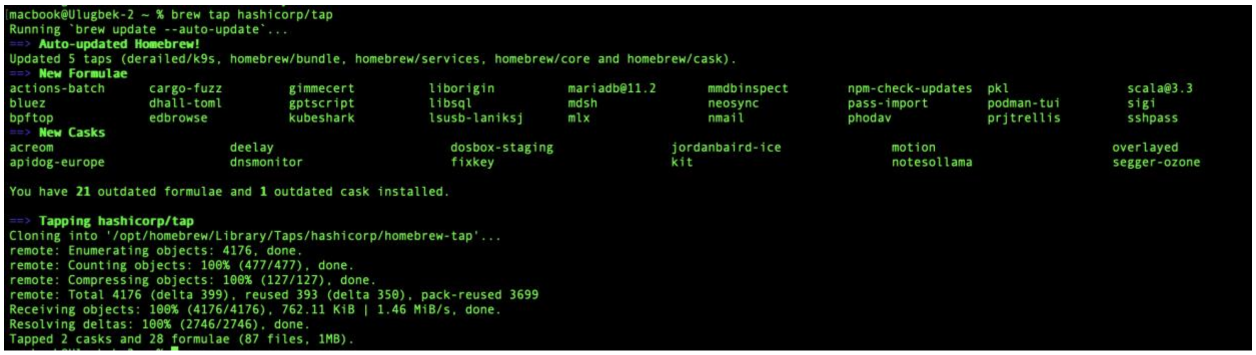
LAB 4

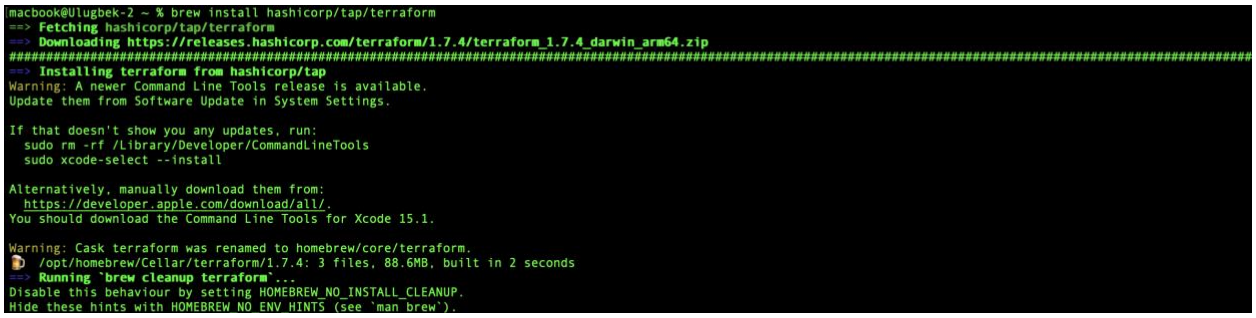
Name: Ulugbek Sattarov
Student ID: 101451916

brew tap hashicorp/tap



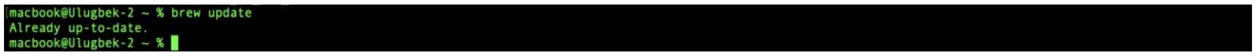
```
macbook@Ulugbek-2 ~ % brew tap hashicorp/tap
Running `brew update --auto-update`...
==> Auto-updated Homebrew!
Updated 5 taps (derailed/k9s, homebrew/bundle, homebrew/services, homebrew/core and homebrew/cask).
==> New Formulae
actions-batch      cargo-fuzz         gimmecert          liborigin          mariadb@11.2       mmdbinspect        npm-check-updates  pkl                scala@3.3
bluez              dhall-toml         gptscript          libsql             mdsh               neosync            pass-import        podman-tui         sigi
bpftop             edbrowse           kubeshark          lsusb-laniksj      mlx                nmail              phodav             prjtrellis         sshpass
==> New Casks
acreom                        deelay                        dosbox-staging                jordanbaird-ice               motion                        overlayed
apidog-europe                 dnsmonitor                    fixkey                        kit                           notesollama                   segger-ozone

You have 21 outdated formulae and 1 outdated cask installed.

==> Tapping hashicorp/tap
Cloning into '/opt/homebrew/Library/Taps/hashicorp/homebrew-tap'...
remote: Enumerating objects: 4176, done.
remote: Counting objects: 100% (477/477), done.
remote: Compressing objects: 100% (127/127), done.
remote: Total 4176 (delta 399), reused 393 (delta 350), pack-reused 3699
Receiving objects: 100% (4176/4176), 762.11 KiB | 1.46 MiB/s, done.
Resolving deltas: 100% (2746/2746), done.
Tapped 2 casks and 28 formulae (87 files, 1MB).
```

