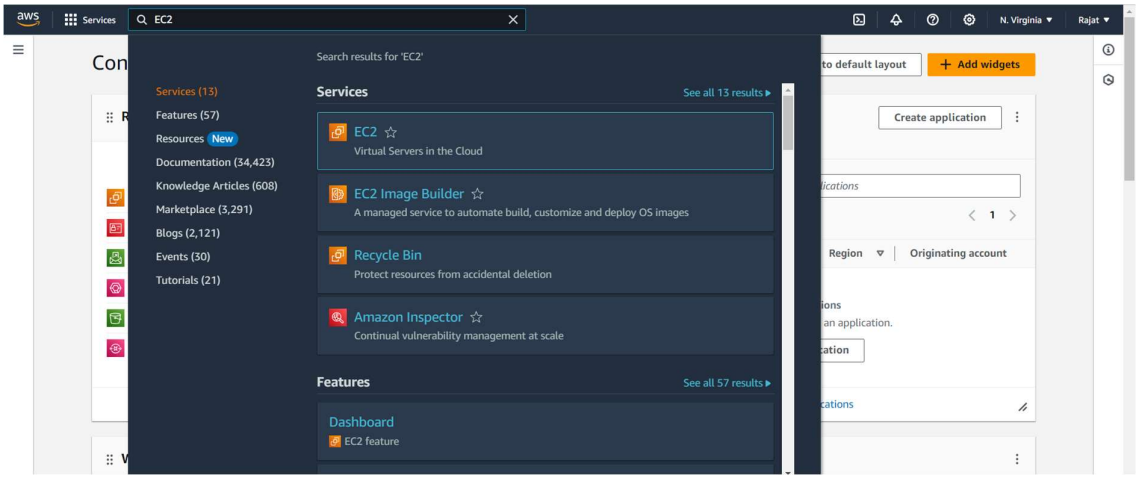
Up until now, this organization used to follow a monolithic architecture with just 2 developers. The product is present on: https://github.com/hshar/website.git

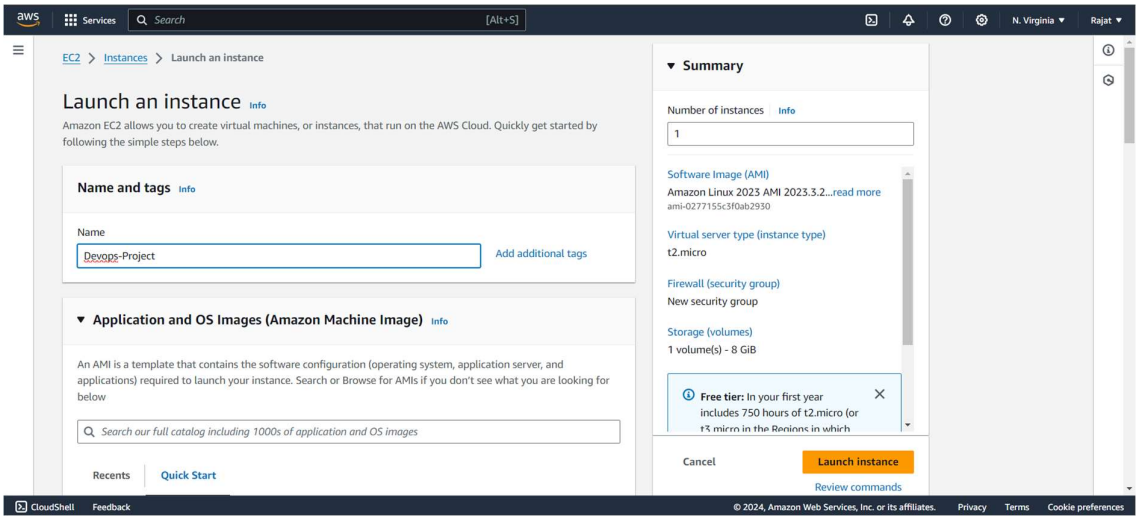
Following are the specifications of the lifecycle:

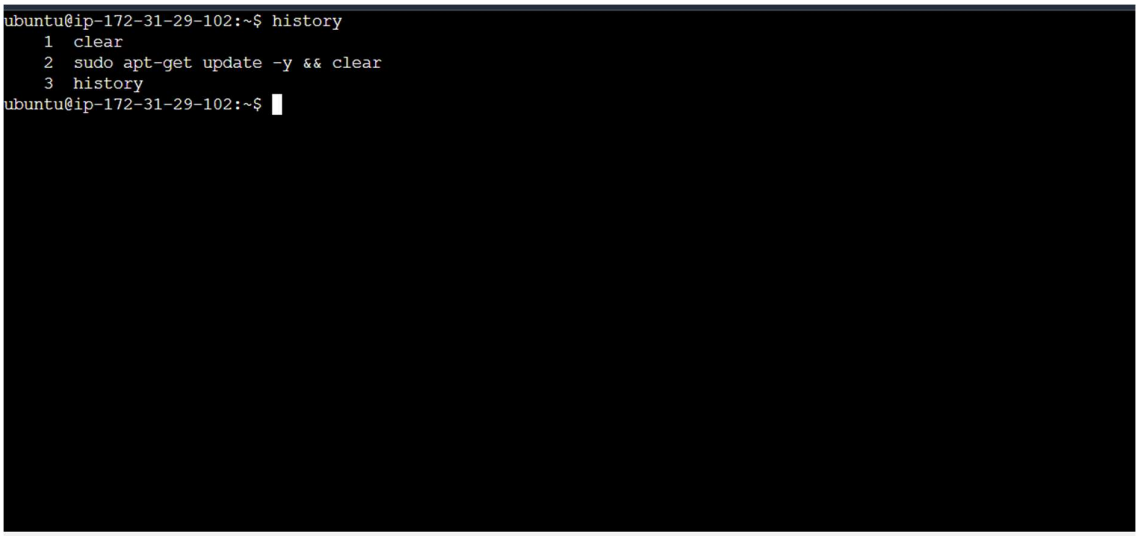
1. Git workflow should be implemented. Since the company follows a monolithic architecture of development, you need to take care of version control. The release should happen only on the 25th of every month.
2. CodeBuild should be triggered once the commits are made in the master branch.
3. The code should be containerized with the help of the Dockerfile. The Dockerfile should be built every time if there is a push to GitHub. Create a custom Docker image using a Dockerfile.
4. As per the requirement in the production server, you need to use the Kubernetes cluster and the containerized code from Docker Hub should be deployed with 2 replicas. Create a NodePort service and configure the same for port 30008.
5. Create a Jenkins Pipeline script to accomplish the above task.
6. For configuration management of the infrastructure, you need to deploy the configuration on the servers to install necessary software and configurations.
7. Using Terraform, accomplish the task of infrastructure creation in the AWS cloud provider.

Architectural Advice:
Softwares to be installed on the respective machines using configuration management.

