

Project 2-

=====

Step1:-

Create 1 instance named "Jenkins_Terraform_Ansible >

Install terraform

```
wget -O- https://apt.releases.hashicorp.com/gpg | sudo gpg --dearmor -o
/usr/share/keyrings/hashicorp-archive-keyring.gpg
echo "deb [signed-by=/usr/share/keyrings/hashicorp-archive-keyring.gpg]
https://apt.releases.hashicorp.com $(lsb_release -cs) main" | sudo tee
/etc/apt/sources.list.d/hashicorp.list
sudo apt update && sudo apt install terraform -y
```

sudo nano main.tf

```
provider "aws" {
  secret_key = ""
  access_key = ""
  region = "us-west-1"
}

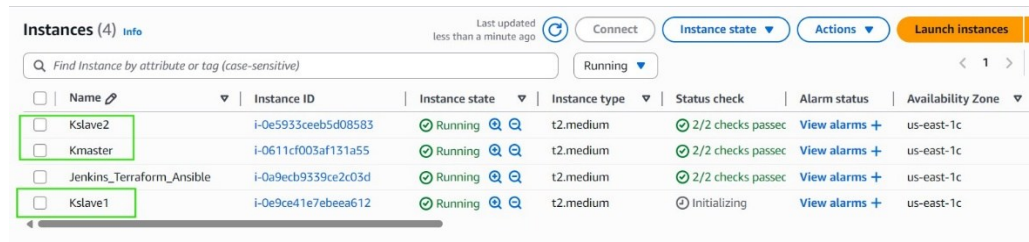
resource "aws_instance" "K8-M" {
  ami = ""
  instance_type = "t2.medium"
  key_name = ""
  tags = {
    Name = "Kmaster"
  }
}

resource "aws_instance" "K8-S1" {
  ami = ""
  instance_type = "t2.medium"
  key_name = ""
  tags = {
    Name = "Kslave1"
  }
}
```

```
}
```

```
resource "aws_instance" "K8-S2" {  
  ami = ""  
  instance_type = "t2.medium"  
  key_name = ""  
  tags = {  
    Name = "Kslave2"  
  }  
}
```

Terraform init
Terraform plan
Terraform apply



Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone
Kslave2	i-0e5933ceeb5d08583	Running	t2.medium	2/2 checks passed	View alarms +	us-east-1c
Kmaster	i-0611cf003af131a55	Running	t2.medium	2/2 checks passed	View alarms +	us-east-1c
Jenkins_Terraform_Ansible	i-0a9ecb9339ce2c03d	Running	t2.medium	2/2 checks passed	View alarms +	us-east-1c
Kslave1	i-0e9ce41e7ebee612	Running	t2.medium	1/2 checks passed	View alarms +	us-east-1c

You will get to see 3 ec2 instances running.

```
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_instance.K8-S1: Creating...  
aws_instance.K8-M: Creating...  
aws_instance.K8-S2: Creating...  
aws_instance.K8-S1: Still creating... [00m10s elapsed]  
aws_instance.K8-M: Still creating... [00m10s elapsed]  
aws_instance.K8-S2: Still creating... [00m10s elapsed]  
aws_instance.K8-S1: Creation complete after 13s [id=i-0e9ce41e7ebee612]  
aws_instance.K8-M: Creation complete after 13s [id=i-0611cf003af131a55]  
aws_instance.K8-S2: Creation complete after 13s [id=i-0e5933ceeb5d08583]  
  
Apply complete! Resources: 3 added, 0 changed, 0 destroyed.  
ubuntu@ip-172-31-28-63:~$
```

i-0a9ecb9339ce2c03d (Jenkins_Terraform_Ansible)
PublicIPs: 54.226.70.54 PrivateIPs: 172.31.28.63

Step 2 :-

Install Ansible on the Jenkins_Terraform_Ansible

```
sudo apt update  
sudo apt install software-properties-common  
sudo add-apt-repository --yes --update ppa:ansible/ansible  
sudo apt install ansible -y
```