brew install hashicorp/tap/terraform

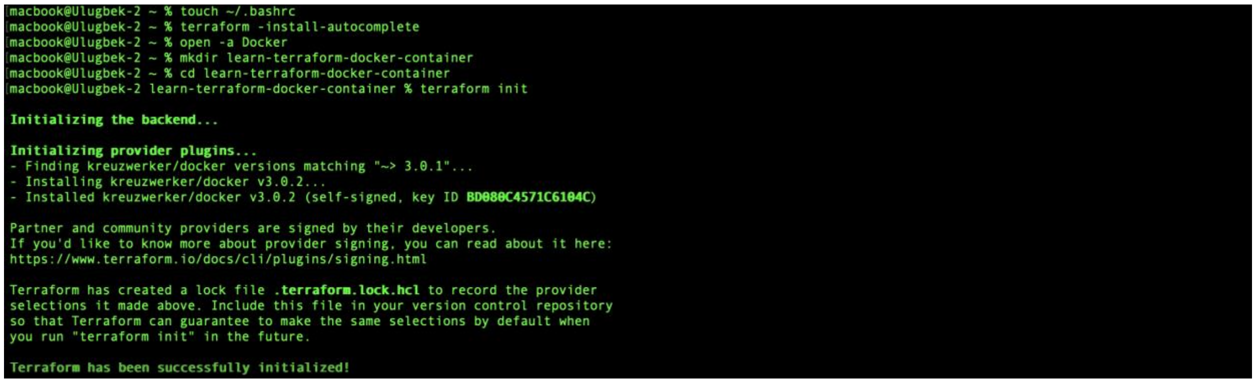


```
macbook@Ulugbek-2 ~ % brew install hashicorp/tap/terraform
==> Fetching hashicorp/tap/terraform
==> Downloading https://releases.hashicorp.com/terraform/1.7.4/terraform_1.7.4_darwin_arm64.zip
########################################################################################################################################################################
==> Installing terraform from hashicorp/tap
Warning: A newer Command Line Tools release is available.
Update them from Software Update in System Settings.

If that doesn't show you any updates, run:
  sudo rm -rf /Library/Developer/CommandLineTools
  sudo xcode-select --install

Alternatively, manually download them from:
  https://developer.apple.com/download/all/.
You should download the Command Line Tools for Xcode 15.1.

Warning: Cask terraform was renamed to homebrew/core/terraform.
🍺  /opt/homebrew/Cellar/terraform/1.7.4: 3 files, 88.6MB, built in 2 seconds
==> Running `brew cleanup terraform`...
Disable this behaviour by setting HOMEBREW_NO_INSTALL_CLEANUP.
Hide these hints with HOMEBREW_NO_ENV_HINTS (see `man brew`).
```

brew update

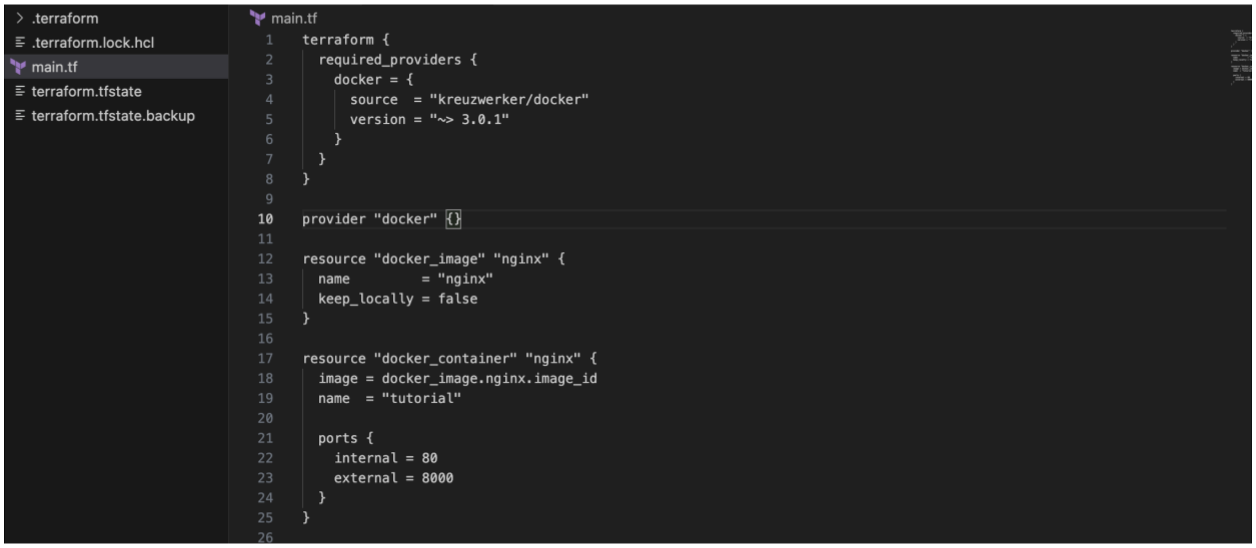


```
macbook@Ulugbek-2 ~ % brew update
Already up-to-date.
macbook@Ulugbek-2 ~ %
```



```
macbook@Ulugbek-2 ~ % touch ~/.bashrc
macbook@Ulugbek-2 ~ % terraform -install-autocomplete
macbook@Ulugbek-2 ~ % open -a Docker
macbook@Ulugbek-2 ~ % mkdir learn-terraform-docker-container
macbook@Ulugbek-2 ~ % cd learn-terraform-docker-container
macbook@Ulugbek-2 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!
```

