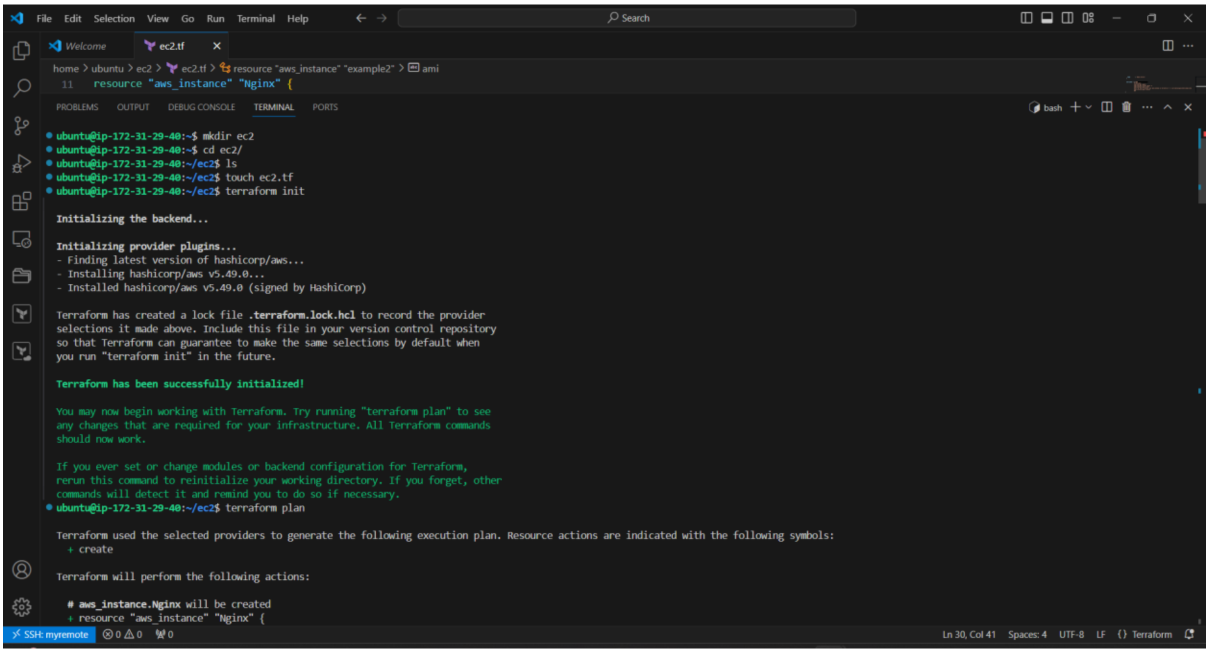
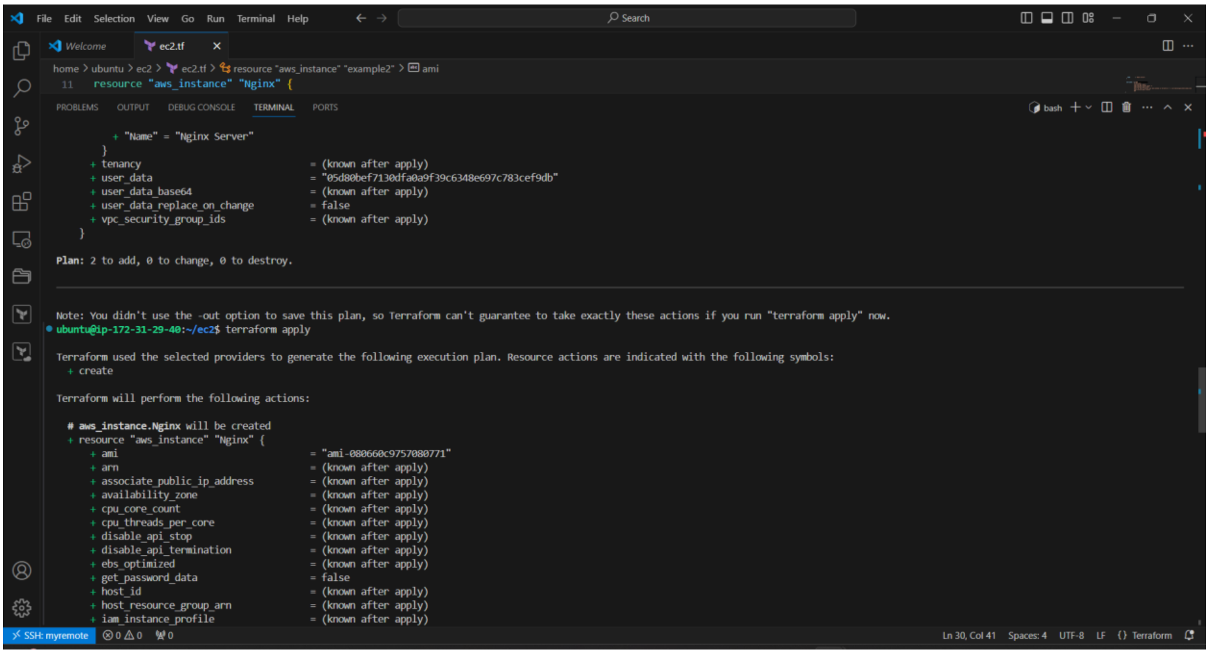


```
11    resource "aws_instance" "Nginx" {
```

```
ubuntu@ip-172-31-29-40:~$ mkdir ec2
ubuntu@ip-172-31-29-40:~$ cd ec2/
ubuntu@ip-172-31-29-40:~/ec2$ ls
ubuntu@ip-172-31-29-40:~/ec2$ touch ec2.tf
ubuntu@ip-172-31-29-40:~/ec2$ terraform init

Initializing the backend...

Initializing provider plugins...
- Finding latest version of hashicorp/aws...
- Installing hashicorp/aws v5.49.0...
- Installed hashicorp/aws v5.49.0 (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-172-31-29-40:~/ec2$ 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.Nginx will be created
  + resource "aws_instance" "Nginx" {
```

