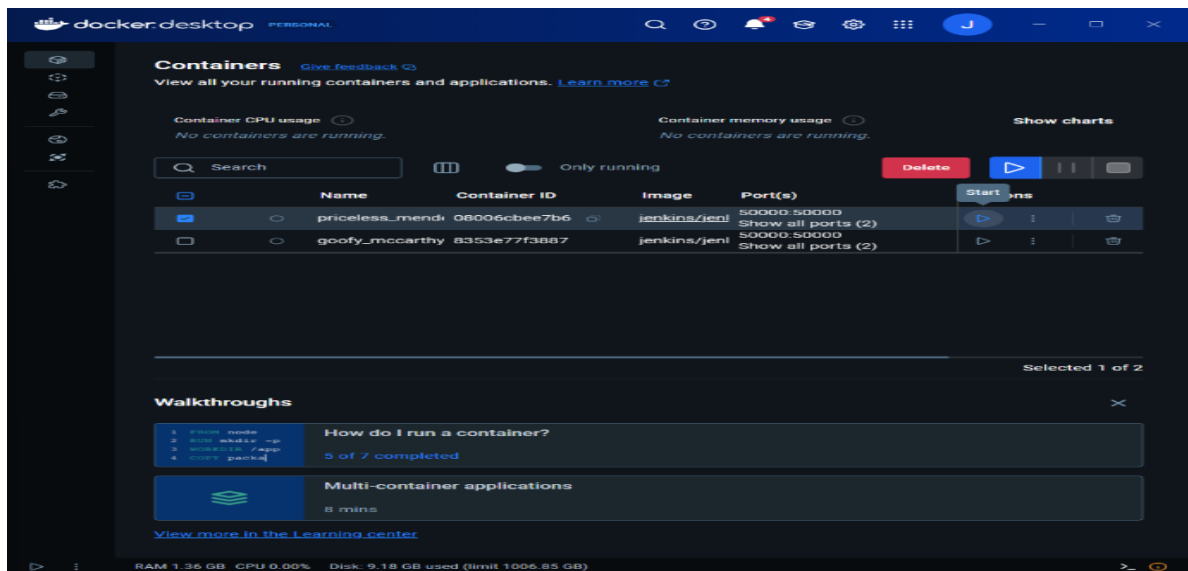
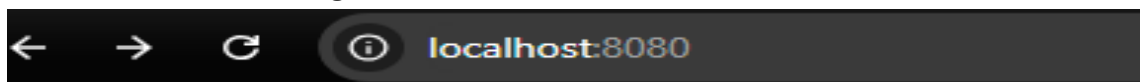


# Jenkins with Terraform Documentation

1. Make sure to have the IAM user for Jenkins in AWS with the AWS Access & Secret Key
  - AWS Access Key - xxxxxxxxxx
  - AWS Secret Key – xxxxxxxxxx
2. Turn on the Jenkins container in either Docker Desktop or CLI



3. Go to web browser and go to localhost:8080



4. Log into Jenkins with Jenkins credentials
5. In Jenkins go to Manage Jenkins>>Credentials
6. Click on system then Global credentials (unrestricted)
7. Click on add credentials
8. For the Kind set to AWS Credentials
9. For ID add whatever label you want
10. Add the Access Key ID and Secret Key

Jenkins Credentials Provider: Jenkins

### Add Credentials

Domain

Global credentials (unrestricted)

Kind

AWS Credentials

Scope ?

Global (Jenkins, nodes, items, all child items, etc)

ID ?

Description ?

Access Key ID ?