Added main.tf file

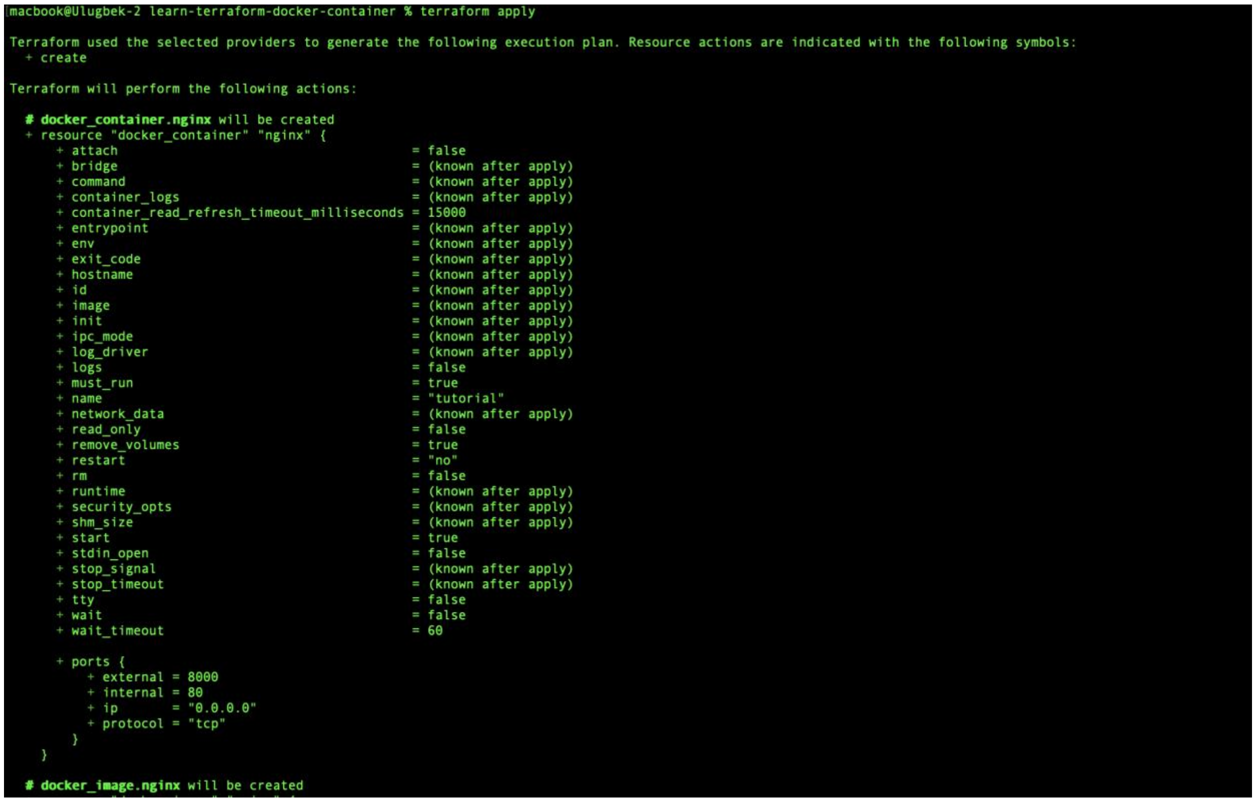


```
> .terraform
  .terraform.lock.hcl
  main.tf
  terraform.tfstate
  terraform.tfstate.backup
```

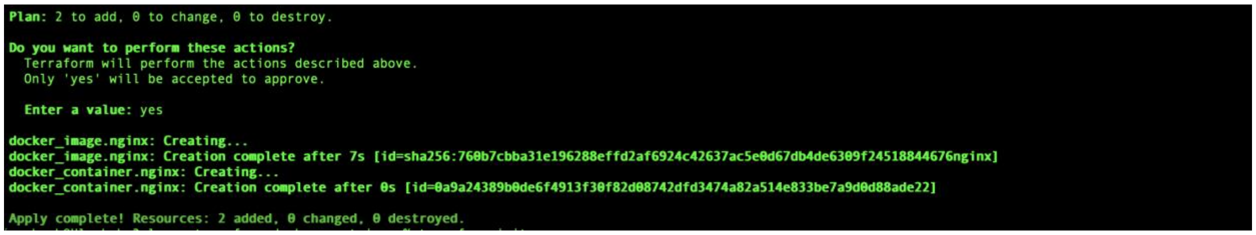
main.tf

```
terraform {
  required_providers {
    docker = {
      source  = "kreuzwerker/docker"
      version = "~> 3.0.1"
    }
  }
}

provider "docker" {}

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

resource "docker_container" "nginx" {
  image = docker_image.nginx.image_id
  name  = "tutorial"

  ports {
    internal = 80
    external = 8000
  }
}
```

