:
```

- Copy all the contents inside and paste it into the terminal and save it using :wq

Start executing the terraform code

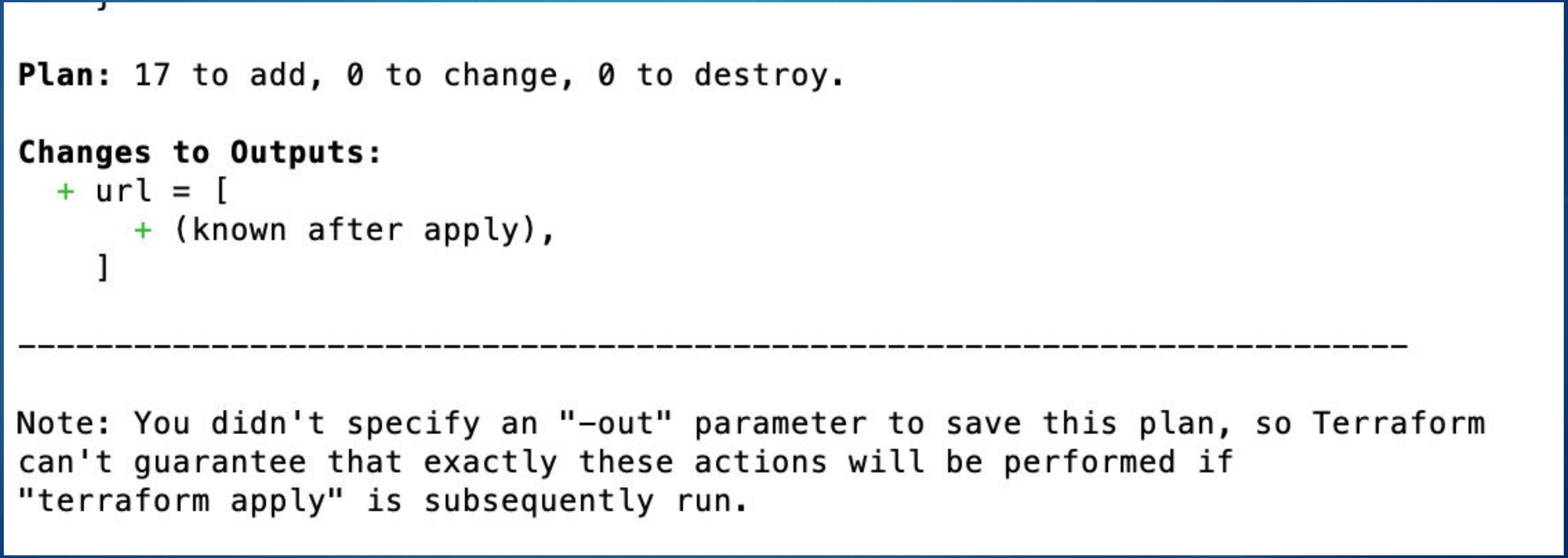
- Change directory into the newly clones git file by running the command cd terraform_elb
- Then start the execution by running the command terraform init



```
[ec2-user@ip-172-31-18-249 terraform_elb]$ terraform init

Initializing the backend...

