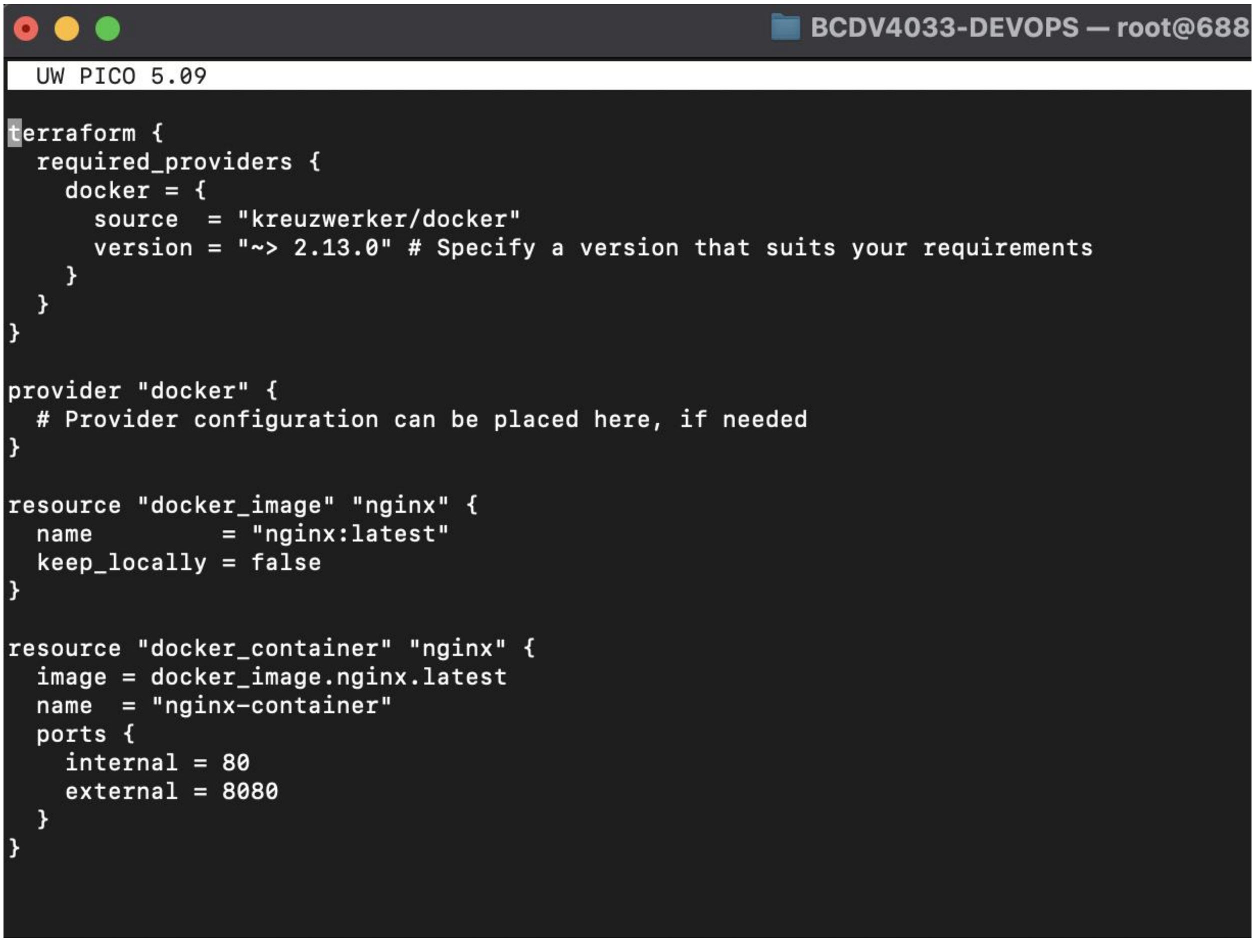
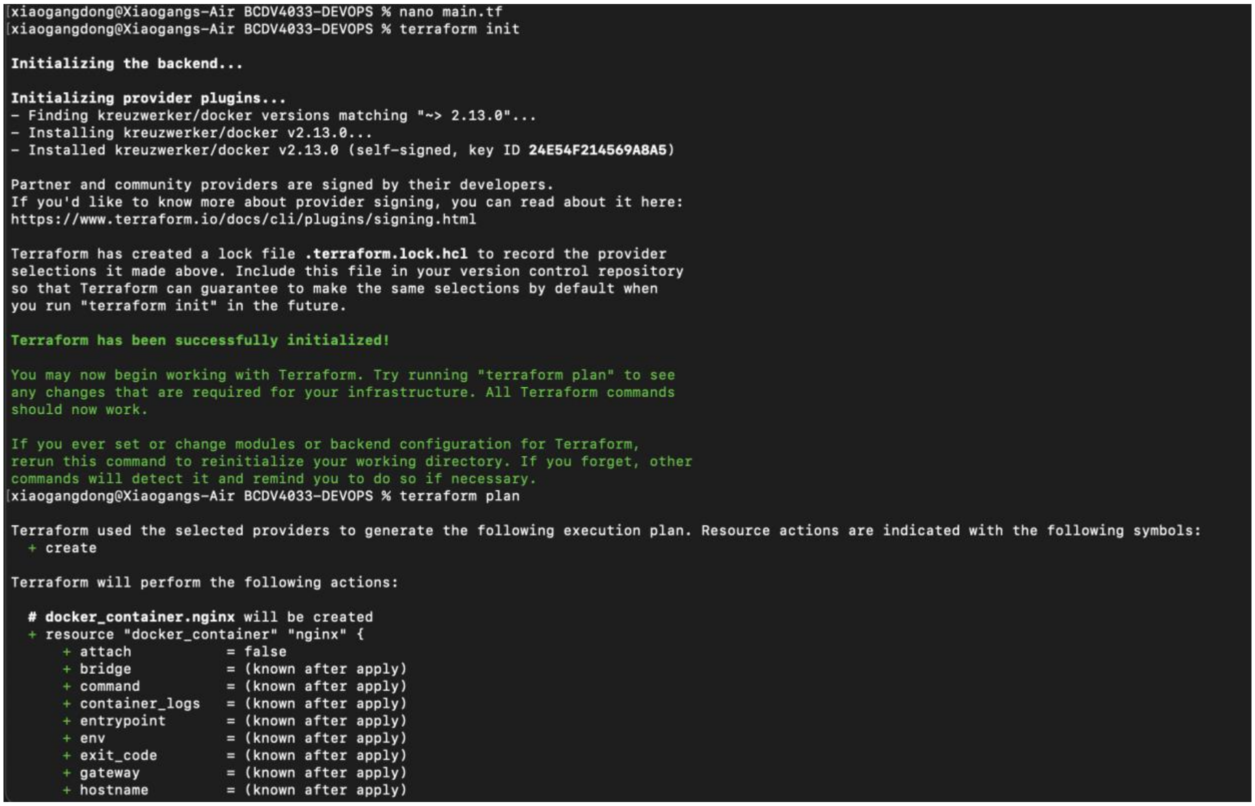
BCDV4033 – LAB4

