# AdvDevOps Lab 6

Sanket More D15A 30

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

# Part A: Creating docker image using terraform

Install docker

```
C:\Users\Sanket More>docker
Usage: docker [OPTIONS] COMMAND
A self-sufficient runtime for containers
Common Commands:
             Create and run a new container from an image
 run
             Execute a command in a running container
 exec
             List containers
  ps
 build
             Build an image from a Dockerfile
             Download an image from a registry
  pull
             Upload an image to a registry
 push
            List images
 images
 login
             Log in to a registry
             Log out from a registry
 logout
             Search Docker Hub for images
 search
 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
             Manage contexts
 context
```

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
          ×
docker.tf > ...
       terraform {
         required providers {
           docker = {
             source = "kreuzwerker/docker"
            version = "2.21.0"
      provider "docker" {
        host = "npipe:///./pipe/docker_engine"
 11
 12
 13
      # Pull the Docker image
      resource "docker image" "ubuntu" {
        name = "ubuntu:latest"
       # Create a Docker container
       resource "docker container" "foo" {
 21
         image = docker image.ubuntu.image id
        name = "foo"
 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
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" {
                              = false
= (known after apply)
       + attach
       + bridge
       + command
                               = (known after apply)
        + container_logs
                               = (known after apply)
                              = (known after apply)
= (known after apply)
         entrypoint
        + env
         exit_code
                                 (known after apply)
(known after apply)
         gateway
                                 (known after apply)
(known after apply)
         hostname
         id
                                 (known after apply)
         image
                                 (known after apply)
         init
                              = (known after apply)
= (known after apply)
= (known after apply)
         ip_address
          ip_prefix_length
          ipc_mode
                                 (known after apply)
```

```
- true
= false
= (known after apply)
= (known after apply)
= false
         + stdin_open
         + stop_signal
         + stop_timeout
         + tty
         + healthcheck (known after apply)
         + labels (known after apply)
   # docker_image.ubuntu will be created
+ resource "docker_image" "ubuntu" {
                             r_inage ubuntu (
= (known after apply)
= (known after apply)
= (known after apply)
= "ubuntu:latest"
= (known after apply)
         + id
            image_id
         + latest
         + name
         + 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

```
logs
                                    false
        + must_run
                                    true
                                 = "foo"
= (known after apply)
        + name
        + network_data
        + read_only
+ remove_volumes
+ restart
                                    false
                                    true
"no"
        + rm
                                    false
        + runtime
                                    (known after apply)
                                    (known after apply)
(known after apply)
        + security_opts
        + shm_size
        + start
+ stdin_open
                                    true
                                    false
(known after apply)
(known after apply)
false
        + stop_signal
+ stop_timeout
        + tty
        + 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=af0512641b95dfece26fa5f29deafb8a8d56bd8b9878a246f46bd694e961e5b5]
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
                       IMAGE ID
             TAG
                                       CREATED
                                                     SIZE
                       619c9b7a9ac5
react-img
                                       2 weeks ago
                                                     320MB
             latest
ubuntu
                       edbfe74c41f8
                                       3 weeks ago
             latest
                                                     78.1MB
```