Initializing provider plugins...
- Finding latest version of hashicorp/aws...
- Installing hashicorp/aws v3.37.0...
- Installed hashicorp/aws v3.37.0 (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.
```

Continue executing the terraform code

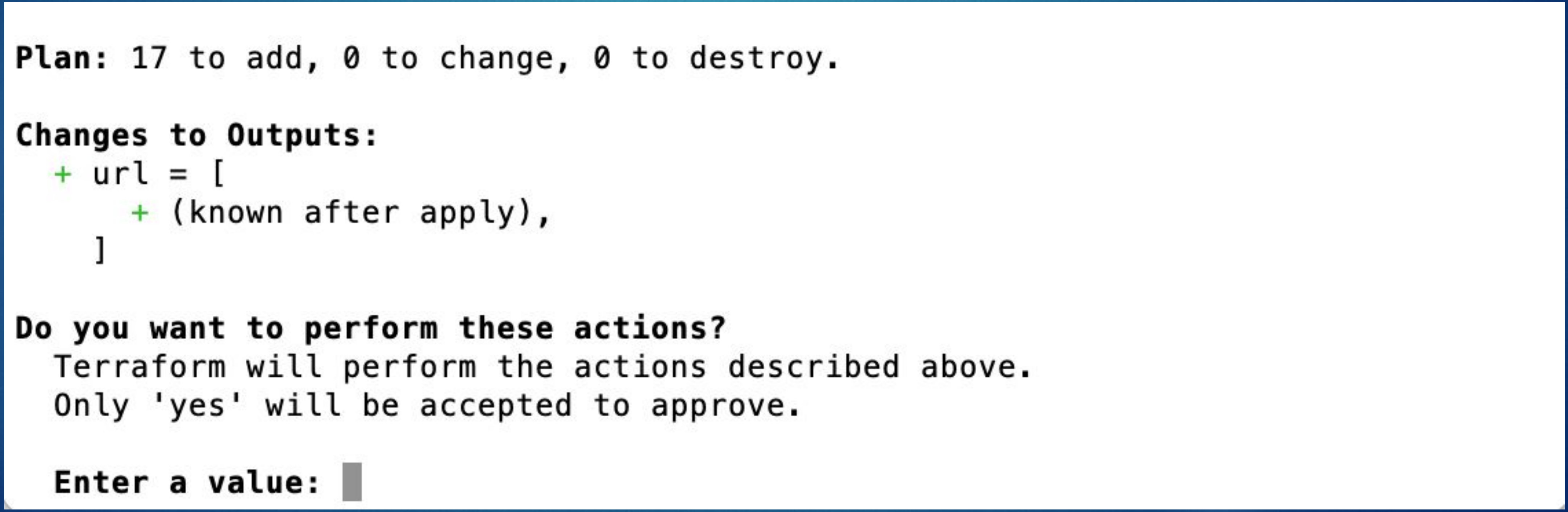
- Then run the command
  terraform plan
- This will show you the number of resources which are going to be added.



```
Plan: 17 to add, 0 to change, 0 to destroy.

Changes to Outputs:
  + url = [
      + (known after apply),
    ]