Worker1: Jenkins, Java







i-08423512d6ebaa351 (Devops-Project)

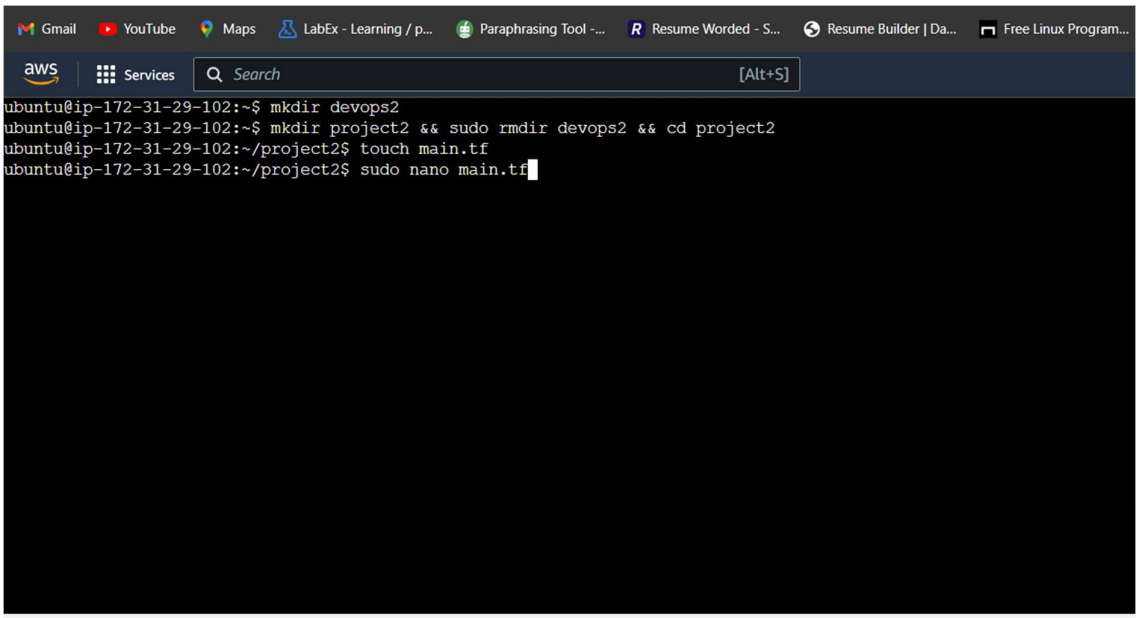
Install Using Script

```
sudo wget https://raw.githubusercontent.com/lerndevops/labs/master/scripts/installTerraform.sh -P /tmp
sudo chmod 755 /tmp/installTerraform.sh
sudo bash /tmp/installTerraform.sh
```

Manual Installation Steps

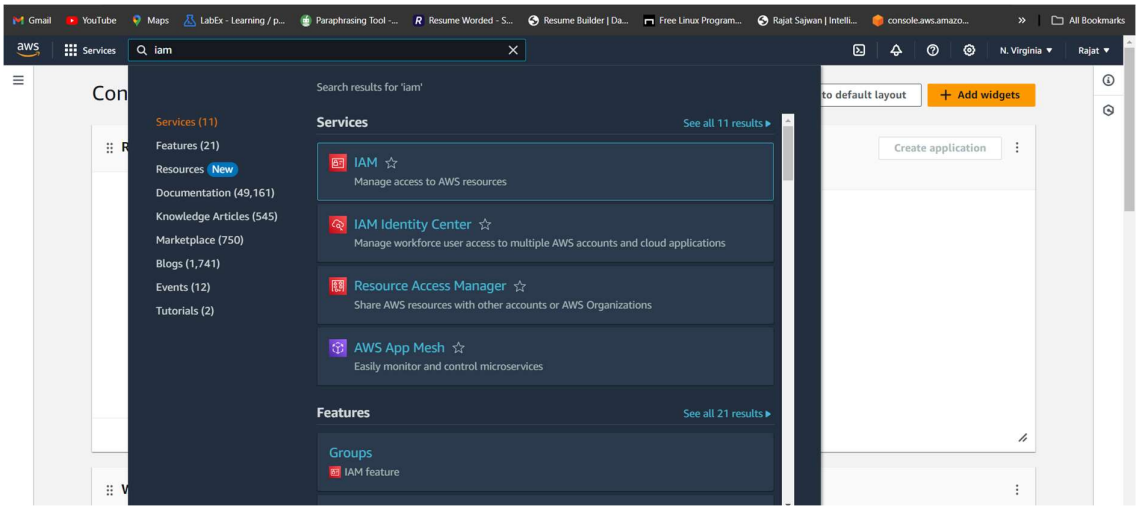
i-08423512d6ebaa351 (Devops-Project)

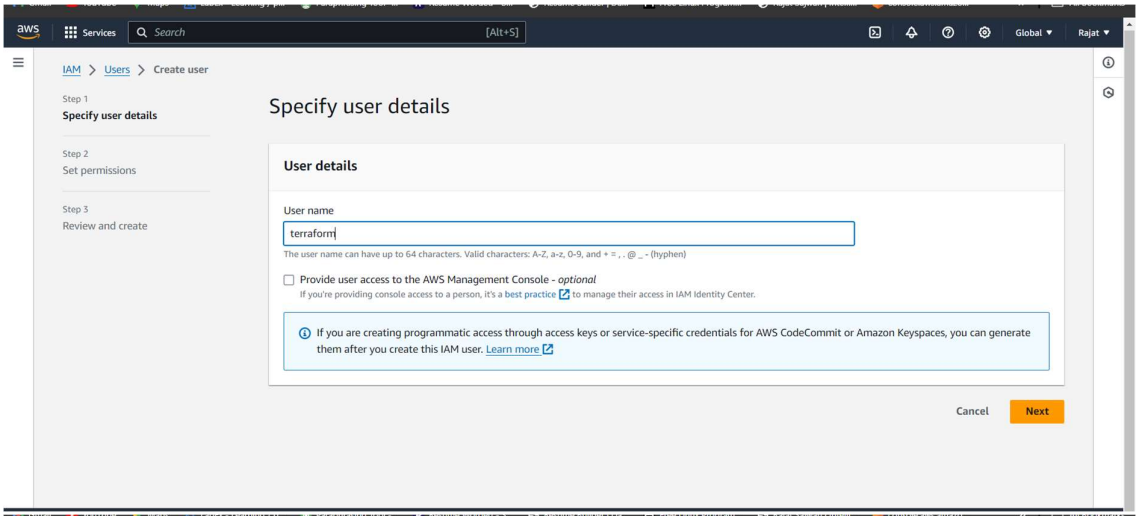
PublicIPs: 34.230.29.92 PrivateIPs: 172.31.29.102