BY Xiaogang Dong 101492108

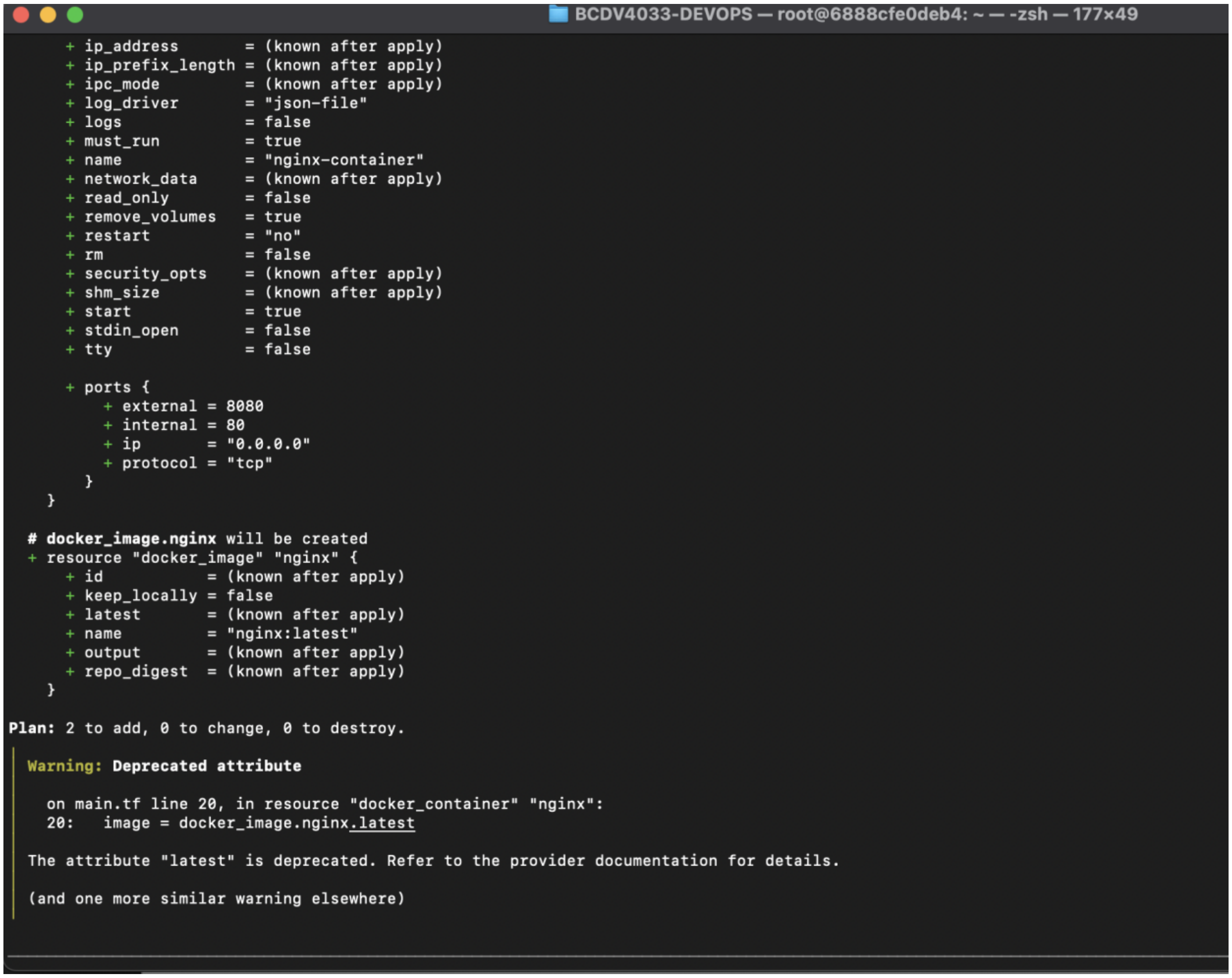
Install Terraform, init and apply



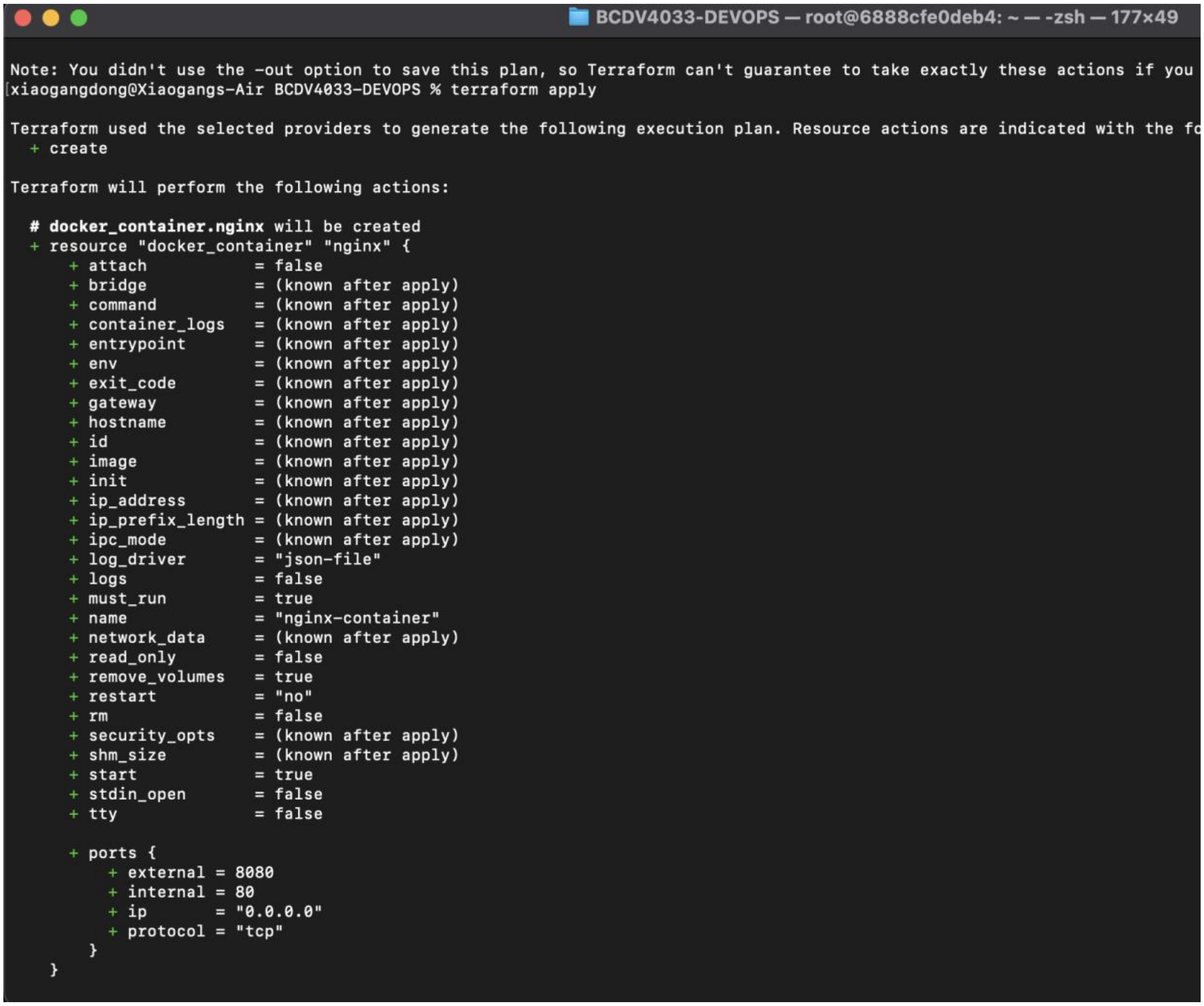
```
  UW PICO 5.09

terraform {
  required_providers {
    docker = {
      source  = "kreuzwerker/docker"
      version = "~> 2.13.0" # Specify a version that suits your requirements
    }
  }
}

provider "docker" {
  # Provider configuration can be placed here, if needed
}

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

resource "docker_container" "nginx" {
  image = docker_image.nginx.latest
  name  = "nginx-container"
  ports {
    internal = 80
    external = 8080
  }
}
```



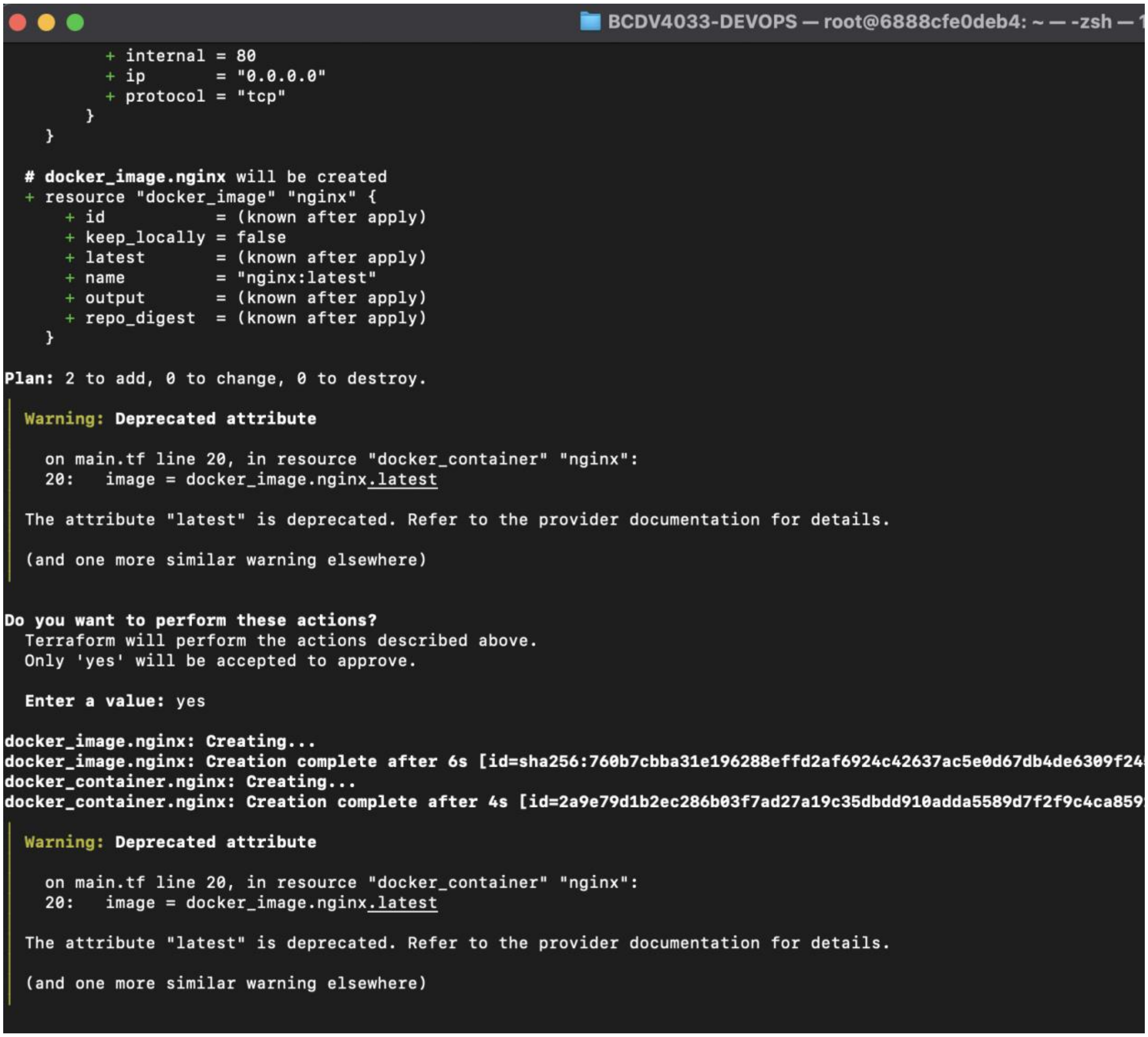
```
xiaogangdong@Xiaogangs-Air BCDV4033-DEVOPS % nano main.tf
xiaogangdong@Xiaogangs-Air BCDV4033-DEVOPS % terraform init

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!

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.
xiaogangdong@Xiaogangs-Air BCDV4033-DEVOPS % 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:

  # 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)
```



