

Assignment: - by Shamik Guha Ray

Successfully deployed the necessary infrastructure setup to execute below assignment:

Problem/question statement

Design an automation deployed with GITHUB/GITLAB pipelines to create load balanced simple nginx hosted on 1 or more virtual machines on AWS or Azure with the following assumption :

1. CIDR retrieve from REST API https://FDQN/vend_ip return

```
{
  "ip_address":"192.168.0.0",
  "subnet_size":"/16"
}
```
2. Create subnets with size /24
3. Generate SSH key for VM credential
4. Take into consideration CSP Best Practices such as security and resiliency
5. Take into consideration coding/scripting practices
6. Leverage on native cloud metrics/logging for error handling
7. Can use bash/terraform/python/powershell for the stack, github or github for the IAC pipeline

Provide evidence of the successful execution

Deployment Steps:

Repository Creation:

- Created a new repository named "github-nginx-terraform-aws".
- GitHub Repo Link is attached [here](#).

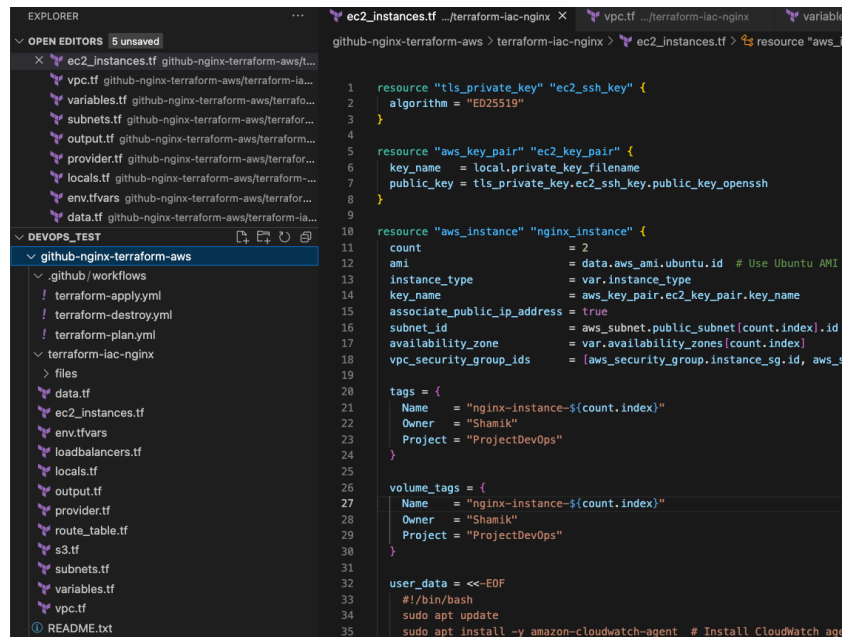
Local Setup:

- Cloned the repository locally.
- Created Terraform configurations.
- Pushed the configurations to the master branch of the repository. The GitHub Actions workflows got initiated upon "git push to master".

```
sguharay@s-guharay-n60cm github-nginx-terraform-aws % git add .
sguharay@s-guharay-n60cm github-nginx-terraform-aws % git commit -m "Deploy nginx server infra to AWS"
[master a68d37c] Deploy nginx server infra to AWS
1 file changed, 4 insertions(+), 4 deletions(-)
sguharay@s-guharay-n60cm github-nginx-terraform-aws %
sguharay@s-guharay-n60cm github-nginx-terraform-aws %
sguharay@s-guharay-n60cm github-nginx-terraform-aws % git push origin master
Enumerating objects: 7, done.
Counting objects: 100% (7/7), done.
Delta compression using up to 10 threads
Compressing objects: 100% (4/4), done.
Writing objects: 100% (4/4), 408 bytes | 408.00 KiB/s, done.
Total 4 (delta 3), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (3/3), completed with 3 local objects.
To github.com:ShamikGray/github-nginx-terraform-aws.git
e021051..a68d37c master -> master
sguharay@s-guharay-n60cm github-nginx-terraform-aws %
```

Terraform Configuration:

- Utilized VS Code to craft the Terraform configuration files.