Secret Access Key

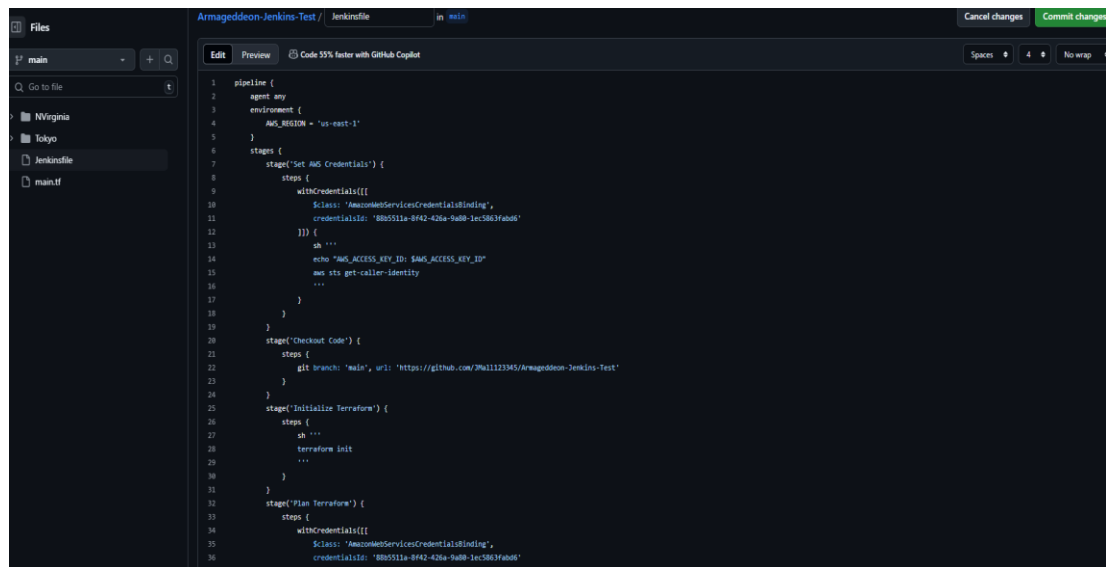
11. Click Create

12. Go back to Dashboard

13. Github Repo

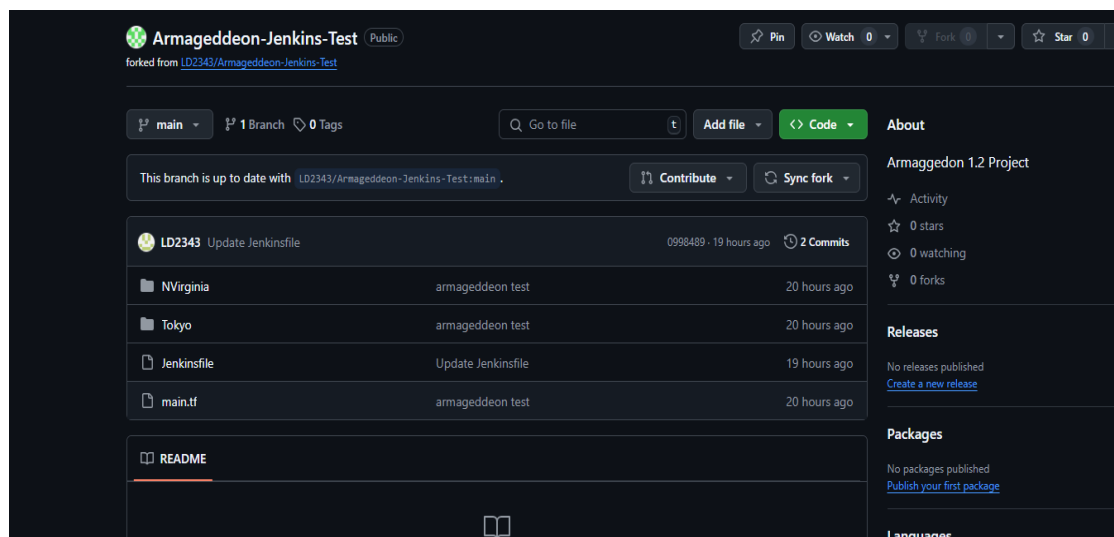
- Create a new repo with existing working TF project
- Create empty repo
- On pc create a folder to git clone empty repo to
- Manually add working tf project
- Manually add the Jenkinsfile
- Update the Jenkinsfile

- update the credentialsId rows to match the ID in Jenkins used for credential creation and the github url to your repo url