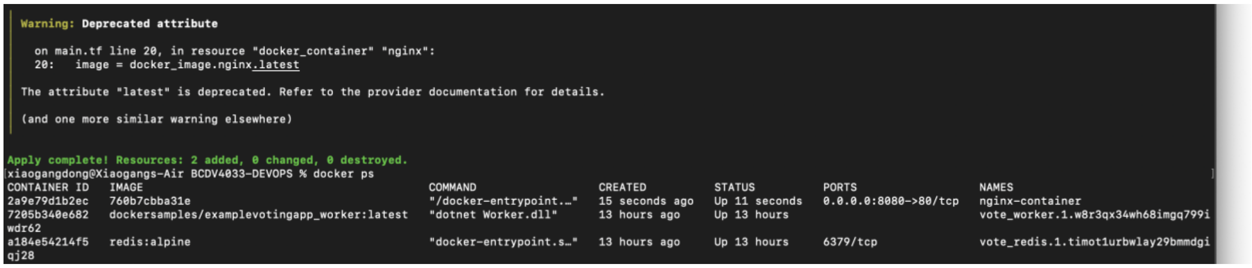
```
      + ip_address        = (known after apply)
      + ip_prefix_length  = (known after apply)
      + ipc_mode          = (known after apply)
      + log_driver        = "json-file"
      + logs              = false
      + must_run          = true
      + name              = "nginx-container"
      + network_data      = (known after apply)
      + read_only         = false
      + remove_volumes    = true
      + restart           = "no"
      + rm                = false
      + security_opts     = (known after apply)
      + shm_size          = (known after apply)
      + start             = true
      + stdin_open        = false
      + tty               = false

      + ports {
          + external = 8080
          + internal = 80
          + ip       = "0.0.0.0"
          + protocol = "tcp"
        }
    }

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

Plan: 2 to add, 0 to change, 0 to destroy.
│
│ Warning: Deprecated attribute
│
│   on main.tf line 20, in resource "docker_container" "nginx":
│   20:   image = docker_image.nginx.latest
│
│ The attribute "latest" is deprecated. Refer to the provider documentation for details.
│
│ (and one more similar warning elsewhere)
│
```



```
Note: You didn't use the -out option to save this plan, so Terraform can't guarantee to take exactly these actions if you
[xiaogangdong@Xiaogangs-Air BCDV4033-DEVOPS % terraform apply

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the fo
  + 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)
      + log_driver       = "json-file"
      + logs             = false
      + must_run         = true
      + name             = "nginx-container"
      + network_data     = (known after apply)
      + read_only        = false
      + remove_volumes   = true
      + restart          = "no"
      + rm               = false
      + security_opts    = (known after apply)
      + shm_size         = (known after apply)
      + start            = true
      + stdin_open       = false
      + tty              = false

      + ports {
          + external = 8080
          + internal = 80
          + ip       = "0.0.0.0"
          + protocol = "tcp"
        }
    }
```



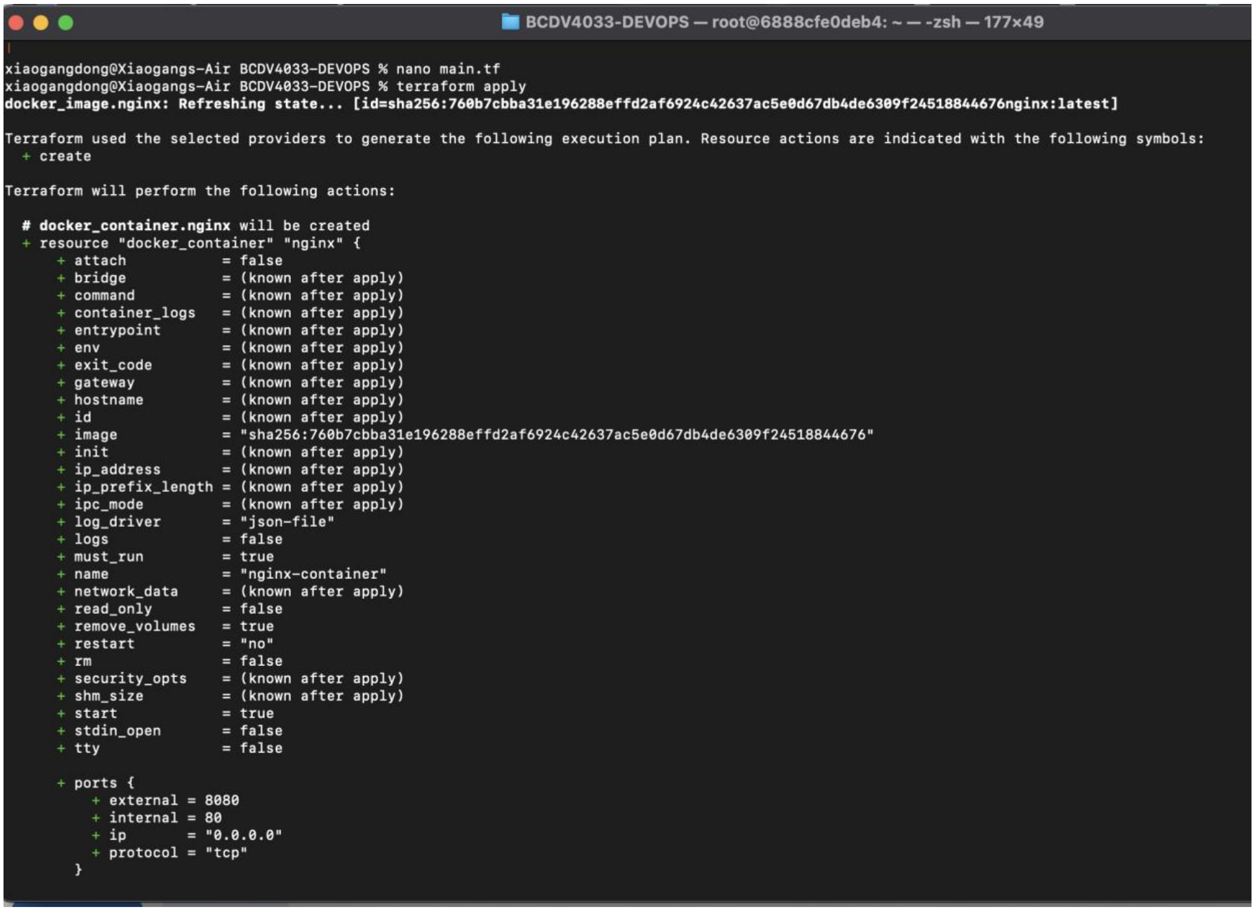
```
          + internal = 80
          + ip       = "0.0.0.0"
          + protocol = "tcp"
        }
    }

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

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

  Warning: Deprecated attribute

    on main.tf line 20, in resource "docker_container" "nginx":
    20:   image = docker_image.nginx.latest

  The attribute "latest" is deprecated. Refer to the provider documentation for details.

  (and one more similar warning elsewhere)


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=sha256:760b7cbba31e196288effd2af6924c42637ac5e0d67db4de6309f24
docker_container.nginx: Creating...
docker_container.nginx: Creation complete after 4s [id=2a9e79d1b2ec286b03f7ad27a19c35dbdd910adda5589d7f2f9c4ca859

  Warning: Deprecated attribute

    on main.tf line 20, in resource "docker_container" "nginx":
    20:   image = docker_image.nginx.latest

  The attribute "latest" is deprecated. Refer to the provider documentation for details.

  (and one more similar warning elsewhere)
```



