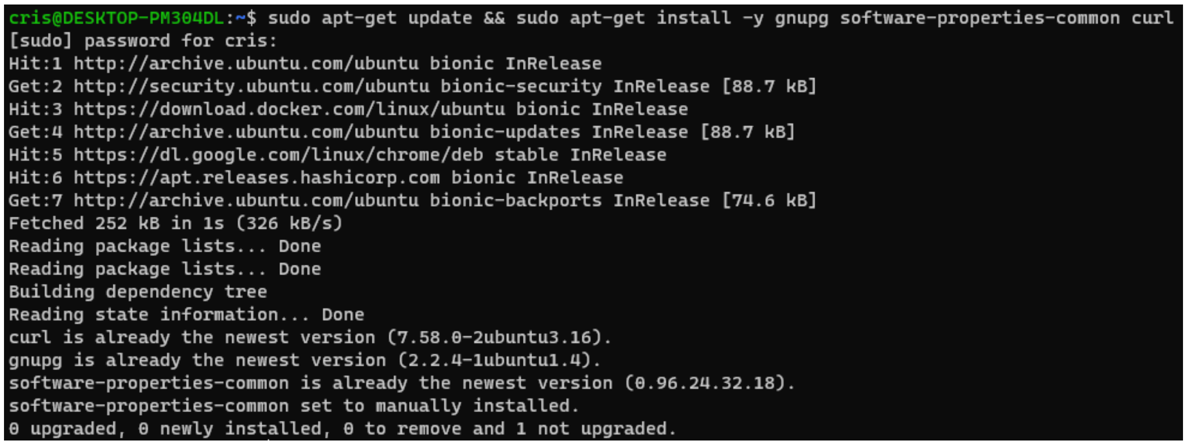
Terraform – Docker provider

- Install terraform – Ubuntu 18.04 on WSL2

Update system, curl packages and Hashicorp repository

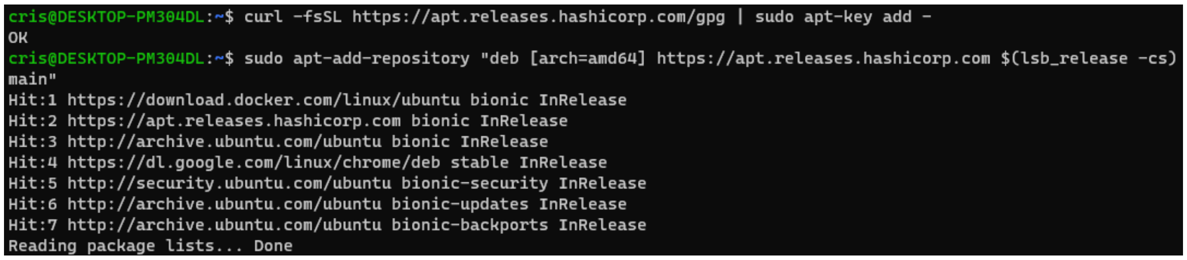
✓ sudo apt-get update && sudo apt-get install -y gnupg software-properties-common curl



```
cris@DESKTOP-PM304DL:~$ sudo apt-get update && sudo apt-get install -y gnupg software-properties-common curl
[sudo] password for cris:
Hit:1 http://archive.ubuntu.com/ubuntu bionic InRelease
Get:2 http://security.ubuntu.com/ubuntu bionic-security InRelease [88.7 kB]
Hit:3 https://download.docker.com/linux/ubuntu bionic InRelease
Get:4 http://archive.ubuntu.com/ubuntu bionic-updates InRelease [88.7 kB]
Hit:5 https://dl.google.com/linux/chrome/deb stable InRelease
Hit:6 https://apt.releases.hashicorp.com bionic InRelease
Get:7 http://archive.ubuntu.com/ubuntu bionic-backports InRelease [74.6 kB]
Fetched 252 kB in 1s (326 kB/s)
Reading package lists... Done
Reading package lists... Done
Building dependency tree
Reading state information... Done
curl is already the newest version (7.58.0-2ubuntu3.16).
gnupg is already the newest version (2.2.4-1ubuntu1.4).
software-properties-common is already the newest version (0.96.24.32.18).
software-properties-common set to manually installed.
0 upgraded, 0 newly installed, 0 to remove and 1 not upgraded.
```

