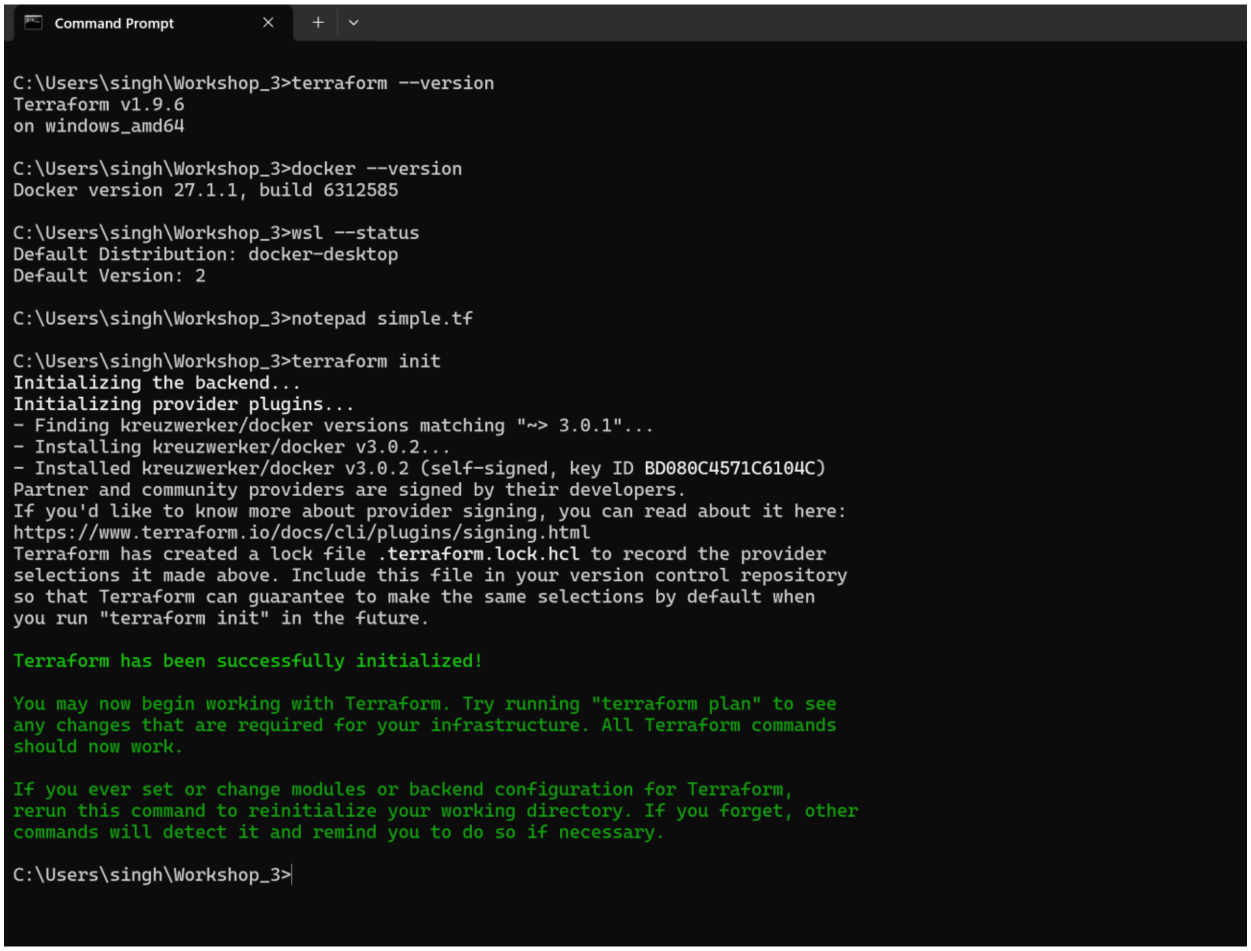
Ankush Singh
WorkShop_3

9/23/2024

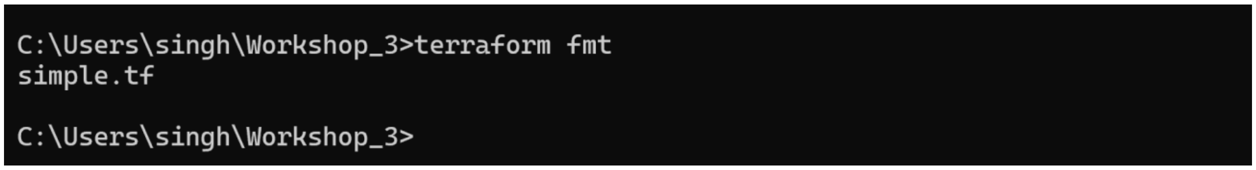
Screenshots_WorkShop_3



Command Prompt

```
C:\Users\singh\Workshop_3>terraform --version
Terraform v1.9.6
on windows_amd64

C:\Users\singh\Workshop_3>docker --version
Docker version 27.1.1, build 6312585

C:\Users\singh\Workshop_3>wsl --status
Default Distribution: docker-desktop
Default Version: 2

C:\Users\singh\Workshop_3>notepad simple.tf

C:\Users\singh\Workshop_3>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.

C:\Users\singh\Workshop_3>
```

Above screenshot showing how user checking terraform and docker tools has been installed. Using notepad to create a simple.tf file. Then performing terraform initializing by using the terraform init.