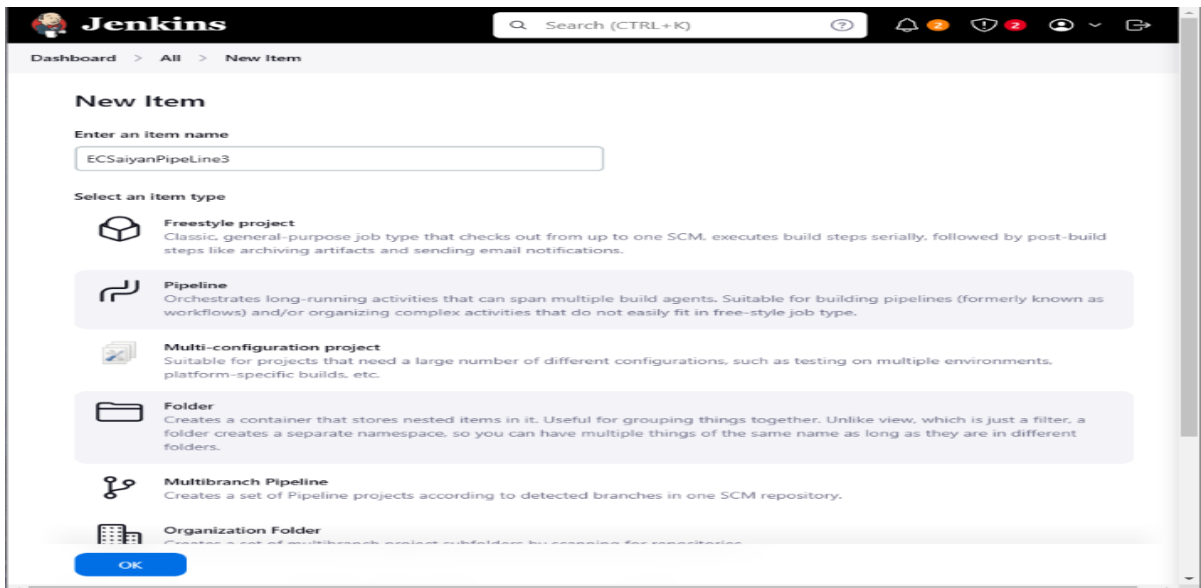


The screenshot shows a code editor with a file named 'Jenkinsfile' open. The left sidebar shows a file tree with 'main', 'NVirginia', 'Tokyo', 'Jenkinsfile', and 'main.tf'. The main editor area shows the following Jenkinsfile content:

```
1 pipeline {
2   agent any
3   environment {
4     AWS_REGION = 'us-east-1'
5   }
6   stages {
7     stage('Set AWS Credentials') {
8       steps {
9         withCredentials([
10           $class: 'AmazonServicesCredentialsBinding',
11           credentialsId: '88b5511a-8442-428a-9a0b-1ec5863fab0d'
12         ]) {
13           sh '''
14             echo "AWS_ACCESS_KEY_ID: $AWS_ACCESS_KEY_ID"
15             aws sts get-caller-identity
16           '''
17         }
18       }
19     }
20     stage('Checkout Code') {
21       steps {
22         git branch: 'main', url: 'https://github.com/LD2343/Armageddeon-Jenkins-Test'
23       }
24     }
25     stage('Initialize Terraform') {
26       steps {
27         sh '''
28           terraform init
29         '''
30       }
31     }
32     stage('Plan Terraform') {
33       steps {
34         withCredentials([
35           $class: 'AmazonServicesCredentialsBinding',
36           credentialsId: '88b5511a-8442-428a-9a0b-1ec5863fab0d'
37         ]) {
38           sh '''
39             terraform plan
40           '''
41         }
42       }
43     }
44   }
45 }
```



- Do the git add . >> git commit -m "message" >> git push
14. Click on New Item
  15. Name the Pipeline and select Pipeline and click ok
  16. Scroll down to Pipeline section