Add Hashicorp key and official repository

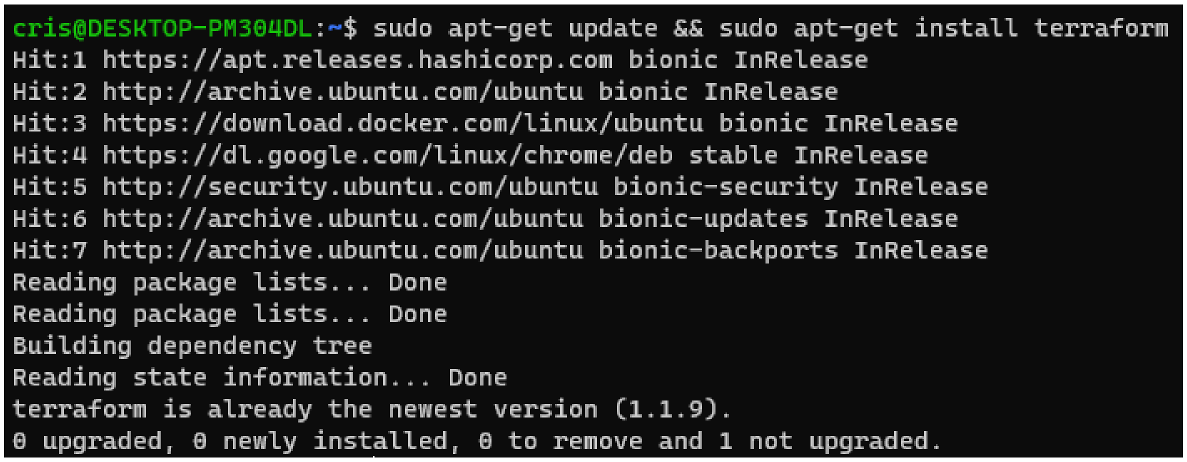
✓ curl -fsSL https://apt.releases.hashicorp.com/gpg | sudo apt-key add –
✓ sudo apt-add-repository "deb [arch=amd64] https://apt.releases.hashicorp.com $(lsb_release -cs) main"



```
cris@DESKTOP-PM304DL:~$ curl -fsSL https://apt.releases.hashicorp.com/gpg | sudo apt-key add -
OK
cris@DESKTOP-PM304DL:~$ sudo apt-add-repository "deb [arch=amd64] https://apt.releases.hashicorp.com $(lsb_release -cs)
main"
Hit:1 https://download.docker.com/linux/ubuntu bionic InRelease
Hit:2 https://apt.releases.hashicorp.com bionic InRelease
Hit:3 http://archive.ubuntu.com/ubuntu bionic InRelease
Hit:4 https://dl.google.com/linux/chrome/deb stable InRelease
Hit:5 http://security.ubuntu.com/ubuntu bionic-security InRelease
Hit:6 http://archive.ubuntu.com/ubuntu bionic-updates InRelease
Hit:7 http://archive.ubuntu.com/ubuntu bionic-backports InRelease
Reading package lists... Done
```

Add repository and install Terraform CLI

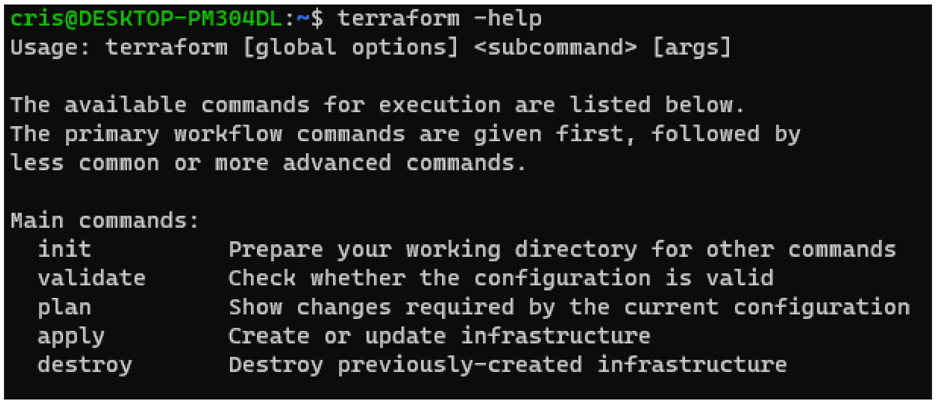
✓ sudo apt-get update && sudo apt-get install terraform



```
cris@DESKTOP-PM304DL:~$ sudo apt-get update && sudo apt-get install terraform
Hit:1 https://apt.releases.hashicorp.com bionic InRelease
Hit:2 http://archive.ubuntu.com/ubuntu bionic InRelease
Hit:3 https://download.docker.com/linux/ubuntu bionic InRelease
Hit:4 https://dl.google.com/linux/chrome/deb stable InRelease
Hit:5 http://security.ubuntu.com/ubuntu bionic-security InRelease
Hit:6 http://archive.ubuntu.com/ubuntu bionic-updates InRelease
Hit:7 http://archive.ubuntu.com/ubuntu bionic-backports InRelease
Reading package lists... Done
Reading package lists... Done
Building dependency tree
Reading state information... Done
terraform is already the newest version (1.1.9).
0 upgraded, 0 newly installed, 0 to remove and 1 not upgraded.
```