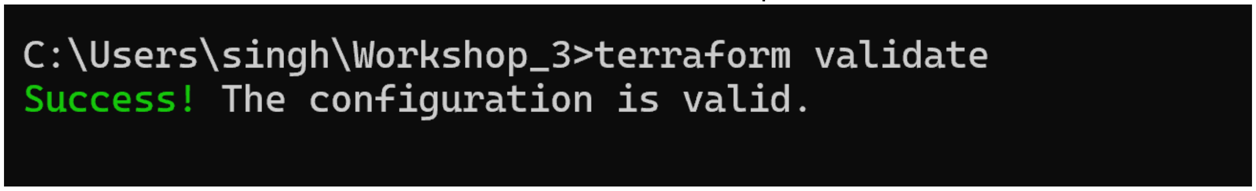
```
C:\Users\singh\Workshop_3>terraform fmt
simple.tf

C:\Users\singh\Workshop_3>
```

terraform fmt command is used to format the terraform configuration (.tf files).

Ankush Singh
WorkShop_3

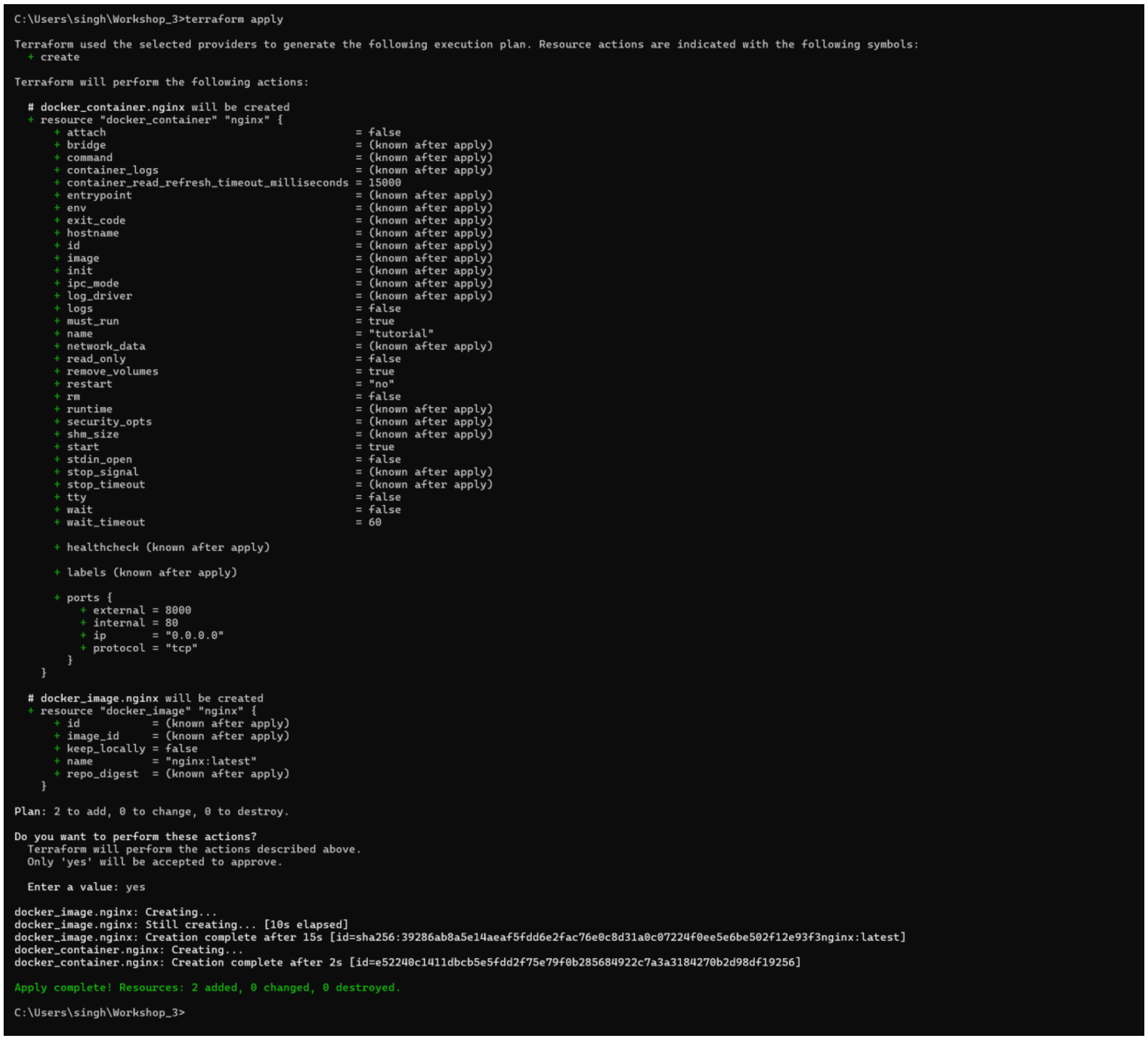
9/23/2024

Screenshots_WorkShop_3



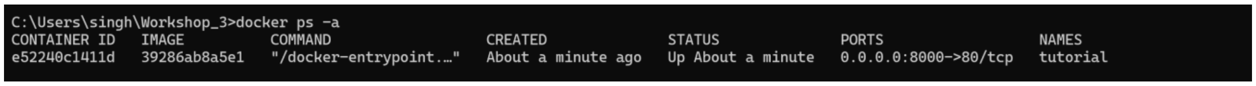
```
C:\Users\singh\Workshop_3>terraform validate
Success! The configuration is valid.
```

Above, we can validate the .tf file by using the terraform validate command.



```
C:\Users\singh\Workshop_3>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

      + healthcheck (known after apply)

      + labels (known after apply)

      + 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:latest"
      + 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: Creation complete after 15s [id=sha256:39286ab8a5e14aeaf5fdd6e2fac76e0c8d31a0c07224f0ee5e6be502f12e93f3nginx:latest]
docker_container.nginx: Creating...
docker_container.nginx: Creation complete after 2s [id=e52240c1411dbcb5e5fdd2f75e79f0b285684922c7a3a3184270b2d98df19256]

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

C:\Users\singh\Workshop_3>
```

Above screenshot, user used terraform apply command to apply the changes required to reach the desired state of the configuration in the .tf file