Step 3:-

Now create a cluster for KMaster

On Master:

```
Ssh-keygen  
Sudo cat id_rsa.pub  
Copy ssh key
```

Go on Kmaster

Cd .ssh
Sudo nano authorized_keys
Paste ssh key

Go on Master:

Cd /etc/ansible
Ls
Sudo nano hosts
[test]
Private Ip of Kmaster
Ansible -m ping all

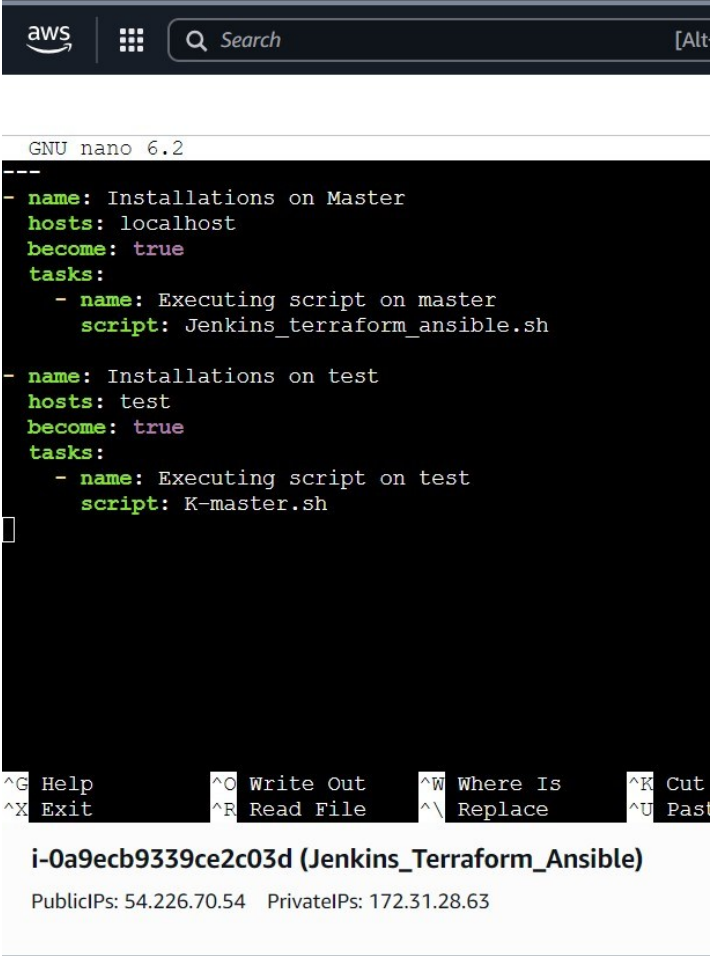
Step 4:-

Playbook Syntax:

Sudo nano playbook.yaml

```
---
- name: Installations on Master
  hosts: localhost
  become: true
  tasks:
    - name: Executing script on master
      script: Jenkins_terraform_ansible.sh

- name: Installations on test
  hosts: test
  become: true
  tasks:
    - name: Executing script on test
      script: K-master.sh
```



```
aws | [Search] | [Alt]
GNU nano 6.2
---
- name: Installations on Master
  hosts: localhost
  become: true
  tasks:
    - name: Executing script on master
      script: Jenkins_terraform_ansible.sh
- name: Installations on test
  hosts: test
  become: true
  tasks:
    - name: Executing script on test
      script: K-master.sh
^G Help      ^O Write Out  ^W Where Is   ^K Cut
^X Exit      ^R Read File  ^\ Replace    ^U Past
i-0a9ecb9339ce2c03d (Jenkins_Terraform_Ansible)
PublicIPs: 54.226.70.54 PrivateIPs: 172.31.28.63
```

