

Name : Devansh Wadhwani

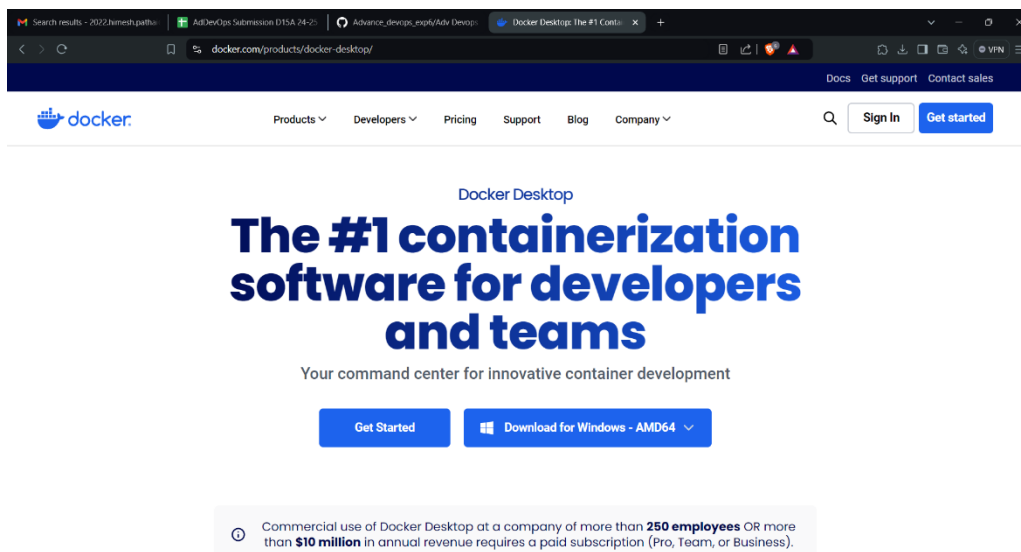
Roll No. : 64

Div. : D15A

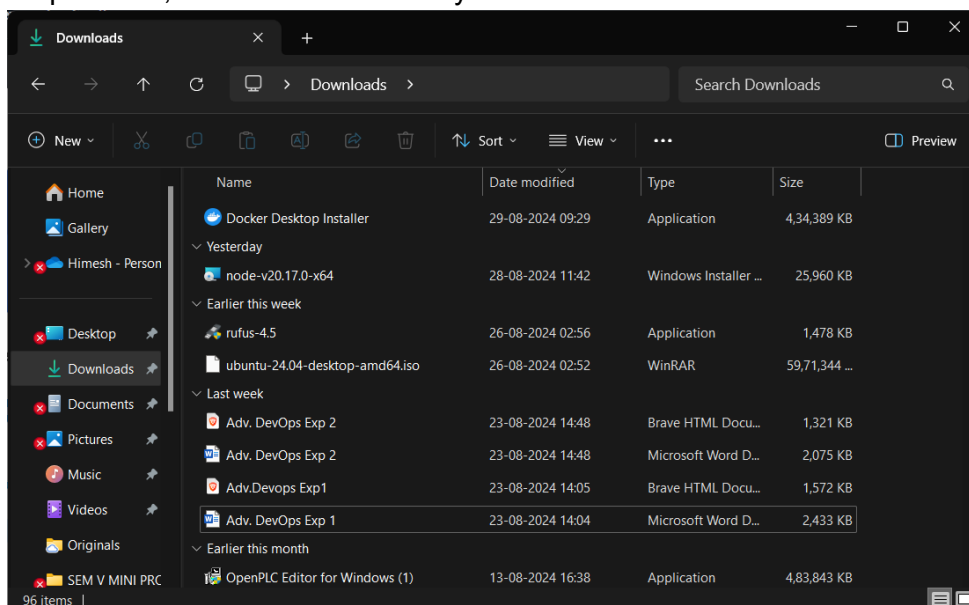