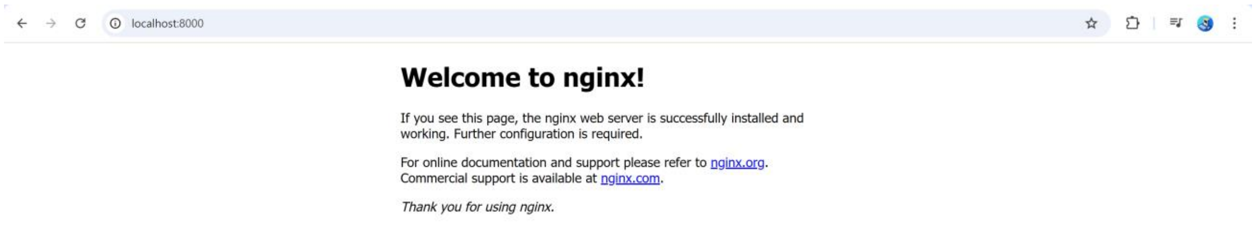
```
C:\Users\singh\Workshop_3>docker ps -a
CONTAINER ID   IMAGE          COMMAND                  CREATED              STATUS              PORTS                  NAMES
e52240c1411d   39286ab8a5e1   "/docker-entrypoint.…"   About a minute ago   Up About a minute   0.0.0.0:8000->80/tcp   tutorial
```

Ankush Singh
WorkShop_3

9/23/2024

Screenshots_WorkShop_3

Above screenshot, user used docker ps -a command to check the container is up and running after terraform apply.



localhost:8000

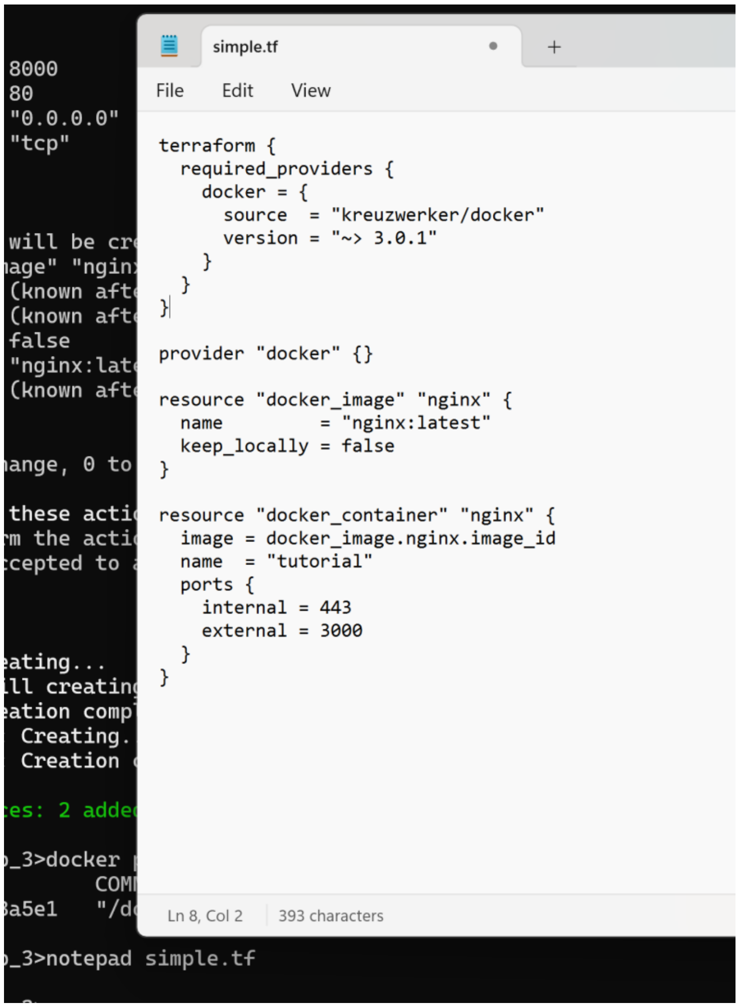
Welcome to nginx!

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.

In the above screenshot we can see that our container is up and running.



```
8000
80
"0.0.0.0"
"tcp"


will be cr
age" "ngin
(known aft
(known aft
false
"nginx:lat
(known aft

ange, 0 to

these acti
m the acti
cepted to


ating...
ll creatin
ation comp
 Creating.
 Creation

es: 2 adde

_3>docker
      COM
a5e1   "/d

_3>notepad simple.tf
```