─────────────────────────────────────────────────────────────────────────

Note: You didn't specify an "-out" parameter to save this plan, so Terraform
can't guarantee that exactly these actions will be performed if
"terraform apply" is subsequently run.
```

Continue executing the terraform code

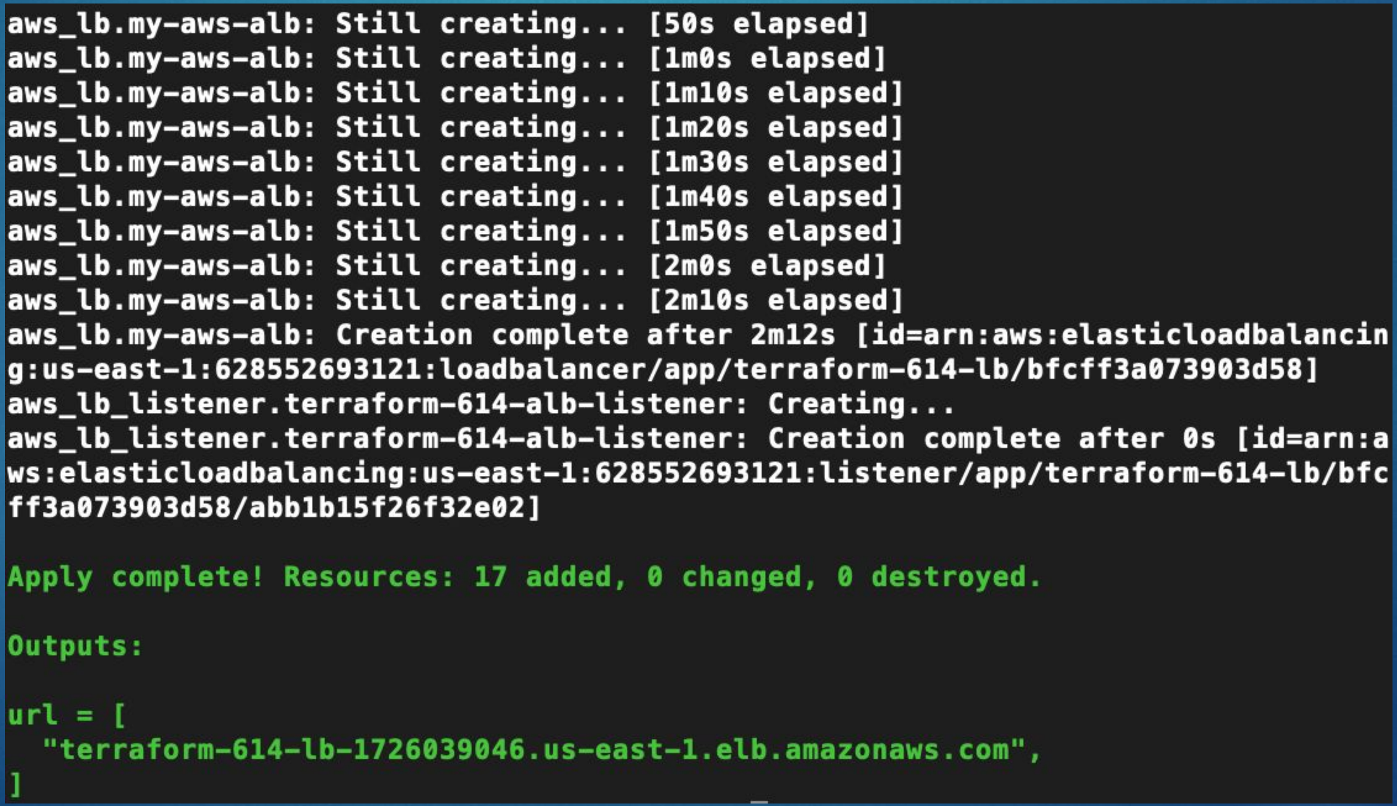
- Then run the command terraform apply
- Enter yes when prompted



```
Plan: 17 to add, 0 to change, 0 to destroy.

Changes to Outputs:
  + url = [
      + (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: █
```

Continue executing the terraform code

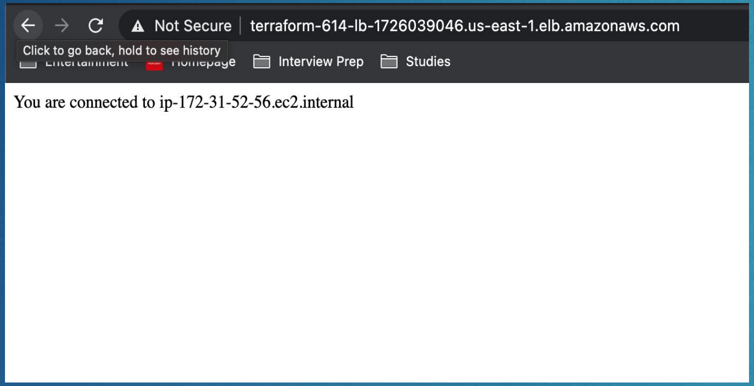
- Finally after the apply is complete you can see that it created 17 resources and since this is a Day 1 Infrastructure it did not create or update any resource.



```
aws_lb.my-aws-alb: Still creating... [50s elapsed]
aws_lb.my-aws-alb: Still creating... [1m0s elapsed]
aws_lb.my-aws-alb: Still creating... [1m10s elapsed]
aws_lb.my-aws-alb: Still creating... [1m20s elapsed]
aws_lb.my-aws-alb: Still creating... [1m30s elapsed]
aws_lb.my-aws-alb: Still creating... [1m40s elapsed]
aws_lb.my-aws-alb: Still creating... [1m50s elapsed]
aws_lb.my-aws-alb: Still creating... [2m0s elapsed]
aws_lb.my-aws-alb: Still creating... [2m10s elapsed]
aws_lb.my-aws-alb: Creation complete after 2m12s [id=arn:aws:elasticloadbalancin
g:us-east-1:628552693121:loadbalancer/app/terraform-614-lb/bfcff3a073903d58]
aws_lb_listener.terraform-614-alb-listener: Creating...
aws_lb_listener.terraform-614-alb-listener: Creation complete after 0s [id=arn:a
ws:elasticloadbalancing:us-east-1:628552693121:listener/app/terraform-614-lb/bfc
ff3a073903d58/abb1b15f26f32e02]

Apply complete! Resources: 17 added, 0 changed, 0 destroyed.