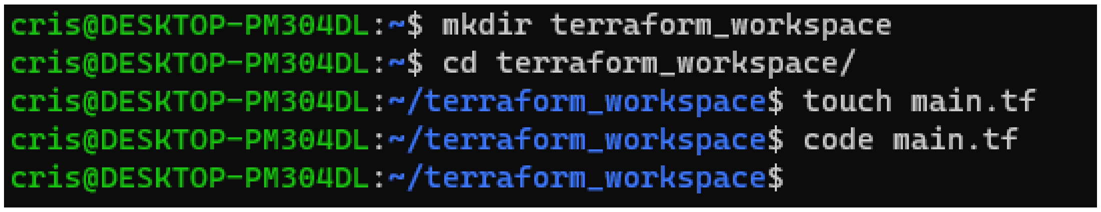
Verify the installation

✓ terraform -help



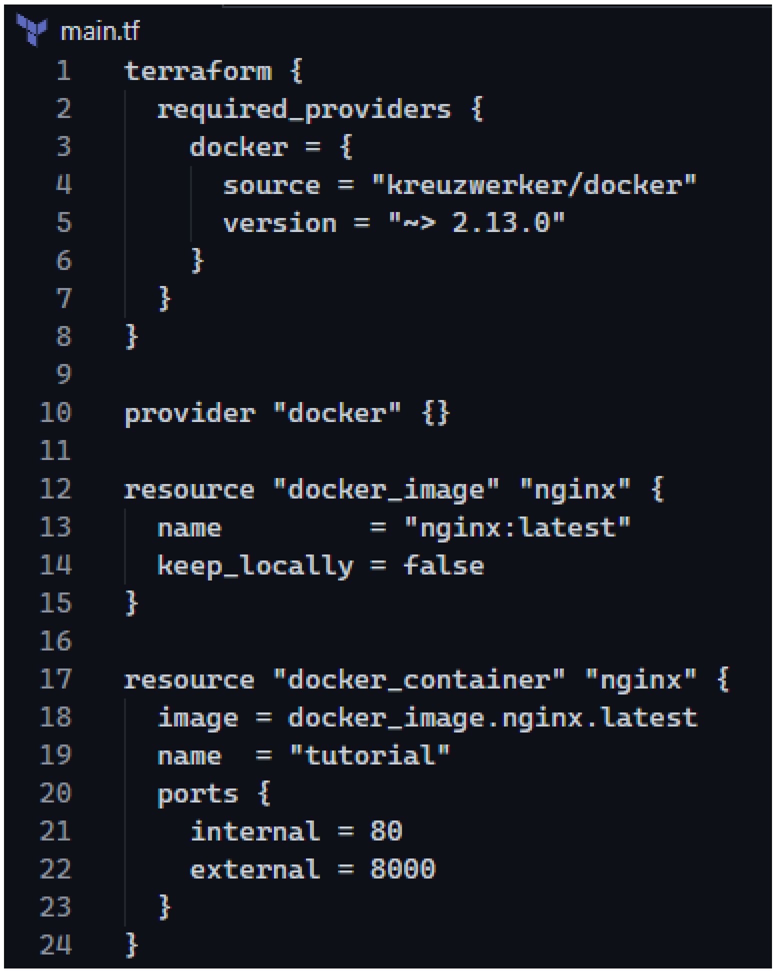
```
cris@DESKTOP-PM304DL:~$ terraform -help
Usage: terraform [global options] <subcommand> [args]

The available commands for execution are listed below.
The primary workflow commands are given first, followed by
less common or more advanced commands.

Main commands:
  init          Prepare your working directory for other commands
  validate      Check whether the configuration is valid
  plan          Show changes required by the current configuration
  apply         Create or update infrastructure
  destroy       Destroy previously-created infrastructure
```