simple.tf

File Edit View

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

provider "docker" {}

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

resource "docker_container" "nginx" {
  image = docker_image.nginx.image_id
  name  = "tutorial"
  ports {
    internal = 443
    external = 3000
  }
}
```

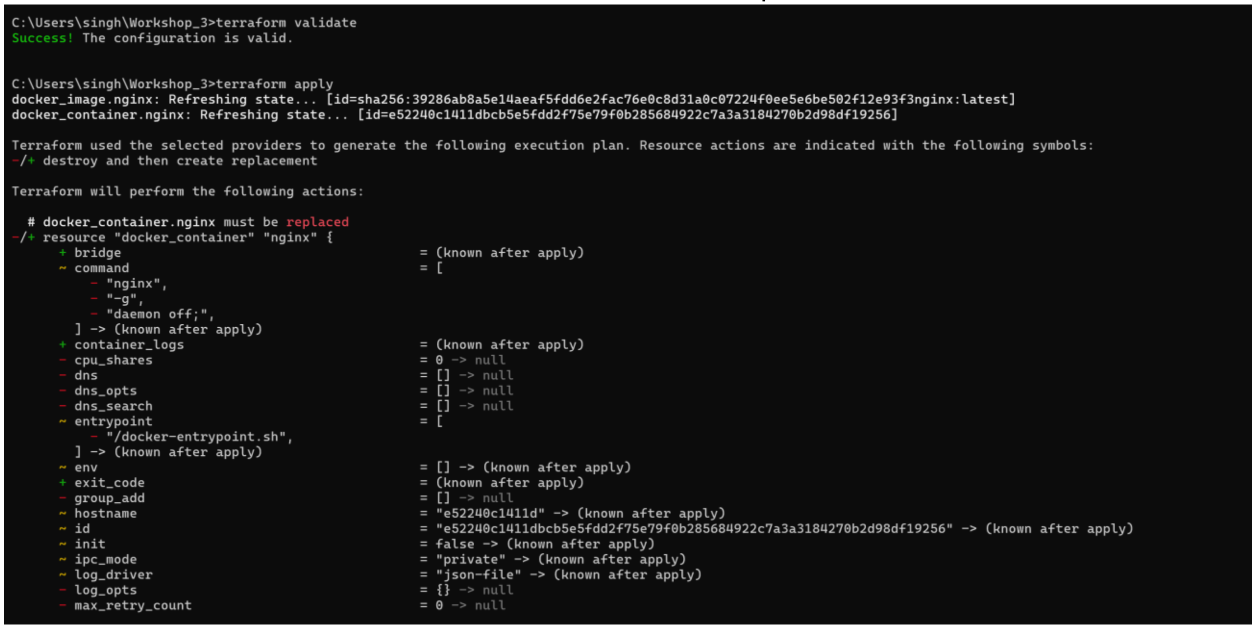
Ln 8, Col 2 | 393 characters

Above screenshot, user opened the simple.tf file using notepad simple.tf command to update the created image with new ports 443 and 3000 for internal and external.

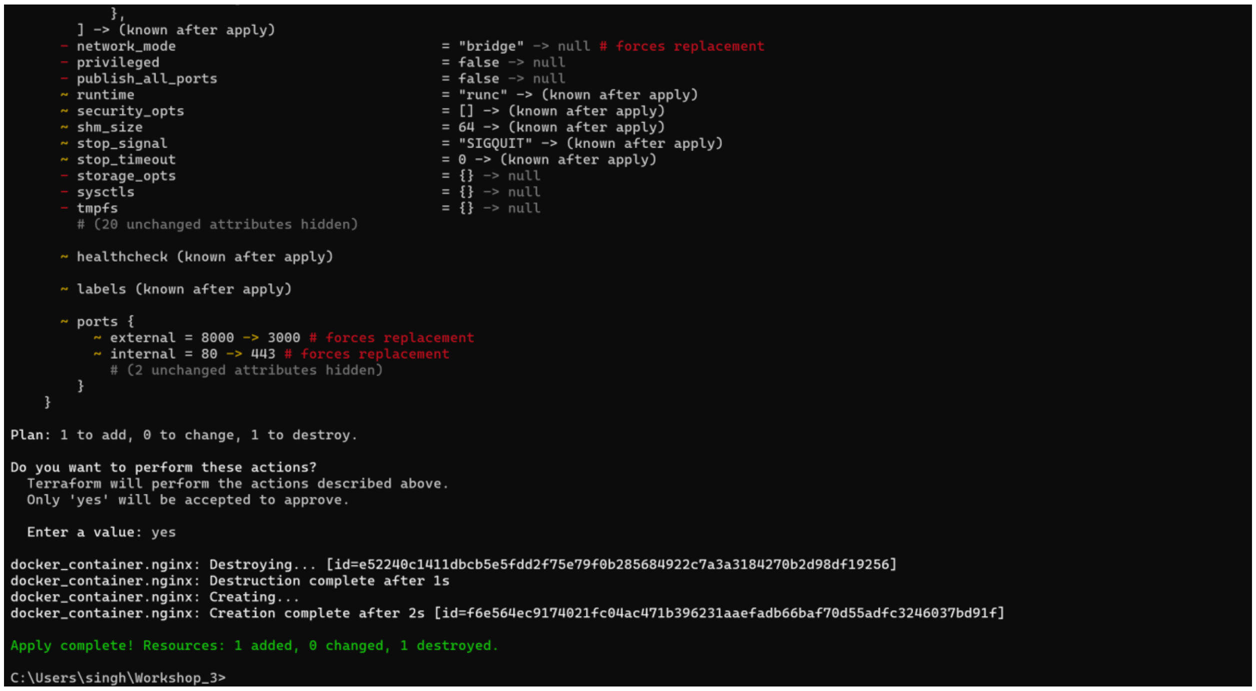
Ankush Singh
WorkShop_3

9/23/2024

Screenshots_WorkShop_3



```
C:\Users\singh\Workshop_3>terraform validate
Success! The configuration is valid.


C:\Users\singh\Workshop_3>terraform apply
docker_image.nginx: Refreshing state... [id=sha256:39286ab8a5e14aeaf5fdd6e2fac76e0c8d31a0c07224f0ee5e6be502f12e93f3nginx:latest]
docker_container.nginx: Refreshing state... [id=e52240c1411dbcb5e5fdd2f75e79f0b285684922c7a3a3184270b2d98df19256]