Outputs:

url = [
  "terraform-614-lb-1726039046.us-east-1.elb.amazonaws.com",
]
```

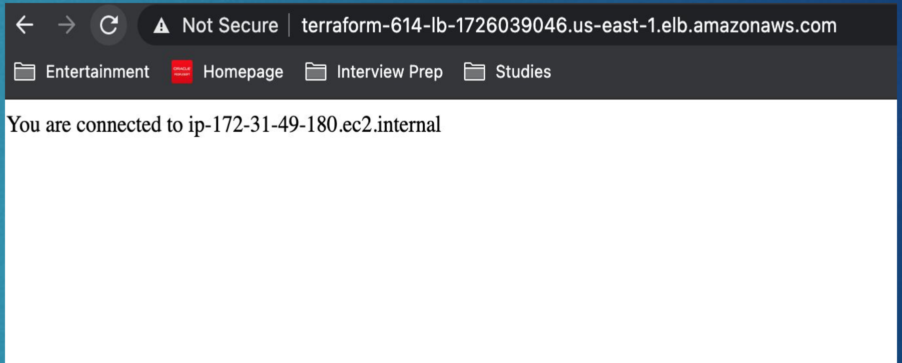
Test the load balancer

- The terraform will output the URL of the load balancer run this URL in a browser and reload it multiple times ensure that you are accessing all four EC2 instances by checking the message on screen.



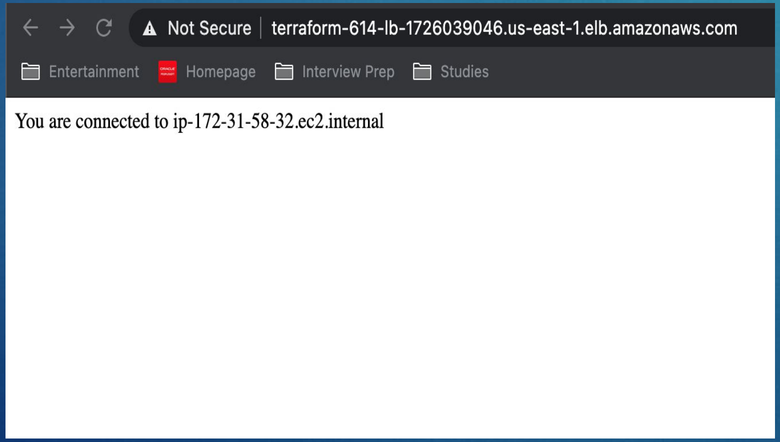
terraform-614-lb-1726039046.us-east-1.elb.amazonaws.com

You are connected to ip-172-31-52-56.ec2.internal



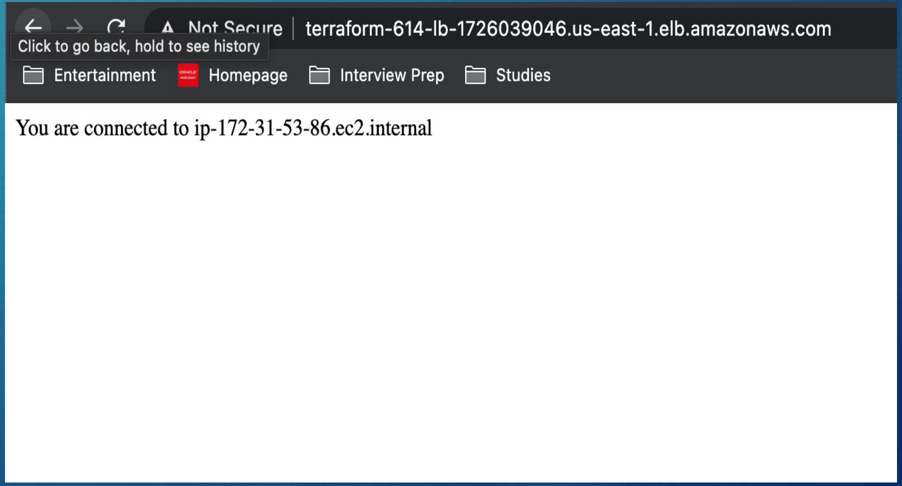
terraform-614-lb-1726039046.us-east-1.elb.amazonaws.com

You are connected to ip-172-31-49-180.ec2.internal