- Create a folder to use as workspace, create a Terraform file (main.tf) to define the infrastructure



```
cris@DESKTOP-PM304DL:~$ mkdir terraform_workspace
cris@DESKTOP-PM304DL:~$ cd terraform_workspace/
cris@DESKTOP-PM304DL:~/terraform_workspace$ touch main.tf
cris@DESKTOP-PM304DL:~/terraform_workspace$ code main.tf
cris@DESKTOP-PM304DL:~/terraform_workspace$
```

- Create required infrastructure definition according to provider’s documentation. In this case docker is being configured as provider, then a docker image is being pulled and launched inside a docker container.



main.tf

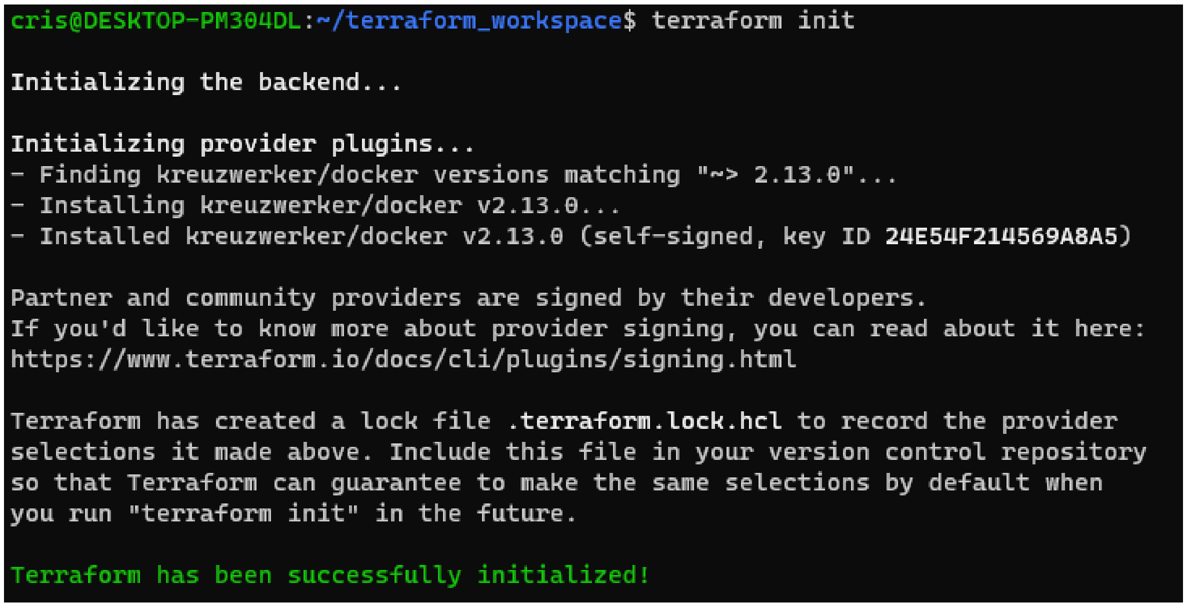
```
terraform {
  required_providers {
    docker = {
      source = "kreuzwerker/docker"
      version = "~> 2.13.0"
    }
  }
}

provider "docker" {}

resource "docker_image" "nginx" {
  name         = "nginx:latest"
  keep_locally = false
}

resource "docker_container" "nginx" {
  image = docker_image.nginx.latest
  name  = "tutorial"
  ports {
    internal = 80
    external = 8000
  }
}
```