Advance DevOps Exp : 6

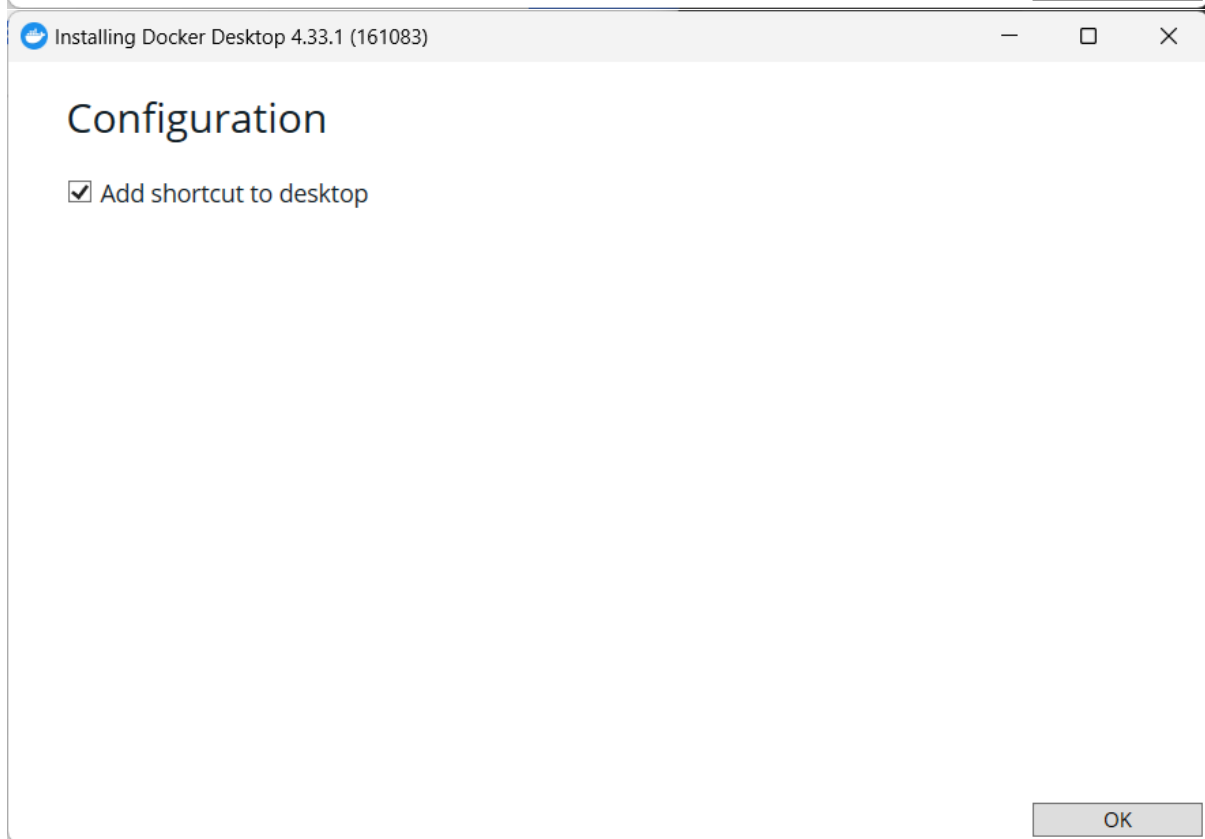
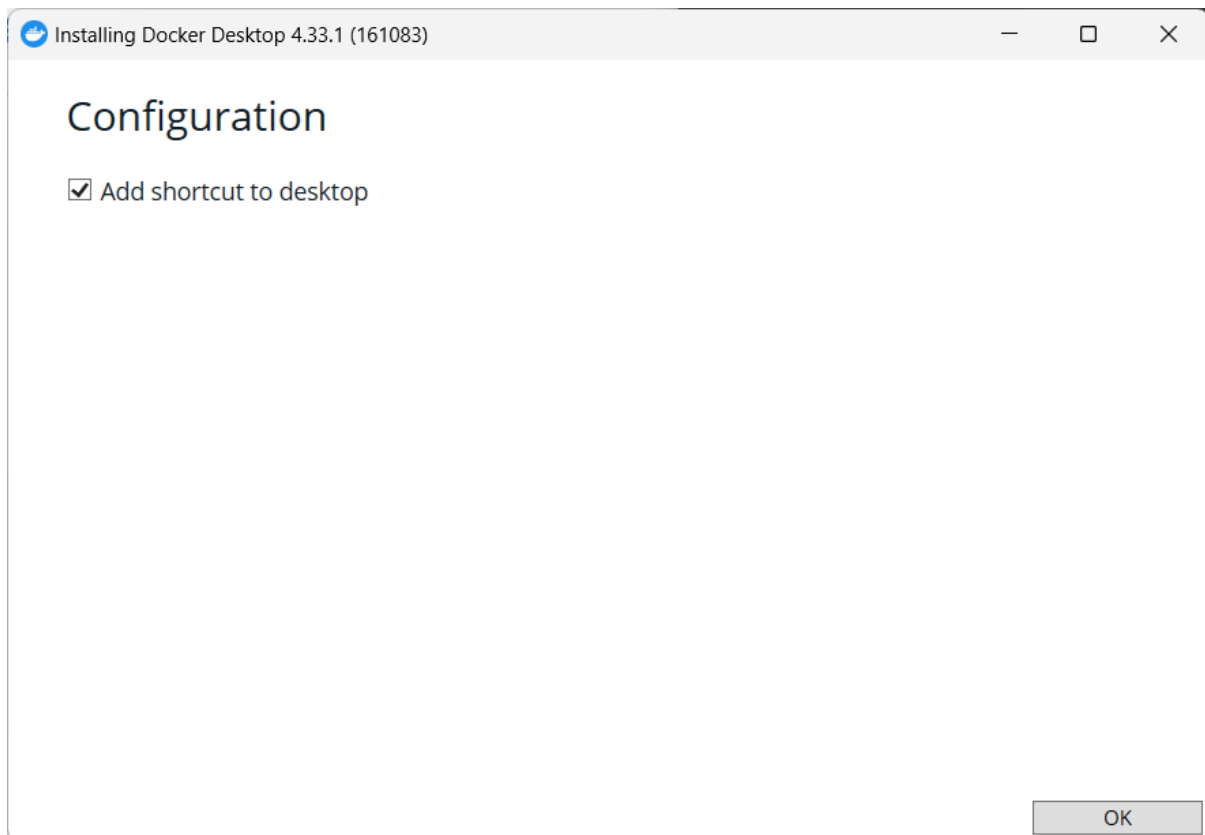
Aim: To Build, change, and destroy AWS / GCP /Microsoft Azure/ DigitalOcean infrastructure Using Terraform. (S3 bucket or Docker)