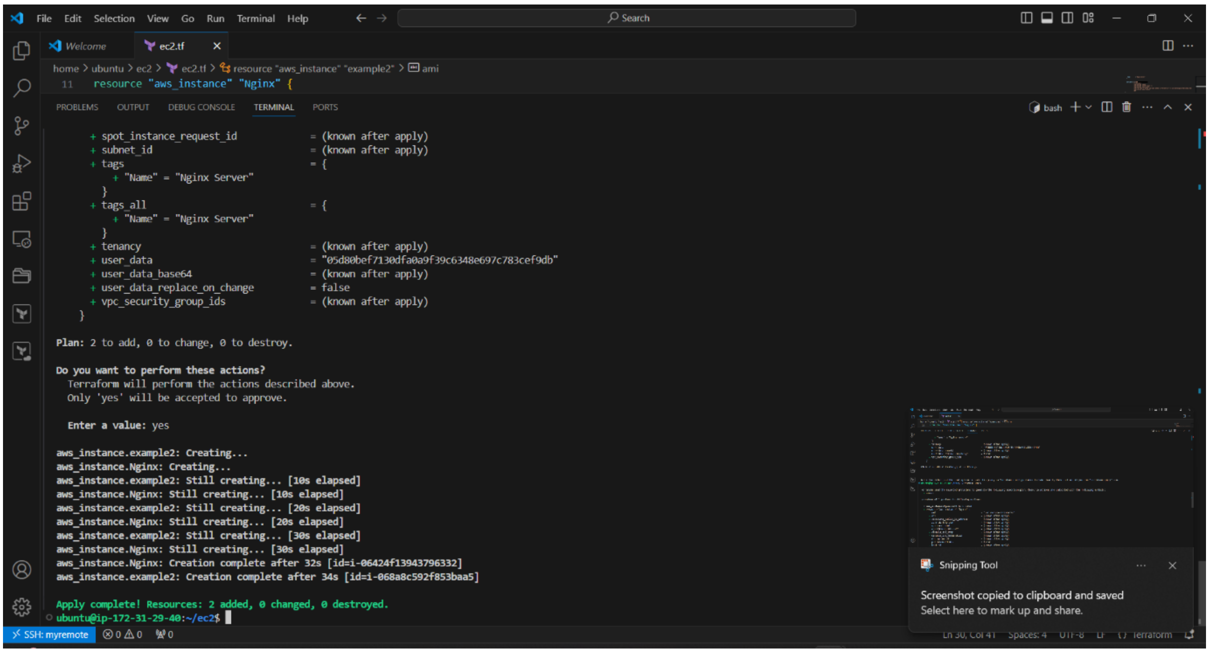
1(terraform init and plan)



```
11    resource "aws_instance" "Nginx" {
```

```
          + "Name" = "Nginx Server"
        }
      + tenancy                              = (known after apply)
      + user_data                            = "05d80bef7130dfa0a9f39c6348e697c783cef9db"
      + user_data_base64                     = (known after apply)
      + user_data_replace_on_change          = false
      + vpc_security_group_ids               = (known after apply)
    }

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

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

Note: You didn't use the -out option to save this plan, so Terraform can't guarantee to take exactly these actions if you run "terraform apply" now.
ubuntu@ip-172-31-29-40:~/ec2$ 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.Nginx will be created
  + resource "aws_instance" "Nginx" {
      + ami                                  = "ami-080660c9757080771"
      + 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_stop                     = (known after apply)
      + disable_api_termination              = (known after apply)
      + ebs_optimized                        = (known after apply)
      + get_password_data                    = false
      + host_id                              = (known after apply)
      + host_resource_group_arn              = (known after apply)
      + iam_instance_profile                 = (known after apply)
```