Note that external port is mapped to port 8000

- Initialize current directory for terraform

Initialization downloads all the providers defined in the configuration file

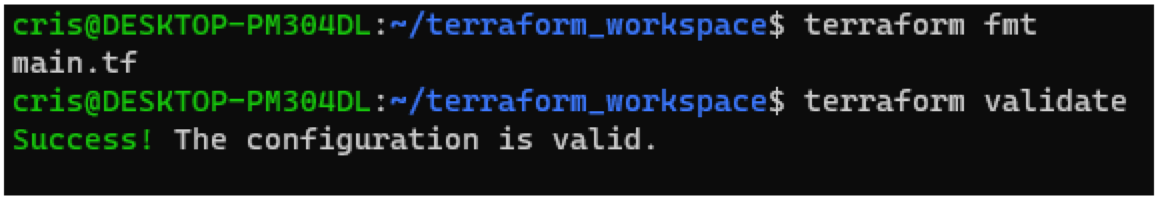
✓ terraform init



```
cris@DESKTOP-PM304DL:~/terraform_workspace$ terraform init

Initializing the backend...

Initializing provider plugins...
- Finding kreuzwerker/docker versions matching "~> 2.13.0"...
- Installing kreuzwerker/docker v2.13.0...
- Installed kreuzwerker/docker v2.13.0 (self-signed, key ID 24E54F214569A8A5)

Partner and community providers are signed by their developers.
If you'd like to know more about provider signing, you can read about it here:
https://www.terraform.io/docs/cli/plugins/signing.html

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!
```

- Make sure everything is configured properly

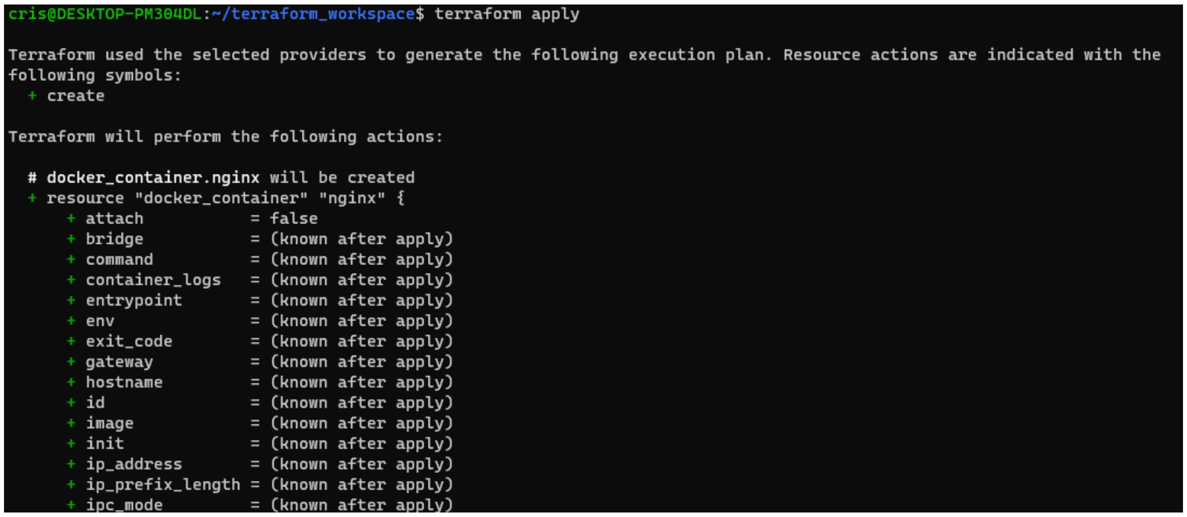
✓ terraform fmt
✓ terraform validate



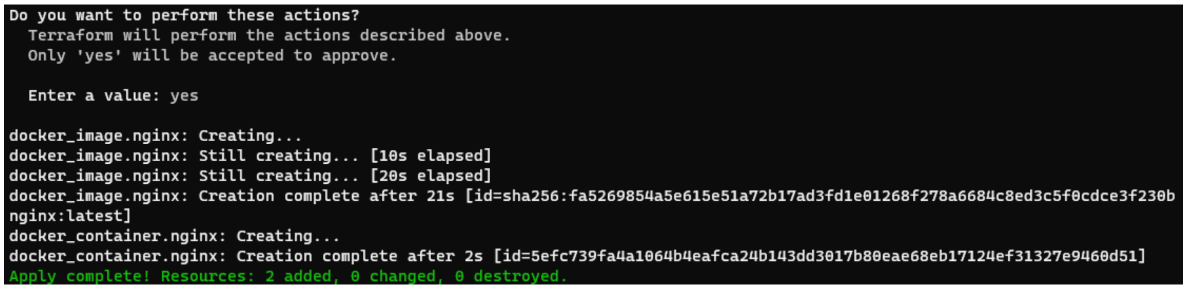
```
cris@DESKTOP-PM304DL:~/terraform_workspace$ terraform fmt
main.tf
cris@DESKTOP-PM304DL:~/terraform_workspace$ terraform validate
Success! The configuration is valid.
```

- Create the configured infrastructure. Make sure docker is initialized and working.

