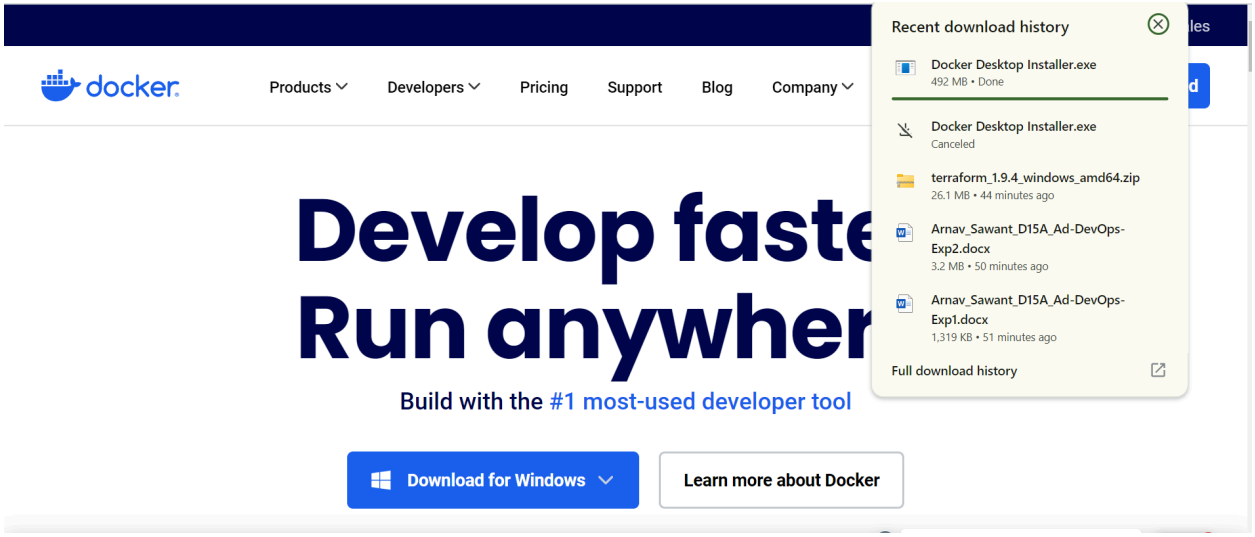
**RIYA VARYANI D15A 64**

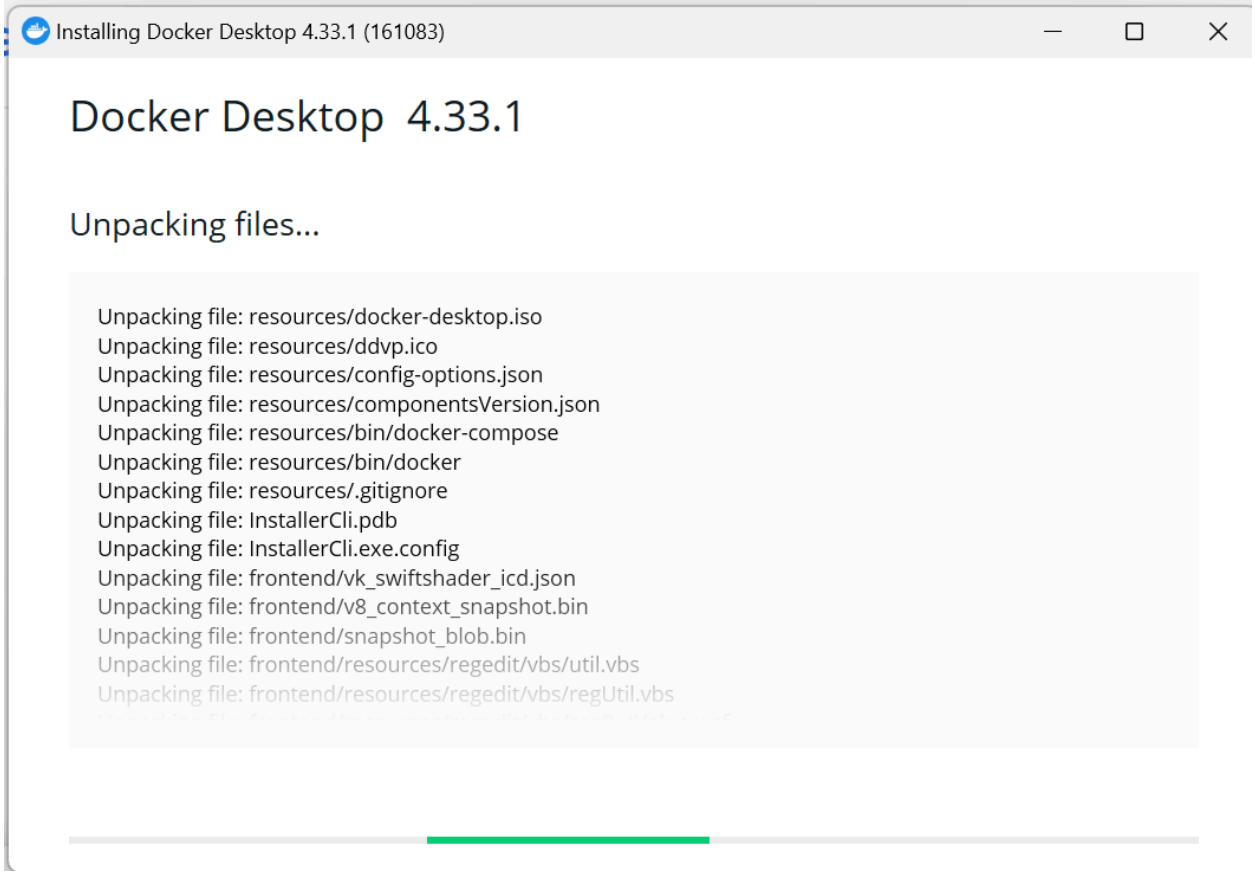
# ADVANCE DEVOPS EXPERIMENT NO 6