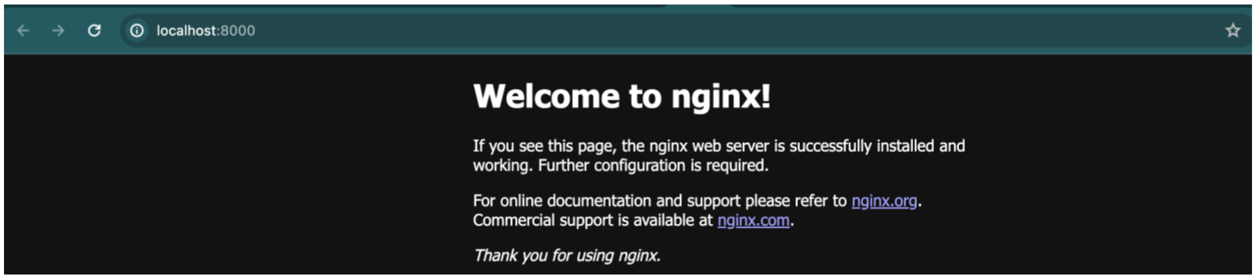
terraform apply



```
macbook@Ulugbek-2 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)
      + image                                       = (known after apply)
      + init                                        = (known after apply)
      + ipc_mode                                    = (known after apply)
      + log_driver                                  = (known after apply)
      + logs                                        = false
      + must_run                                    = true
      + name                                        = "tutorial"
      + network_data                                = (known after apply)
      + read_only                                   = false
      + remove_volumes                              = true
      + restart                                     = "no"
      + rm                                          = false
      + runtime                                     = (known after apply)
      + security_opts                               = (known after apply)
      + shm_size                                    = (known after apply)
      + start                                       = true
      + stdin_open                                  = false
      + 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
```



```
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: Creation complete after 7s [id=sha256:760b7cbba31e196288effd2af6924c42637ac5e0d67db4de6309f24518844676nginx]
docker_container.nginx: Creating...
docker_container.nginx: Creation complete after 0s [id=0a9a24389b0de6f4913f30f82d08742dfd3474a82a514e833be7a9d0d88ade22]

Apply complete! Resources: 2 added, 0 changed, 0 destroyed.
```



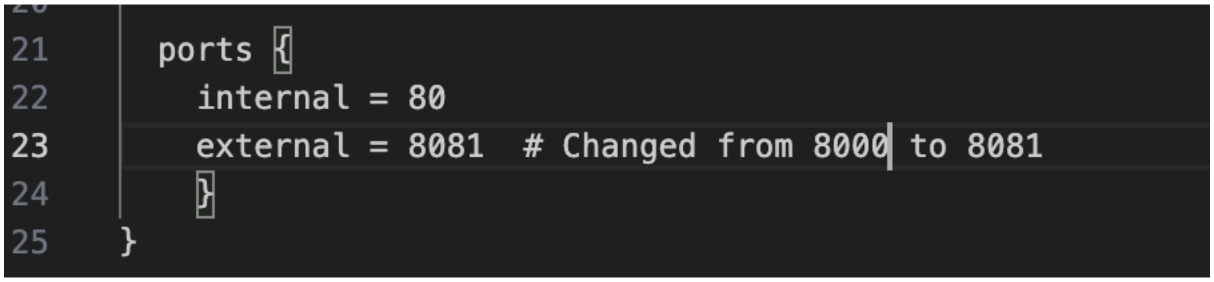
Welcome to nginx!

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.

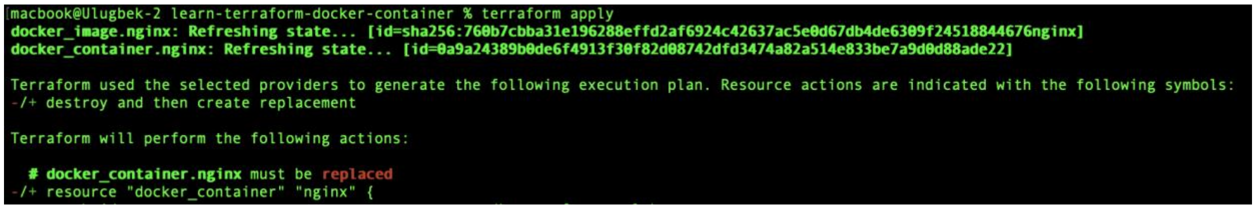
Thank you for using nginx.