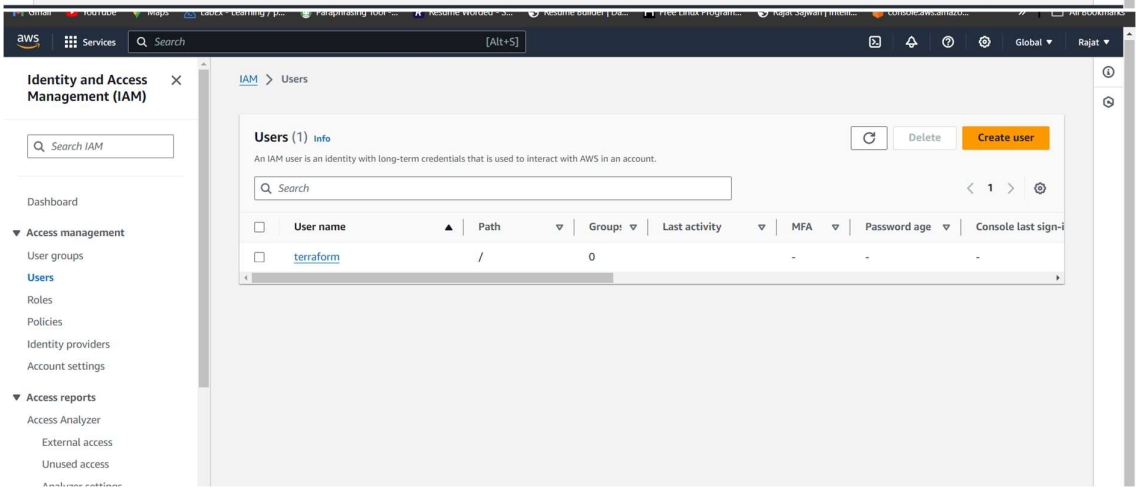


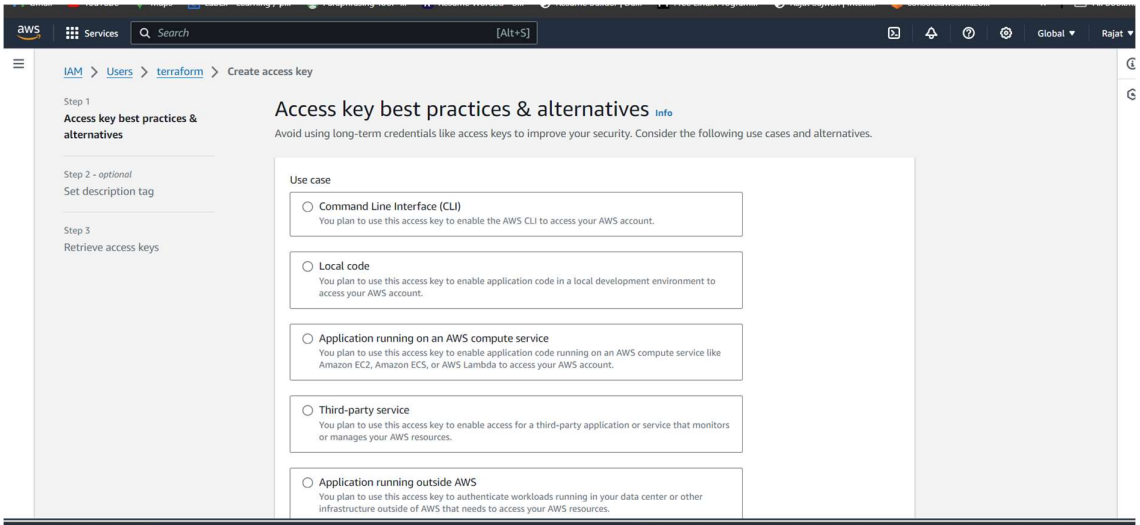
i-08423512d6ebaa351 (Devops-Project)

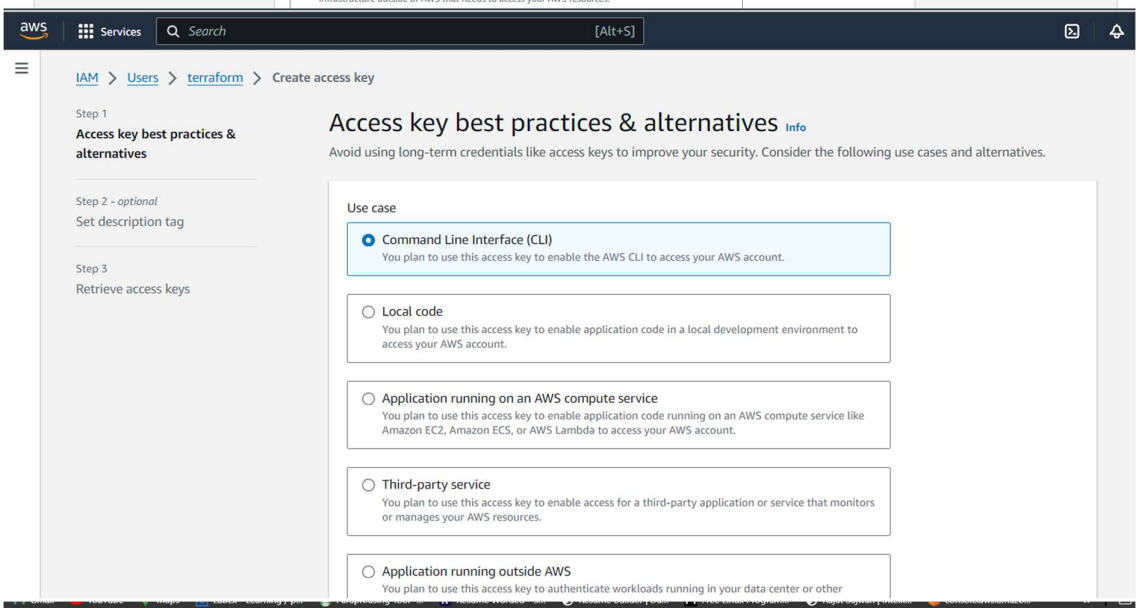
PublicIPs: 34.230.29.92 PrivateIPs: 172.31.29.102

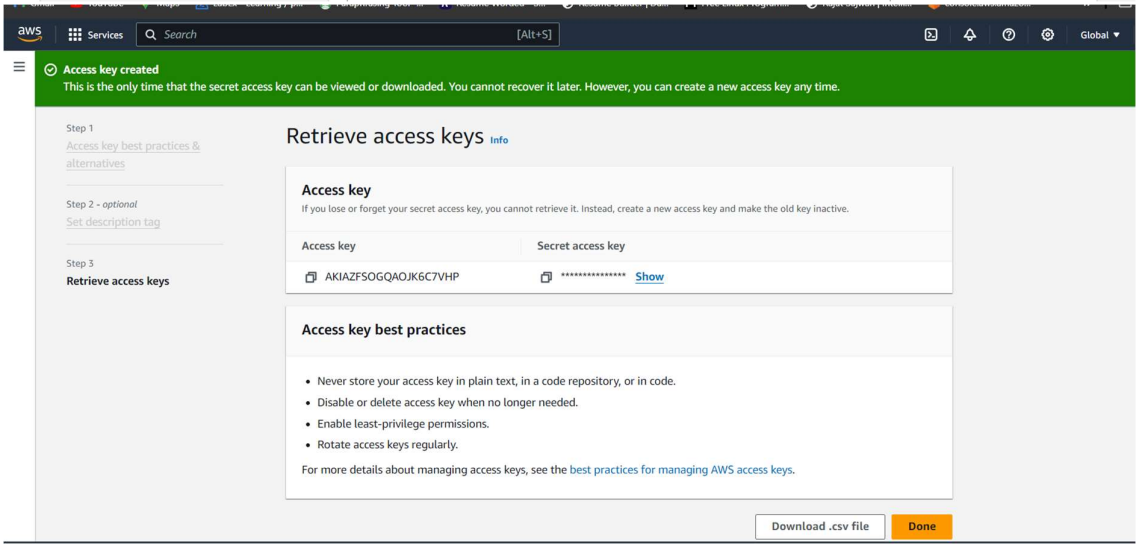

Gmail
YouTube
Maps
LabEx - Learning / p...
Paraphrasing Tool -...
Resume Worded - S...
Resume Builder | Da...
Free Linux Program...
Rajat Sajwan | Intelli...
console.aws.amazo...
All Bookmarks
aws
Services
iam
N. Virginia
Rajat
Search results for 'iam'
Services (11)
Features (21)
Resources New
Documentation (49,161)
Knowledge Articles (545)
Marketplace (750)
Blogs (1,741)
Events (12)
Tutorials (2)
Services
See all 11 results
IAM
Manage access to AWS resources
IAM Identity Center
Manage workforce user access to multiple AWS accounts and cloud applications
Resource Access Manager
Share AWS resources with other accounts or AWS Organizations
AWS App Mesh
Easily monitor and control microservices
Features
See all 21 results
Groups
IAM feature
to default layout
Add widgets
Create application