The screenshot shows the VS Code interface with the Explorer and Open Editors views. The Explorer view shows a project structure for 'github-nginx-terraform-aws' with files like 'vpc.tf', 'variables.tf', 'subnets.tf', 'output.tf', 'provider.tf', 'locals.tf', 'env.tfvars', 'data.tf', 'ec2_instances.tf', 'loadbalancers.tf', 'route_table.tf', 's3.tf', 'subnets.tf', 'variables.tf', and 'vpc.tf'. The Open Editors view shows the 'ec2_instances.tf' file with the following Terraform configuration:

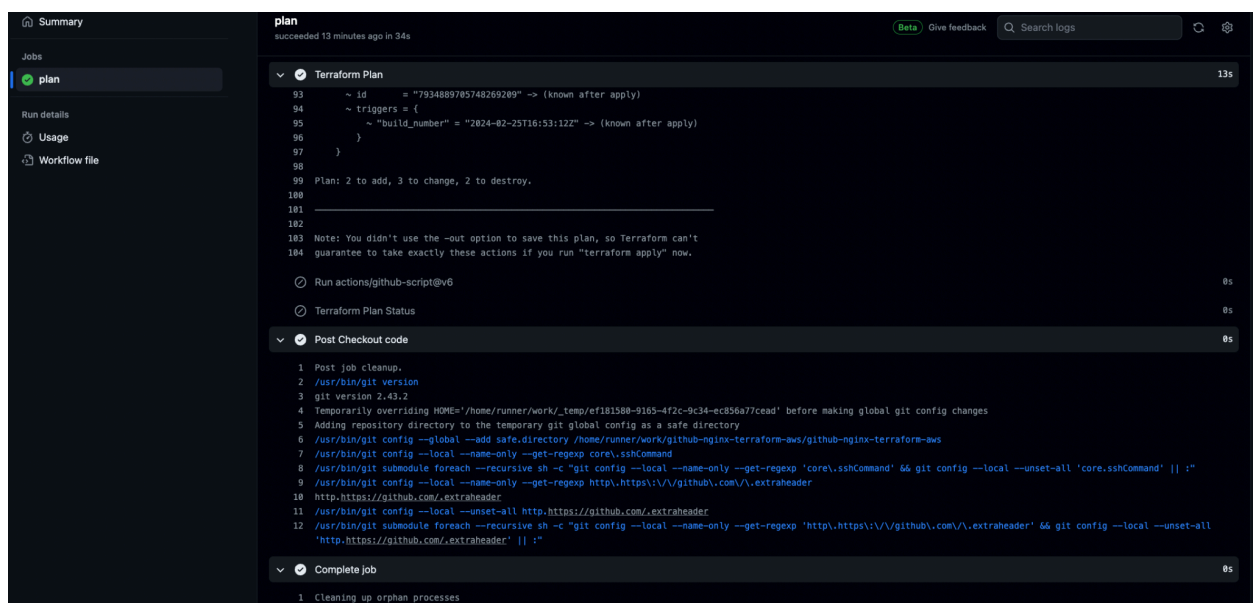
```
1 resource "tls_private_key" "ec2_ssh_key" {
2   algorithm = "ED25519"
3 }
4
5 resource "aws_key_pair" "ec2_key_pair" {
6   key_name   = local.private_key_filename
7   public_key = tls_private_key.ec2_ssh_key.public_key_openssh
8 }
9
10 resource "aws_instance" "nginx_instance" {
11   count          = 2
12   ami            = data.aws_ami.ubuntu.id # Use Ubuntu AMI
13   instance_type  = var.instance_type
14   key_name       = aws_key_pair.ec2_key_pair.key_name
15   associate_public_ip_address = true
16   subnet_id      = aws_subnet.public_subnet[count.index].id
17   availability_zone = var.availability_zones[count.index]
18   vpc_security_group_ids = [aws_security_group.instance_sg.id, aws_security_group.elb_sg.id]
19
20   tags = {
21     Name        = "nginx-instance-${count.index}"
22     Owner       = "Shamik"
23     Project     = "ProjectDevOps"
24   }
25
26   volume_tags = {
27     Name        = "nginx-instance-${count.index}"
28     Owner       = "Shamik"
29     Project     = "ProjectDevOps"
30   }
31
32   user_data = <<-EOF
33   #!/bin/bash
34   sudo apt update
35   sudo apt install -y amazon-cloudwatch-agent # Install CloudWatch agent
```

Workflow Utilization:

- Implemented two separate workflows: Terraform Plan and Terraform Apply.