changed the external port mapping from 8000 to 8081

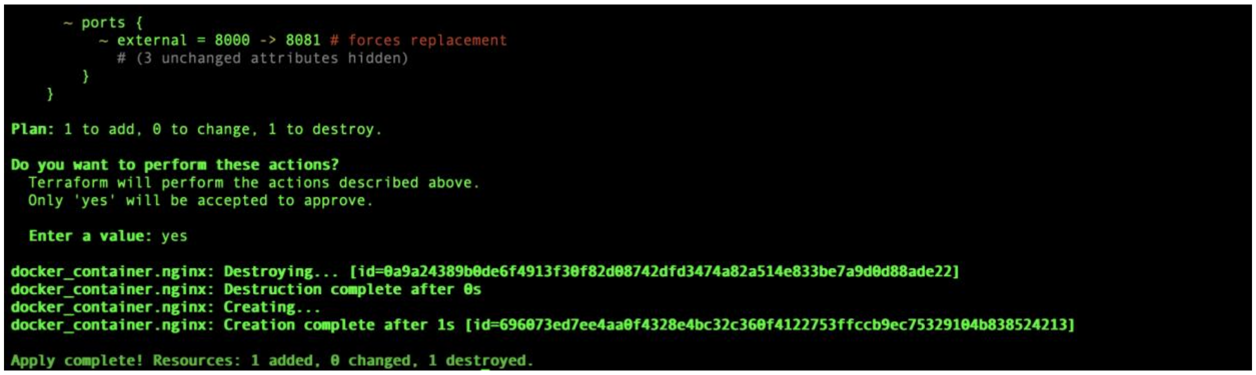


```
    ports {
      internal = 80
      external = 8081   # Changed from 8000 to 8081
    }
}
```

‘terraform apply’



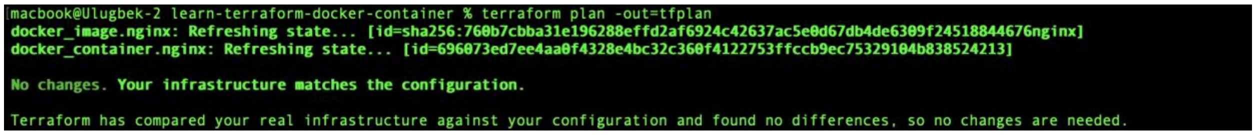
```
macbook@Ulugbek-2 learn-terraform-docker-container % terraform apply
docker_image.nginx: Refreshing state... [id=sha256:760b7cbba31e196288effd2af6924c42637ac5e0d67db4de6309f24518844676nginx]
docker_container.nginx: Refreshing state... [id=0a9a24389b0de6f4913f30f82d08742dfd3474a82a514e833be7a9d0d88ade22]

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols:
-/+ destroy and then create replacement

Terraform will perform the following actions:

  # docker_container.nginx must be replaced
-/+ resource "docker_container" "nginx" {
```



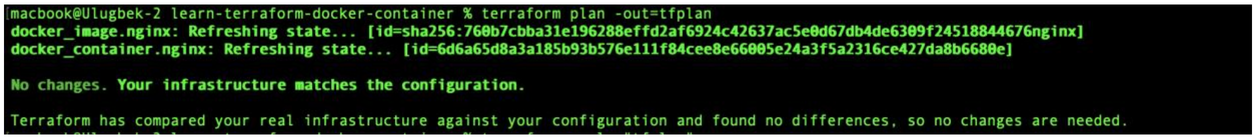
```
      ~ ports {
          ~ external = 8000 -> 8081 # forces replacement
            # (3 unchanged attributes hidden)
        }
    }

Plan: 1 to add, 0 to change, 1 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_container.nginx: Destroying... [id=0a9a24389b0de6f4913f30f82d08742dfd3474a82a514e833be7a9d0d88ade22]
docker_container.nginx: Destruction complete after 0s
docker_container.nginx: Creating...
docker_container.nginx: Creation complete after 1s [id=696073ed7ee4aa0f4328e4bc32c360f4122753ffccb9ec75329104b838524213]

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

Creating Plan

‘terraform plan -out=tfplan’ to create a plan file named ‘tfplan’.

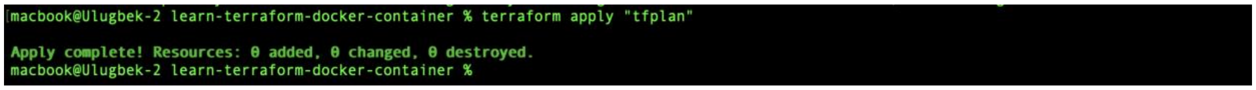


```
macbook@Ulugbek-2 learn-terraform-docker-container % terraform plan -out=tfplan
docker_image.nginx: Refreshing state... [id=sha256:760b7cbba31e196288effd2af6924c42637ac5e0d67db4de6309f24518844676nginx]
docker_container.nginx: Refreshing state... [id=696073ed7ee4aa0f4328e4bc32c360f4122753ffccb9ec75329104b838524213]

No changes. Your infrastructure matches the configuration.