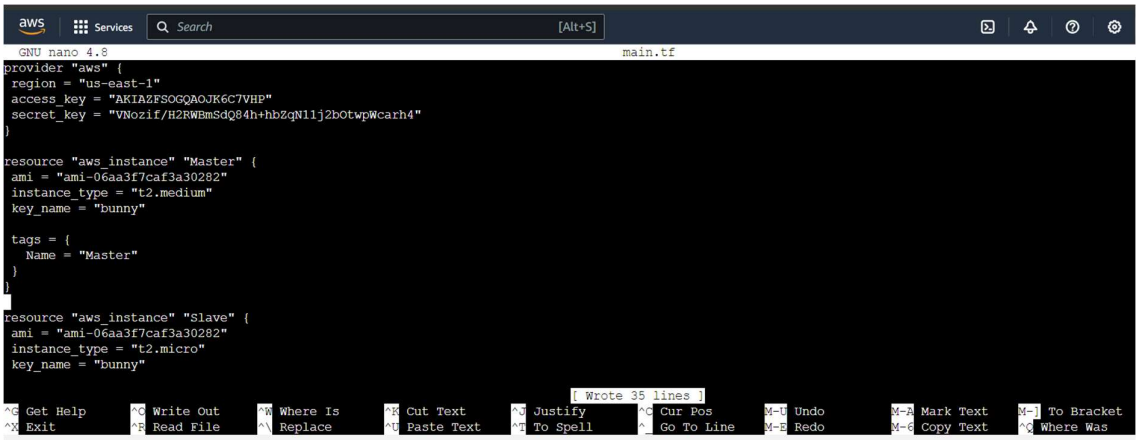

aws
Services
Search
[Alt+S]
Global
Rajat
IAM > Users > Create user
Step 1
Specify user details
Step 2
Set permissions
Step 3
Review and create
Specify user details
User details
User name
terraform
The user name can have up to 64 characters. Valid characters: A-Z, a-z, 0-9, and + = , . @ _ - (hyphen)
Provide user access to the AWS Management Console - optional
If you're providing console access to a person, it's a best practice to manage their access in IAM Identity Center.
If you are creating programmatic access through access keys or service-specific credentials for AWS CodeCommit or Amazon Keyspaces, you can generate them after you create this IAM user. Learn more
Cancel
Next


aws
Services
Search
[Alt+S]
Global
Rajat
Identity and Access Management (IAM)
Search IAM
Dashboard
Access management
User groups
Users
Roles
Policies
Identity providers
Account settings
Access reports
Access Analyzer
External access
Unused access
IAM > Users
Users (1) Info
An IAM user is an identity with long-term credentials that is used to interact with AWS in an account.
Delete
Create user
Search
User name
Path
Groups
Last activity
MFA
Password age
Console last sign-i
terraform
/
0
-
-
-


aws
Services
Search
[Alt+S]
Global
Rajat
IAM > Users > terraform > Create access key
Step 1
Access key best practices & alternatives
Step 2 - optional
Set description tag
Step 3
Retrieve access keys
Access key best practices & alternatives Info
Avoid using long-term credentials like access keys to improve your security. Consider the following use cases and alternatives.
Use case
Command Line Interface (CLI)
You plan to use this access key to enable the AWS CLI to access your AWS account.
Local code
You plan to use this access key to enable application code in a local development environment to access your 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.

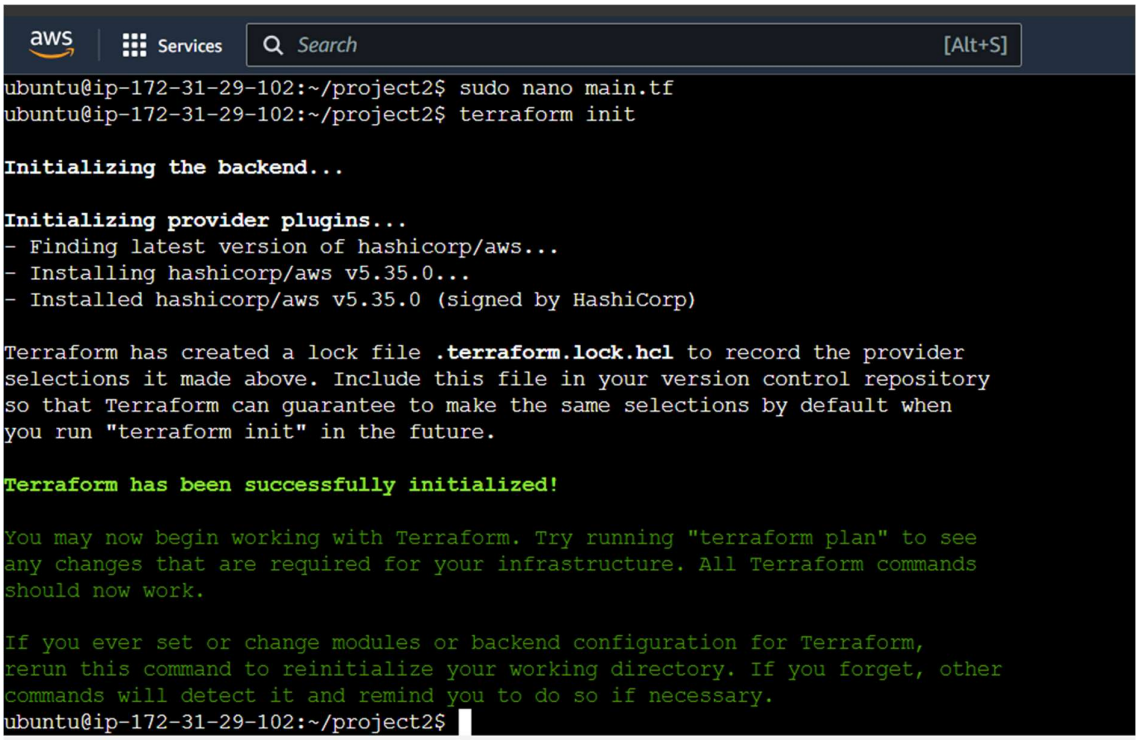

aws
Services
Search
[Alt+S]
IAM > Users > terraform > Create access key
Step 1
Access key best practices & alternatives
Step 2 - optional
Set description tag
Step 3
Retrieve access keys
Access key best practices & alternatives Info
Avoid using long-term credentials like access keys to improve your security. Consider the following use cases and alternatives.
Use case
Command Line Interface (CLI)
You plan to use this access key to enable the AWS CLI to access your AWS account.
Local code
You plan to use this access key to enable application code in a local development environment to access your 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


aws
Services
Search
[Alt+S]
Global
Access key created
This is the only time that the secret access key can be viewed or downloaded. You cannot recover it later. However, you can create a new access key any time.
Step 1
Access key best practices & alternatives
Step 2 - optional
Set description tag
Step 3
Retrieve access keys
Retrieve access keys Info
Access key
If you lose or forget your secret access key, you cannot retrieve it. Instead, create a new access key and make the old key inactive.
Access key
Secret access key
AKIAZFSOGQAOJK6C7VHP
********** Show
Access key best practices
Never store your access key in plain text, in a code repository, or in code.
Disable or delete access key when no longer needed.
Enable least-privilege permissions.
Rotate access keys regularly.
For more details about managing access keys, see the best practices for managing AWS access keys.
Download .csv file
Done



