

# AdvDevOps Lab 6

Sanket More  
D15A 30

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

## Part A: Creating docker image using terraform

1. Install docker

```
C:\Users\Sanket More>docker
```

```
Usage:  docker [OPTIONS] COMMAND
```

```
A self-sufficient runtime for containers
```

### Common Commands:

run	Create and run a new container from an image
exec	Execute a command in a running container
ps	List containers
build	Build an image from a Dockerfile
pull	Download an image from a registry
push	Upload an image to a registry
images	List images
login	Log in to a registry
logout	Log out from a registry
search	Search Docker Hub for images
version	Show the Docker version information
info	Display system-wide information

### Management Commands:

builder	Manage builds
buildx*	Docker Buildx
checkpoint	Manage checkpoints
compose*	Docker Compose
container	Manage containers
context	Manage contexts

```
C:\Users\Sanket More>docker --version  
Docker version 27.0.3, build 7d4bcd8
```

2. Create a new folder Docker, inside it, create a file docker.tf

```
docker.tf  X
docker.tf > ...
1  terraform {
2    required_providers {
3      docker = {
4        source = "kreuzwerker/docker"
5        version = "2.21.0"
6      }
7    }
8  }
9
10 provider "docker" {
11   host = "npipe:////./pipe/docker_engine"
12 }
13
14 # Pull the Docker image
15 resource "docker_image" "ubuntu" {
16   name = "ubuntu:latest"
17 }
18
19 # Create a Docker container
20 resource "docker_container" "foo" {
21   image = docker_image.ubuntu.image_id
22   name  = "foo"
23 }
24 |
```

### 3. Terraform init

```
C:\Users\siddi\OneDrive\Desktop\lab-works\terraform_scripts>cd Docker

C:\Users\siddi\OneDrive\Desktop\lab-works\terraform_scripts\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.
```

### 4. Terraform plan

```
C:\Users\siddi\OneDrive\Desktop\lab-works\terraform_scripts\Docker>terraform plan

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with
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          = (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       = (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.

---

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.

## 5. Check docker images before applying

```

C:\Users\siddi\OneDrive\Desktop\lab-works\terraform_scripts\Docker>docker images
REPOSITORY    TAG       IMAGE ID       CREATED        SIZE
react-img     latest    619c9b7a9ac5   2 weeks ago    320MB

C:\Users\siddi\OneDrive\Desktop\lab-works\terraform_scripts\Docker>|

```

## 6. Terraform apply

```

C:\Users\siddi\OneDrive\Desktop\lab-works\terraform_scripts\Docker>terraform apply
docker_image.ubuntu: Refreshing state... [id=sha256:edbf74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598aubuntu: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.foo will be created
+ resource "docker_container" "foo" {
+   attach      = false
+   bridge      = (known after apply)
+   command     = [
+     "tail",
+     "-f",
+     "/dev/null",
+   ]
+   container_logs = (known after apply)
+   entrypoint   = (known after apply)
}

```

```

+ logs           = false
+ must_run       = true
+ name           = "foo"
+ 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

+ healthcheck (known after apply)

+ labels (known after apply)
}

Plan: 1 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_container.foo: Creating...
docker_container.foo: Creation complete after 1s [id=af0512641b95dfce26fa5f29deafb8a8d56bd8b9878a246f46bd694e961e5b5]

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

C:\Users\siddi\OneDrive\Desktop\lab-works\terraform_scripts\Docker>

```

## 7. Docker images after apply

```

C:\Users\siddi\OneDrive\Desktop\lab-works\terraform_scripts\Docker>docker images
REPOSITORY    TAG       IMAGE ID       CREATED        SIZE
react-img     latest    619c9b7a9ac5   2 weeks ago    320MB
ubuntu        latest    edbfe74c41f8   3 weeks ago    78.1MB

```

## 8. Terraform destroy

```
C:\Users\siddi\OneDrive\Desktop\lab-works\terraform_scripts\Docker>terraform destroy
docker_image.ubuntu: Refreshing state... [id=sha256:edbfe74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598aubuntu:latest]
docker_container.foo: Refreshing state... [id=af0512641b95dfce26fa5f29deafb8a8d56bd8b9878a246f46bd694e961e5b5]

