```
PS C:\Users\nemer\learn-terraform-docker-container> ni main.tf


    Directory: C:\Users\nemer\learn-terraform-docker-container


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a----        10/30/2023   3:19 PM              0 main.tf


PS C:\Users\nemer\learn-terraform-docker-container> terraform init

Initializing the backend...

Initializing provider plugins...
- Finding kreuzwerker/docker versions matching "~> 3.0.1"...
- Installing kreuzwerker/docker v3.0.2...
- Installed kreuzwerker/docker v3.0.2 (self-signed, key ID BD080C4571C6104C)

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!

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.
PS C:\Users\nemer\learn-terraform-docker-container> 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)
      + container_read_refresh_timeout_milliseconds = 15000
      + entrypoint                                  = (known after apply)
      + env                                         = (known after apply)
      + exit_code                                   = (known after apply)
      + hostname                                    = (known after apply)
      + id                                          = (known after apply)
```

```
        + stop_signal                         = (known after apply)
        + stop_timeout                        = (known after apply)
        + tty                                 = false
        + wait                                = false
        + wait_timeout                        = 60

        + ports {
            + external = 8000
            + internal = 80
            + ip       = "0.0.0.0"
            + protocol = "tcp"
          }
      }

  # docker_image.nginx will be created
  + resource "docker_image" "nginx" {
      + id            = (known after apply)
      + image_id      = (known after apply)
      + keep_locally  = false
      + name          = "nginx"
      + repo_digest   = (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:
```
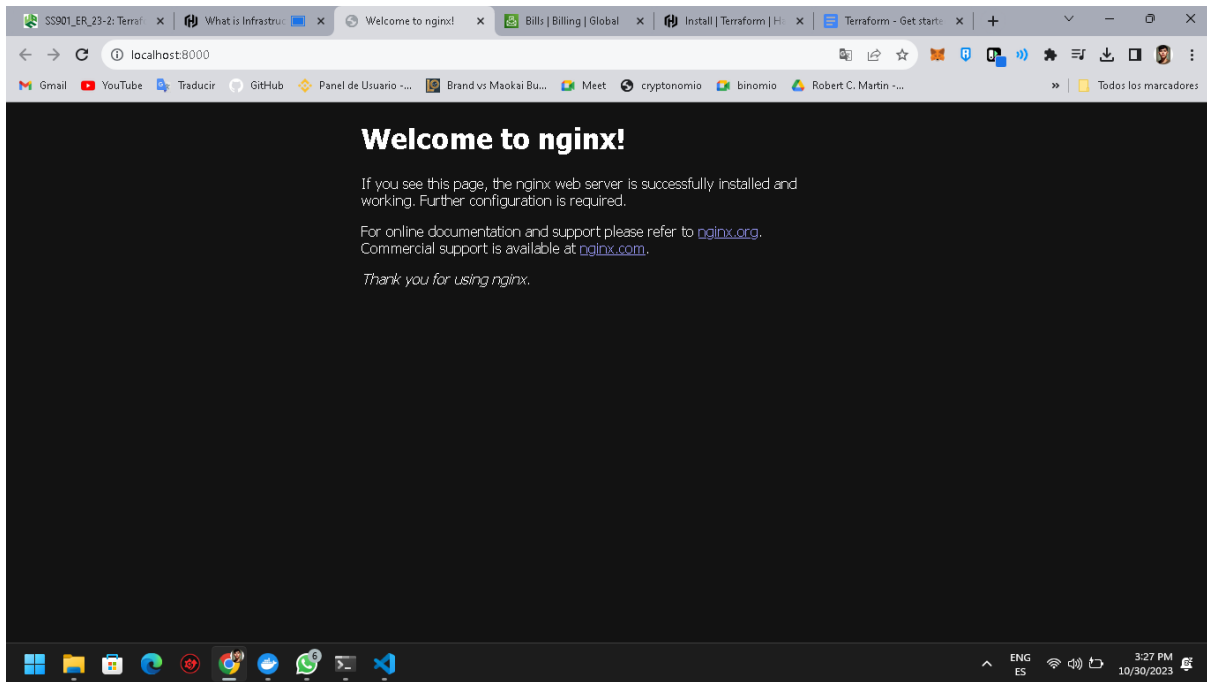
```
    }

  # docker_image.nginx will be created
  + resource "docker_image" "nginx" {
      + id            = (known after apply)
      + image_id      = (known after apply)
      + keep_locally  = false
      + name          = "nginx"
      + repo_digest   = (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

docker_image.nginx: Creating...
docker_image.nginx: Still creating... [10s elapsed]
docker_image.nginx: Still creating... [20s elapsed]
docker_image.nginx: Still creating... [30s elapsed]
docker_image.nginx: Creation complete after 36s [id=sha256:593aee2afb642798b83a85306d2625fd7f089c0a1242c7e75a237846d80aa
2a0nginx]
docker_container.nginx: Creating...
docker_container.nginx: Creation complete after 5s [id=ca209e143db4d66ac26505b1865b95b06e4b633038803aa5420336e09efab341]

Apply complete! Resources: 2 added, 0 changed, 0 destroyed.
PS C:\Users\nemer\learn-terraform-docker-container>
```