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" {
      + bridge                                      = (known after apply)
      ~ command                                     = [
          - "nginx",
          - "-g",
          - "daemon off;",
        ] -> (known after apply)
      + container_logs                              = (known after apply)
      - cpu_shares                                  = 0 -> null
      - dns                                         = [] -> null
      - dns_opts                                    = [] -> null
      - dns_search                                  = [] -> null
      ~ entrypoint                                  = [
          - "/docker-entrypoint.sh",
        ] -> (known after apply)
      ~ env                                         = [] -> (known after apply)
      + exit_code                                   = (known after apply)
      - group_add                                   = [] -> null
      ~ hostname                                    = "e52240c1411d" -> (known after apply)
      ~ id                                          = "e52240c1411dbcb5e5fdd2f75e79f0b285684922c7a3a3184270b2d98df19256" -> (known after apply)
      ~ init                                        = false -> (known after apply)
      ~ ipc_mode                                    = "private" -> (known after apply)
      ~ log_driver                                  = "json-file" -> (known after apply)
      - log_opts                                    = {} -> null
      - max_retry_count                             = 0 -> null
```

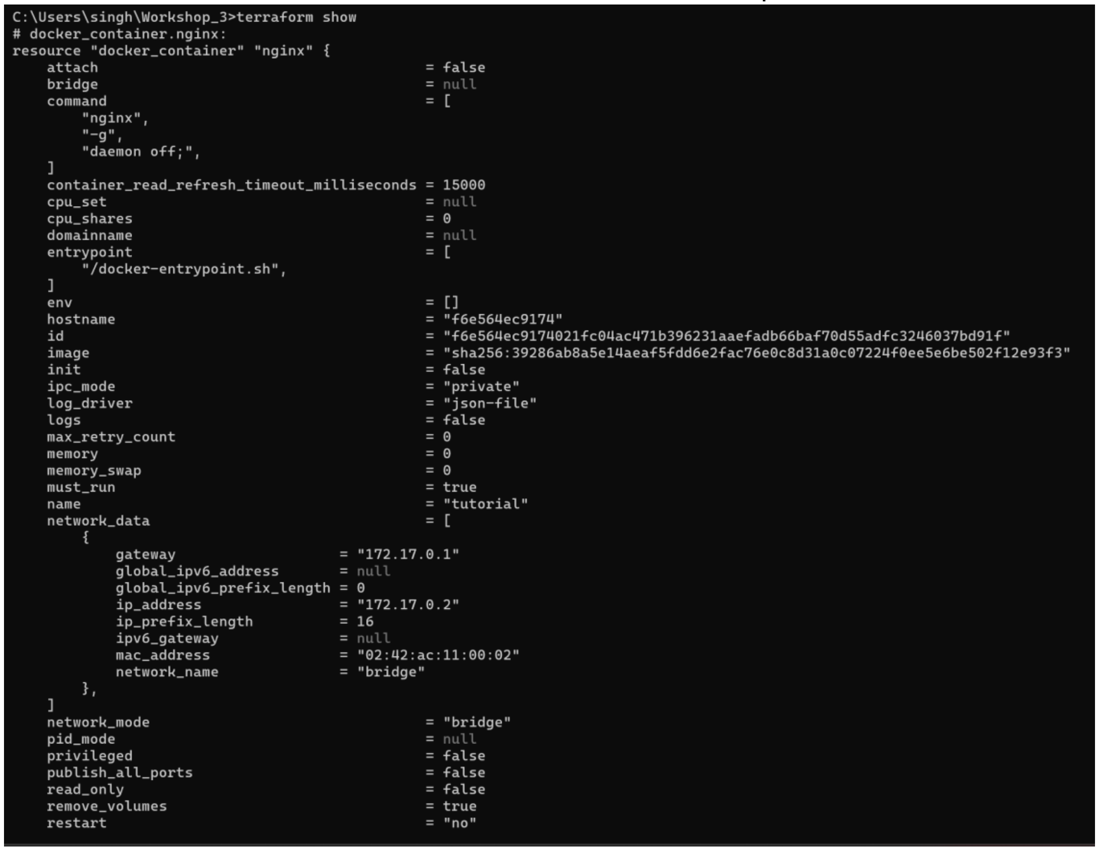


```
          },
        ] -> (known after apply)
      - network_mode                                = "bridge" -> null # forces replacement
      - privileged                                  = false -> null
      - publish_all_ports                           = false -> null
      ~ runtime                                     = "runc" -> (known after apply)
      ~ security_opts                               = [] -> (known after apply)
      ~ shm_size                                    = 64 -> (known after apply)
      ~ stop_signal                                 = "SIGQUIT" -> (known after apply)
      ~ stop_timeout                                = 0 -> (known after apply)
      - storage_opts                                = {} -> null
      - sysctls                                     = {} -> null
      - tmpfs                                       = {} -> null
        # (20 unchanged attributes hidden)

      ~ healthcheck (known after apply)

      ~ labels (known after apply)

      ~ ports {
          ~ external = 8000 -> 3000 # forces replacement
          ~ internal = 80 -> 443 # forces replacement
            # (2 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=e52240c1411dbcb5e5fdd2f75e79f0b285684922c7a3a3184270b2d98df19256]
docker_container.nginx: Destruction complete after 1s
docker_container.nginx: Creating...
docker_container.nginx: Creation complete after 2s [id=f6e564ec9174021fc04ac471b396231aaefadb66baf70d55adfc3246037bd91f]