2(terraform apply)



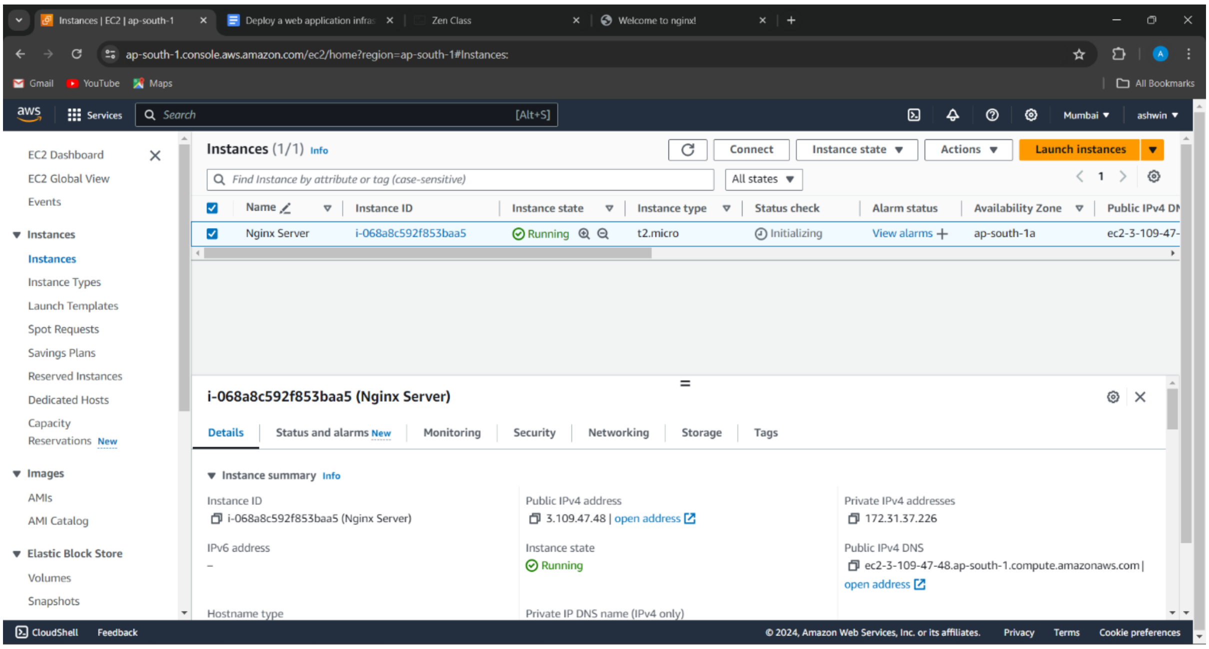
File Edit Selection View Go Run Terminal Help
Search
Welcome
ec2.tf
home > ubuntu > ec2 > ec2.tf > resource "aws_instance" "example2" > ami
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
bash
Snipping Tool
SSH: myremote
Ln 30, Col 41 Spaces: 4 UTF-8 LF {} Terraform

```
resource "aws_instance" "Nginx" {
```

```
      + spot_instance_request_id             = (known after apply)
      + subnet_id                            = (known after apply)
      + tags                                 = {
          + "Name" = "Nginx Server"
        }
      + tags_all                             = {
          + "Name" = "Nginx Server"
        }
      + tenancy                              = (known after apply)
      + user_data                            = "05d80bef7130dfa0a9f39c6348e697c783cef9db"
      + user_data_base64                     = (known after apply)
      + user_data_replace_on_change          = false
      + vpc_security_group_ids               = (known after apply)
    }

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

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.example2: Creating...
aws_instance.Nginx: Creating...
aws_instance.example2: Still creating... [10s elapsed]
aws_instance.Nginx: Still creating... [10s elapsed]
aws_instance.example2: Still creating... [20s elapsed]
aws_instance.Nginx: Still creating... [20s elapsed]
aws_instance.example2: Still creating... [30s elapsed]
aws_instance.Nginx: Still creating... [30s elapsed]
aws_instance.Nginx: Creation complete after 32s [id=i-06424f13943796332]
aws_instance.example2: Creation complete after 34s [id=i-068a8c592f853baa5]

Apply complete! Resources: 2 added, 0 changed, 0 destroyed.
ubuntu@ip-172-31-29-40:~/ec2$ 
```