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.


```
    + image_id     = (known after apply)
    + keep_locally = false
    + name         = "nginx"
    + repo_digest  = (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

docker_image.nginx: Creating...
docker_image.nginx: Still creating... [10s elapsed]
docker_image.nginx: Still creating... [20s elapsed]
docker_image.nginx: Still creating... [30s elapsed]
docker_image.nginx: Creation complete after 36s [id=sha256:593aee2afb642798b83a85306d2625fd7f089c0a1242c7e75a237846d80aa
2a0nginx]
docker_container.nginx: Creating...
docker_container.nginx: Creation complete after 5s [id=ca209e143db4d66ac26505b1865b95b06e4b633038803aa5420336e09efab341]

Apply complete! Resources: 2 added, 0 changed, 0 destroyed.
PS C:\Users\nemer\learn-terraform-docker-container> docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED          STATUS          PORTS                   N
AMES
ca209e143db4   593aee2afb64   "/docker-entrypoint.…"   About a minute ago   Up About a minute   0.0.0.0:8000->80/tcp   t
utorial
PS C:\Users\nemer\learn-terraform-docker-container> terraform destroy
```

```
             - internal = 80 -> null
             - ip       = "0.0.0.0" -> null
             - protocol = "tcp" -> null
           }
       }

     # docker_image.nginx will be destroyed
   - resource "docker_image" "nginx" {
       - id            = "sha256:593aee2afb642798b83a85306d2625fd7f089c0a1242c7e75a237846d80aa2a0nginx" -> null
       - image_id      = "sha256:593aee2afb642798b83a85306d2625fd7f089c0a1242c7e75a237846d80aa2a0" -> null
       - keep_locally  = false -> null
       - name          = "nginx" -> null
       - repo_digest   = "nginx@sha256:add4792d930c25dd2abf2ef9ea79de578097a1c175a16ab25814332fe33622de" -> null
     }

Plan: 0 to add, 0 to change, 2 to 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=ca209e143db4d66ac26505b1865b95b06e4b633038803aa5420336e09efab341]
docker_container.nginx: Destruction complete after 0s
docker_image.nginx: Destroying... [id=sha256:593aee2afb642798b83a85306d2625fd7f089c0a1242c7e75a237846d80aa2a0nginx]
docker_image.nginx: Destruction complete after 1s

Destroy complete! Resources: 2 destroyed.
PS C:\Users\nemer\learn-terraform-docker-container>
```

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\nemer> aws --version
aws-cli/2.13.30 Python/3.11.6 Windows/10 exe/AMD64 prompt/off
```

```
PS C:\Users\nemer> aws configure
AWS Access Key ID [None]: 
AWS Secret Access Key [None]: 
Default region name [None]:
Default output format [None]:
PS C:\Users\nemer>
```

```
PS C:\Users\nemer> mkdir learn-terraform-aws-instance


    Directory: C:\Users\nemer


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d-----         10/30/2023   4:03 PM                learn-terraform-aws-instance