By following these steps, the necessary infrastructure for the assignment was successfully deployed. Below is the screenshot of Github Actions Deployment:

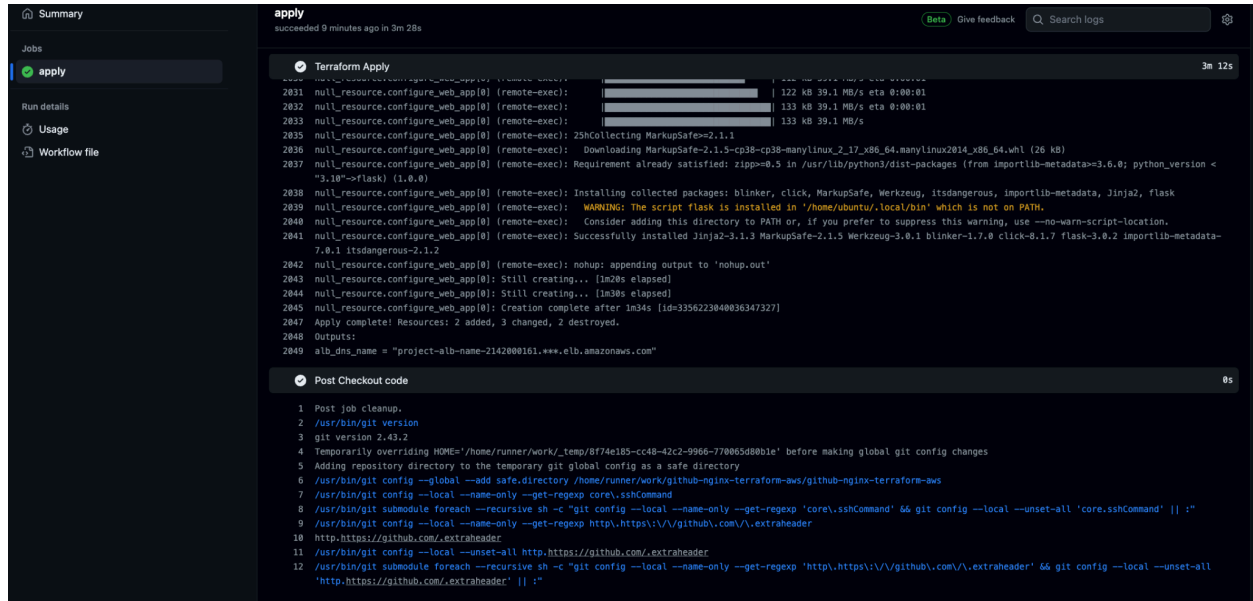
Terraform Plan - GitHub Actions Workflow:



The screenshot shows the GitHub Actions workflow for 'Terraform Plan'. The workflow is named 'plan' and is triggered by a push event to the 'main' branch. The workflow consists of the following steps:

- Terraform Plan**: This step runs the Terraform plan command. The output shows the plan details, including the number of resources to be added, changed, or destroyed. The step is marked as 'succeeded'.
- Post Checkout code**: This step runs a series of commands to set up the environment, including installing Git, setting up the repository, and configuring the SSH key.
- Complete job**: This step runs the 'echo' command to indicate the end of the job.