terraform-614-lb-1726039046.us-east-1.elb.amazonaws.com

You are connected to ip-172-31-58-32.ec2.internal

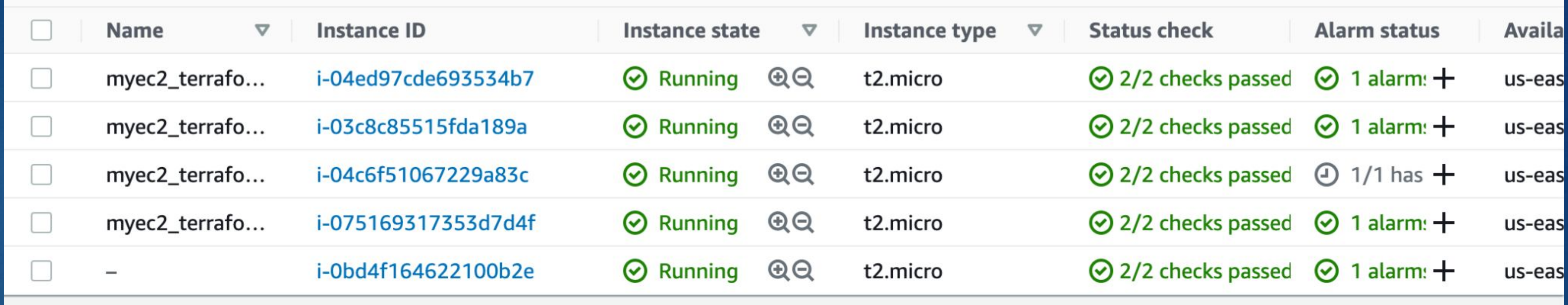


terraform-614-lb-1726039046.us-east-1.elb.amazonaws.com

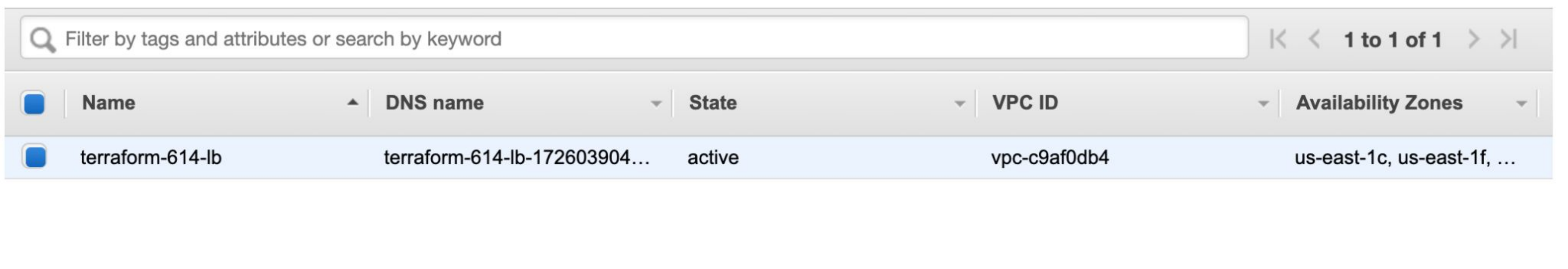
You are connected to ip-172-31-53-86.ec2.internal

Verify the EC2 instances and Load balancers

- Inside the AWS search for EC2 and verify that there are 5 ec2 instances running and 1 load balancer running.



□	Name	Instance ID	Instance state		Instance type	Status check	Alarm status		Availa
□	myec2_terrafo...	i-04ed97cde693534b7	✓ Running	⊕⊖	t2.micro	✓ 2/2 checks passed	✓ 1 alarm:	+	us-eas