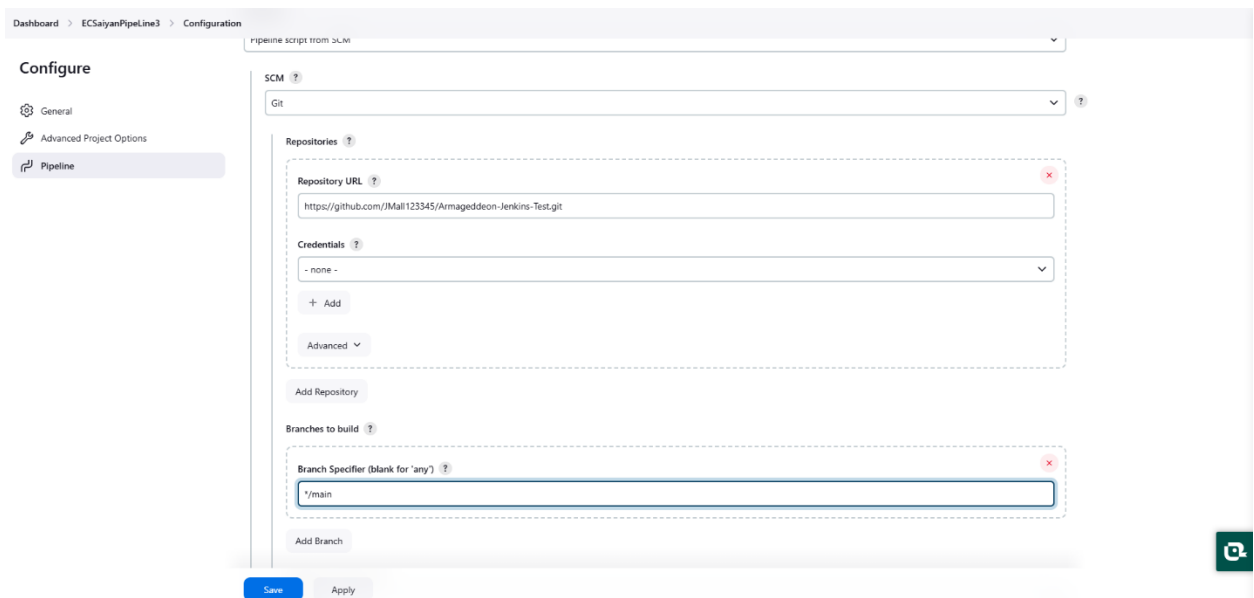


17. For Definition select Pipeline script from SCM

18. For SCM select Git

19. Add the Repo URL from github

20. Under Branch change it to main instead of master



21. Click Save

22. Now need to install terraform and AWS CLI inside the container

- Open new gitbash
- Enter the following cmd
- docker ps ### to get running containers
- See the 1st 3 characters of the container id
- Add this cmd

- docker exec -it --user root <container name or 1st 3 characters of container
- id> bash
- Hit enter this cmd
- apt update && apt install -y awscli
- Once install complete do this cmds
- mkdir -p /home/jenkins/bin

```
08006cbee7b6 jenkins/jenkins:lts-jdk11 "/usr/bin/tini -- /u..." 11 days ago
Up About an hour 0.0.0.0:8080->8080/tcp, 0.0.0.0:50000->50000/tcp priceles
s_mendel

OMALL@Jason2 MINGW64 ~/Documents/Jenkins
$ docker exec -it --user root 080 bash
root@08006cbee7b6:/#
root@08006cbee7b6:/# apt update && apt install -y awscli
Hit:1 http://deb.debian.org/debian bookworm InRelease
Get:2 http://deb.debian.org/debian bookworm-updates InRelease [55.4 kB]
Hit:3 http://deb.debian.org/debian-security bookworm-security InRelease
Hit:4 https://packagecloud.io/github/git-lfs/debian bookworm InRelease
Fetched 55.4 kB in 1s (69.6 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
25 packages can be upgraded. Run 'apt list --upgradable' to see them.
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
awscli is already the newest version (2.9.19-1).
0 upgraded, 0 newly installed, 0 to remove and 25 not upgraded.
root@08006cbee7b6:/# mkdir -p /home/jenkins/bin
root@08006cbee7b6:/#
```