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" {
  - attach      = false -> null
  - command     = [
    - "tail",
    - "-f",
    - "/dev/null",
  ] -> null
  - cpu_shares  = 0 -> null
  - dns         = [] -> null
  - dns_opts    = [] -> null
  - dns_search  = [] -> null
  - entrypoint  = [] -> null
  - env         = [] -> null
  - gateway     = "172.17.0.1" -> null
  - group_add   = [] -> null
  - hostname    = "af0512641b95" -> null
  - id          = "af0512641b95dfce26fa5f29deafb8a8d56bd8b9878a246f46bd694e961e5b5" -> null
  - image       = "sha256:edbfe74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598a" -> null
  - init        = false -> null
  - ip_address  = "172.17.0.2" -> null
  - ip_prefix_length = 16 -> null
  - ipc_mode    = "private" -> null
  - links       = [] -> null
  - log_driver  = "json-file" -> null
}
```

```
# docker_image.ubuntu will be destroyed
- resource "docker_image" "ubuntu" {
  - id          = "sha256:edbfe74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598aubuntu:latest" -> null
  - image_id    = "sha256:edbfe74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598a" -> null
  - latest      = "sha256:edbfe74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598a" -> null
  - name        = "ubuntu:latest" -> null
  - repo_digest = "ubuntu@sha256:8a37d68f4f73ebf3d4efafbcf66379bf3728902a8038616808f04e34a9ab63ee" -> 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.foo: Destroying... [id=af0512641b95dfce26fa5f29deafb8a8d56bd8b9878a246f46bd694e961e5b5]
docker_container.foo: Destruction complete after 1s
docker_image.ubuntu: Destroying... [id=sha256:edbfe74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598aubuntu:latest]
docker_image.ubuntu: Destruction complete after 0s

Destroy complete! Resources: 2 destroyed.

C:\Users\siddi\OneDrive\Desktop\lab-works\terraform_scripts\Docker>
```

## 9. Docker images after apply

```
C:\Users\siddi\OneDrive\Desktop\lab-works\terraform_scripts\Docker>docker images
REPOSITORY    TAG       IMAGE ID       CREATED        SIZE
react-img     latest    619c9b7a9ac5   2 weeks ago   320MB

C:\Users\siddi\OneDrive\Desktop\lab-works\terraform_scripts\Docker>
```

## Part B: Terraform and S3 -

Step 1: Create access keys and secret key for IAM user



AWS account.

☒ **Application running on an AWS compute service**  
You plan to use this access key to enable application code running on an AWS compute service like Amazon EC2, Amazon ECS, or AWS Lambda to access your AWS account.

☐ **Third-party service**  
You plan to use this access key to enable access for a third-party application or service that monitors or manages your AWS resources.

☐ **Application running outside AWS**  
You plan to use this access key to authenticate workloads running in your data center or other infrastructure outside of AWS that needs to access your AWS resources.

☐ **Other**  
Your use case is not listed here.

 **Alternative recommended**  
Assign an IAM role to compute resources like EC2 instances or Lambda functions to automatically supply temporary credentials to enable access. [Learn more](#) 

Step 2 : Type below code in main.tf in editor for aws and terraform connection and environment creation .

Code -

```
terraform {  
  required_providers {  
    aws = {  
      source = "hashicorp/aws"  
      version = "~> 5.0"  
    }  
  }  
}  
  
# Configure the AWS Provider  
provider "aws" {  
  region = "us-east-1"
```



```
access_key = ""

secret_key = ""

}

resource "aws_s3_bucket" "bucket" {

bucket = "bucket-pranav-123"

tags = {

Name = "My bucket"

}

}
```

```
s3 > main.tf
1 terraform {
2   required_providers {
3     aws = {
4       source  = "hashicorp/aws"
5       version = "~> 5.0"
6     }
7   }
8 }
9
10 # Configure the AWS Provider
11 provider "aws" {
12   region = "us-east-1"
13   access_key = ""
14   secret_key = ""
15 }
16
17
18
19 resource "aws_s3_bucket" "bucket" {
20   bucket = "bucket-pranav-123"
21
22   tags = {
23     Name = "My bucket"
24   }
25 }
26 }
```



Step 3 : Type terraform init command in powershell.

```
(base) PS C:\Users\sbpol\Documents\terraform_scripts\docker\s3> terraform init
Initializing the backend...
Initializing provider plugins...
- Finding hashicorp/aws versions matching "~> 5.0"...
- Installing hashicorp/aws v5.63.1...
- Installed hashicorp/aws v5.63.1 (signed by HashiCorp)
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.
(base) PS C:\Users\sbpol\Documents\terraform_scripts\docker\s3>
```

Step 4 : Type terraform plan command in powershell.