#### 8. Terraform destroy

```
C:\Users\siddi\OneDrive\Desktop\lab-works\terraform_scripts\Docker>terraform destroy
docker_image.ubuntu: Refreshing state... [id=sha256:edbfe74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598aubun
docker_container.foo: Refreshing state... [id=af0512641b95dfece26fa5f29deafb8a8d56bd8b9878a246f46bd694e961e5b5]
Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the
following symbols:
    destrov
Terraform will perform the following actions:
  = [
          command
             "tail",
- "-f",
- "/dev/null",
                              = 0 -> null
= [] -> null
          cpu_shares
          dns
          dns_opts
          dns_search
          entrypoint
          env
                              = "172.17.0.1" -> null
= [] -> null
= "af0512641b95" -> null
          gateway
          group_add
          hostname
                               = "af0512641b95dfece26fa5f29deafb8a8d56bd8b9878a246f46bd694e961e5b5" -> null
          id
                               = "sha256:edbfe74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598a"
          image
          init
                               = "172.17.0.2" -> null
          ip_address
          ip_prefix_length = 16 -> null
                                 "private" -> null
[] -> null
          ipc_mode
          links
          log_driver
                                  "json-file" -> null
```

```
# docker_image.ubuntu will be destroyed
- resource "docker_image" "ubuntu" {
    - id = "sha256:edbfe74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598aubuntu:latest" -> null
                          = "sha256:edbfe74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598a" -> nul
           image_id
                          = "sha256:edbfe74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598a" -> null
           latest
                          = "ubuntu:latest"
           name
           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: ves
docker_container.foo: Destroying... [id=af0512641b95dfece26fa5f29deafb8a8d56bd8b9878a246f46bd694e961e5b5]
docker_container.foo: Destruction complete after 1s
docker_image.ubuntu: Destroying... [id=sha256:edbfe74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598aubuntu:lat
estl
docker_image.ubuntu: Destruction complete after 0s
C:\Users\siddi\OneDrive\Desktop\lab-works\terraform_scripts\Docker>
```

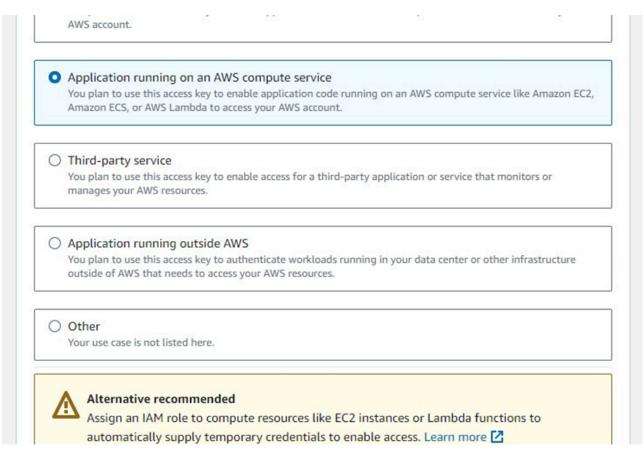
#### 9. Docker images after apply

```
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



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
       terraform {
         required_providers {
           aws = {
             source = "hashicorp/aws"
            version = "~> 5.0"
       # Configure the AWS Provider
       provider "aws" {
 11
         region = "us-east-1"
 12
         access key = ""
 13
        secret key = ""
 15
 17
       resource "aws_s3_bucket" "bucket" {
         bucket = "bucket-pranav-123"
 21
 22
         tags = {
           Name = "My bucket"
 23
         }
 25
```

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) DS C:\Useraclashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoellashoel
```

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:
Terraform will perform the following actions:
   # aws_s3_bucket.terr will be created
+ resource "aws_s3_bucket" "terr" {
                                                                      {
    (known after apply)
    = (known after apply)
    = (known after apply)
    = "my-tf-test-bucket"
    (known after apply)
    = (known after apply)
               arce aws_s3_bucket"
acceleration_status
acl
             arn
bucket
bucket_domain_name
bucket_prefix = (known after apply)
bucket_regional_domain_name = (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)
= (known after apply)
= (known after apply)
= (known after apply)
               arn
bucket
               tags
+ "Environment" = "Dev"
+ "Name" = "My bucket"
               }
tags_all = {
    "Environment" = "Dev"
    + "Name" = "My bucket"
                                                                     = (known after apply)
= (known after apply)
               website_domain
            + website endpoint
            + cors_rule (known after apply)
            + grant (known after apply)
            + lifecycle_rule (known after apply)
            + logging (known after apply)
```

```
}
tags_all
+ "Name" = "My bucket"
          }
website_domain
website_endpoint
                                              = (known after apply)
= (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]
```