- curl -fsSL [https://releases.hashicorp.com/terraform/1.5.7/terraform\\_1.5.7\\_linux\\_amd64.zip](https://releases.hashicorp.com/terraform/1.5.7/terraform_1.5.7_linux_amd64.zip) -o /home/jenkins/terraform.zip

```
root@08006cbee7b6:/# curl -fsSL https://releases.hashicorp.com/terraform/1.5.7/
erraform_1.5.7_linux_amd64.zip -o /home/jenkins/terraform.zip
root@08006cbee7b6:/#
```

- unzip /home/jenkins/terraform.zip -d /home/jenkins/bin  
rm /home/jenkins/terraform.zip

```
root@08006cbee7b6:/# unzip /home/jenkins/terraform.zip -d /home/jenkins/bin
Archive: /home/jenkins/terraform.zip
  inflating: /home/jenkins/bin/terraform
root@08006cbee7b6:/#
```

- export PATH="/home/jenkins/bin:\$PATH"
- Now check if terraform is installed
- terraform —version

```

root@08006cbee7b6:/# curl -fsSL https://releases.hashicorp.com/terraform/1.5.7/t
erraform_1.5.7_linux_amd64.zip -o /home/jenkins/terraform.zip
root@08006cbee7b6:/# unzip /home/jenkins/terraform.zip -d /home/jenkins/bin
Archive: /home/jenkins/terraform.zip
  inflating: /home/jenkins/bin/terraform
root@08006cbee7b6:/# rm /home/jenkins/terraform.zip
root@08006cbee7b6:/# export PATH="/home/jenkins/bin:$PATH"
bash: export: `PATH="/home/jenkins/bin:/opt/java/openjdk/bin:/usr/local/sbin:/usr
/local/bin:/usr/sbin:/usr/bin:/sbin:/bin': not a valid identifier
root@08006cbee7b6:/# export PATH="/home/jenkins/bin:$PATH"
root@08006cbee7b6:/# terraform -version
Terraform has no command named "-version". Did you mean "version"?

To see all of Terraform's top-level commands, run:
  terraform -help

root@08006cbee7b6:/# terraform version
Terraform v1.5.7
on linux_amd64

Your version of Terraform is out of date! The latest version
is 1.10.5. You can update by downloading from https://www.terraform.io/downloads
.html
root@08006cbee7b6:/#