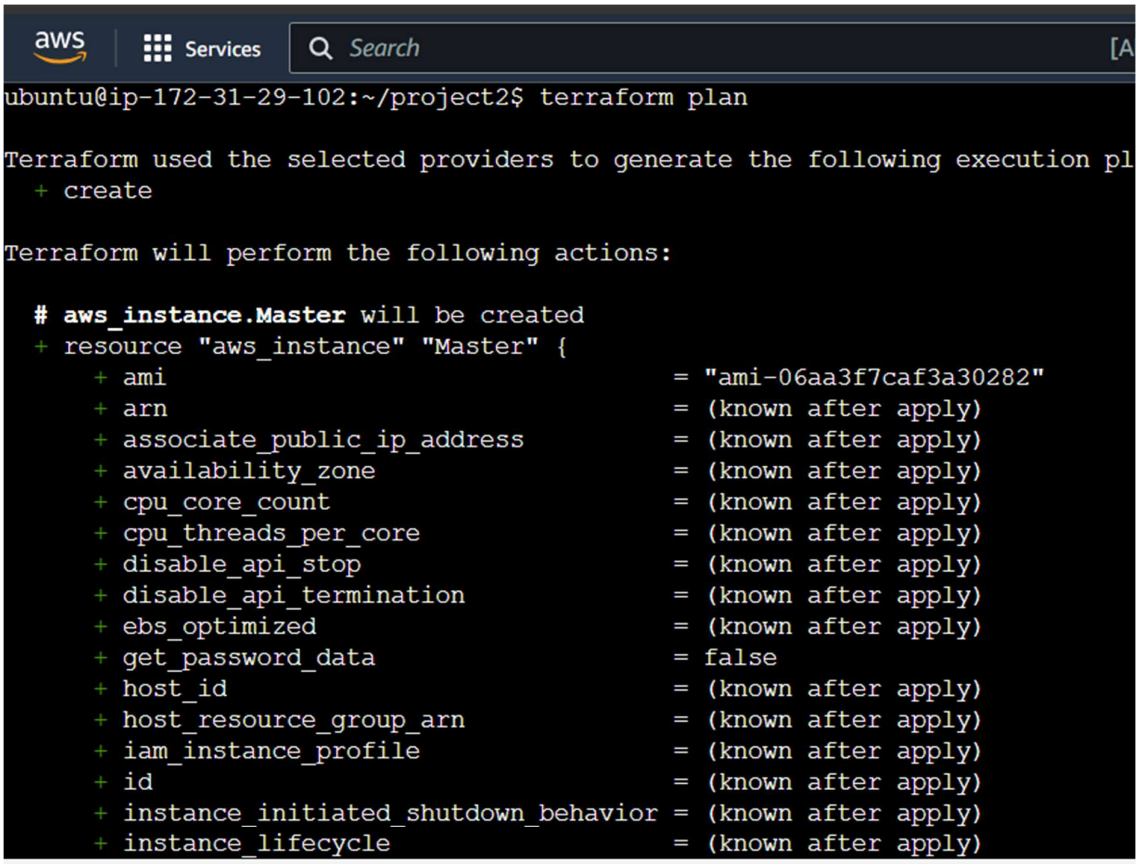
i-08423512d6ebaa351 (Devops-Project)

PublicIPs: 34.230.29.92 PrivateIPs: 172.31.29.102



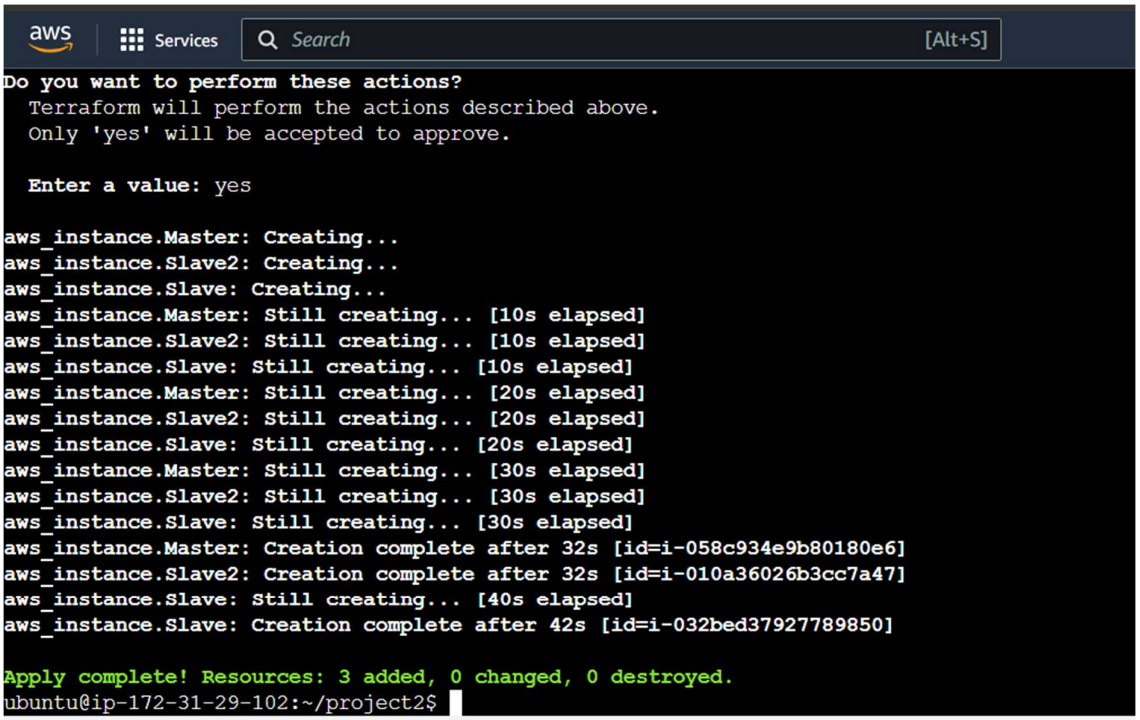
i-08423512d6ebaa351 (Devops-Project)

PublicIPs: 34.230.29.92 PrivateIPs: 172.31.29.102



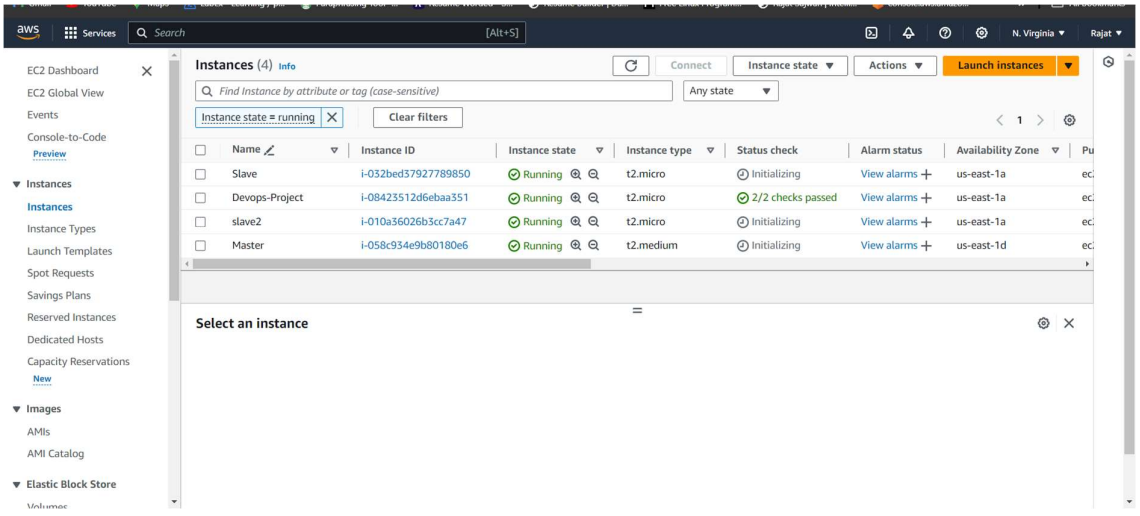
i-08423512d6ebaa351 (Devops-Project)

PublicIPs: 34.230.29.92 PrivateIPs: 172.31.29.102



i-08423512d6ebaa351 (Devops-Project)