□	myec2_terrafo...	i-03c8c85515fda189a	✓ Running	⊕⊖	t2.micro	✓ 2/2 checks passed	✓ 1 alarm:	+	us-eas
□	myec2_terrafo...	i-04c6f51067229a83c	✓ Running	⊕⊖	t2.micro	✓ 2/2 checks passed	◷ 1/1 has	+	us-eas
□	myec2_terrafo...	i-075169317353d7d4f	✓ Running	⊕⊖	t2.micro	✓ 2/2 checks passed	✓ 1 alarm:	+	us-eas
□	–	i-0bd4f164622100b2e	✓ Running	⊕⊖	t2.micro	✓ 2/2 checks passed	✓ 1 alarm:	+	us-eas



Filter by tags and attributes or search by keyword

1 to 1 of 1

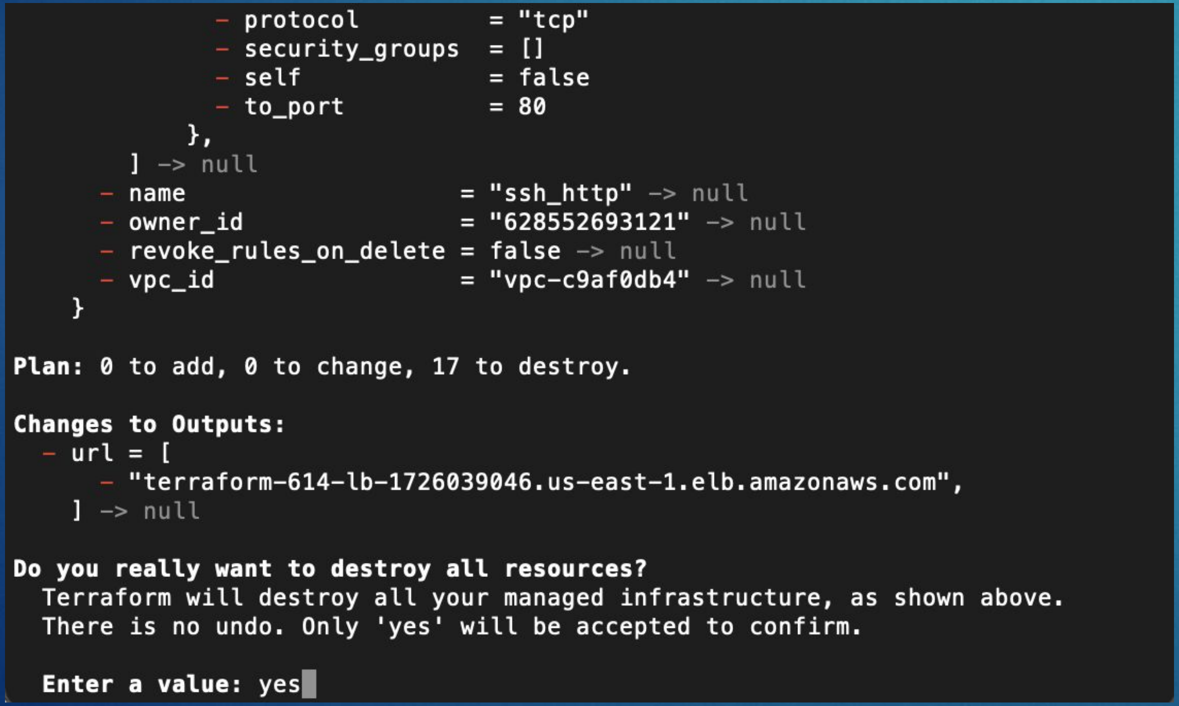
☑	Name	DNS name	State	VPC ID	Availability Zones
☑	terraform-614-lb	terraform-614-lb-172603904...	active	vpc-c9af0db4	us-east-1c, us-east-1f, ...

Congratulations