```
(base) PS C:\Users\sbpol\Documents\terraform_scripts\docker\s3> 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:

# aws_s3_bucket.terr will be created
+ resource "aws_s3_bucket" "terr" {
+   acceleration_status      = (known after apply)
+   acl                      = (known after apply)
+   arn                     = (known after apply)
+   bucket                  = "my-tf-test-bucket"
+   bucket_domain_name      = (known after apply)
+   bucket_prefix           = (known after apply)
+   bucket_regional_domain_name = (known after apply)
+   force_destroy           = false
+   hosted_zone_id          = (known after apply)
+   id                     = (known after apply)
+   object_lock_enabled     = (known after apply)
+   policy                  = (known after apply)
+   region                 = (known after apply)
+   request_payer           = (known after apply)
+   tags                    = {
+     "Environment" = "Dev"
+     "Name"        = "My bucket"
+   }
+   tags_all              = {
+     "Environment" = "Dev"
+     "Name"        = "My bucket"
+   }
+   website_domain        = (known after apply)
+   website_endpoint      = (known after apply)

+ cors_rule (known after apply)

+ grant (known after apply)

+ lifecycle_rule (known after apply)

+ logging (known after apply)
```

Step 5 : Type terraform apply command in powershell.

```
}
+ tags_all = {
+   + "Name" = "My bucket"
+ }
+ website_domain = (known after apply)
+ website_endpoint = (known after apply)
+ cors_rule (known after apply)
+ grant (known after apply)
+ lifecycle_rule (known after apply)
+ logging (known after apply)
+ object_lock_configuration (known after apply)
+ replication_configuration (known after apply)
+ server_side_encryption_configuration (known after apply)
+ versioning (known after apply)
+ website (known after apply)
}

Plan: 1 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

aws_s3_bucket.bucket: Creating...
aws_s3_bucket.bucket: Creation complete after 5s [id=bucket-pranav-123]

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

```
}
+ tags_all = {
+   + "Name" = "My bucket"
+ }
+ website_domain = (known after apply)
+ website_endpoint = (known after apply)
+ cors_rule (known after apply)
+ grant (known after apply)
+ lifecycle_rule (known after apply)
+ logging (known after apply)
+ object_lock_configuration (known after apply)
+ replication_configuration (known after apply)
+ server_side_encryption_configuration (known after apply)
+ versioning (known after apply)
+ website (known after apply)
}

Plan: 1 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

aws_s3_bucket.bucket: Creating...
aws_s3_bucket.bucket: Creation complete after 5s [id=bucket-pranav-123]

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

Step 6 : AWS s3 before and after the bucket creation using terraform.

BEFORE -

General purpose buckets (3) <span>Info</span> <span>All AWS Regions</span>					
Buckets are containers for data stored in S3.					
<input type="text" value="Find buckets by name"/>					
<span>&lt; 1 &gt;</span> <span>⚙</span>					
	Name ▲	AWS Region ▼	IAM Access Analyzer	Creation date ▼	
<input type="radio"/>	<a href="#">codepipeline-eu-north-1-21903055499</a>	Europe (Stockholm) eu-north-1	<a href="#">View analyzer for eu-north-1</a>	August 21, 2024, 16:26:15 (UTC+05:30)	
<input type="radio"/>	<a href="#">elasticbeanstalk-eu-north-1-869935102438</a>	Europe (Stockholm) eu-north-1	<a href="#">View analyzer for eu-north-1</a>	August 15, 2024, 16:27:46 (UTC+05:30)	

AFTER -

General purpose buckets (3) <span>Info</span> <span>All AWS Regions</span>					
Buckets are containers for data stored in S3.					
<input type="text" value="Find buckets by name"/>					
<span>&lt; 1 &gt;</span> <span>⚙</span>					
	Name ▲	AWS Region ▼	IAM Access Analyzer	Creation date ▼	
<input type="radio"/>	<a href="#">codepipeline-eu-north-1-21903055499</a>	Europe (Stockholm) eu-north-1	<a href="#">View analyzer for eu-north-1</a>	August 21, 2024, 16:26:15 (UTC+05:30)	
<input type="radio"/>	<a href="#">elasticbeanstalk-eu-north-1-869935102438</a>	Europe (Stockholm) eu-north-1	<a href="#">View analyzer for eu-north-1</a>	August 15, 2024, 16:27:46 (UTC+05:30)	
<input type="radio"/>	<a href="#">sanket-bucket1</a>	Europe (Stockholm) eu-north-1	<a href="#">View analyzer for eu-north-1</a>	August 19, 2024, 17:22:35 (UTC+05:30)	

Step 7 : Upload file to the bucket using terraform .

CODE -