Step 1: Download Docker form www.docker.com

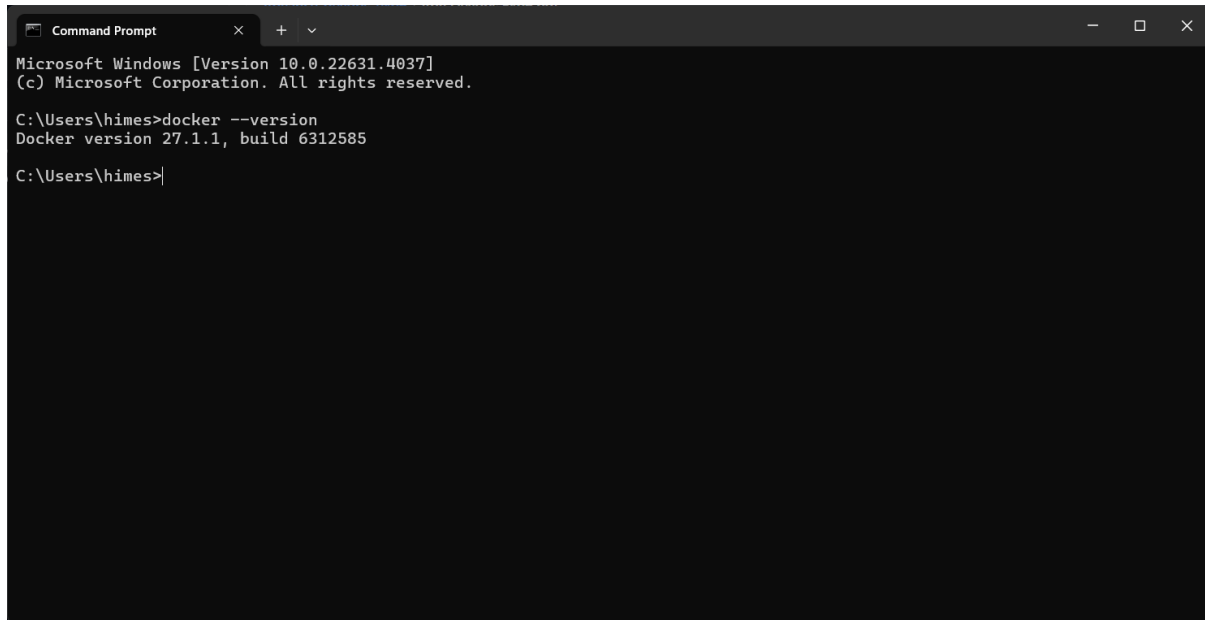


Step 2: Now, Docker is successfully downloaded.





Step 3: Open Command Prompt and enter the command `docker --version`, to check whether the docker is successfully installed.

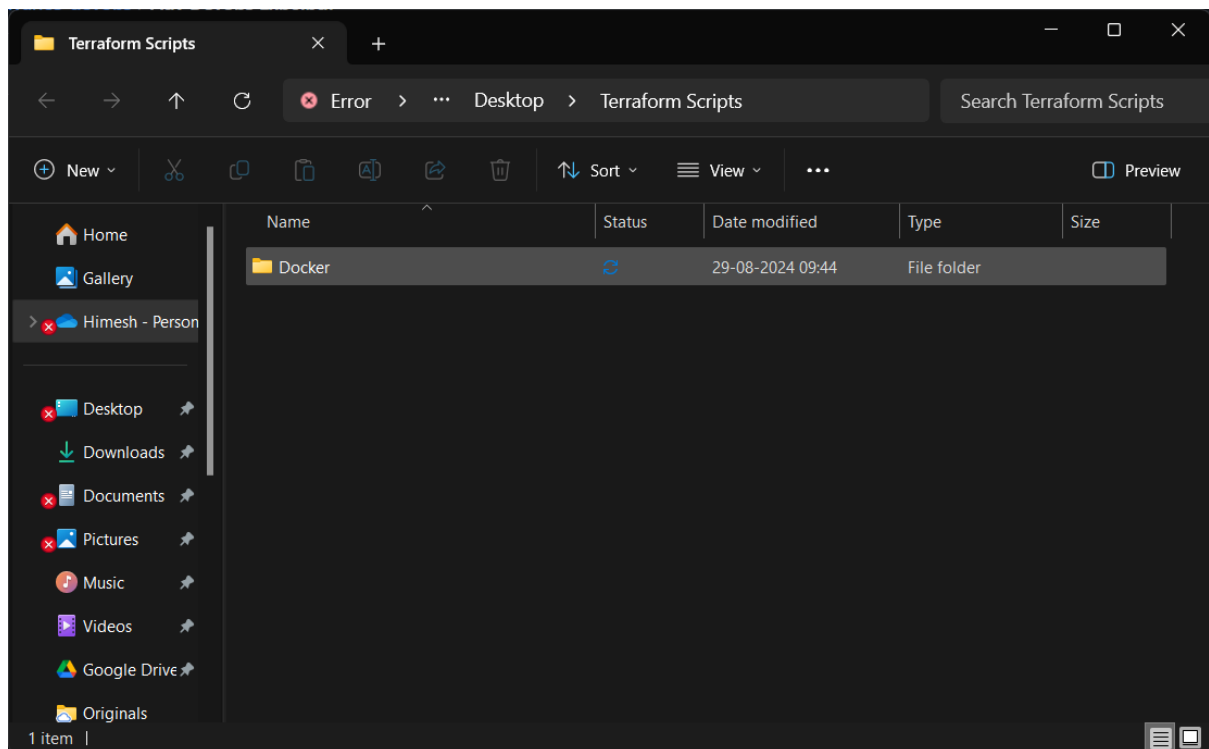


```
Microsoft Windows [Version 10.0.22631.4037]
(c) Microsoft Corporation. All rights reserved.

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

C:\Users\himes>
```

Step 4: Create a folder `Terraform_scripts` and inside it create a folder named `Docker`.