Terraform has compared your real infrastructure against your configuration and found no differences, so no changes are needed.
```



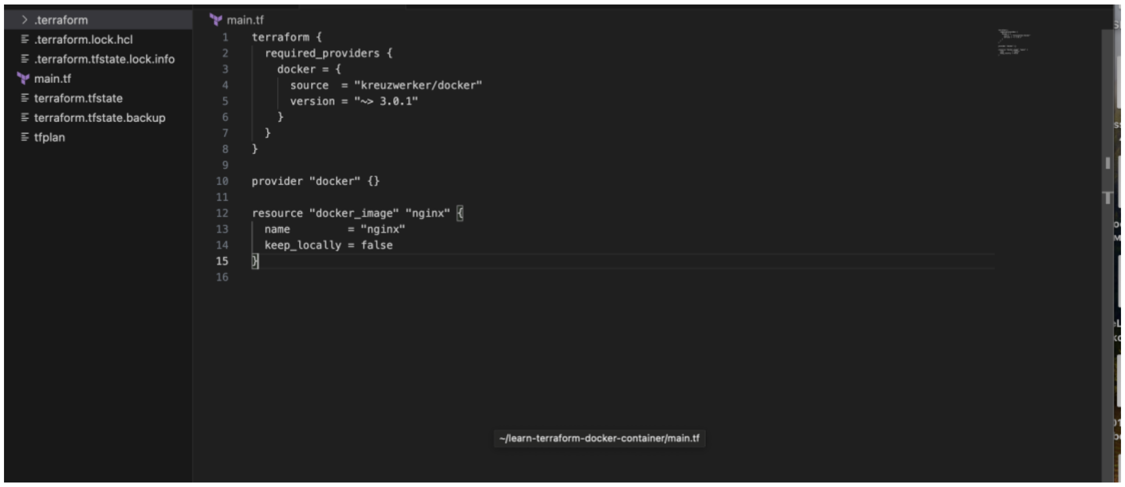
```
macbook@Ulugbek-2 learn-terraform-docker-container % terraform plan -out=tfplan
docker_image.nginx: Refreshing state... [id=sha256:760b7cbba31e196288effd2af6924c42637ac5e0d67db4de6309f24518844676nginx]
docker_container.nginx: Refreshing state... [id=6d6a65d8a3a185b93b576e111f84cee8e66005e24a3f5a2316ce427da8b6680e]

No changes. Your infrastructure matches the configuration.

Terraform has compared your real infrastructure against your configuration and found no differences, so no changes are needed.
```

Terraform apply ‘tplan’



```
macbook@Ulugbek-2 learn-terraform-docker-container % terraform apply "tfplan"

Apply complete! Resources: 0 added, 0 changed, 0 destroyed.
macbook@Ulugbek-2 learn-terraform-docker-container %
```

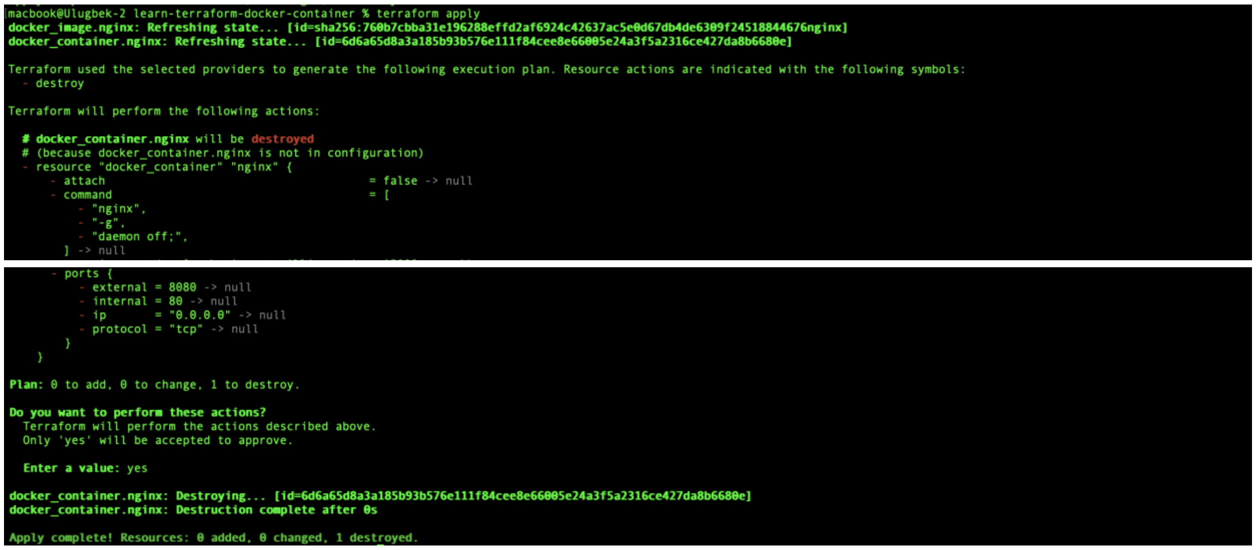
Destructive Changes



```
terraform {
  required_providers {
    docker = {
      source  = "kreuzwerker/docker"
      version = "~> 3.0.1"
    }
  }
}