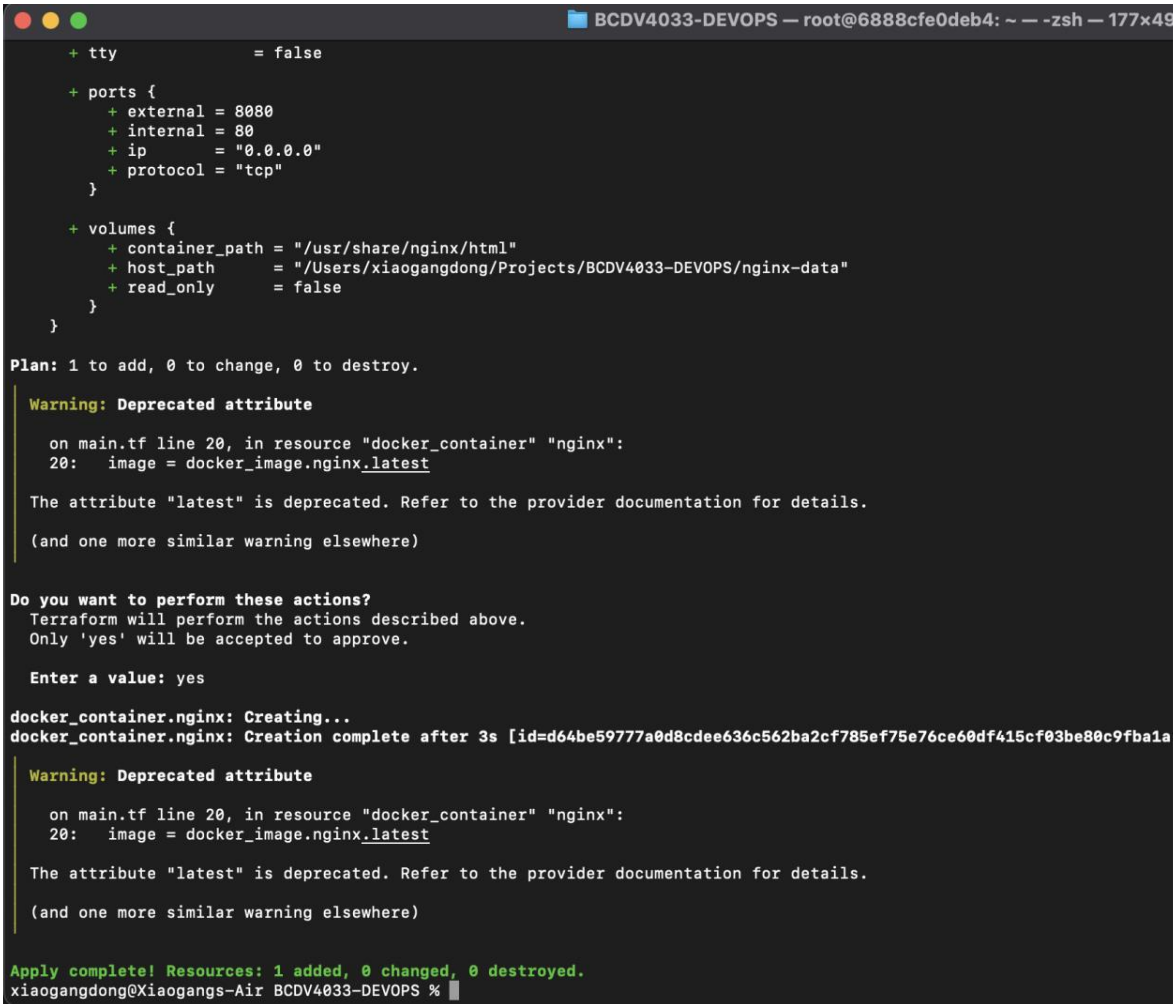
```
  Warning: Deprecated attribute

    on main.tf line 20, in resource "docker_container" "nginx":
    20:   image = docker_image.nginx.latest

  The attribute "latest" is deprecated. Refer to the provider documentation for details.

  (and one more similar warning elsewhere)


Apply complete! Resources: 2 added, 0 changed, 0 destroyed.
xiaogangdong@Xiaogangs-Air BCDV4033-DEVOPS % docker ps
CONTAINER ID   IMAGE                                       COMMAND                  CREATED          STATUS          PORTS                  NAMES
2a9e79d1b2ec   760b7cbba31e                                "/docker-entrypoint.…"   15 seconds ago   Up 11 seconds   0.0.0.0:8080->80/tcp   nginx-container
7205b340e682   dockersamples/examplevotingapp_worker:latest   "dotnet Worker.dll"      13 hours ago     Up 13 hours                            vote_worker.1.w8r3qx34wh68imgq799i
wdr62
a184e54214f5   redis:alpine                                "docker-entrypoint.s…"   13 hours ago     Up 13 hours     6379/tcp               vote_redis.1.timot1urbwlay29bmmdgi
qj28
```

Change infrastructure (added volume)

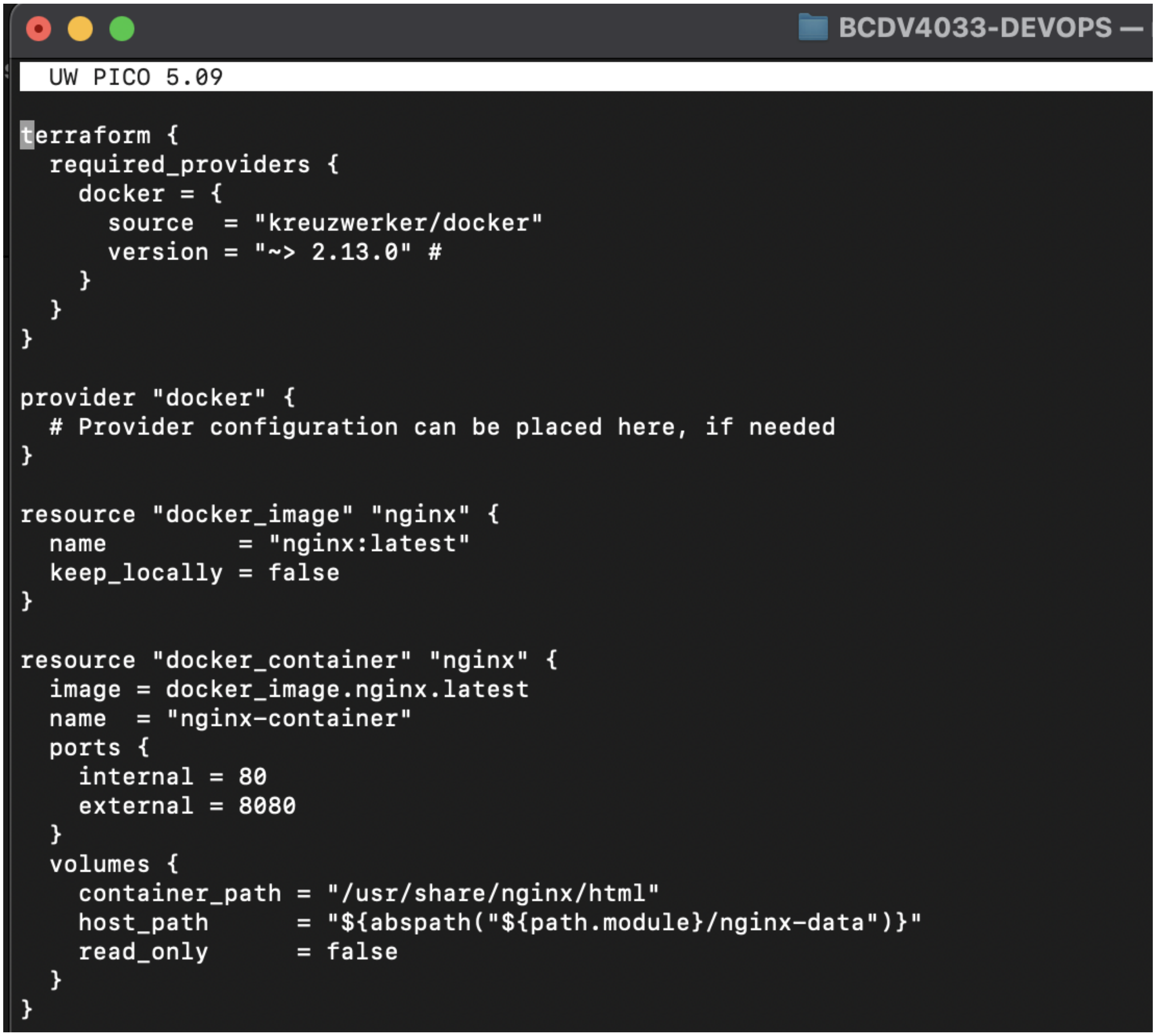
Create a plan and use the plan to make changes to the resource



```
xiaogangdong@Xiaogangs-Air BCDV4033-DEVOPS % nano main.tf
xiaogangdong@Xiaogangs-Air BCDV4033-DEVOPS % terraform apply
docker_image.nginx: Refreshing state... [id=sha256:760b7cbba31e196288effd2af6924c42637ac5e0d67db4de6309f24518844676nginx:latest]

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            = "sha256:760b7cbba31e196288effd2af6924c42637ac5e0d67db4de6309f24518844676"
      + init             = (known after apply)
      + ip_address       = (known after apply)
      + ip_prefix_length = (known after apply)
      + ipc_mode         = (known after apply)
      + log_driver       = "json-file"
      + logs             = false
      + must_run         = true
      + name             = "nginx-container"
      + network_data     = (known after apply)
      + read_only        = false
      + remove_volumes   = true
      + restart          = "no"
      + rm               = false
      + security_opts    = (known after apply)
      + shm_size         = (known after apply)
      + start            = true
      + stdin_open       = false
      + tty              = false

      + ports {
          + external = 8080
          + internal = 80
          + ip       = "0.0.0.0"
          + protocol = "tcp"
        }
```