✓ terraform apply



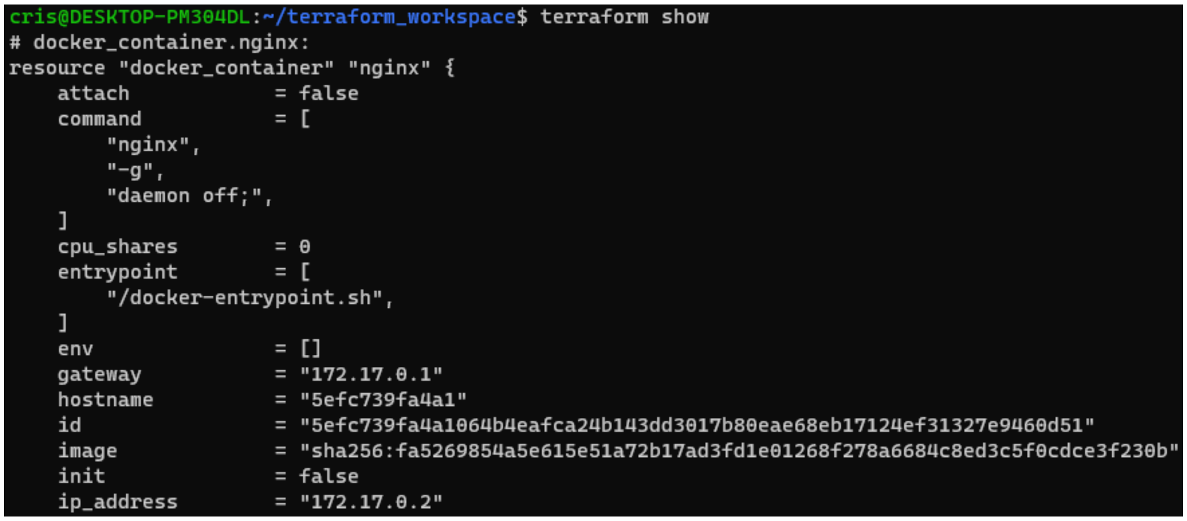
```
cris@DESKTOP-PM304DL:~/terraform_workspace$ terraform apply

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:

  # docker_container.nginx will be created
  + resource "docker_container" "nginx" {
      + attach           = false
      + bridge           = (known after apply)
      + command          = (known after apply)
      + container_logs   = (known after apply)
      + entrypoint       = (known after apply)
      + env              = (known after apply)
      + exit_code        = (known after apply)
      + gateway          = (known after apply)
      + hostname         = (known after apply)
      + id               = (known after apply)
      + image            = (known after apply)
      + init             = (known after apply)
      + ip_address       = (known after apply)
      + ip_prefix_length = (known after apply)
      + ipc_mode         = (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

docker_image.nginx: Creating...
docker_image.nginx: Still creating... [10s elapsed]
docker_image.nginx: Still creating... [20s elapsed]
docker_image.nginx: Creation complete after 21s [id=sha256:fa5269854a5e615e51a72b17ad3fd1e01268f278a6684c8ed3c5f0cdce3f230b
nginx:latest]
docker_container.nginx: Creating...
docker_container.nginx: Creation complete after 2s [id=5efc739fa4a1064b4eafca24b143dd3017b80eae68eb17124ef31327e9460d51]
Apply complete! Resources: 2 added, 0 changed, 0 destroyed.
```

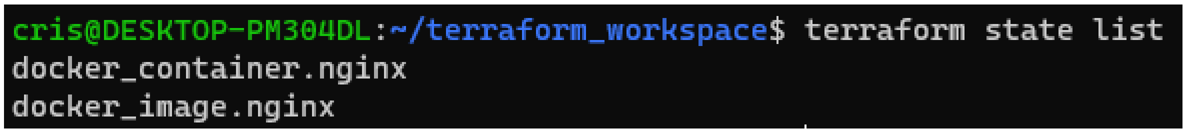
- Make sure everything is working properly