PublicIPs: 34.230.29.92 PrivateIPs: 172.31.29.102



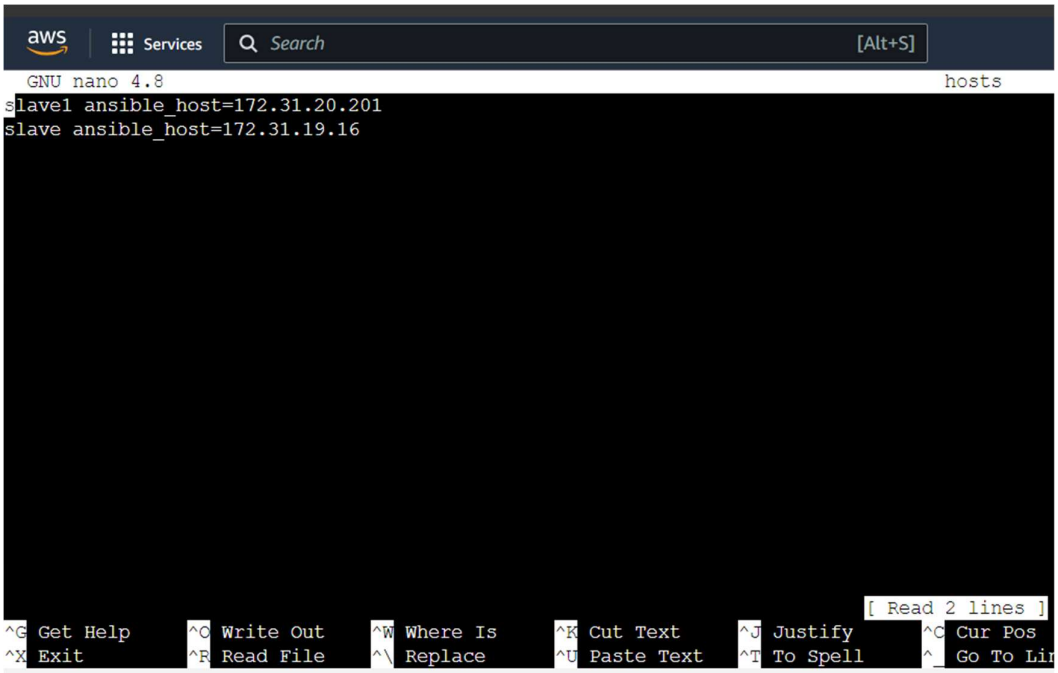

```
$ sudo apt update
$ sudo apt install software-properties-common
$ sudo add-apt-repository --yes --update ppa:ansible/ansible
$ sudo apt install ansible
```

```
ubuntu@ip-172-31-95-176:~$ history
    1  clear
    2  sudo apt update -y
    3  clear
    4  ssh-keygen
    5  cd .ssh/
    6  ls
    7  cat id_rsa.pub
    8  clear
    9  ansible -version
   10  sudo apt install software-properties-common
   11  sudo add-apt-repository --yes --update ppa:ansible/ansib
   12  sudo apt install ansible && ansible -version
   13  sudo apt install ansible -y
   14  ansible -version
   15  clear
   16  ansible --version
   17  ssh-keygen
   18  sudo cat /home/ubuntu/.ssh/id_rsa
   19  clear
   20  ssh-keygen
   21  sudo cat /home/ubuntu/.ssh/id_rsa
   22  sudo cat /home/ubuntu/.ssh/id_rsa.pub
   23  history
ubuntu@ip-172-31-95-176:~$
```

i-058c934e9b80180e6 (Master)

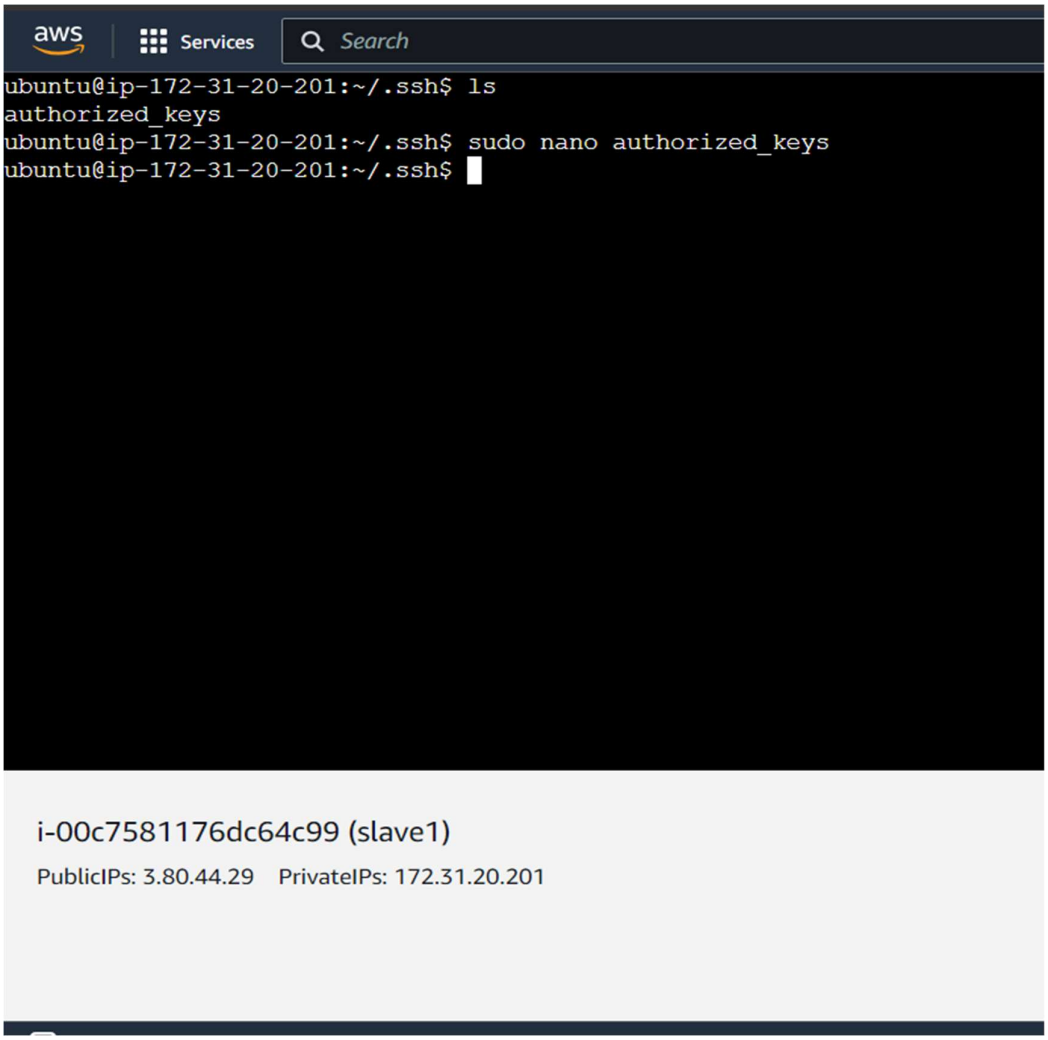
PublicIPs: 54.86.205.38 PrivateIPs: 172.31.95.176