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

C:\Users\singh\Workshop_3>
```

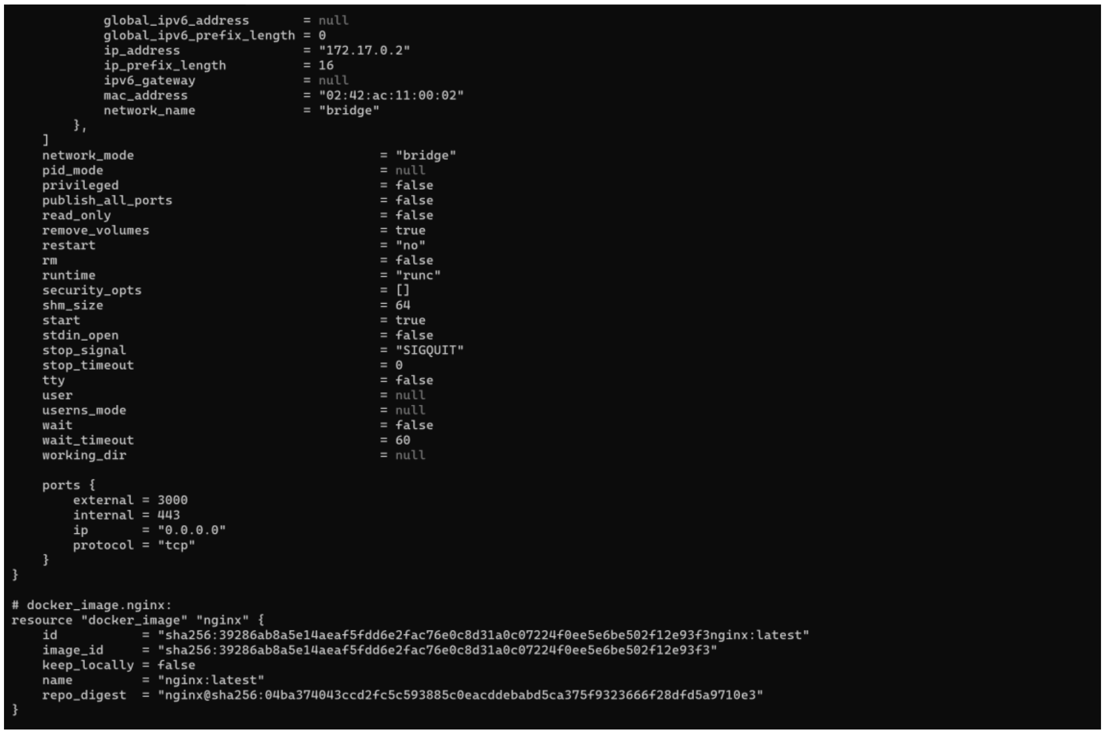
Ankush Singh
WorkShop_3

9/23/2024

Screenshots_WorkShop_3



```
C:\Users\singh\Workshop_3>terraform show
# docker_container.nginx:
resource "docker_container" "nginx" {
    attach                                      = false
    bridge                                      = null
    command                                     = [
        "nginx",
        "-g",
        "daemon off;",
    ]
    container_read_refresh_timeout_milliseconds = 15000
    cpu_set                                     = null
    cpu_shares                                  = 0
    domainname                                  = null
    entrypoint                                  = [
        "/docker-entrypoint.sh",
    ]
    env                                         = []
    hostname                                    = "f6e564ec9174"
    id                                          = "f6e564ec9174021fc04ac471b396231aaefadb66baf70d55adfc3246037bd91f"
    image                                       = "sha256:39286ab8a5e14aeaf5fdd6e2fac76e0c8d31a0c07224f0ee5e6be502f12e93f3"
    init                                        = false
    ipc_mode                                    = "private"
    log_driver                                  = "json-file"
    logs                                        = false
    max_retry_count                             = 0
    memory                                      = 0
    memory_swap                                 = 0
    must_run                                    = true
    name                                        = "tutorial"
    network_data                                = [
        {
            gateway                   = "172.17.0.1"
            global_ipv6_address       = null
            global_ipv6_prefix_length = 0
            ip_address                = "172.17.0.2"
            ip_prefix_length          = 16
            ipv6_gateway              = null
            mac_address               = "02:42:ac:11:00:02"
            network_name              = "bridge"
        },
    ]
    network_mode                                = "bridge"
    pid_mode                                    = null
    privileged                                  = false
    publish_all_ports                           = false
    read_only                                   = false
    remove_volumes                              = true
    restart                                     = "no"
```

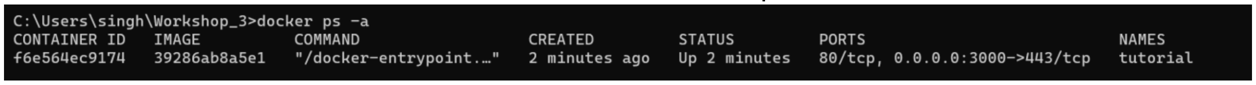


```
            global_ipv6_address       = null
            global_ipv6_prefix_length = 0
            ip_address                = "172.17.0.2"
            ip_prefix_length          = 16
            ipv6_gateway              = null
            mac_address               = "02:42:ac:11:00:02"
            network_name              = "bridge"
        },
    ]
    network_mode                                = "bridge"
    pid_mode                                    = null
    privileged                                  = false
    publish_all_ports                           = false
    read_only                                   = false
    remove_volumes                              = true
    restart                                     = "no"
    rm                                          = false
    runtime                                     = "runc"
    security_opts                               = []
    shm_size                                    = 64
    start                                       = true
    stdin_open                                  = false
    stop_signal                                 = "SIGQUIT"
    stop_timeout                                = 0
    tty                                         = false
    user                                        = null
    userns_mode                                 = null
    wait                                        = false
    wait_timeout                                = 60
    working_dir                                 = null

    ports {
        external = 3000
        internal = 443
        ip       = "0.0.0.0"
        protocol = "tcp"
    }
}