Step 5:-

Jenkins_terraform_ansible.sh

sudo apt update

sudo apt install openjdk-17-jre -y

sudo wget -O /usr/share/keyrings/jenkins-keyring.asc \

`https://pkg.jenkins.io/debian-stable/jenkins.io-2023.key`

`echo deb [signed-by=/usr/share/keyrings/jenkins-keyring.asc] \`

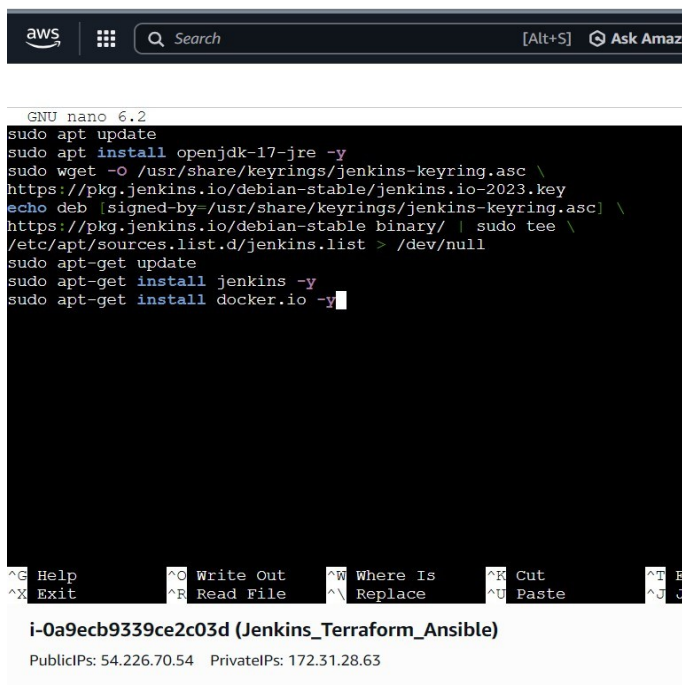
`https://pkg.jenkins.io/debian-stable binary/ | sudo tee \`

`/etc/apt/sources.list.d/jenkins.list > /dev/null sudo`

`apt-get update`

`sudo apt-get install jenkins -y`

`sudo apt-get install docker.io -y`



```
aws | Search [Alt+S] Ask Amazon Q
GNU nano 6.2
sudo apt update
sudo apt install openjdk-17-jre -y
sudo wget -O /usr/share/keyrings/jenkins-keyring.asc \
https://pkg.jenkins.io/debian-stable/jenkins.io-2023.key
echo deb [signed-by=/usr/share/keyrings/jenkins-keyring.asc] \
https://pkg.jenkins.io/debian-stable binary/ | sudo tee \
/etc/apt/sources.list.d/jenkins.list > /dev/null
sudo apt-get update
sudo apt-get install jenkins -y
sudo apt-get install docker.io -y
i-0a9ecb9339ce2c03d (Jenkins_Terraform_Ansible)
PublicIPs: 54.226.70.54 PrivateIPs: 172.31.28.63
```

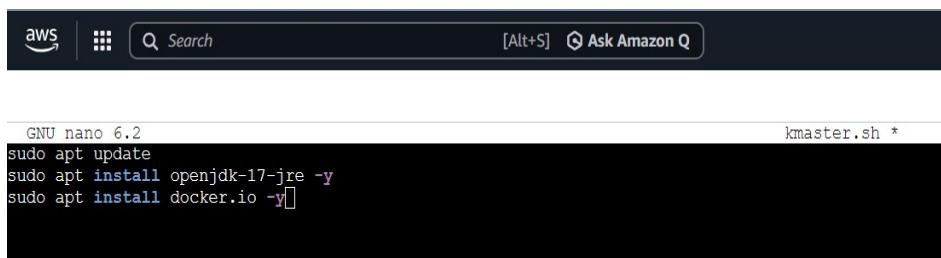
Step 6:-

Kmaster.sh

`sudo apt update`

`sudo apt install openjdk-17-jre -y`

`sudo apt install docker.io -y`



```
aws | Search [Alt+S] Ask Amazon Q
GNU nano 6.2 kmaster.sh *
sudo apt update
sudo apt install openjdk-17-jre -y
sudo apt install docker.io -y
```