provider "docker" {}

resource "docker_image" "nginx" {
  name         = "nginx"
  keep_locally = false
}
```

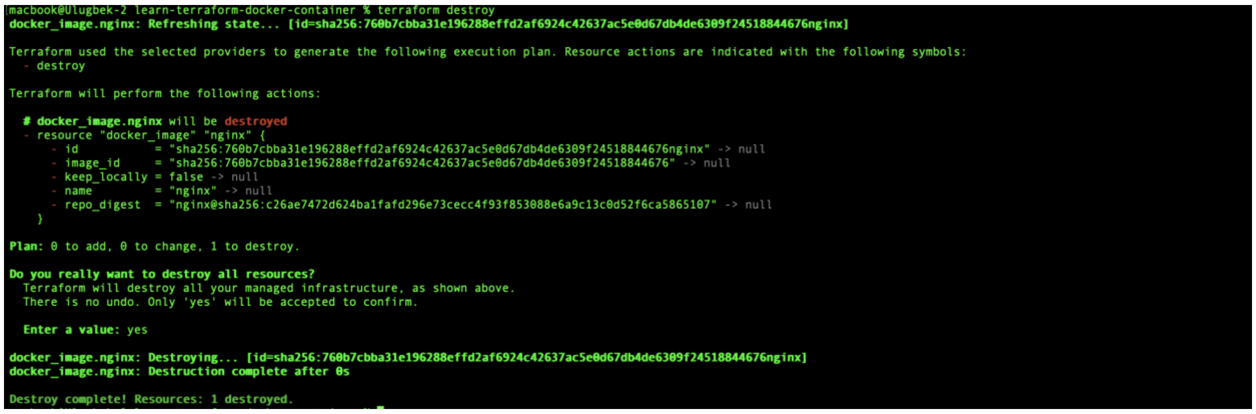
Removed the ‘docker_container’ resource

terraform apply



```
macbook@Ulugbek-2 learn-terraform-docker-container % terraform apply
docker_image.nginx: Refreshing state... [id=sha256:760b7cbba31e196288effd2af6924c42637ac5e0d67db4de6309f24518844676nginx]
docker_container.nginx: Refreshing state... [id=6d6a65d8a3a185b93b576e111f84cee8e66005e24a3f5a2316ce427da8b6680e]

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols:
  - destroy

Terraform will perform the following actions:

  # docker_container.nginx will be destroyed
  # (because docker_container.nginx is not in configuration)
  - resource "docker_container" "nginx" {
      - attach                                      = false -> null
      - command                                     = [
          - "nginx",
          - "-g",
          - "daemon off;",
        ] -> null
      - ports {
          - external = 8080 -> null
          - internal = 80 -> null
          - ip       = "0.0.0.0" -> null
          - protocol = "tcp" -> null
        }
    }

Plan: 0 to add, 0 to change, 1 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_container.nginx: Destroying... [id=6d6a65d8a3a185b93b576e111f84cee8e66005e24a3f5a2316ce427da8b6680e]
docker_container.nginx: Destruction complete after 0s

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

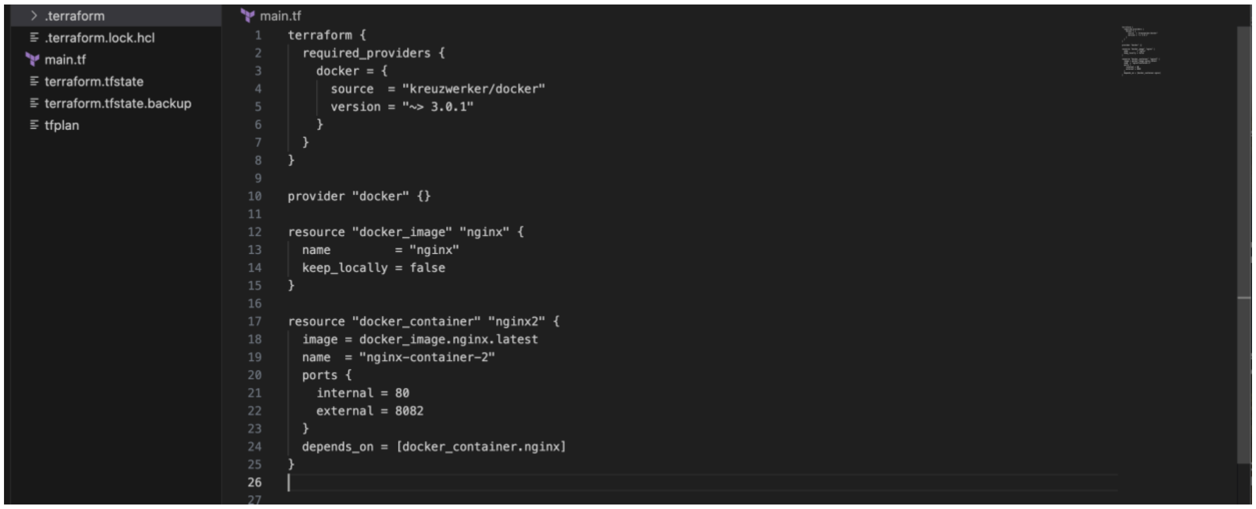
terraform destroy



```
macbook@Ulugbek-2 learn-terraform-docker-container % terraform destroy
docker_image.nginx: Refreshing state... [id=sha256:760b7cbba31e196288effd2af6924c42637ac5e0d67db4de6309f24518844676nginx]

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols:
  - destroy

Terraform will perform the following actions:

  # docker_image.nginx will be destroyed
  - resource "docker_image" "nginx" {
      - id           = "sha256:760b7cbba31e196288effd2af6924c42637ac5e0d67db4de6309f24518844676nginx" -> null
      - image_id     = "sha256:760b7cbba31e196288effd2af6924c42637ac5e0d67db4de6309f24518844676" -> null
      - keep_locally = false -> null
      - name         = "nginx" -> null
      - repo_digest  = "nginx@sha256:c26ae7472d624ba1fafd296e73cecc4f93f853088e6a9c13c0d52f6ca5865107" -> null
    }

Plan: 0 to add, 0 to change, 1 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_image.nginx: Destroying... [id=sha256:760b7cbba31e196288effd2af6924c42637ac5e0d67db4de6309f24518844676nginx]
docker_image.nginx: Destruction complete after 0s

Destroy complete! Resources: 1 destroyed.
```