A. Creating docker image using terraform

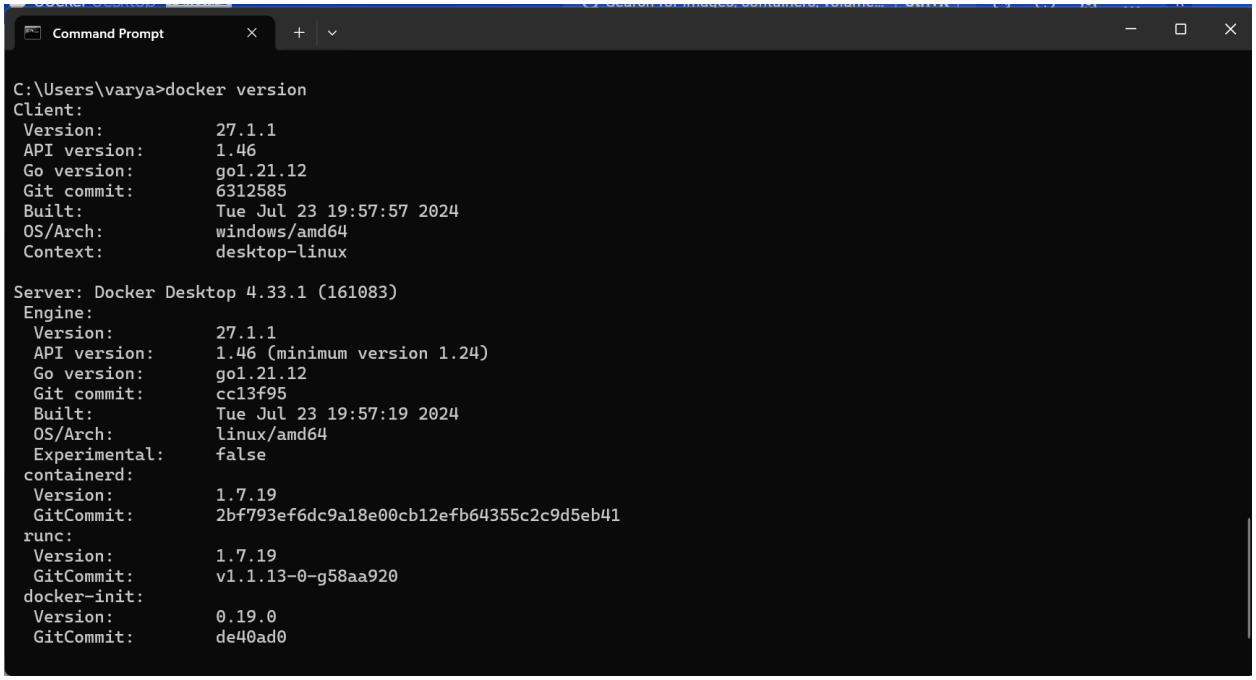
1) Download and Install Docker Desktop from https://www.docker.com/