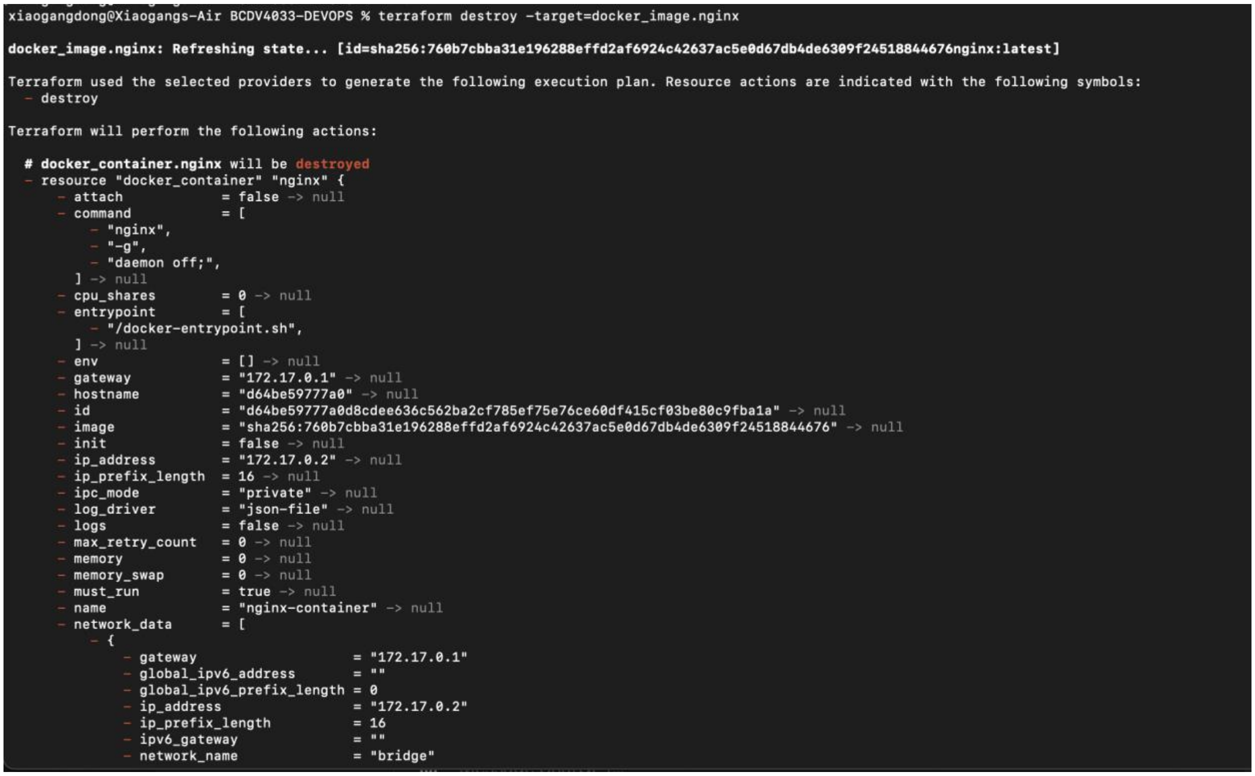
```
      + tty               = false

      + ports {
          + external = 8080
          + internal = 80
          + ip       = "0.0.0.0"
          + protocol = "tcp"
        }

      + volumes {
          + container_path = "/usr/share/nginx/html"
          + host_path      = "/Users/xiaogangdong/Projects/BCDV4033-DEVOPS/nginx-data"
          + read_only      = false
        }
    }

Plan: 1 to add, 0 to change, 0 to destroy.
│
│ Warning: Deprecated attribute
│
│   on main.tf line 20, in resource "docker_container" "nginx":
│   20:   image = docker_image.nginx.latest
│
│ The attribute "latest" is deprecated. Refer to the provider documentation for details.
│
│ (and one more similar warning elsewhere)
│

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: Creating...
docker_container.nginx: Creation complete after 3s [id=d64be59777a0d8cdee636c562ba2cf785ef75e76ce60df415cf03be80c9fba1a
│
│ Warning: Deprecated attribute
│
│   on main.tf line 20, in resource "docker_container" "nginx":
│   20:   image = docker_image.nginx.latest
│
│ The attribute "latest" is deprecated. Refer to the provider documentation for details.
│
│ (and one more similar warning elsewhere)
│

Apply complete! Resources: 1 added, 0 changed, 0 destroyed.
xiaogangdong@Xiaogangs-Air BCDV4033-DEVOPS % 
```



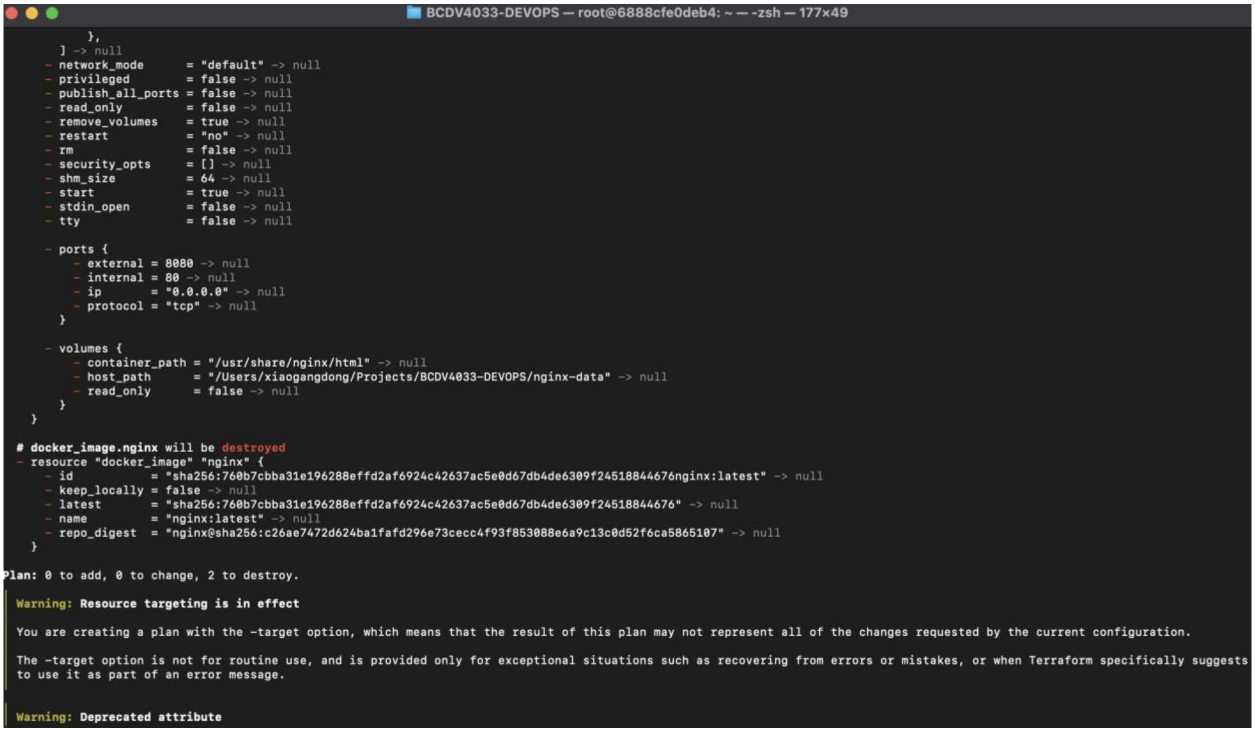
```
terraform {
  required_providers {
    docker = {
      source  = "kreuzwerker/docker"
      version = "~> 2.13.0" #
    }
  }
}

provider "docker" {
  # Provider configuration can be placed here, if needed
}

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

resource "docker_container" "nginx" {
  image = docker_image.nginx.latest
  name  = "nginx-container"
  ports {
    internal = 80
    external = 8080
  }
  volumes {
    container_path = "/usr/share/nginx/html"
    host_path      = "${abspath("${path.module}/nginx-data")}"
    read_only      = false
  }
}
```

Make destructive changes (like removing one of the docker images)