Step 7:-

RUN PLAYBOOK :

ansible-playbook play.yaml --syntax-check

ansible-playbook play.yaml --check

ansible-playbook play.yaml

```
ubuntu@ip-172-31-20-201:/etc/ansible$ sudo nano play.yaml
ubuntu@ip-172-31-20-201:/etc/ansible$ ansible-playbook play.yaml

PLAY [Installations on Master] *****

TASK [Gathering Facts] *****
ok: [localhost]

TASK [Executing script on master] *****
changed: [localhost]

PLAY RECAP *****
localhost                : ok=2    changed=1    unreachable=0    failed=0    skipped=0

ubuntu@ip-172-31-20-201:/etc/ansible$
```

i-006ec4ade90e24b2a (jenkins-terra-ansible)

PublicIPs: 54.164.117.172 PrivateIPs: 172.31.20.201

Step 8 :-

Open the Github repository and fork it.

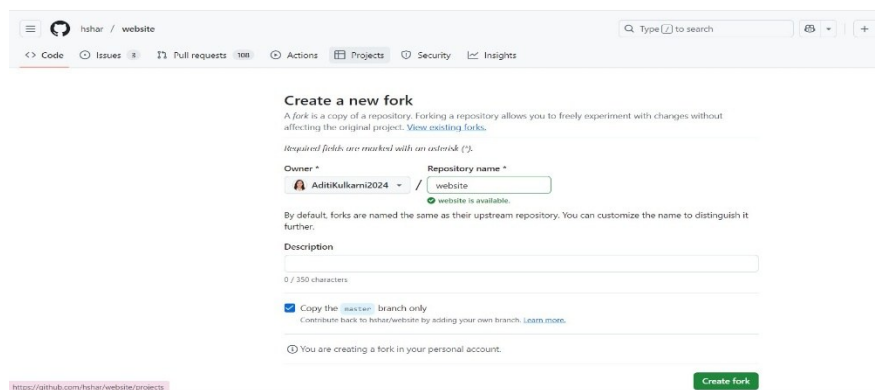
Clone this new repository > Go to folder > Create a Dockerfile