```

23. Ok go to the pipeline and run it


```
    [32m+0[0m[0m private_key_pem          = (sensitive value)
    [32m+0[0m[0m private_key_pem_pkcs8      = (sensitive value)
    [32m+0[0m[0m public_key_fingerprint_md5   = (known after apply)
    [32m+0[0m[0m public_key_fingerprint_sha256 = (known after apply)
    [32m+0[0m[0m public_key_openssh          = (known after apply)
    [32m+0[0m[0m public_key_pem              = (known after apply)
    [32m+0[0m[0m rsa_bits                    = 2048
  }

[1mPlan:[0m 56 to add, 0 to change, 0 to destroy.
[0m[90m







Saved the plan to: tfplan

To perform exactly these actions, run the following command to apply:
    terraform apply "tfplan"
[Pipeline] }
[Pipeline] // withCredentials
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (Apply Terraform)
[Pipeline] input
Approve Terraform Apply?
Deploy or Abort
Approved by Jason Mallard
[Pipeline] withCredentials
Masking supported pattern matches of $AWS_ACCESS_KEY_ID or $AWS_SECRET_ACCESS_KEY
[Pipeline] {
[Pipeline] sh
+ export AWS_ACCESS_KEY_ID=****
+ export AWS_SECRET_ACCESS_KEY=****
+ terraform apply -auto-approve tfplan
[0m[1mmodule.NVirginia.tls_private_key.NVirginiaLinux: Creating...[0m[0m
[0m[1mmodule.Tokyo.tls_private_key.ToykoLinux: Creating...[0m[0m
[0m[1mmodule.NVirginia.tls_private_key.NVirginiaLinux: Creation complete after 0s [id=d4e5fd7d4b1c3c5b0ce4a313e49c20b92116e0c9][0m
[0m[1mmodule.NVirginia.data.tls_public_key.NVirginiaLinux: Reading...[0m[0m
```


```
+ terraform apply -auto-approve tfplan
[0m[1mmodule.Tokyo.aws_autoscaling_group.tokyo_asg: Creating...[0m[0m
[0m[1mmodule.Tokyo.aws_autoscaling_group.tokyo_asg: Still creating... [30s elapsed][0m[0m
[0m[1mmodule.Tokyo.aws_autoscaling_group.tokyo_asg: Still creating... [20s elapsed][0m[0m
[0m[1mmodule.Tokyo.aws_autoscaling_group.tokyo_asg: Still creating... [30s elapsed][0m[0m
[0m[1mmodule.Tokyo.aws_autoscaling_group.tokyo_asg: Still creating... [40s elapsed][0m[0m
[0m[1mmodule.Tokyo.aws_autoscaling_group.tokyo_asg: Still creating... [50s elapsed][0m[0m
[0m[1mmodule.Tokyo.aws_autoscaling_group.tokyo_asg: Still creating... [1m0s elapsed][0m[0m
[0m[1mmodule.Tokyo.aws_autoscaling_group.tokyo_asg: Creation complete after 1m10s [id=tokyo-auto-scaling-group-20250202220657736500000001][0m
[0m[1mmodule.Tokyo.aws_autoscaling_policy.tokyo_scaling_policy: Creating...[0m[0m
[0m[1mmodule.Tokyo.aws_autoscaling_policy.tokyo_scaling_policy: Creation complete after 2s [id=tokyo-cpu-target][0m
[0m[1m[32m
Apply complete! Resources: 2 added, 0 changed, 0 destroyed.
[0m
[Pipeline] }
[Pipeline] // withCredentials
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (Declarative: Post Actions) (hide)
[Pipeline] echo
Terraform deployment completed successfully!
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```


 **Jenkins**


Search (CTRL+K)


    Jason Mallard   log out


Dashboard >


 New Item

 Build History

 Project Relationship

 Check File Fingerprint










 Manage Jenkins

 My Views

Build Queue

View build for this queue

Add description

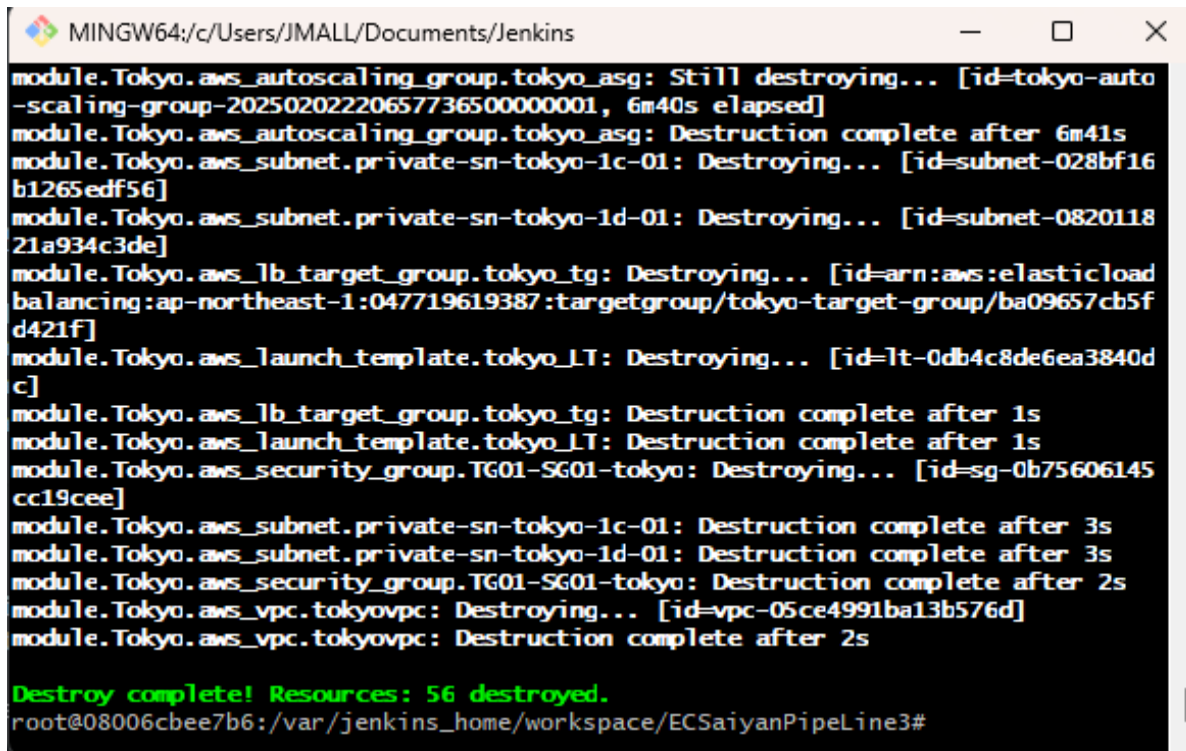
S	W	Name	Last Success	Last Failure	Last Duration
		ECSaiyanPipeline3	2 min 7 sec #9	6 min 34 sec #8	1 min 46 sec 
		ECSaiyans2	N/A	N/A	N/A 
		ECSaiyansJenkins	22 hr #4	22 hr #3	4 min 21 sec 

# Tear Down

## How to tear Down Jenkins Terraform:

1. To view the active terraform configuration use the exec or ssh of you container and change directories into the path below:

- `cd var/jenkins_home/workspace/<PIPELINENAME>`
2. Input AWS Credentials into the CLI of your container Login to destroy:
- `export AWS_ACCESS_KEY_ID="xxxxxxx"`
  - `export AWS_SECRET_ACCESS_KEY="xxxxxxx"`
  - `export AWS_REGION="xxxxxxx"`
3. Run Terraform Destroy
4. `terraform destroy`