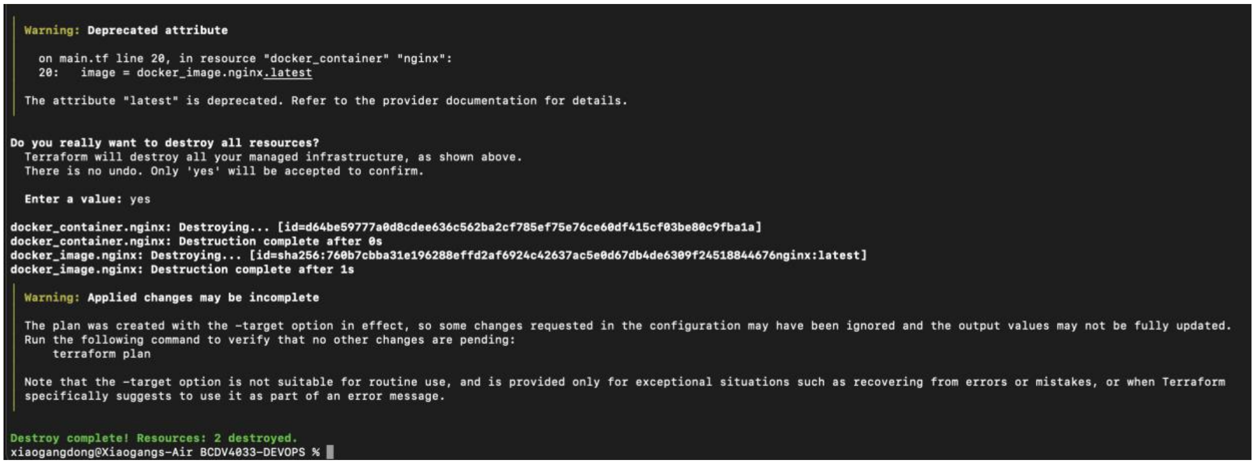
Destory the complete resource.



```
xiaogangdong@Xiaogangs-Air BCDV4033-DEVOPS % terraform destroy -target=docker_image.nginx

docker_image.nginx: Refreshing state... [id=sha256:760b7cbba31e196288effd2af6924c42637ac5e0d67db4de6309f24518844676nginx:latest]

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
  - resource "docker_container" "nginx" {
      - attach            = false -> null
      - command           = [
          - "nginx",
          - "-g",
          - "daemon off;",
        ] -> null
      - cpu_shares        = 0 -> null
      - entrypoint        = [
          - "/docker-entrypoint.sh",
        ] -> null
      - env               = [] -> null
      - gateway           = "172.17.0.1" -> null
      - hostname          = "d64be59777a0" -> null
      - id                = "d64be59777a0d8cdee636c562ba2cf785ef75e76ce60df415cf03be80c9fba1a" -> null
      - image             = "sha256:760b7cbba31e196288effd2af6924c42637ac5e0d67db4de6309f24518844676" -> null
      - init              = false -> null
      - ip_address        = "172.17.0.2" -> null
      - ip_prefix_length  = 16 -> null
      - ipc_mode          = "private" -> null
      - log_driver        = "json-file" -> null
      - logs              = false -> null
      - max_retry_count   = 0 -> null
      - memory            = 0 -> null
      - memory_swap       = 0 -> null
      - must_run          = true -> null
      - name              = "nginx-container" -> null
      - network_data      = [
          - {
              - gateway                   = "172.17.0.1"
              - global_ipv6_address       = ""
              - global_ipv6_prefix_length = 0
              - ip_address                = "172.17.0.2"
              - ip_prefix_length          = 16
              - ipv6_gateway              = ""
              - network_name              = "bridge"
```

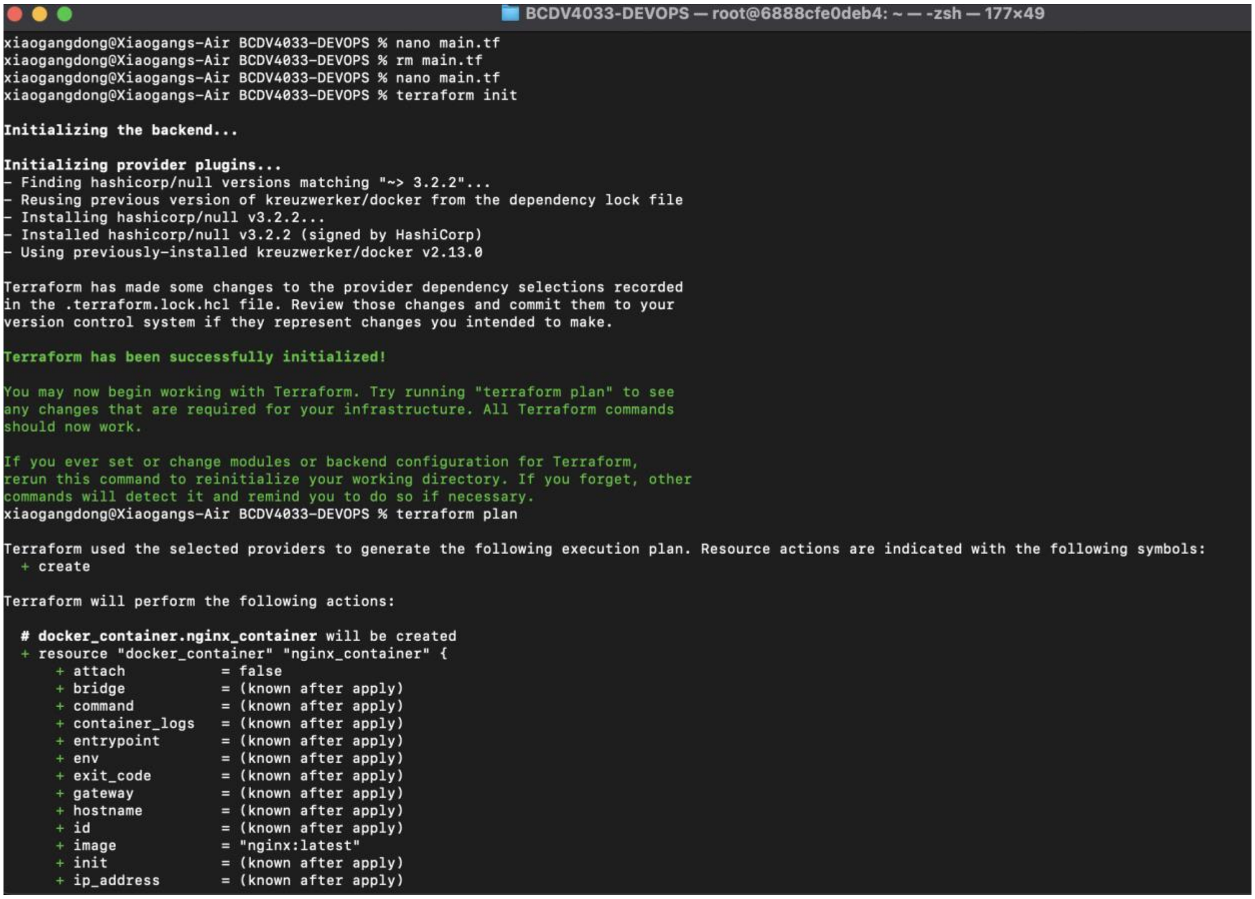


```
            },
        ] -> null
      - network_mode      = "default" -> null
      - privileged        = false -> null
      - publish_all_ports = false -> null
      - read_only         = false -> null
      - remove_volumes    = true -> null
      - restart           = "no" -> null
      - rm                = false -> null
      - security_opts     = [] -> null
      - shm_size          = 64 -> null
      - start             = true -> null
      - stdin_open        = false -> null
      - tty               = false -> null

      - ports {
          - external = 8080 -> null
          - internal = 80 -> null
          - ip       = "0.0.0.0" -> null
          - protocol = "tcp" -> null
        }

      - volumes {
          - container_path = "/usr/share/nginx/html" -> null
          - host_path      = "/Users/xiaogangdong/Projects/BCDV4033-DEVOPS/nginx-data" -> null
          - read_only      = false -> null
        }
    }

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

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

  Warning: Resource targeting is in effect

  You are creating a plan with the -target option, which means that the result of this plan may not represent all of the changes requested by the current configuration.

  The -target option is not for routine use, and is provided only for exceptional situations such as recovering from errors or mistakes, or when Terraform specifically suggests
  to use it as part of an error message.


  Warning: Deprecated attribute
```