Step 5: create a new folder named 'Terraform' in the 'TerraformScripts' folder. Then create a new terraform_script.tf file using vs code.

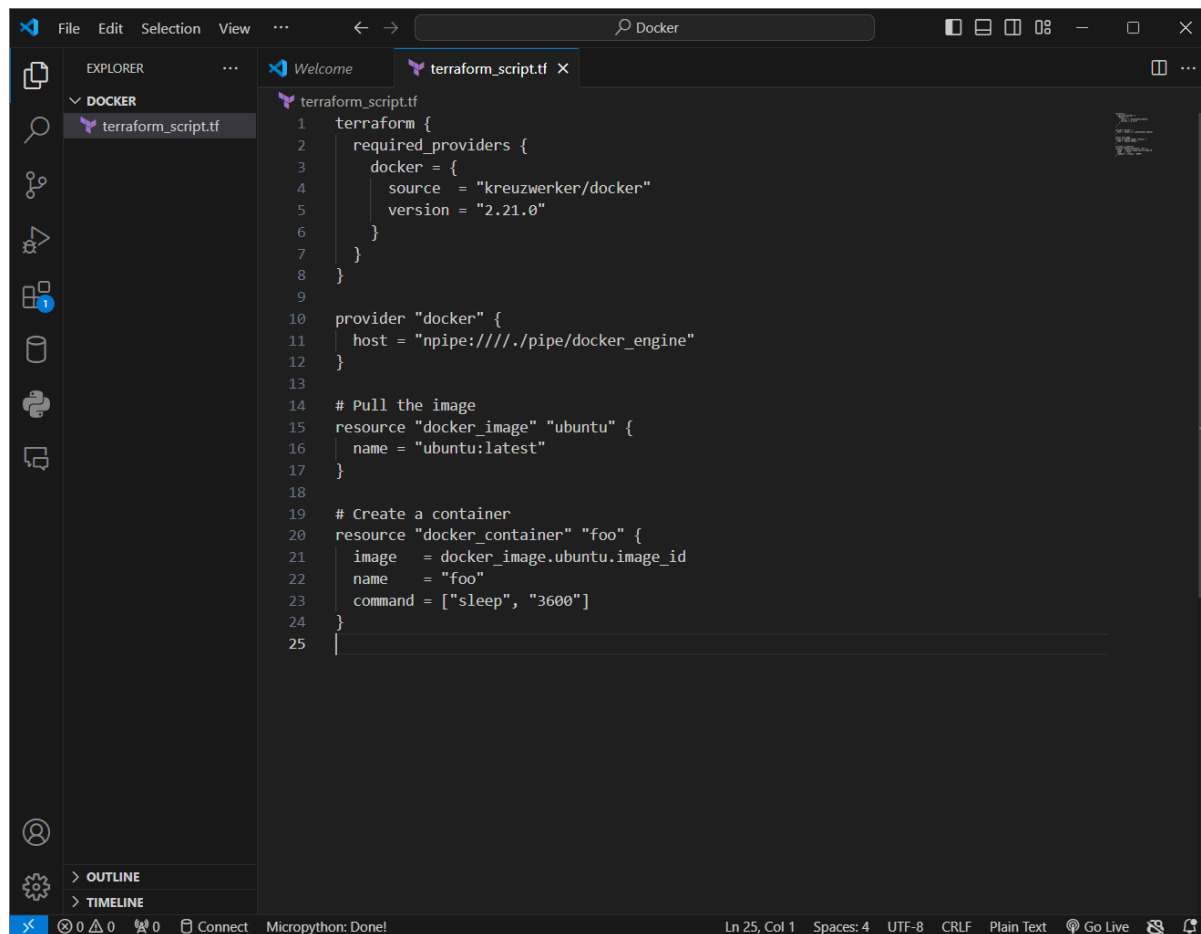
Run the following script in the VS Code.

```
terraform {
  required_providers {
    docker = {
      source = "kreuzwerker/docker"
      version = "2.21.0"
    }
  }
}

provider "docker" {
  host = "npipe:////./pipe/docker_engine"
}

# Pull the image
resource "docker_image" "ubuntu" {
  name = "ubuntu:latest"
}

# Create a container resource
"docker_container" "foo" { image =
docker_image.ubuntu.image_id
  name = "foo"
  command = ["sleep", "3600"]
}
```