```
MINGW64:/c:/Users/JMALL/Documents/Jenkins
module.Tokyo.aws_autoscaling_group.tokyo_asg: Still destroying... [id=tokyo-auto-scaling-group-20250202220657736500000001, 6m40s elapsed]
module.Tokyo.aws_autoscaling_group.tokyo_asg: Destruction complete after 6m41s
module.Tokyo.aws_subnet.private-sn-tokyo-1c-01: Destroying... [id=subnet-028bf16b1265edf56]
module.Tokyo.aws_subnet.private-sn-tokyo-1d-01: Destroying... [id=subnet-082011821a934c3de]
module.Tokyo.aws_lb_target_group.tokyo_tg: Destroying... [id=arn:aws:elasticloadbalancing:ap-northeast-1:047719619387:targetgroup/tokyo-target-group/ba09657cb5fd421f]
module.Tokyo.aws_launch_template.tokyo_LT: Destroying... [id=lt-0db4c8de6ea3840dc]
module.Tokyo.aws_lb_target_group.tokyo_tg: Destruction complete after 1s
module.Tokyo.aws_launch_template.tokyo_LT: Destruction complete after 1s
module.Tokyo.aws_security_group.TG01-SG01-tokyo: Destroying... [id=sg-0b75606145cc19cee]
module.Tokyo.aws_subnet.private-sn-tokyo-1c-01: Destruction complete after 3s
module.Tokyo.aws_subnet.private-sn-tokyo-1d-01: Destruction complete after 3s
module.Tokyo.aws_security_group.TG01-SG01-tokyo: Destruction complete after 2s
module.Tokyo.aws_vpc.tokyovpc: Destroying... [id=vpc-05ce4991ba13b576d]
module.Tokyo.aws_vpc.tokyovpc: Destruction complete after 2s

Destroy complete! Resources: 56 destroyed.
root@08006cbee7b6:/var/jenkins_home/workspace/ECSaiyanPipeLine3#
```

YOUR DONE!