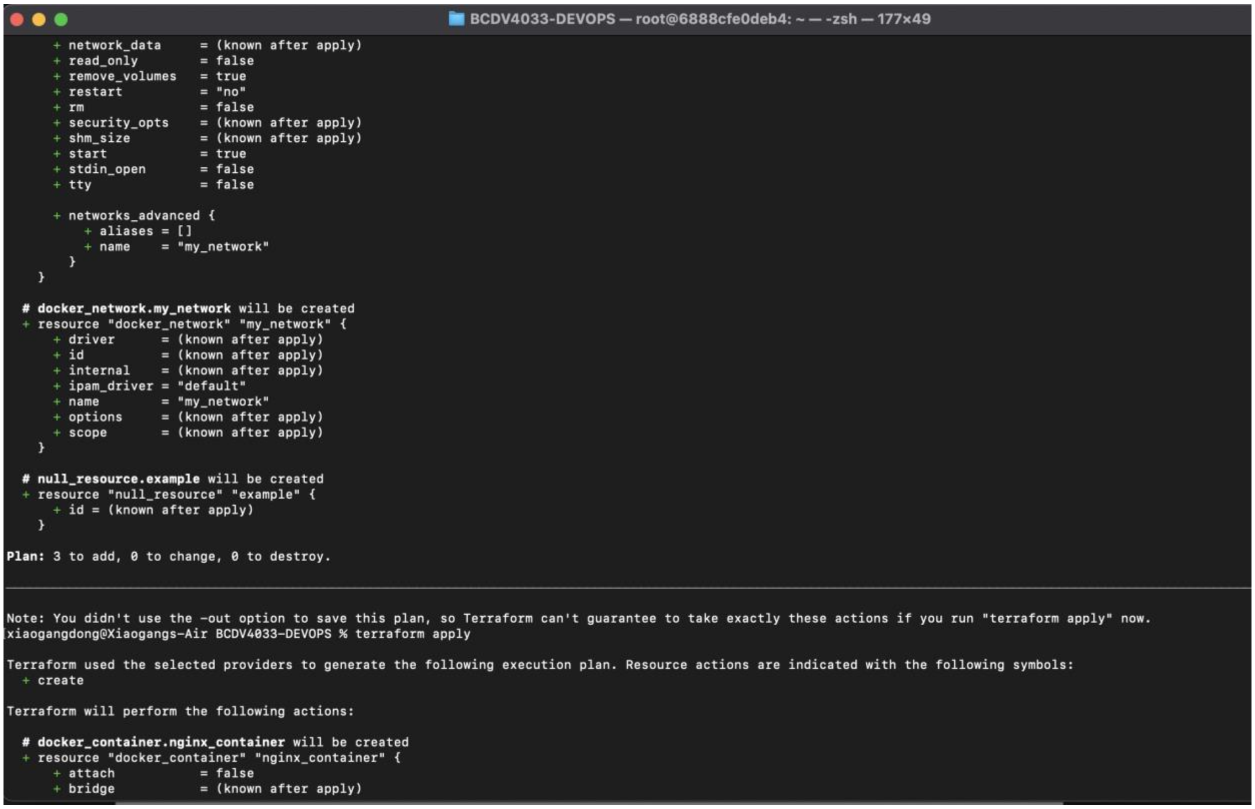


```
│ Warning: Deprecated attribute
│
│   on main.tf line 20, in resource "docker_container" "nginx":
│   20:   image = docker_image.nginx.latest
│
│ The attribute "latest" is deprecated. Refer to the provider documentation for details.


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=d64be59777a0d8cdee636c562ba2cf785ef75e76ce60df415cf03be80c9fba1a]
docker_container.nginx: Destruction complete after 0s
docker_image.nginx: Destroying... [id=sha256:760b7cbba31e196288effd2af6924c42637ac5e0d67db4de6309f24518844676nginx:latest]
docker_image.nginx: Destruction complete after 1s
│ Warning: Applied changes may be incomplete
│
│ The plan was created with the -target option in effect, so some changes requested in the configuration may have been ignored and the output values may not be fully updated.
│ Run the following command to verify that no other changes are pending:
│     terraform plan
│
│ Note that the -target option is not suitable for routine use, and is provided only for exceptional situations such as recovering from errors or mistakes, or when Terraform
│ specifically suggests to use it as part of an error message.


Destroy complete! Resources: 2 destroyed.
xiaogangdong@Xiaogangs-Air BCDV4033-DEVOPS % 
```

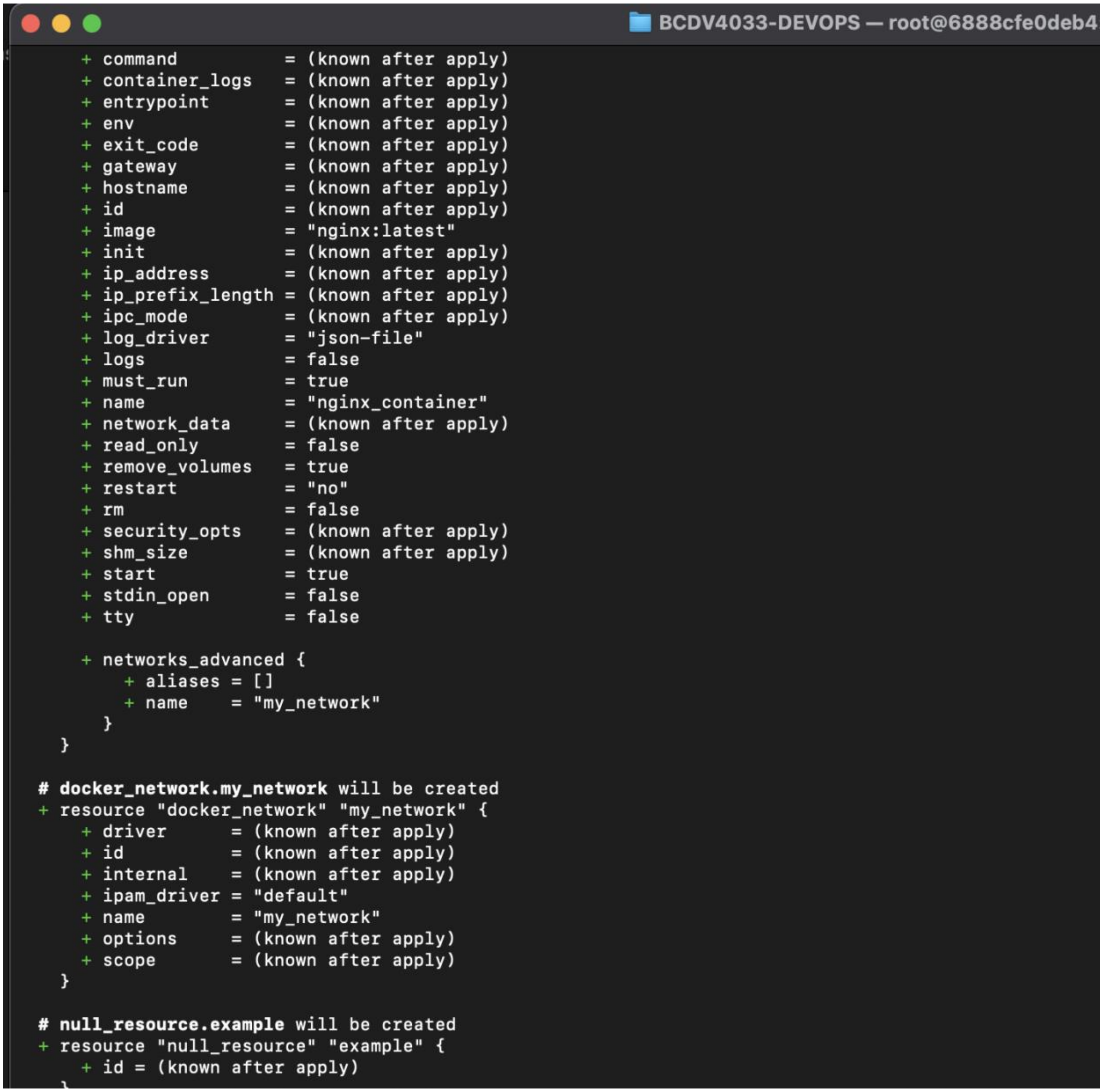
Create resource with dependencies (implicit and explicit).



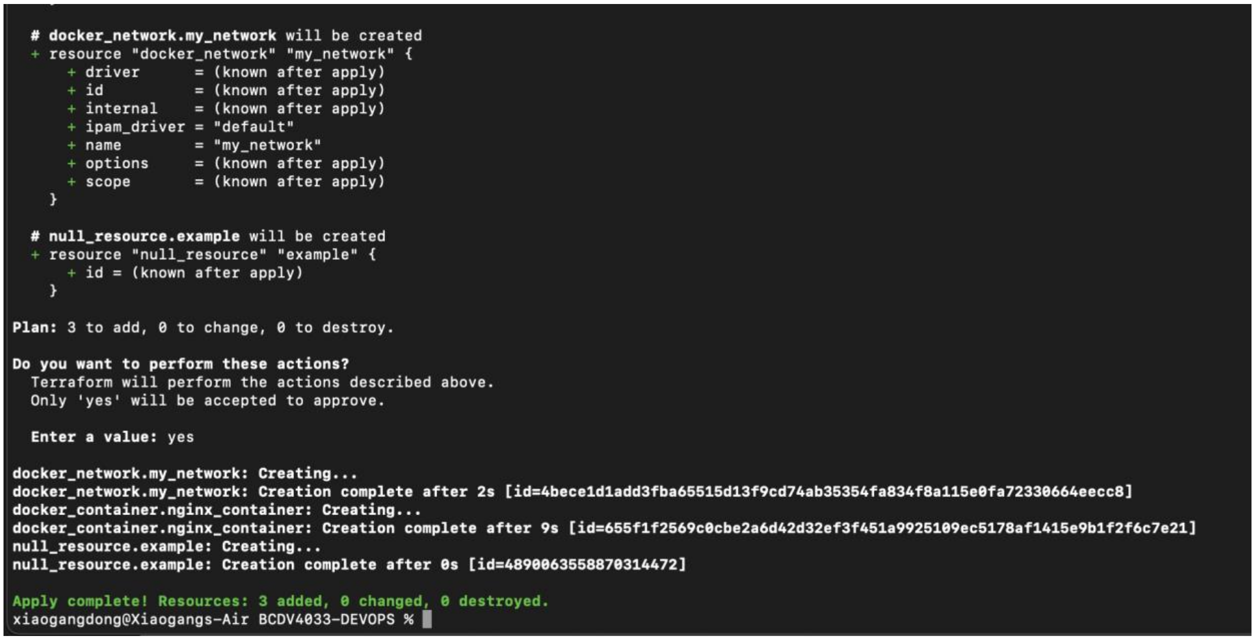
```
xiaogangdong@Xiaogangs-Air BCDV4033-DEVOPS % nano main.tf
xiaogangdong@Xiaogangs-Air BCDV4033-DEVOPS % rm main.tf
xiaogangdong@Xiaogangs-Air BCDV4033-DEVOPS % nano main.tf
xiaogangdong@Xiaogangs-Air BCDV4033-DEVOPS % terraform init

Initializing the backend...

Initializing provider plugins...
- Finding hashicorp/null versions matching "~> 3.2.2"...
- Reusing previous version of kreuzwerker/docker from the dependency lock file
- Installing hashicorp/null v3.2.2...
- Installed hashicorp/null v3.2.2 (signed by HashiCorp)
- Using previously-installed kreuzwerker/docker v2.13.0

Terraform has made some changes to the provider dependency selections recorded
in the .terraform.lock.hcl file. Review those changes and commit them to your
version control system if they represent changes you intended to make.

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.
xiaogangdong@Xiaogangs-Air BCDV4033-DEVOPS % 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:

  # docker_container.nginx_container will be created
  + resource "docker_container" "nginx_container" {
      + 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            = "nginx:latest"
      + init             = (known after apply)
      + ip_address       = (known after apply)
```