Step 6: Open Windows Explorer and run the following command terraform init, terraform plan, terraform apply, terraform destroy, terraform provider, terraform validate, terraform state list and docker images.

```
Command Prompt
C:\Users\devansh\Drive\Desktop\TerraformScripts>cd docker
C:\Users\himes\OneDrive\Desktop\TerraformScripts\DOCKER>terraform init
Initializing the backend...
Initializing provider plugins...
- Finding kreuzwerker/docker versions matching "2.21.0"...
- Installing kreuzwerker/docker v2.21.0...
- Installed kreuzwerker/docker v2.21.0 (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\devansh\Drive\Desktop\TerraformScripts\DOCKER>
```

```
Command Prompt
C:\Users\himes\OneDrive\Desktop\TerraformScripts\Docker>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.foo will be created
+ resource "docker_container" "foo" {
+   attach      = false
+   bridge      = (known after apply)
+   command     = [
+     "sleep",
+     "3600",
+   ]
+   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    = (known after apply)
}
```

```
Command Prompt
C:\Users\himes\OneDrive\Desktop\TerraformScripts\Docker>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.foo will be created
+ resource "docker_container" "foo" {
+   attach      = false
+   bridge      = (known after apply)
+   command     = [
+     "sleep",
+     "3600",
+   ]
+   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    = (known after apply)
+   logs         = false
}
```

```
Command Prompt
+ 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

+ healthcheck (known after apply)

+ labels (known after apply)
}

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

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

```
Command Prompt
+ 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

+ healthcheck (known after apply)

+ labels (known after apply)
}

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

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

```
Command Prompt
}

# docker_image.ubuntu will be created
+ resource "docker_image" "ubuntu" {
  + id          = (known after apply)
  + image_id    = (known after apply)
  + latest      = (known after apply)
  + name        = "ubuntu:latest"
  + output      = (known after apply)
  + 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.ubuntu: Creating...
docker_image.ubuntu: Still creating... [10s elapsed]
docker_image.ubuntu: Creation complete after 15s [id=sha256:edbfe74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598aubuntu:latest]
docker_container.foo: Creating...
docker_container.foo: Creation complete after 1s [id=a5c5d23b0cb0e6c71d8b5ecdc395a48bc3fdaf608fc3058df946209f0886952b]

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

C:\Users\devansh\Drive\Desktop\TerraformScripts\Docker>
```

```
Command Prompt

C:\Users\himes\OneDrive\Desktop\TerraformScripts\Docker>terraform providers

Providers required by configuration:
└─ provider[registry.terraform.io/kreuzwerker/docker] 2.21.0

Providers required by state:
    provider[registry.terraform.io/kreuzwerker/docker]

C:\Users\devansh\Drive\Desktop\TerraformScripts\Docker>
```



```
C:\Users\devansh\Drive\Desktop\TerraformScripts\Docker>terraform validate
Success! The configuration is valid.
```

```
C:\Users\devansh\Drive\Desktop\TerraformScripts\Docker>
```

```
C:\Users\devansh\Drive\Desktop\TerraformScripts\Docker>terraform state list
docker_container.foo
docker_image.ubuntu
```

```
C:\Users\devansh\Drive\Desktop\TerraformScripts\Docker>SS|
```

```
C:\Users\devansh\Drive\Desktop\TerraformScripts>docker images
REPOSITORY    TAG       IMAGE ID       CREATED        SIZE
ubuntu        latest    edbfe74c41f8   3 weeks ago    78.1MB
```

```
C:\Users\devansh\Drive\Desktop\TerraformScripts>S|
```