# docker_image.nginx:
resource "docker_image" "nginx" {
    id           = "sha256:39286ab8a5e14aeaf5fdd6e2fac76e0c8d31a0c07224f0ee5e6be502f12e93f3nginx:latest"
    image_id     = "sha256:39286ab8a5e14aeaf5fdd6e2fac76e0c8d31a0c07224f0ee5e6be502f12e93f3"
    keep_locally = false
    name         = "nginx:latest"
    repo_digest  = "nginx@sha256:04ba374043ccd2fc5c593885c0eacddebabd5ca375f9323666f28dfd5a9710e3"
}
```

Ankush Singh
WorkShop_3

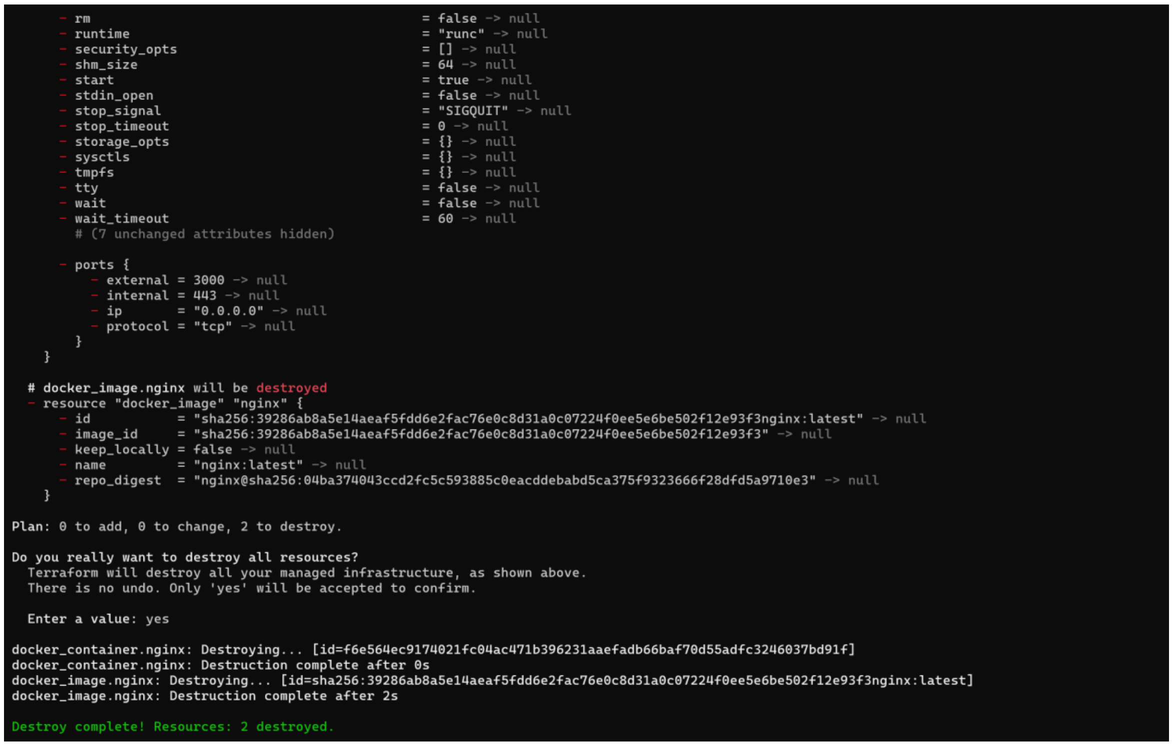
9/23/2024

Screenshots_WorkShop_3

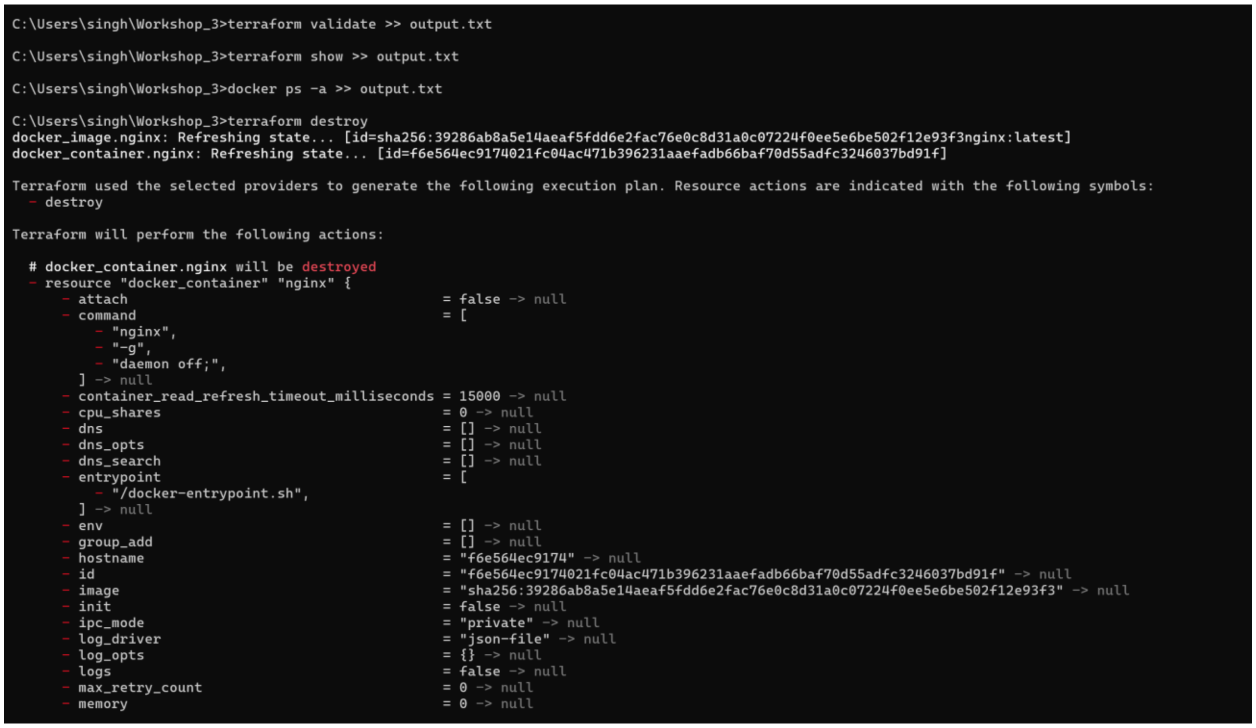


```
C:\Users\singh\Workshop_3>docker ps -a
CONTAINER ID   IMAGE          COMMAND                  CREATED         STATUS         PORTS                              NAMES
f6e564ec9174   39286ab8a5e1   "/docker-entrypoint.…"   2 minutes ago   Up 2 minutes   80/tcp, 0.0.0.0:3000->443/tcp      tutorial
```

Above screenshot, confirmed the container 3000 ->443.