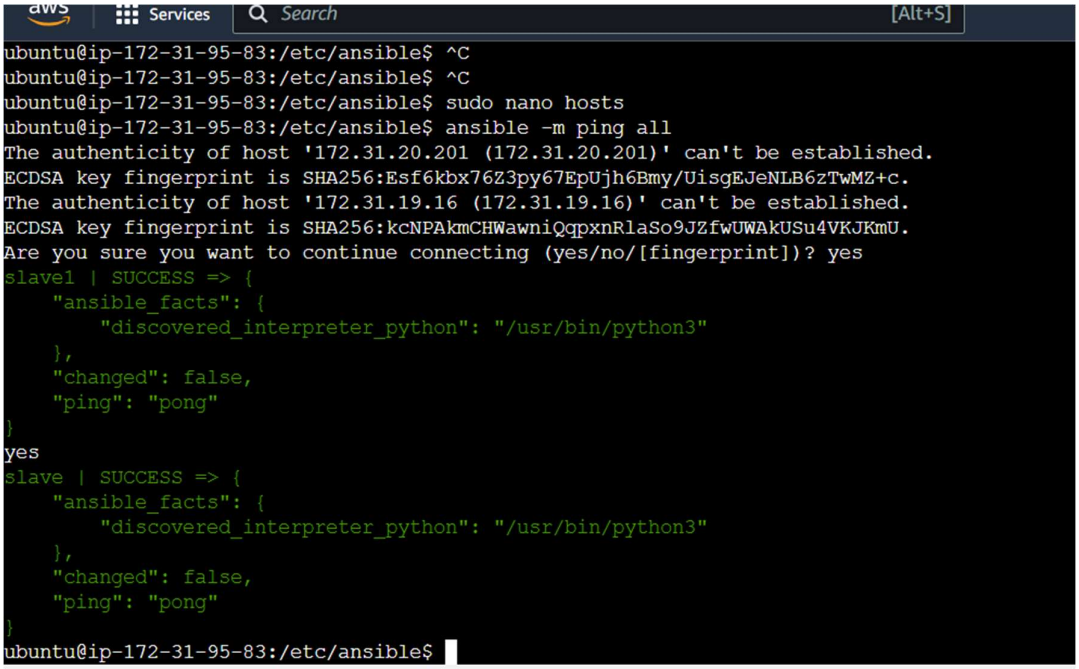
```
Nje2g30TmVcRPdAAAAF3VidW50dUBpcC0xNzItMzEtOTUtMTc2AQID
-----END OPENSSH PRIVATE KEY-----
ubuntu@ip-172-31-95-176:~$ sudo cat /home/ubuntu/.ssh/id_rsa.pub
ssh-rsa AAAAB3Nzac1yc2EAAAADAQABAAABgQDNY2d4v3se1CCCtZd3r2MkDHosVZyulEZsP8zgtxHtYG1ljQfXSOXhgE9xiA6hNWCjFDZiHre8MRetIFr4FtK6tiLRQEPN8BsUVUcnYUg+F6ojSuV4VgW2dsbzNzdOAz
+yZHGQUZagbTHfSSOxmAPdEC8Ku7pxRLgWyCDlaNPsRdSHlo45kuXTnNQq3ls6l+l5q2LZBEFd2DlljqiHEZSlJquH7HACWKzKPfdYVOIamU+rrKoMSojPrLThuqP/SAnQzQChFsJtlwfKUq5IHxe7m5xF2+f6SoXP9eIs
5MU//c5A2OEG85DbKMBZROHA5RTlFycSucX+piEKxilgxFdZ+bviZFcq4Hts4XCSb9tL7gT+772eFCZcWsmGaY400FsqV9VgbiC0SotWwYXFxyuSU/FhgI+n5rmr0w0VXJqYEq+S7QJANZiZF+1FivA0/ebjzaLnliUEnA
7nj74jeNgzItI3Xq6YKgrJJMpelJj3ObxywqjNqaiwlAf78wGguP0= ubuntu@ip-172-31-95-176
ubuntu@ip-172-31-95-176:~$ history
    1  clear
```



i-0193e197b29689058 (Master)

PublicIPs: 44.202.237.175 PrivateIPs: 172.31.95.83


aws
Services
Search
ubuntu@ip-172-31-20-201:~/.ssh$ ls
authorized_keys
ubuntu@ip-172-31-20-201:~/.ssh$ sudo nano authorized_keys
ubuntu@ip-172-31-20-201:~/.ssh$
i-00c7581176dc64c99 (slave1)
PublicIPs: 3.80.44.29 PrivateIPs: 172.31.20.201

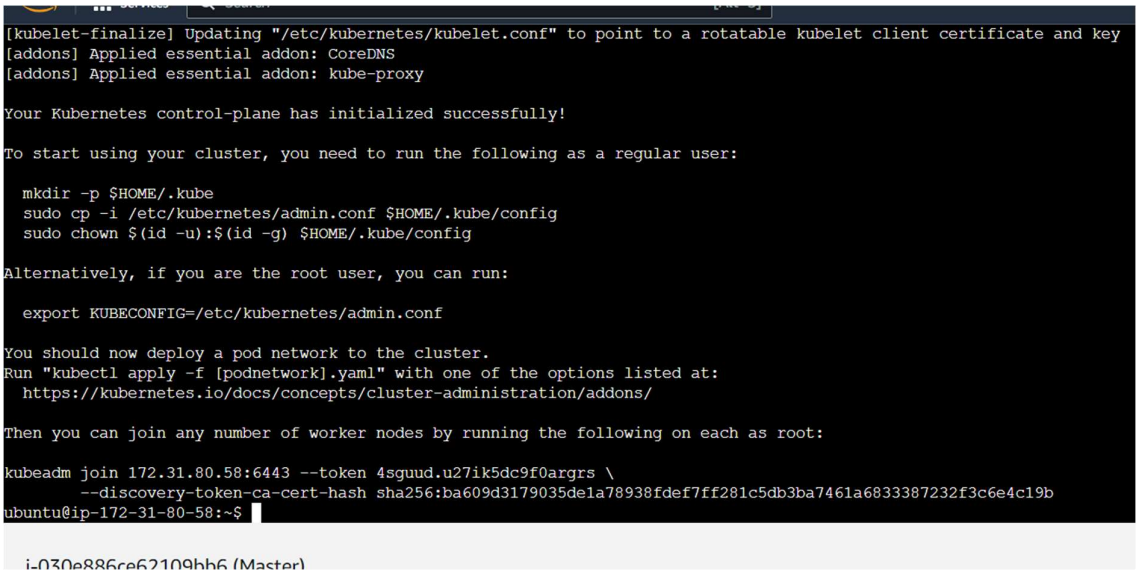



```
ubuntu@ip-172-31-80-58:~$ history
    1  sudo apt-get update -y && clear
    2  sudo wget https://raw.githubusercontent.com/lerndevops/labs/master/scripts/installDocker.sh -P /tmp
    3  sudo chmod 755 /tmp/installDocker.sh
    4  sudo bash /tmp/installDocker.sh
    5  sudo systemctl restart docker.service
    6  sudo wget https://raw.githubusercontent.com/lerndevops/labs/master/scripts/installCRIDockerd.sh -P /tmp
    7  sudo chmod 755 /tmp/installCRIDockerd.sh
    8  sudo bash /tmp/installCRIDockerd.sh
    9  sudo systemctl restart cri-docker.service && clear
   10  sudo wget https://raw.githubusercontent.com/lerndevops/labs/master/scripts/installK8S.sh -P /tmp
   11  sudo chmod 755 /tmp/installK8S.sh
   12  sudo bash /tmp/installK8S.sh
   13  clear
   14  cri-dockerd --version
   15  kubeadm version -o short
   16  clear
   17  history
ubuntu@ip-172-31-80-58:~$
```

```
ubuntu@ip-172-31-80-58:~$  docker -v
Docker version 25.0.3, build 4debf41
ubuntu@ip-172-31-80-58:~$ cri-dockerd --version
cri-dockerd 0.3.4 (e88b1605)
ubuntu@ip-172-31-80-58:~$ kubeadm version -o short
v1.29.1
ubuntu@ip-172-31-80-58:~$  kubelet --version
Kubernetes v1.29.1
ubuntu@ip-172-31-80-58:~$   kubectl version --client
Client Version: v1.29.1
Kustomize Version: v5.0.4-0.20230601165947-6ce0bf390ce3
ubuntu@ip-172-31-80-58:~$
```