PS C:\Users\nemer> cd learn-terraform-aws-instance
PS C:\Users\nemer\learn-terraform-aws-instance> touch main.tf
touch : The term 'touch' is not recognized as the name of a cmdlet, function, script file, or operable program. Check
the spelling of the name, or if a path was included, verify that the path is correct and try again.
At line:1 char:1
+ touch main.tf
+ ~~~~~
    + CategoryInfo          : ObjectNotFound: (touch:String) [], CommandNotFoundException
    + FullyQualifiedErrorId : CommandNotFoundException

PS C:\Users\nemer\learn-terraform-aws-instance> ni main.tf


    Directory: C:\Users\nemer\learn-terraform-aws-instance


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a----         10/30/2023   4:04 PM              0 main.tf
```

```
Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a----         10/30/2023   4:04 PM              0 main.tf

PS C:\Users\nemer\learn-terraform-aws-instance> terraform init

Initializing the backend...

Initializing provider plugins...
- Finding hashicorp/aws versions matching "~> 4.16"...
- Installing hashicorp/aws v4.67.0...
- Installed hashicorp/aws v4.67.0 (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.
PS C:\Users\nemer\learn-terraform-aws-instance>
```

```
PS C:\Users\nemer\learn-terraform-aws-instance> 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.app_server will be created
  + resource "aws_instance" "app_server" {
      + ami                                  = "ami-830c94e3"
      + 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)
      + 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                             = (known after apply)
```

```
      + tags                         = {
          + "Name" = "ExampleAppServerInstance"
        }
      + tags_all                     = {
          + "Name" = "ExampleAppServerInstance"
        }
      + tenancy                      = (known after apply)
      + user_data                    = (known after apply)
      + user_data_base64             = (known after apply)
      + user_data_replace_on_change  = false
      + vpc_security_group_ids       = (known after apply)
    }

Plan: 1 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.app_server: Creating...
aws_instance.app_server: Still creating... [10s elapsed]
aws_instance.app_server: Still creating... [20s elapsed]
aws_instance.app_server: Still creating... [30s elapsed]
aws_instance.app_server: Still creating... [40s elapsed]
aws_instance.app_server: Creation complete after 45s [id=i-0dd89e2bc3fef71c9]

Apply complete! Resources: 1 added, 0 changed, 0 destroyed.
PS C:\Users\nemer\learn-terraform-aws-instance>
```

```
PS C:\Users\nemer\learn-terraform-aws-instance> terraform show
# aws_instance.app_server:
resource "aws_instance" "app_server" {
    ami                                  = "ami-830c94e3"
    arn                                  = "arn:aws:ec2:us-west-2:453285446265:instance/i-0dd89e2bc3fef71c9"
    associate_public_ip_address          = true
    availability_zone                    = "us-west-2b"
    cpu_core_count                       = 1
    cpu_threads_per_core                 = 1
    disable_api_stop                     = false
    disable_api_termination              = false
    ebs_optimized                        = false
    get_password_data                    = false
    hibernation                          = false
    id                                   = "i-0dd89e2bc3fef71c9"
    instance_initiated_shutdown_behavior = "stop"
    instance_state                       = "running"
    instance_type                        = "t2.micro"
    ipv6_address_count                   = 0
    ipv6_addresses                       = []
    monitoring                           = false
    placement_partition_number           = 0
    primary_network_interface_id         = "eni-04d0fa9c62f270dac"
    private_dns                          = "ip-172-31-17-66.us-west-2.compute.internal"
    private_ip                           = "172.31.17.66"
    public_dns                           = "ec2-34-216-219-5.us-west-2.compute.amazonaws.com"
    public_ip                            = "34.216.219.5"
    secondary_private_ips                = []
    security_groups                      = [
        "default",
```

```
    maintenance_options {
        auto_recovery = "default"
    }

    metadata_options {
        http_endpoint               = "enabled"
        http_put_response_hop_limit = 1
        http_tokens                 = "optional"
        instance_metadata_tags      = "disabled"
    }

    private_dns_name_options {
        enable_resource_name_dns_a_record    = false
        enable_resource_name_dns_aaaa_record = false
        hostname_type                        = "ip-name"
    }

    root_block_device {
        delete_on_termination = true
        device_name           = "/dev/sda1"
        encrypted             = false
        iops                  = 0
        tags                  = {}
        throughput            = 0
        volume_id             = "vol-086efbeb5a7e4b298"
        volume_size           = 8
        volume_type           = "standard"
    }
}
PS C:\Users\nemer\learn-terraform-aws-instance>
```

```
}
PS C:\Users\nemer\learn-terraform-aws-instance> terraform state list
aws_instance.app_server
PS C:\Users\nemer\learn-terraform-aws-instance> |
```