Step 1: Check the docker functionality



**RIYA VARYANI D15A 64**



```
C:\Users\varya>docker version
Client:
 Version:           27.1.1
 API version:       1.46
 Go version:        go1.21.12
 Git commit:        6312585
 Built:             Tue Jul 23 19:57:57 2024
 OS/Arch:           windows/amd64
 Context:           desktop-linux

Server: Docker Desktop 4.33.1 (161083)
 Engine:
  Version:          27.1.1
  API version:      1.46 (minimum version 1.24)
  Go version:       go1.21.12
  Git commit:       cc13f95
  Built:            Tue Jul 23 19:57:19 2024
  OS/Arch:          linux/amd64
  Experimental:     false
 containerd:
  Version:          1.7.19
  GitCommit:        2bf793ef6dc9a18e00cb12efb64355c2c9d5eb41
 runc:
  Version:          1.7.19
  GitCommit:        v1.1.13-0-g58aa920
 docker-init:
  Version:          0.19.0
  GitCommit:        de40ad0
```

**Now, create a folder named 'Terraform Scripts' in which we save our different types of**
**scripts which will be further used in this experiment.**

**Step 2:** Firstly create a new folder named 'Docker' in the 'TerraformScripts' folder. Then
create a new dteocker.tf file using Atom editor and write the followingcontents into it to
create a Ubuntu Linux container.

Script:

```
terraform

{ required_providers
{docker = {
source = "kreuzwerker/docker"
version = "2.21.0"
}
}
}
provider "docker" {
host = "npipe:////.//pipe//docker_engine"
}
```

**RIYA VARYANI D15A 64**

# Pulls the image
resource "docker_image" "ubuntu"
{name = "ubuntu:latest"
}
# Create a container
resource "docker_container" "foo"
{ image =
docker_image.ubuntu.image_idname =
"foo"
}

docker.tf

File Edit View

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

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

# Pulls the Ubuntu image from Docker Hub
resource "docker_image" "ubuntu" {
  name = "ubuntu:latest"
}

# Creates and runs a Docker container with the Ubuntu image
resource "docker_container" "foo" {
  image   = docker_image.ubuntu.image_id
  name    = "foo"
  command = ["sleep", "infinity"]  # Keeps the container running indefinitely
}
```

**Step 3**: Execute Terraform Init command to initialize the resources.

**RIYA VARYANI D15A 64**

```
C:\Users\varya>cd C:\Users\varya\TerraformScripts\Docker

C:\Users\varya\TerraformScripts\Docker>terraform init
Initializing the backend...
Initializing provider plugins...
- Reusing previous version of kreuzwerker/docker from the depend
ency lock file
- Using previously-installed kreuzwerker/docker v2.21.0

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 Terra
form commands
should now work.

If you ever set or change modules or backend configuration for T
erraform,
rerun this command to reinitialize your working directory. If yo
u forget, other
commands will detect it and remind you to do so if necessary.
```

**Step 4**: Execute Terraform plan to see the available resources

```
C:\Users\varya\TerraformScripts\Docker>terraform plan
docker_image.ubuntu: Refreshing state... [id=sha256:edbfe74c41f8
a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598aubuntu:lates
t]

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",
          + "infinity",
        ]
      + 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)
```

**RIYA VARYANI D15A 64**

**Step 5**: Execute Terraform apply to apply the configuration, which will automatically
create and run the Ubuntu Linux container based on our configuration.
Using command :
“terraform apply”

```
C:\Users\varya\TerraformScripts\Docker>terraform apply
docker_image.ubuntu: Refreshing state... [id=sha256:edbfe74c41f8
a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598aubuntu:lates
t]

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",
          + "infinity",
        ]
```

Docker images, Before Executing Apply step:

```
C:\Users\varya\TerraformScripts\Docker>docker images
REPOSITORY   TAG       IMAGE ID       CREATED       SIZE
ubuntu       latest    edbfe74c41f8   2 weeks ago   78.1MB
```

**Step 6**: Execute Terraform destroy to delete the configuration, which will automatically
delete the Ubuntu Container.

**RIYA VARYANI D15A 64**

```
C:\Users\varya\TerraformScripts\Docker>terraform destroy
docker_image.ubuntu: Refreshing state... [id=sha256:edbfe74c41f8
a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598aubuntu:lates
t]
docker_container.foo: Refreshing state... [id=1dcd9c4a8bbccf62fe
a18099efddb1af6fdcf0a85deab004c503949cb7d16554]

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.foo will be destroyed
  - resource "docker_container" "foo" {
```

Docker images After Executing Destroy step

```
C:\Users\varya\TerraformScripts\Docker>docker images
REPOSITORY   TAG       IMAGE ID   CREATED   SIZE
```