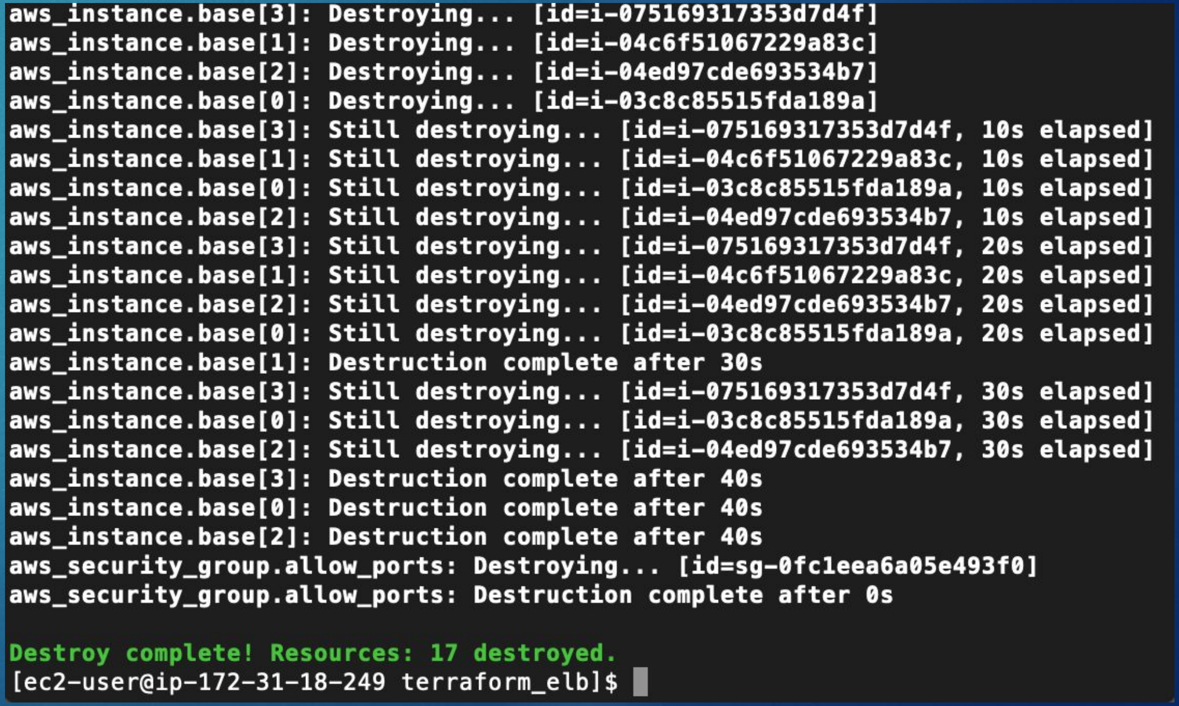
- You have created four EC2 instances over a load balancer and is running a static webpage in each of them.

Now to destroy all the Infrastructure

- Then run the command
  terraform destroy
- Enter yes when prompted



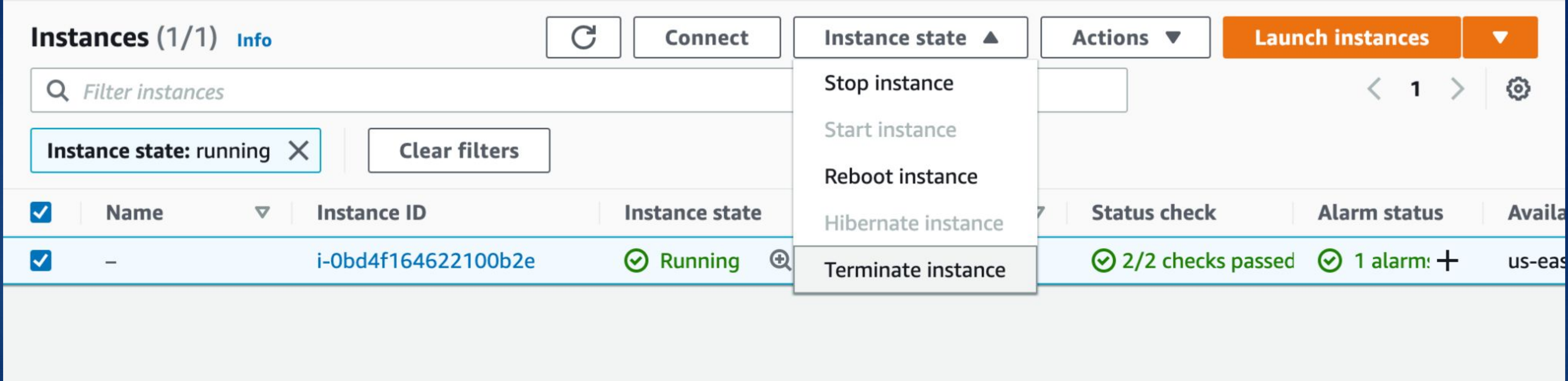
```
       - protocol         = "tcp"
       - security_groups  = []
       - self             = false
       - to_port          = 80
     },
    ] -> null
   - name                   = "ssh_http" -> null
   - owner_id               = "628552693121" -> null
   - revoke_rules_on_delete = false -> null
   - vpc_id                 = "vpc-c9af0db4" -> null
  }