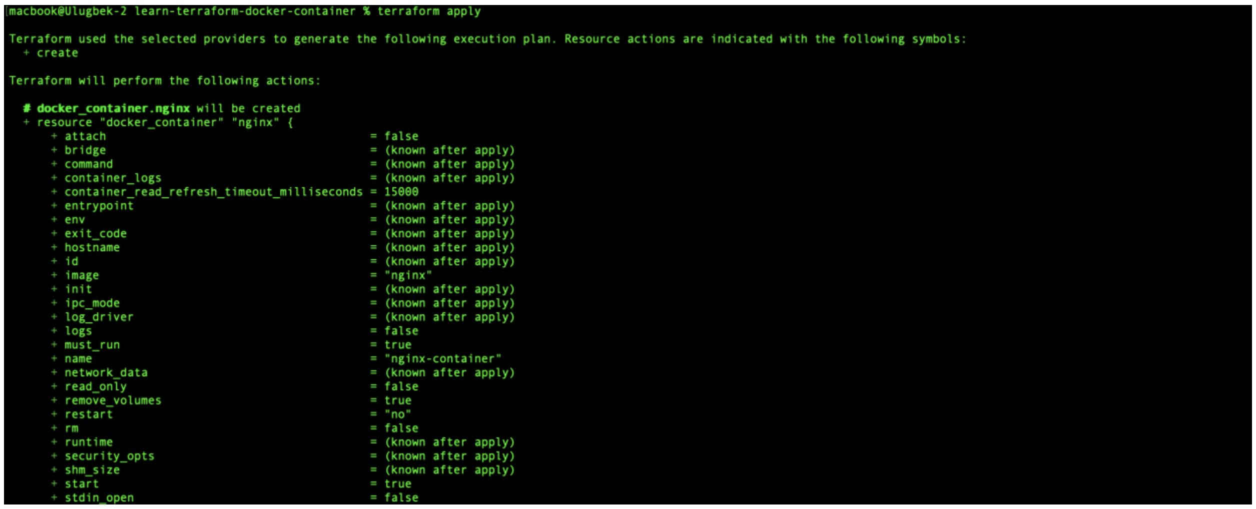
Create Resources with Dependencies

Added a new resource to main.tf file that depends on the existing docker_image resource

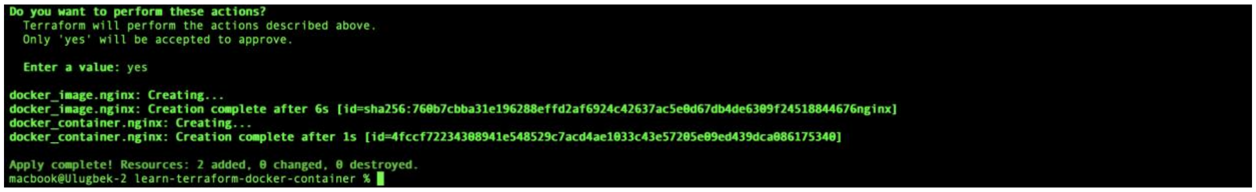


```
terraform {
  required_providers {
    docker = {
      source  = "kreuzwerker/docker"
      version = "~> 3.0.1"
    }
  }
}

provider "docker" {}

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

resource "docker_container" "nginx2" {
  image = docker_image.nginx.latest
  name  = "nginx-container-2"
  ports {
    internal = 80
    external = 8082
  }
  depends_on = [docker_container.nginx]
}

```

terraform apply



```
macbook@Ulugbek-2 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)
      + image                                       = "nginx"
      + init                                        = (known after apply)
      + ipc_mode                                    = (known after apply)
      + log_driver                                  = (known after apply)
      + logs                                        = false
      + must_run                                    = true
      + name                                        = "nginx-container"
      + network_data                                = (known after apply)
      + read_only                                   = false
      + remove_volumes                              = true
      + restart                                     = "no"
      + rm                                          = false
      + runtime                                     = (known after apply)
      + security_opts                               = (known after apply)
      + shm_size                                    = (known after apply)
      + start                                       = true
      + stdin_open                                  = false
```



```
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: Creation complete after 6s [id=sha256:760b7cbba31e196288effd2af6924c42637ac5e0d67db4de6309f24518844676nginx]
docker_container.nginx: Creating...
docker_container.nginx: Creation complete after 1s [id=4fccf72234308941e548529c7acd4ae1033c43e57205e09ed439dca086175340]

Apply complete! Resources: 2 added, 0 changed, 0 destroyed.
macbook@Ulugbek-2 learn-terraform-docker-container % 
```