✓ terraform show



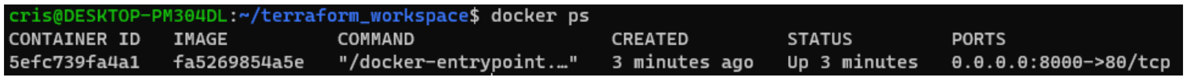
```
cris@DESKTOP-PM304DL:~/terraform_workspace$ terraform show
# docker_container.nginx:
resource "docker_container" "nginx" {
    attach            = false
    command           = [
        "nginx",
        "-g",
        "daemon off;",
    ]
    cpu_shares        = 0
    entrypoint        = [
        "/docker-entrypoint.sh",
    ]
    env               = []
    gateway           = "172.17.0.1"
    hostname          = "5efc739fa4a1"
    id                = "5efc739fa4a1064b4eafca24b143dd3017b80eae68eb17124ef31327e9460d51"
    image             = "sha256:fa5269854a5e615e51a72b17ad3fd1e01268f278a6684c8ed3c5f0cdce3f230b"
    init              = false
    ip_address        = "172.17.0.2"
```

✓ terraform state list



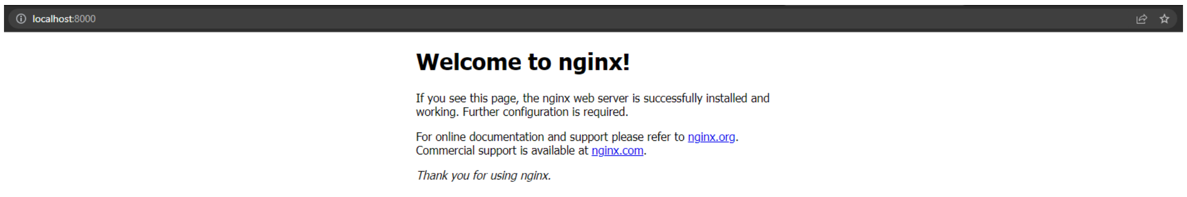
```
cris@DESKTOP-PM304DL:~/terraform_workspace$ terraform state list
docker_container.nginx
docker_image.nginx
```

✓ docker ps



```
cris@DESKTOP-PM304DL:~/terraform_workspace$ docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED          STATUS         PORTS
5efc739fa4a1   fa5269854a5e   "/docker-entrypoint.…"   3 minutes ago    Up 3 minutes   0.0.0.0:8000->80/tcp
```

✓ Browser



Welcome to nginx!

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

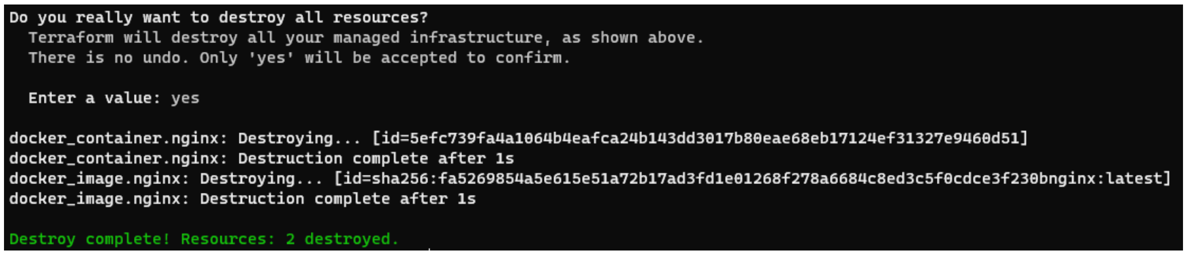
For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

Thank you for using nginx.

Everything is working properly!

- If you want to shut down the infrastructure

✓ terraform 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

docker_container.nginx: Destroying... [id=5efc739fa4a1064b4eafca24b143dd3017b80eae68eb17124ef31327e9460d51]
docker_container.nginx: Destruction complete after 1s
docker_image.nginx: Destroying... [id=sha256:fa5269854a5e615e51a72b17ad3fd1e01268f278a6684c8ed3c5f0cdce3f230bnginx:latest]
docker_image.nginx: Destruction complete after 1s

Destroy complete! Resources: 2 destroyed.
```