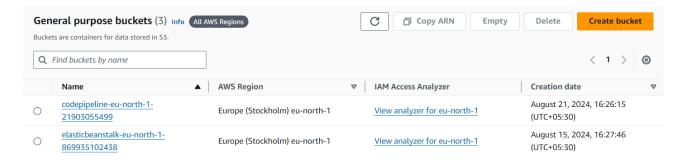
```
tags_all
                                         = {
              "Name" = "My bucket"
       website_domainwebsite_endpoint
                                         = (known after apply)
                                         = (known after apply)
       + cors_rule (known after apply)
       + 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.

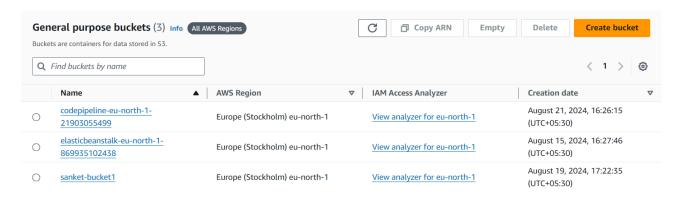
Only 'yes' will be accepted to approve.
  Enter a value: yes
aws_s3_bucket.bucket: Creating...
aws_s3_bucket.bucket: Creation complete after 5s [id=bucket-pranav-123]
Apply complete! Resources: 1 added, 0 changed, 0 destroyed.
```

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

#### BEFORE -



#### AFTER -



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/sbpol/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/sbpol/Documents/terraform_scripts/docker/s3/hello.txt"
}
```

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

```
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" no (base) PS ("Ubservalopholocuments) terraform siptiodicents) 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:

**Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols:

**Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols:

**Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols:

**Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols:

**Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols:

**Terraform used the selected providers the following execution plan. Resource actions are indicated with the following symbols:

**Terraform used the selected providers the following execution plan. Resource actions are indicated with the following symbols:

**Terraform used the selected providers the following execution plan. Resource actions are indicated with the following symbols:

**Terraform used the selected providers the following execution plan. Resource actions are indicated with the following symbols:

**Terraform used the selected providers the following execution plan. Resource actions are indicated with the following symbols:

**Terraform used the selected providers and indicated with the following symbols:

**Terraform used the selected providers and indicated with the following symbols:

**Terraform used the selected providers and indicated with the following symbols:

**Terra
```

```
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" f

use the aws_s3_object resource instead

Warning: Argument is deprecated

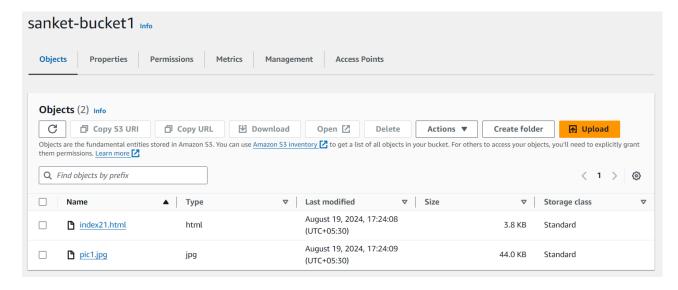
with aws_s3_bucket_object.file,
on main.tf line 29, in resource "aws_s3_bucket_object" "file":
29: bucket = aws_s3_bucket.bucket.id

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



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

```
(base) PS C:\Users\sbpol\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
- bucket
- bucket
- bucket_domain_name
- bucket_domain_name
- bucket_regional_domain_name
- force_destroy
- hosted_zone_id
- bucket_gand = "ZSAQBSTGFY3STF" -> null
- bucket_gand = "ZSAQBSTGFY3STF" -> null
- bucket_gand = "Jucket_gand = "Jucket_ga
                                                                                                                                                = false -> null
= "Z3AQBSTGFYJSTF" -> null
= "bucket-pranav-123" -> null
                               object_lock_enabled
region
request_payer
                                                                                                                                                = false -> null
= "us-east-1" -> null
= "BucketOwner" -> null
                                tags
- "Name" = "My bucket"
                                 = {
                                 # (3 unchanged attributes hidden)
                                                id = "10def03d73e09d8adda11bfe68e632f70a83a37758b74ea6e933dafd0250c850" -> null permissions = [
                                grant {
- id
                                                                  "FULL_CONTROL",
                                                ] -> null
                                                                                               = "CanonicalUser" -> null
                                 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>
```