```
terraform {  
  
  required_providers {  
  
    aws = {  
  
      source = "hashicorp/aws"  
  
      version = "~> 5.0" } } }  
  
# Configure the AWS Provider  
  
provider "aws" {  
  
  region = "us-east-1"  
  
  access_key = ""  
  
  secret_key = "" }  
resource "aws_s3_bucket" "bucket" {  
  
  bucket = "bucket-pranav-123"  
  
  tags = {  
  
    Name = "My bucket" } }
```

```
resource "aws_s3_bucket_object" "file" {

bucket = aws_s3_bucket.bucket.id

key = "hello.txt"

source = "C:/Users/sbp01/Documents/terraform_scripts/docker/s3/hello.txt"}
```

```
resource "aws_s3_bucket" "bucket" {
  bucket = "bucket-pranav-123"

  tags = {
    Name = "My bucket"
  }
}

resource "aws_s3_bucket_object" "file" {
  bucket = aws_s3_bucket.bucket.id
  key    = "hello.txt"
  source = "C:/Users/sbp01/Documents/terraform_scripts/docker/s3/hello.txt"
}
```

Step 8 : Terraform plan and apply command to apply the changes for file .

```
(base) PS C:\Users\sbp01\Documents\terraform_scripts\docker\s3> terraform plan
aws_s3_bucket.bucket: Refreshing state... [id=bucket-pranav-123]

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following
+ create

Terraform will perform the following actions:

# aws_s3_bucket_object.file will be created
+ resource "aws_s3_bucket_object" "file" {
  + acl                = "private"
  + arn                = (known after apply)
  + bucket             = "bucket-pranav-123"
  + bucket_key_enabled = (known after apply)
  + content_type       = (known after apply)
  + etag               = (known after apply)
  + force_destroy      = false
  + id                 = (known after apply)
  + key                = "hello.txt"
  + kms_key_id         = (known after apply)
  + server_side_encryption = (known after apply)
  + source              = "C:/Users/sbp01/Documents/terraform_scripts/docker/s3/hello.txt"
  + storage_class       = (known after apply)
  + tags_all           = (known after apply)
  + version_id         = (known after apply)
}

Plan: 1 to add, 0 to change, 0 to destroy.

Warning: Deprecated Resource

  with aws_s3_bucket_object.file,
  on main.tf line 28, in resource "aws_s3_bucket_object" "file":
  28: resource "aws_s3_bucket_object" "file" {

use the aws_s3_object resource instead

(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 run "terraform apply" now.  
(base) PS C:\Users\sbpol\Documents\terraform\_scripts\docker\s3> terraform apply  
aws\_s3\_bucket.bucket: Refreshing state... [id=bucket-pranav-123]

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:

```
# aws_s3_bucket_object.file will be created
+ resource "aws_s3_bucket_object" "file" {
  + acl                = "private"
  + arn                = (known after apply)
  + bucket             = "bucket-pranav-123"
  + bucket_key_enabled = (known after apply)
  + content_type       = (known after apply)
  + etag               = (known after apply)
  + force_destroy      = false
  + id                 = (known after apply)
  + key                = "hello.txt"
  + kms_key_id         = (known after apply)
  + server_side_encryption = (known after apply)
  + source              = "C:/Users/sbpol/Documents/terraform_scripts/docker/s3/hello.txt"
  + storage_class       = (known after apply)
  + tags_all           = (known after apply)
  + version_id         = (known after apply)
}
```

Plan: 1 to add, 0 to change, 0 to destroy.