Terraform Apply - GitHub Action Workflow:



The screenshot displays a GitHub Actions workflow run for the 'apply' job. The left sidebar shows the 'Summary' tab with a green checkmark for the 'apply' job. The main panel shows the workflow logs for the 'Terraform Apply' step, which succeeded 9 minutes ago in 3m 28s. The logs show the execution of Terraform commands on a remote host, including downloading and installing various packages like MarkupSafe, Werkzeug, and Jinja2. A warning is shown about the flask script location. The final output shows the creation of an ALB resource with the name 'project-alb-name-2142000161'. Below the logs, the 'Post Checkout code' step is shown, which performs cleanup and git configuration tasks.

```
2831 null_resource.configure_web_app[0] (remote-exec): | 122 KB 39.1 MB/s eta 0:00:01
2832 null_resource.configure_web_app[0] (remote-exec): | 133 KB 39.1 MB/s eta 0:00:01
2833 null_resource.configure_web_app[0] (remote-exec): | 133 KB 39.1 MB/s
2835 null_resource.configure_web_app[0] (remote-exec): 25hCollecting MarkupSafe<=2.1.1
2836 null_resource.configure_web_app[0] (remote-exec): Downloading MarkupSafe-2.1.1-cp38-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (26 KB)
2837 null_resource.configure_web_app[0] (remote-exec): Requirement already satisfied: zipp>=0.5 in /usr/lib/python3/dist-packages (from importlib-metadata==3.6.0; python_version <
"3.10">=flask) (1.0.0)
2838 null_resource.configure_web_app[0] (remote-exec): Installing collected packages: blinker, click, MarkupSafe, Werkzeug, itsdangerous, importlib-metadata, Jinja2, flask
2839 null_resource.configure_web_app[0] (remote-exec): WARNING: The script flask is installed in '/home/ubuntu/.local/bin' which is not on PATH.
2840 null_resource.configure_web_app[0] (remote-exec): Consider adding this directory to PATH or, if you prefer to suppress this warning, use --no-warn-script-location.
2841 null_resource.configure_web_app[0] (remote-exec): Successfully installed Jinja2-3.1.3 MarkupSafe-2.1.5 Werkzeug-3.0.1 blinker-1.7.0 click-8.1.7 flask-3.0.2 importlib-metadata-
7.0.1 itsdangerous-2.1.2
2842 null_resource.configure_web_app[0] (remote-exec): nohup: appending output to 'nohup.out'
2843 null_resource.configure_web_app[0]: Still creating... [1m28s elapsed]
2844 null_resource.configure_web_app[0]: Still creating... [1m30s elapsed]
2845 null_resource.configure_web_app[0]: Creation complete after 1m34s [id=3356223040036347327]
2847 Apply complete! Resources: 2 added, 3 changed, 2 destroyed.
2848 Outputs:
2849 alb_dns_name = "project-alb-name-2142000161.***.elb.amazonaws.com"
```

```
1 Post job cleanup.
2 /usr/bin/git version
3 git version 2.43.2
4 Temporarily overriding HOME='/home/runner/work/_temp/8f74c185-cc48-42c2-9966-770065d80b1e' before making global git config changes
5 Adding repository directory to the temporary git global config as a safe directory
6 /usr/bin/git config --global --add safe.directory /home/runner/work/github-nginx-terraform-aws/github-nginx-terraform-aws
7 /usr/bin/git config --local --name-only --get-regexp core.sshCommand
8 /usr/bin/git submodule foreach --recursive sh -c "git config --local --name-only --get-regexp 'core.sshCommand' && git config --local --unset-all 'core.sshCommand' || :"
9 /usr/bin/git config --local --name-only --get-regexp http:http://v/github.com/\extraheader
10 http:https://github.com/extraheader
11 /usr/bin/git config --local --unset-all http:https://github.com/extraheader
12 /usr/bin/git submodule foreach --recursive sh -c "git config --local --name-only --get-regexp 'http:https://v/github.com/\extraheader' && git config --local --unset-all
'http:https://github.com/extraheader' || :"
```

Post-Deployment Validation:

Infrastructure Verification:

- Ensured the correct provisioning of all infrastructure resources post-deployment.

Accessibility Testing:

- Tested accessibility to nginx servers using the load balancer's DNS name.

Multi-AZ Availability Testing:

- Validated the load balancer's ability to distribute traffic across instances in multiple availability zones.