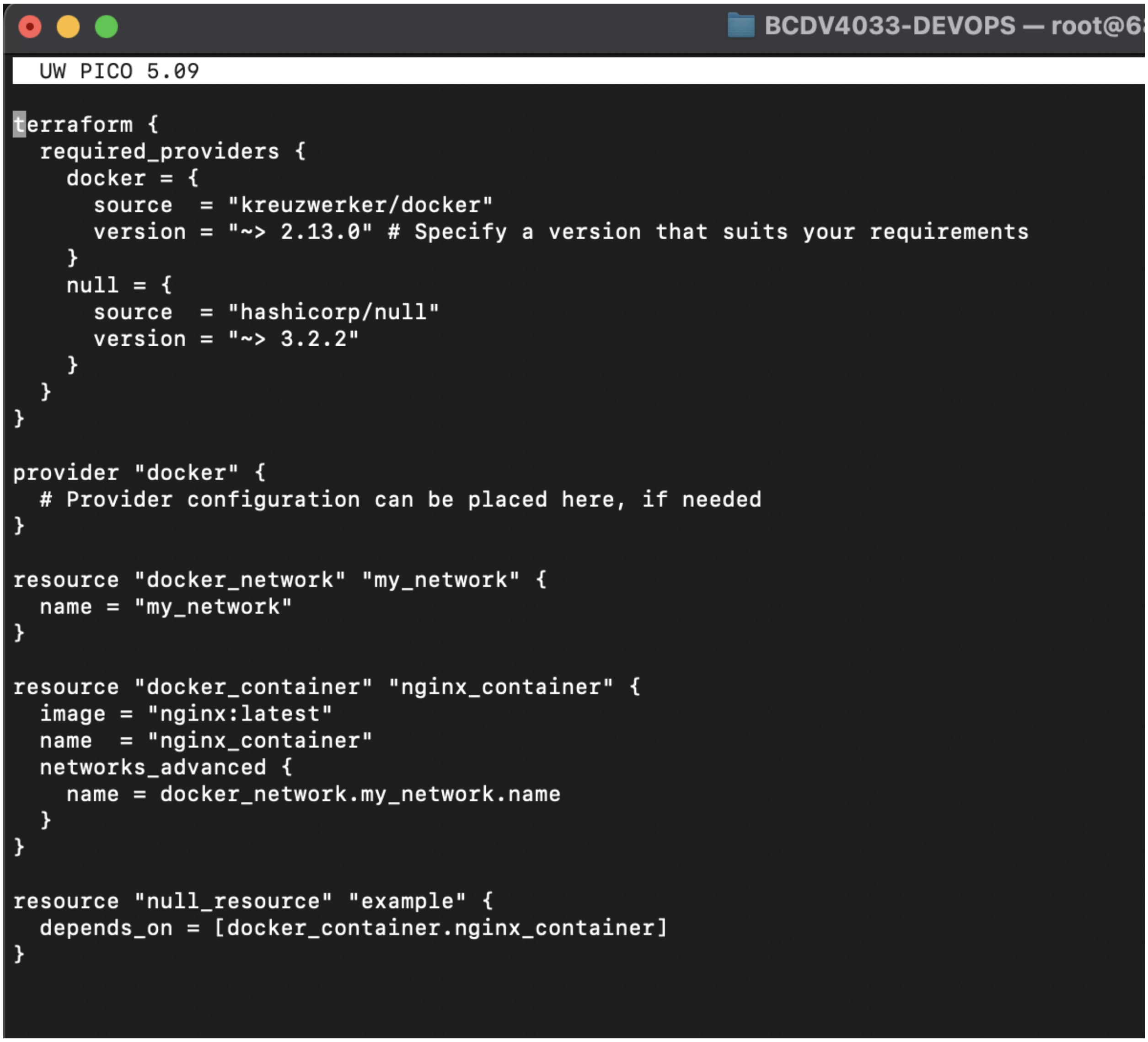
```
      + network_data     = (known after apply)
      + read_only        = false
      + remove_volumes   = true
      + restart          = "no"
      + rm               = false
      + security_opts    = (known after apply)
      + shm_size         = (known after apply)
      + start            = true
      + stdin_open       = false
      + tty              = false

      + networks_advanced {
          + aliases = []
          + name    = "my_network"
        }
    }

  # docker_network.my_network will be created
  + resource "docker_network" "my_network" {
      + driver      = (known after apply)
      + id          = (known after apply)
      + internal    = (known after apply)
      + ipam_driver = "default"
      + name        = "my_network"
      + options     = (known after apply)
      + scope       = (known after apply)
    }

  # null_resource.example will be created
  + resource "null_resource" "example" {
      + id = (known after apply)
    }

Plan: 3 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.
xiaogangdong@Xiaogangs-Air BCDV4033-DEVOPS % 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_container will be created
  + resource "docker_container" "nginx_container" {
      + 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             = "nginx:latest"
      + init              = (known after apply)
      + ip_address        = (known after apply)
      + ip_prefix_length  = (known after apply)
      + ipc_mode          = (known after apply)
      + log_driver        = "json-file"
      + logs              = false
      + must_run          = true
      + name              = "nginx_container"
      + network_data      = (known after apply)
      + read_only         = false
      + remove_volumes    = true
      + restart           = "no"
      + rm                = false
      + security_opts     = (known after apply)
      + shm_size          = (known after apply)
      + start             = true
      + stdin_open        = false
      + tty               = false

      + networks_advanced {
          + aliases = []
          + name    = "my_network"
        }
    }

  # docker_network.my_network will be created
  + resource "docker_network" "my_network" {
      + driver      = (known after apply)
      + id          = (known after apply)
      + internal    = (known after apply)
      + ipam_driver = "default"
      + name        = "my_network"
      + options     = (known after apply)
      + scope       = (known after apply)
    }

  # null_resource.example will be created
  + resource "null_resource" "example" {
      + id = (known after apply)
    }
```



```
  # docker_network.my_network will be created
  + resource "docker_network" "my_network" {
      + driver      = (known after apply)
      + id          = (known after apply)
      + internal    = (known after apply)
      + ipam_driver = "default"
      + name        = "my_network"
      + options     = (known after apply)
      + scope       = (known after apply)
    }

  # null_resource.example will be created
  + resource "null_resource" "example" {
      + id = (known after apply)
    }

Plan: 3 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_network.my_network: Creating...
docker_network.my_network: Creation complete after 2s [id=4bece1d1add3fba65515d13f9cd74ab35354fa834f8a115e0fa72330664eecc8]
docker_container.nginx_container: Creating...
docker_container.nginx_container: Creation complete after 9s [id=655f1f2569c0cbe2a6d42d32ef3f451a9925109ec5178af1415e9b1f2f6c7e21]
null_resource.example: Creating...
null_resource.example: Creation complete after 0s [id=4890063558870314472]

Apply complete! Resources: 3 added, 0 changed, 0 destroyed.
xiaogangdong@Xiaogangs-Air BCDV4033-DEVOPS % 
```



```
  UW PICO 5.09

terraform {
  required_providers {
    docker = {
      source  = "kreuzwerker/docker"
      version = "~> 2.13.0" # Specify a version that suits your requirements
    }
    null = {
      source  = "hashicorp/null"
      version = "~> 3.2.2"
    }
  }
}

provider "docker" {
  # Provider configuration can be placed here, if needed
}

resource "docker_network" "my_network" {
  name = "my_network"
}

resource "docker_container" "nginx_container" {
  image = "nginx:latest"
  name  = "nginx_container"
  networks_advanced {
    name = docker_network.my_network.name
  }
}

resource "null_resource" "example" {
  depends_on = [docker_container.nginx_container]
}
```