Warning: Deprecated Resource

with aws\_s3\_bucket\_object.file,  
on main.tf line 28, in resource "aws\_s3\_bucket\_object" "file":  
28: resource "aws\_s3\_bucket\_object" "file" {

use the aws\_s3\_object resource instead

(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

aws\_s3\_bucket\_object.file: Creating...

aws\_s3\_bucket\_object.file: Creation complete after 1s [id=hello.txt]

Warning: Deprecated Resource

with aws\_s3\_bucket\_object.file,  
on main.tf line 28, in resource "aws\_s3\_bucket\_object" "file":  
28: resource "aws\_s3\_bucket\_object" "file" {

use the aws\_s3\_object resource instead

Warning: Argument is deprecated

with aws\_s3\_bucket\_object.file,  
on main.tf line 29, in resource "aws\_s3\_bucket\_object" "file":  
29: bucket = aws\_s3\_bucket.bucket.id

Use the aws\_s3\_object resource instead

(and one more similar warning elsewhere)

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

(base) PS C:\Users\sbpol\Documents\terraform\_scripts\docker\s3>

Step 9 : s3 bucket before and after execution of upload

sanket-bucket1

Info

Objects

Properties

Permissions

Metrics

Management

Access Points

Objects (2)

Info

Copy S3 URI

Copy URL

Download

Open

Delete

Actions



Create folder

Upload

Objects are the fundamental entities stored in Amazon S3. You can use [Amazon S3 inventory](#) to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. [Learn more](#)

Find objects by prefix

< 1 > ⚙

<input type="checkbox"/>	Name	Type	Last modified	Size	Storage class
<input type="checkbox"/>	 <a href="#">index21.html</a>	html	August 19, 2024, 17:24:08 (UTC+05:30)	3.8 KB	Standard
<input type="checkbox"/>	 <a href="#">pic1.jpg</a>	jpg	August 19, 2024, 17:24:09 (UTC+05:30)	44.0 KB	Standard

Step 10 : Terraform destroy command to destroy the s3 bucket.

```
(base) PS C:\Users\sbp01\Documents\terraform_scripts\docker\s3> terraform destroy
aws_s3_bucket.bucket: Refreshing state... [id=bucket-pranav-123]
aws_s3_bucket_object.file: Refreshing state... [id=hello.txt]

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:

# aws_s3_bucket.bucket will be destroyed
- resource "aws_s3_bucket" "bucket" {
  - arn                = "arn:aws:s3:::bucket-pranav-123" -> null
  - bucket             = "bucket-pranav-123" -> null
  - bucket_domain_name = "bucket-pranav-123.s3.amazonaws.com" -> null
  - bucket_regional_domain_name = "bucket-pranav-123.s3.us-east-1.amazonaws.com" -> null
  - force_destroy      = false -> null
  - hosted_zone_id     = "Z3AQBSTGFYJSTF" -> null
  - id                 = "bucket-pranav-123" -> null
  - object_lock_enabled = false -> null
  - region             = "us-east-1" -> null
  - request_payer      = "BucketOwner" -> null
  - tags               = {
    - "Name" = "My bucket"
  } -> null
  - tags_all           = {
    - "Name" = "My bucket"
  } -> null
  # (3 unchanged attributes hidden)

- grant {
  - id          = "10def03d73e09d8adda11bfe68e632f70a83a37758b74ea6e933dafd0250c850" -> null
  - permissions = [
    - "FULL_CONTROL",
  ] -> null
  - type        = "CanonicalUser" -> null
  # (1 unchanged attribute hidden)
}

- server_side_encryption_configuration {
```

```
    }  
  }  
  - versioning {  
    - enabled = false -> null  
    - mfa_delete = false -> null  
  }  
}
```

Plan: 0 to add, 0 to change, 1 to destroy.

**Warning: Deprecated Resource**

with aws\_s3\_bucket\_object.file,  
on main.tf line 28, in resource "aws\_s3\_bucket\_object" "file":  
28: resource "aws\_s3\_bucket\_object" "file" {

use the aws\_s3\_object resource instead

**Warning: Argument is deprecated**

with aws\_s3\_bucket\_object.file,  
on main.tf line 30, in resource "aws\_s3\_bucket\_object" "file":  
30: key = "hello.txt"

Use the aws\_s3\_object resource instead

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

aws\_s3\_bucket.bucket: Destroying... [id=bucket-pranav-123]

aws\_s3\_bucket.bucket: Destruction complete after 1s

**Destroy complete! Resources: 1 destroyed.**

(base) PS C:\Users\sbpol\Documents\terraform\_scripts\docker\s3>