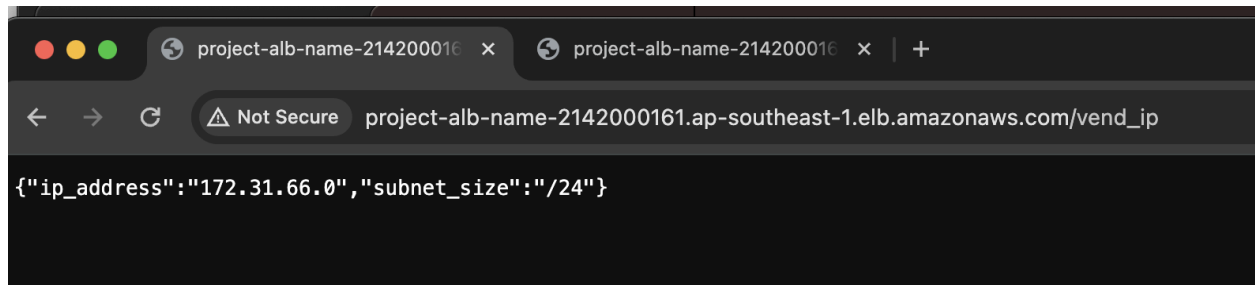
Steps Taken:

Accessed the following DNS resolver to perform post-deployment validation:

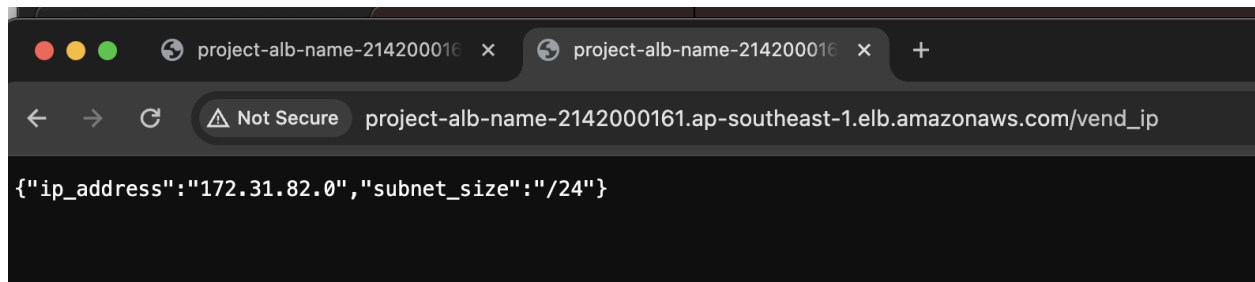
"http://project-alb-name-2142000161.ap-southeast-1.elb.amazonaws.com/vend_ip"

Screenshot of the Content Served by Our Two Nginx EC2 Servers:

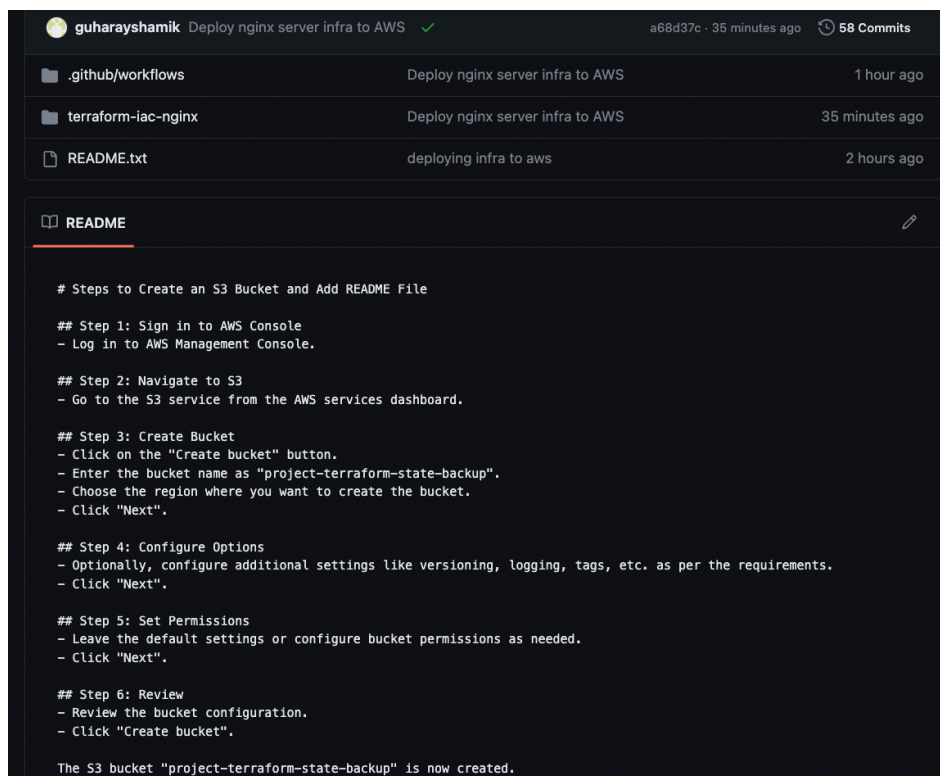
1)