Plan: 0 to add, 0 to change, 17 to destroy.

Changes to Outputs:
 - url = [
   - "terraform-614-lb-1726039046.us-east-1.elb.amazonaws.com",
  ] -> null

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.base[3]: Destroying... [id=i-075169317353d7d4f]
aws_instance.base[1]: Destroying... [id=i-04c6f51067229a83c]
aws_instance.base[2]: Destroying... [id=i-04ed97cde693534b7]
aws_instance.base[0]: Destroying... [id=i-03c8c85515fda189a]
aws_instance.base[3]: Still destroying... [id=i-075169317353d7d4f, 10s elapsed]
aws_instance.base[1]: Still destroying... [id=i-04c6f51067229a83c, 10s elapsed]
aws_instance.base[0]: Still destroying... [id=i-03c8c85515fda189a, 10s elapsed]
aws_instance.base[2]: Still destroying... [id=i-04ed97cde693534b7, 10s elapsed]
aws_instance.base[3]: Still destroying... [id=i-075169317353d7d4f, 20s elapsed]
aws_instance.base[1]: Still destroying... [id=i-04c6f51067229a83c, 20s elapsed]
aws_instance.base[2]: Still destroying... [id=i-04ed97cde693534b7, 20s elapsed]
aws_instance.base[0]: Still destroying... [id=i-03c8c85515fda189a, 20s elapsed]
aws_instance.base[1]: Destruction complete after 30s
aws_instance.base[3]: Still destroying... [id=i-075169317353d7d4f, 30s elapsed]
aws_instance.base[0]: Still destroying... [id=i-03c8c85515fda189a, 30s elapsed]
aws_instance.base[2]: Still destroying... [id=i-04ed97cde693534b7, 30s elapsed]
aws_instance.base[3]: Destruction complete after 40s
aws_instance.base[0]: Destruction complete after 40s
aws_instance.base[2]: Destruction complete after 40s
aws_security_group.allow_ports: Destroying... [id=sg-0fc1eea6a05e493f0]
aws_security_group.allow_ports: Destruction complete after 0s

Destroy complete! Resources: 17 destroyed.
[ec2-user@ip-172-31-18-249 terraform_elb]$
```

Destroy all the Infrastructure

- Don’t forget to terminate your EC2 instance which was used for installing the terraform



Instances (1/1) Info

Instance state: running

☑	Name	Instance ID	Instance state	Status check	Alarm status	Availa
☑	–	i-0bd4f164622100b2e	Running	2/2 checks passed	1 alarm: +	us-eas

All done!