```
      - rm                                          = false -> null
      - runtime                                     = "runc" -> null
      - security_opts                               = [] -> null
      - shm_size                                    = 64 -> null
      - start                                       = true -> null
      - stdin_open                                  = false -> null
      - stop_signal                                 = "SIGQUIT" -> null
      - stop_timeout                                = 0 -> null
      - storage_opts                                = {} -> null
      - sysctls                                     = {} -> null
      - tmpfs                                       = {} -> null
      - tty                                         = false -> null
      - wait                                        = false -> null
      - wait_timeout                                = 60 -> null
        # (7 unchanged attributes hidden)

      - ports {
          - external = 3000 -> null
          - internal = 443 -> null
          - ip       = "0.0.0.0" -> null
          - protocol = "tcp" -> null
        }
    }

  # docker_image.nginx will be destroyed
  - resource "docker_image" "nginx" {
      - id           = "sha256:39286ab8a5e14aeaf5fdd6e2fac76e0c8d31a0c07224f0ee5e6be502f12e93f3nginx:latest" -> null
      - image_id     = "sha256:39286ab8a5e14aeaf5fdd6e2fac76e0c8d31a0c07224f0ee5e6be502f12e93f3" -> null
      - keep_locally = false -> null
      - name         = "nginx:latest" -> null
      - repo_digest  = "nginx@sha256:04ba374043ccd2fc5c593885c0eacddebabd5ca375f9323666f28dfd5a9710e3" -> 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=f6e564ec9174021fc04ac471b396231aaefadb66baf70d55adfc3246037bd91f]
docker_container.nginx: Destruction complete after 0s
docker_image.nginx: Destroying... [id=sha256:39286ab8a5e14aeaf5fdd6e2fac76e0c8d31a0c07224f0ee5e6be502f12e93f3nginx:latest]
docker_image.nginx: Destruction complete after 2s

Destroy complete! Resources: 2 destroyed.
```

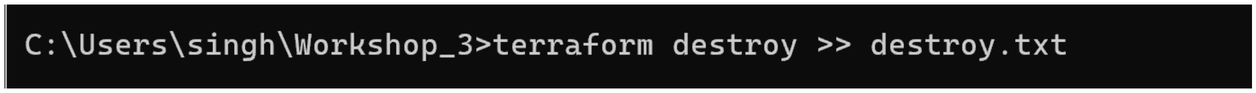


```
C:\Users\singh\Workshop_3>terraform validate >> output.txt

C:\Users\singh\Workshop_3>terraform show >> output.txt

C:\Users\singh\Workshop_3>docker ps -a >> output.txt

C:\Users\singh\Workshop_3>terraform destroy
docker_image.nginx: Refreshing state... [id=sha256:39286ab8a5e14aeaf5fdd6e2fac76e0c8d31a0c07224f0ee5e6be502f12e93f3nginx:latest]
docker_container.nginx: Refreshing state... [id=f6e564ec9174021fc04ac471b396231aaefadb66baf70d55adfc3246037bd91f]

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
      - container_read_refresh_timeout_milliseconds = 15000 -> null
      - cpu_shares                                  = 0 -> null
      - dns                                         = [] -> null
      - dns_opts                                    = [] -> null
      - dns_search                                  = [] -> null
      - entrypoint                                  = [
          - "/docker-entrypoint.sh",
        ] -> null
      - env                                         = [] -> null
      - group_add                                   = [] -> null
      - hostname                                    = "f6e564ec9174" -> null
      - id                                          = "f6e564ec9174021fc04ac471b396231aaefadb66baf70d55adfc3246037bd91f" -> null
      - image                                       = "sha256:39286ab8a5e14aeaf5fdd6e2fac76e0c8d31a0c07224f0ee5e6be502f12e93f3" -> null
      - init                                        = false -> null
      - ipc_mode                                    = "private" -> null
      - log_driver                                  = "json-file" -> null
      - log_opts                                    = {} -> null
      - logs                                        = false -> null
      - max_retry_count                             = 0 -> null
      - memory                                      = 0 -> null
```

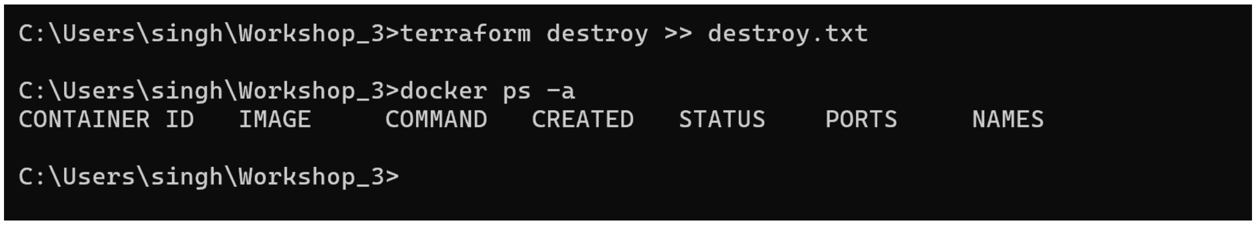
Ankush Singh
WorkShop_3

9/23/2024

Screenshots_WorkShop_3



```
C:\Users\singh\Workshop_3>terraform destroy >> destroy.txt
```



```
C:\Users\singh\Workshop_3>terraform destroy >> destroy.txt

C:\Users\singh\Workshop_3>docker ps -a
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES

C:\Users\singh\Workshop_3>
```

Above screenshots, after modifying the port numbers user just reapplied the configuration where as user used terraform validate, terraform apply, terraform show, docker ps -a. Confirmed the updated container. After, user exported the outputs to text files using terraform validate >> output.txt, terraform show >> output.txt, docker ps -a >> output.txt. User used the terraform destroy command to destroy the created infrastructure. After that exported the output using terraform destroy >> destroy.txt command.