Dockerfile syntax

FROM ubuntu

RUN apt update

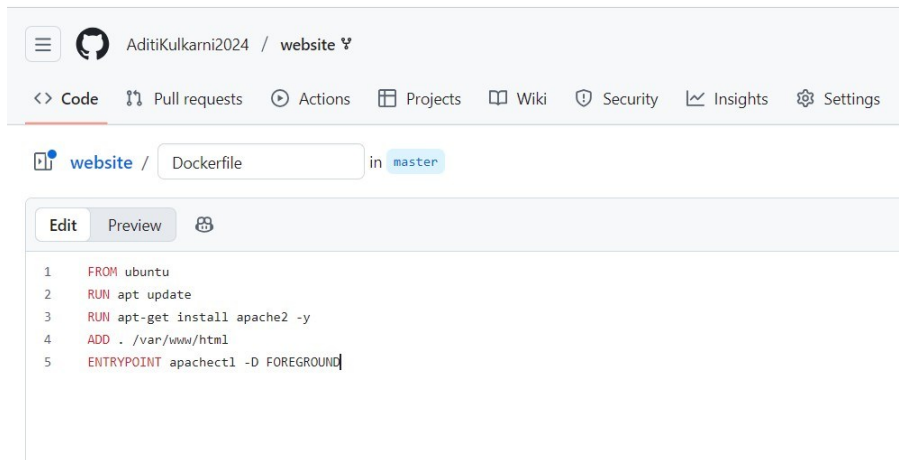
RUN apt-get install apache2 -y



The screenshot shows the GitHub interface for creating a new fork of the repository 'hahar / website'. The page title is 'Create a new fork'. Below the title, there is a description: 'A fork is a copy of a repository. Forking a repository allows you to freely experiment with changes without affecting the original project. View existing forks.' There is a note: 'Required fields are marked with an asterisk (*)'. The 'Owner' field is set to 'AditiKulkarni2024'. The 'Repository name' field is set to 'website', with a green checkmark indicating 'website is available'. Below this, there is a 'Description' field with a character count '0 / 350 characters'. There is a checkbox 'Copy the master branch only' which is checked, with a link 'Learn more.' below it. At the bottom, there is a note 'You are creating a fork in your personal account.' and a green 'Create fork' button. The URL at the bottom is 'https://github.com/hahar/website/projects'.

ADD ./var/www/html

ENTRYPOINT apachectl -D FOREGROUND



The screenshot shows a GitHub repository interface for a user named AditiKulkarni2024. The repository is named 'website'. The file 'Dockerfile' is selected, and the 'master' branch is active. The Dockerfile content is as follows:

```
1 FROM ubuntu
2 RUN apt update
3 RUN apt-get install apache2 -y
4 ADD ./var/www/html
5 ENTRYPOINT apachectl -D FOREGROUND
```

Step 9:- Create deploy.yml file

apiVersion: apps/v1

kind: Deployment

metadata:

name: custom-deployment

labels:

app: custom

spec:

replicas: 2

selector:

matchLabels:

app: custom

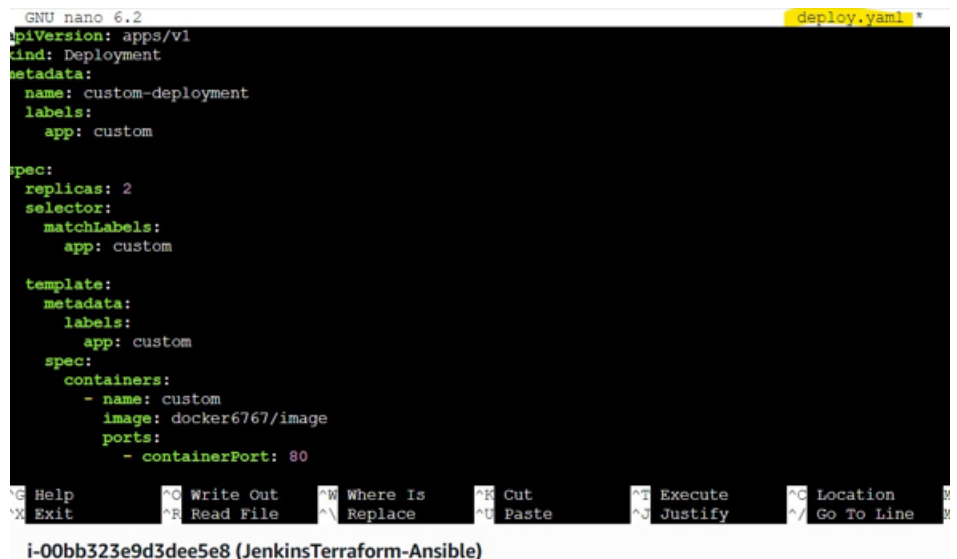
template:

metadata:

labels:

app: custom

spec:



The screenshot shows a terminal window with the GNU nano 6.2 editor. The file being edited is 'deploy.yml'. The content of the file is as follows:

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: custom-deployment
  labels:
    app: custom
spec:
  replicas: 2
  selector:
    matchLabels:
      app: custom
  template:
    metadata:
      labels:
        app: custom
    spec:
      containers:
        - name: custom
          image: docker6767/image
          ports:
            - containerPort: 80
```

The terminal window also shows a prompt at the bottom: 'i-00bb323e9d3dee5e8 (JenkinsTerraform-Ansible)'.

containers:

- name: custom

image: docker6767/image

ports:

- containerPort: 80

Create SVC.yml

apiVersion: v1

kind: Service

metadata:

name: my-custom-deployment

spec:

type: NodePort

ports:

- targetPort: 80

port: 80

nodePort: 30008

selector:

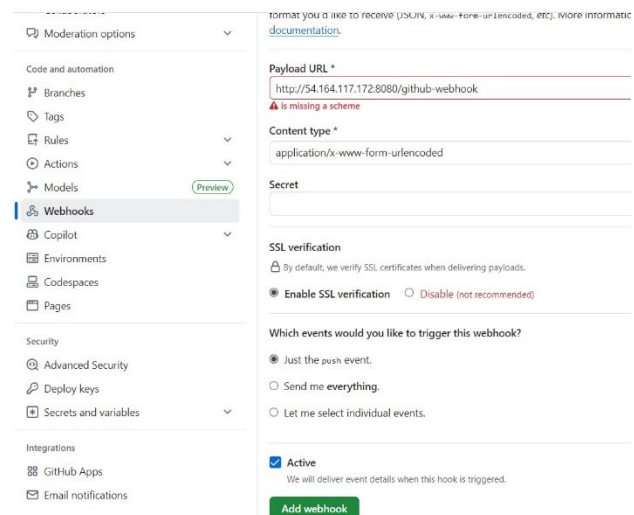
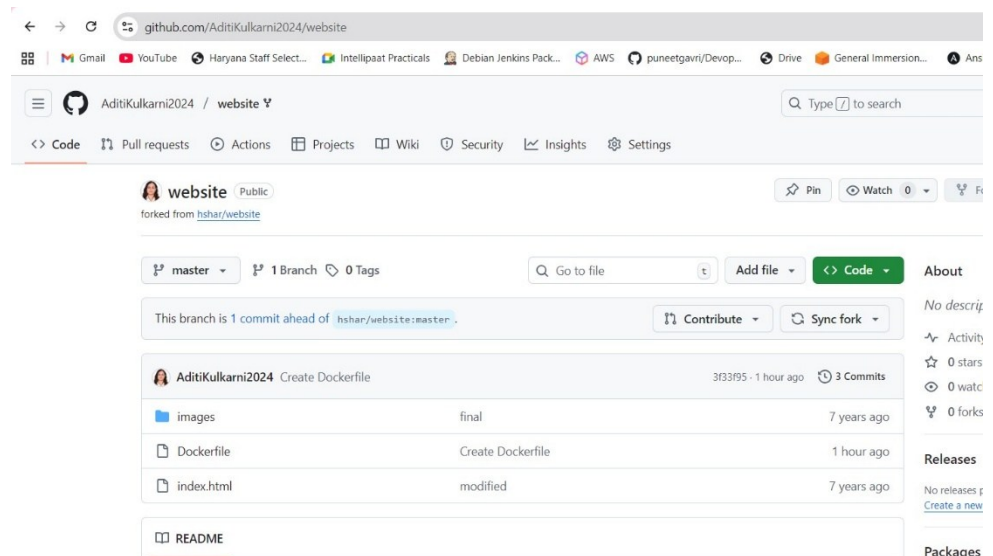
app: custom

Git status

Git add .

Git commit -m "add dockerfile"

Git branch > you will get a master branch



Step 10:-

Create Nodes-

← → ↻ Not secure 54.164.117.172:8080/manage/computer/ ☆

Gmail YouTube Haryana Staff Select... Intelliptat Practicals Debian Jenkins Pack... AWS puneetgavri/Devop... Drive General Immersion... Ansible Document >>

Jenkins / Manage Jenkins ▾ / Nodes 🔍

Nodes

[+ New Node](#) [Configure Monitors](#) [↻](#)

S	Name ↓	Architecture	Clock Difference	Free Disk Space	Free Swap Space	Free Temp Space	Response Time
	Built-In Node	Linux (amd64)	In sync	3.24 GiB	0 B	3.24 GiB	0ms
	slave	Linux (amd64)	In sync	3.24 GiB	0 B	3.24 GiB	97ms
Data obtained		0.44 sec	0.44 sec	0.35 sec	0.35 sec	0.36 sec	0.35 sec

Icon: S M **L** Legend

Create a Pipeline job

Jenkins / project-job ▾ / Configuration

Configure

- General
- Triggers**
- Pipeline
- Advanced

☒ GitHub hook trigger for GITScm polling ?
☐ Poll SCM ?
☐ Trigger builds remotely (e.g., from scripts) ?

Pipeline
Define your Pipeline using Groovy directly or pull it from source control.

Definition

Pipeline script