Screenshot copied to clipboard and saved
Select here to mark up and share.

3(terraform script output)



Instances (1/1) Info

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DN
Nginx Server	i-068a8c592f853baa5	Running	t2.micro	Initializing	View alarms +	ap-south-1a	ec2-3-109-47-

i-068a8c592f853baa5 (Nginx Server)

Details | Status and alarms New | Monitoring | Security | Networking | Storage | Tags

Instance summary Info

Instance ID
i-068a8c592f853baa5 (Nginx Server)

Public IPv4 address
3.109.47.48 | open address

Private IPv4 addresses
172.31.37.226

IPv6 address
–

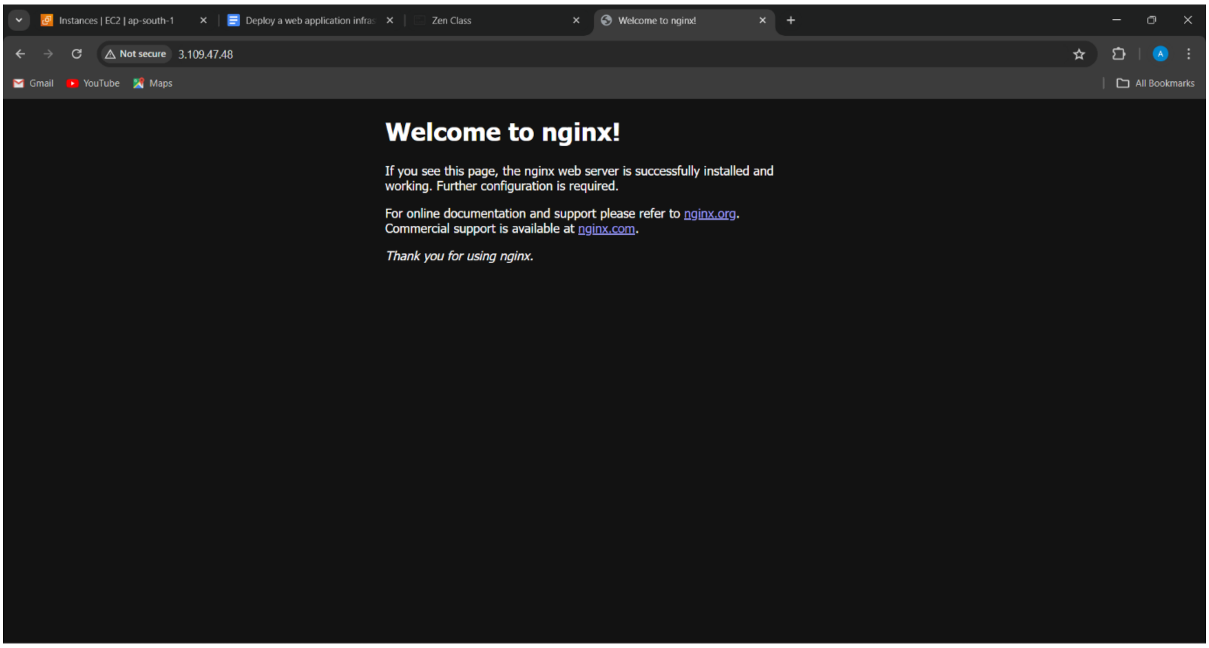
Instance state
Running

Public IPv4 DNS
ec2-3-109-47-48.ap-south-1.compute.amazonaws.com | open address

Hostname type

Private IP DNS name (IPv4 only)

4(created a ec2 with nginx in mumbai region)



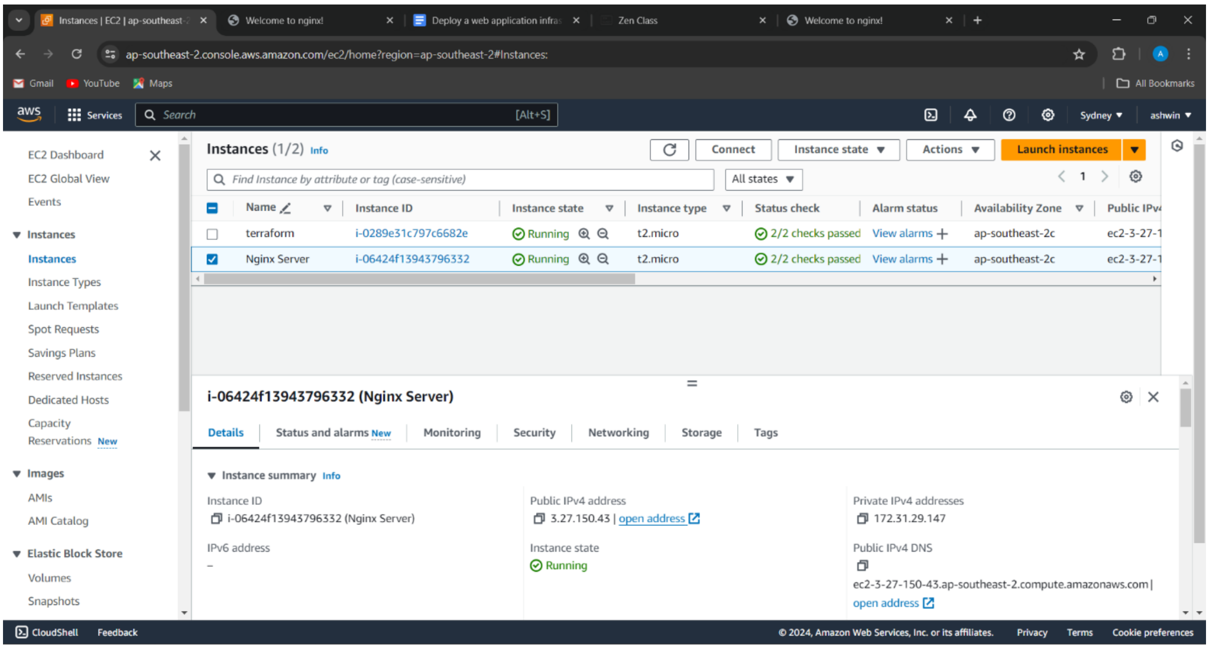
Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

Thank you for using nginx.