2)



Attached the README containing the steps to setup the s3 bucket before triggering the terraform init/plan/apply.



Please find the screenshots of AWS resources deployed using the workflow:

Instances (2) info

Any state ▾

Instance state = running ✕

Clear filters

Refresh

Connect

Instance state ▾

Actions ▾

Launch instances ▾

1

< > ⌕

<input type="checkbox"/>	Name ↗	Instance ID	Instance state	Instance type	Status check	Alarm status
<input type="checkbox"/>	nginx-instance-1	i-0fc9e51dbccc6c94d	Running	t2.micro	2/2 checks passed	View alarms
<input type="checkbox"/>	nginx-instance-0	i-056f6b6fabf44cc8	Running	t2.micro	2/2 checks passed	View alarms

EC2 > Load balancers

Load balancers (1/1)

Elastic Load Balancing scales your load balancer capacity automatically in response to changes in incoming traffic.

<

1

>

ⓘ

<input checked="" type="checkbox"/>	Name	DNS name	State	VPC ID	Availability Zones
<input checked="" type="checkbox"/>	project-alb-name	project-alb-name-2142000161.ap-southeast-1.elb.amazonaws.com	Active	vpc-07cde10be030ade69	2 Availability Zones

Load balancer: project-alb-name

Details

Listeners and rules

Network mapping

Security

Monitoring

Integrations

Attributes

Tags

Listeners and rules (1) Info

A listener checks for connection requests on its configured protocol and port. Traffic received by the listener is routed according to the default action and any additional rules.

<

1

>

ⓘ

<input type="checkbox"/>	Protocol:Port	Default action	Rules	ARN	Security policy	Default SSL/TLS certificate	mTLS
<input type="checkbox"/>	HTTP:80	<div>Forward to target group</div> <ul style="list-style-type: none"> Shamik-proj-EC2-Target-Group 🔗: 1 (100%) Group-level stickiness: Off 	1 rule	ARN	Not applicable	Not applicable	Not applicable

Shamik-proj-EC2-Target-Group

Actions

Introducing Automatic Target Weights (ATW) to increase application availability

Automatic Target Weights is achieved by turning on anomaly mitigation, which provides responsive, dynamic distribution of traffic to targets based on anomaly detection results. All HTTP/HTTPS target groups now include anomaly detection by default. [Learn more](#)

Details

arn:aws:elasticloadbalancing:ap-southeast-1:847787987541:targetgroup/Shamik-proj-EC2-Target-Group/bcbcd919d85acfe2

Target type Instance	Protocol : Port HTTP: 80	Protocol version HTTP1	VPC vpc-07cde10be030ade69		
IP address type IPv4	Load balancer project-alb-name				

2

Total targets

2
Healthy

0
Unhealthy

0
Unused

0
Initial

0
Draining

0 Anomalous

► Distribution of targets by Availability Zone (AZ)

Select values in this table to see corresponding filters applied to the Registered targets table below.

Targets

Monitoring

Health checks

Attributes

Tags

Health check settings

Edit

Protocol HTTP	Path /	Port 80	Healthy threshold 2 consecutive health check successes
Unhealthy threshold 2 consecutive health check failures	Timeout 5 seconds	Interval 30 seconds	Success codes 200