```
Script ?
1 pipeline {
2   agent none
3
4   environment {
5     // Make sure the credentials ID is correct and doesn't have line breaks
6     DOCKERHUB_CREDENTIALS = credentials('a33c448b-a34e-42f2-b98d-e9e7e890a018')
7   }
8
9   stages {
```

[Save](#) [Apply](#)

```
pipeline{
  agent none
  environment {

    DOCKERHUB_CREDENTIALS=credentials('

  }

  stages{
    stage('Hello'){
      agent{

        label 'KMaster'

      }

      steps{

        echo 'Hello World'

      }

    }

    stage('git'){
      agent{
        label 'KMaster'

      }

      steps{

        git'URL'
```

```

    }
}

stage('docker') {
    agent {
        label 'KMaster'
    }
    steps {

```

```

        sh 'sudo docker build /home/ubuntu/jenkins/workspace/FinalProject -t
docker6767/image'

```

```

        sh 'sudo echo $DOCKERHUB_CREDENTIALS_PSW | sudo docker login -u
$DOCKERHUB_CREDENTIALS_USR --password-stdin'

```

```

        sh 'sudo docker push docker6767/image'

```

```

    }
}

stage('Kubernetes') {

    agent {

        label 'KMaster'

    }


    steps {

```

```

        sh 'kubectl create -f deploy.yml'

```


Jenkins / project-job

Status

[Changes](#)
[Build Now](#)
[Configure](#)
[Delete Pipeline](#)
[GitHub](#)
[Stages](#)
[Rename](#)
[Pipeline Syntax](#)
[GitHub Hook Log](#)

project-job

Permalinks

Builds

Today

#2 6:34 pm

#1 6:31 pm

```
sh 'kubectl create -f svc.yml'
```

```
}
```

```
}
```

```
}
```

```
}
```

```
aws [Search] [Alt+S] Ask Amazon Q United States (N. Virginia) Account ID: 1791:

Apply complete! Resources: 3 added, 0 changed, 0 destroyed.
ubuntu@ip-172-31-16-21:~$ history
 1 sudo apt update -y
 2 sudo apt-get update && sudo apt-get install -y gnupg software-properties-common
 3 wget -O- https://apt.releases.hashicorp.com/gpg | gpg --dearmor | sudo tee /usr/share/keyrings/hashicorp-archive-keyring.gpg > /dev/null
 4 gpg --no-default-keyring --keyring /usr/share/keyrings/hashicorp-archive-keyring.gpg --fingerprint
 5 echo "deb [arch=$(dpkg --print-architecture) signed-by=/usr/share/keyrings/hashicorp-archive-keyring.gpg] https://apt.releases.hashicorp.com $(grep -oP '
U_CODENAME=)' /etc/os-release || lsb_release -cs) main" | sudo tee /etc/apt/sources.list.d/hashicorp.list
 6 sudo apt update
 7 sudo apt-get install terraform
 8 terraform --version
 9 sudo nanomain.tf
10 sudo nano main.tf
11 terraform init
12 terraform plan
13 sudo nano main.tf
14 terraform init
15 terraform plan
16 sudo nano main.tf
17 terraform init
18 terraform plan
19 sudo nano main.tf
20 terraform init
21 terraform plan
22 terraform apply
23 history

i-0eb1c0ac734bf93c8 (Jenkins-Terraform-Ansible)
PublicIPs: 54.208.117.223 PrivateIPs: 172.31.16.21
```