5(nginx output)



Instances (1/2) Info

	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4
□	terraform	i-0289e31c797c6682e	Running	t2.micro	2/2 checks passed	View alarms +	ap-southeast-2c	ec2-3-27-1
☑	Nginx Server	i-06424f13943796332	Running	t2.micro	2/2 checks passed	View alarms +	ap-southeast-2c	ec2-3-27-1

i-06424f13943796332 (Nginx Server)

Instance summary Info

Instance ID
i-06424f13943796332 (Nginx Server)

Public IPv4 address
3.27.150.43 | open address

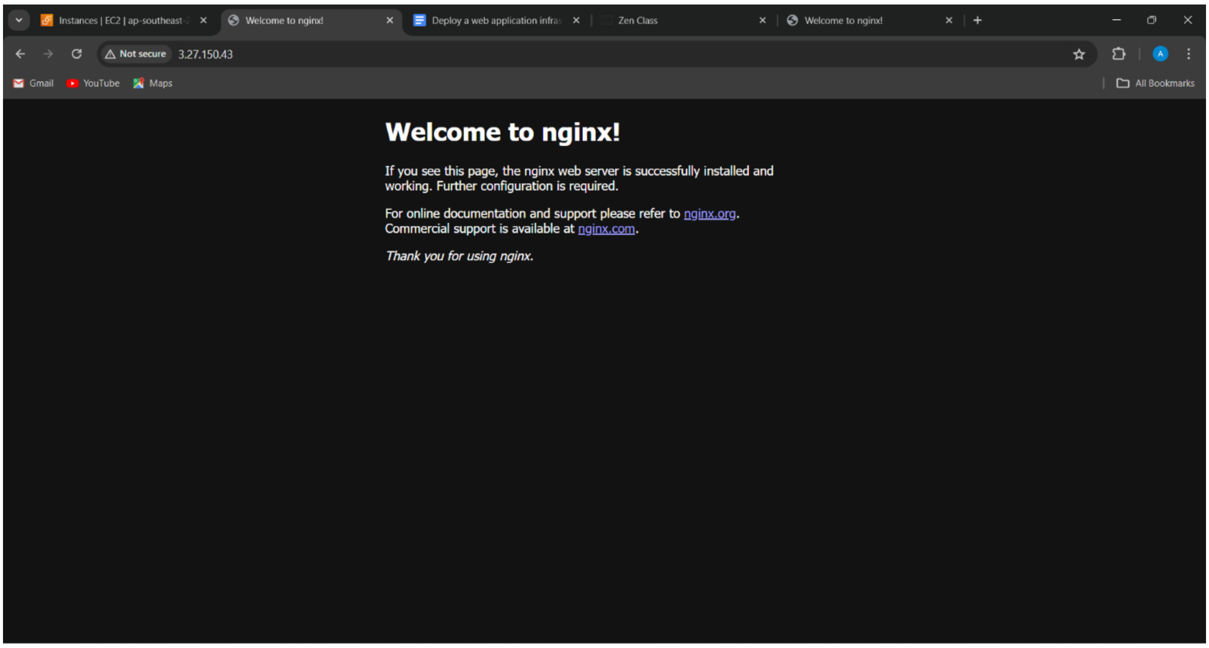
Private IPv4 addresses
172.31.29.147

IPv6 address
–

Instance state
Running

Public IPv4 DNS
ec2-3-27-150-43.ap-southeast-2.compute.amazonaws.com | open address

6(created a ec2 with nginx in sydney region)



Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to nginx.org. Commercial support is available at nginx.com.

Thank you for using nginx.

7(nginx output 2)