i-030e886ce62109bb6 (Master)

```
ubuntu@ip-172-31-80-58:~$  sudo kubeadm init --cri-socket unix:///var/run/cri-dockerd.sock --ignore-preflight-errors=all
[init] Using Kubernetes version: v1.29.1
[preflight] Running pre-flight checks
[preflight] Pulling images required for setting up a Kubernetes cluster
[preflight] This might take a minute or two, depending on the speed of your internet connection
[preflight] You can also perform this action in beforehand using 'kubeadm config images pull'
W0212 13:05:30.079023    6079 checks.go:835] detected that the sandbox image "registry.k8s.io/pause:3.6" of the container runtime is inconsistent with tha
beadm. It is recommended that using "registry.k8s.io/pause:3.9" as the CRI sandbox image.
[certs] Using certificateDir folder "/etc/kubernetes/pki"
[certs] Generating "ca" certificate and key
[certs] Generating "apiserver" certificate and key
[certs] apiserver serving cert is signed for DNS names [ip-172-31-80-58 kubernetes kubernetes.default kubernetes.default.svc kubernetes.default.svc.cluste
 IPs [10.96.0.1 172.31.80.58]
[certs] Generating "apiserver-kubelet-client" certificate and key
[certs] Generating "front-proxy-ca" certificate and key
[certs] Generating "front-proxy-client" certificate and key
[certs] Generating "etcd/ca" certificate and key
[certs] Generating "etcd/server" certificate and key
[certs] etcd/server serving cert is signed for DNS names [ip-172-31-80-58 localhost] and IPs [172.31.80.58 127.0.0.1 ::1]
[certs] Generating "etcd/peer" certificate and key
[certs] etcd/peer serving cert is signed for DNS names [ip-172-31-80-58 localhost] and IPs [172.31.80.58 127.0.0.1 ::1]
[certs] Generating "etcd/healthcheck-client" certificate and key
[certs] Generating "apiserver-etcd-client" certificate and key
[certs] Generating "sa" key and public key
[kubeconfig] Using kubeconfig folder "/etc/kubernetes"
```


[kubelet-finalize] Updating "/etc/kubernetes/kubelet.conf" to point to a rotatable kubelet client certificate and key
[addons] Applied essential addon: CoreDNS
[addons] Applied essential addon: kube-proxy
Your Kubernetes control-plane has initialized successfully!
To start using your cluster, you need to run the following as a regular user:
mkdir -p $HOME/.kube
sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
sudo chown $(id -u):$(id -g) $HOME/.kube/config
Alternatively, if you are the root user, you can run:
export KUBECONFIG=/etc/kubernetes/admin.conf
You should now deploy a pod network to the cluster.
Run "kubectl apply -f [podnetwork].yaml" with one of the options listed at:
https://kubernetes.io/docs/concepts/cluster-administration/addons/
Then you can join any number of worker nodes by running the following on each as root:
kubeadm join 172.31.80.58:6443 --token 4sguud.u27ik5dc9f0argrs \
--discovery-token-ca-cert-hash sha256:ba609d3179035de1a78938fdef7ff281c5db3ba7461a6833387232f3c6e4c19b
ubuntu@ip-172-31-80-58:~$
i-030e886ce62109bb6 (Master)

```
ubuntu@ip-172-31-80-58:~$ sudo mkdir -p $HOME/.kube
ubuntu@ip-172-31-80-58:~$ sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
ubuntu@ip-172-31-80-58:~$ sudo chown $(id -u):$(id -g) $HOME/.kube/config
ubuntu@ip-172-31-80-58:~$ kubectl get node
NAME              STATUS     ROLES           AGE    VERSION
ip-172-31-80-58   NotReady   control-plane   5m1s   v1.29.1
ubuntu@ip-172-31-80-58:~$
```

```
ip-172-31-80-58   NotReady   control-plane   5m1s   v1.29.1
ubuntu@ip-172-31-80-58:~$ kubectl apply -f https://raw.githubusercontent.com/projectcalico/calico/v3.24.1/manifests/calico.yaml
poddisruptionbudget.policy/calico-kube-controllers created
serviceaccount/calico-kube-controllers created
serviceaccount/calico-node created
```

```
ubuntu@ip-172-31-80-58:~$ kubectl get node
NAME              STATUS   ROLES           AGE   VERSION
ip-172-31-80-58   Ready    control-plane   9m    v1.29.1
ubuntu@ip-172-31-80-58:~$
```

i-030e886ce62109bb6 (Master)

Install Kubernetes Using Script

Step1: On Master Node Only

```
## Install Docker
sudo wget https://raw.githubusercontent.com/lerndevops/labs/master/scripts/installDocker.sh -P /tmp
sudo chmod 755 /tmp/installDocker.sh
sudo bash /tmp/installDocker.sh
sudo systemctl restart docker.service

## Install CRI-Docker
sudo wget https://raw.githubusercontent.com/lerndevops/labs/master/scripts/installCRIDockerd.sh -P /tmp
sudo chmod 755 /tmp/installCRIDockerd.sh
sudo bash /tmp/installCRIDockerd.sh
sudo systemctl restart cri-docker.service

## Install kubeadm,kubelet,kubectl
sudo wget https://raw.githubusercontent.com/lerndevops/labs/master/scripts/installK8S.sh -P /tmp
sudo chmod 755 /tmp/installK8S.sh
sudo bash /tmp/installK8S.sh

# Validate

   docker -v
   cri-dockerd --version
   kubeadm version -o short
   kubelet --version
   kubectl version --client

## Initialize kubernetes Master Node

   sudo kubeadm init --cri-socket unix:///var/run/cri-dockerd.sock --ignore-preflight-errors=all

   sudo mkdir -p $HOME/.kube
   sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
   sudo chown $(id -u):$(id -g) $HOME/.kube/config

   ## install networking driver -- Weave/flannel/canal/calico etc...

   ## below installs calico networking driver

   kubectl apply -f https://raw.githubusercontent.com/projectcalico/calico/v3.24.1/manifests/calico.yaml

   # Validate:  kubectl get nodes
```

Step2: On All Worker Nodes

```
## Install Docker
sudo wget https://raw.githubusercontent.com/lerndevops/labs/master/scripts/installDocker.sh -P /tmp
sudo chmod 755 /tmp/installDocker.sh
sudo bash /tmp/installDocker.sh
sudo systemctl restart docker.service

## Install CRI-Docker
sudo wget https://raw.githubusercontent.com/lerndevops/labs/master/scripts/installCRIDockerd.sh -P /tmp
sudo chmod 755 /tmp/installCRIDockerd.sh
sudo bash /tmp/installCRIDockerd.sh
sudo systemctl restart cri-docker.service

## Install kubeadm,kubelet,kubectl
sudo wget https://raw.githubusercontent.com/lerndevops/labs/master/scripts/installK8S.sh -P /tmp
sudo chmod 755 /tmp/installK8S.sh
sudo bash /tmp/installK8S.sh


# Validate

   docker -v
   cri-dockerd --version
   kubeadm version -o short
   kubelet --version
   kubectl version --client

## Run Below on Master Node to get join token

kubeadm token create --print-join-command

    copy the kubeadm join token from master &
           ensure to add --cri-socket unix:///var/run/cri-dockerd.sock as below &
           ensure to add sudo
           then run on worker nodes

    Ex: sudo kubeadm join 10.128.15.231:6443  --token mks3y2.v03tyyru0gy12mbt \
           --discovery-token-ca-cert-hash sha256:3de23d42c7002be0893339fbe558ee75e14399e11f22e3f0b34351077b7c4b56 --cri-socket
```

```
ubuntu@ip-172-31-95-31:~$ history
    1  sudo apt-get update -y && clear
    2  sudo wget https://raw.githubusercontent.com/lerndevops/labs/master/scripts/installDocker.sh -P /tmp
    3  sudo chmod 755 /tmp/installDocker.sh
    4  sudo bash /tmp/installDocker.sh
    5  sudo systemctl restart docker.service
    6  clear
    7  history
ubuntu@ip-172-31-95-31:~$
```

i-08445b544fbb946f9 (slave1)

```
ubuntu@ip-172-31-95-31:~$    docker -v
Docker version 25.0.3, build 4debf41
ubuntu@ip-172-31-95-31:~$    cri-dockerd --version
cri-dockerd 0.3.4 (e88b1605)
ubuntu@ip-172-31-95-31:~$    kubeadm version -o short
v1.29.1
ubuntu@ip-172-31-95-31:~$    kubelet --version
Kubernetes v1.29.1
ubuntu@ip-172-31-95-31:~$    kubectl version --client
Client Version: v1.29.1
Kustomize Version: v5.0.4-0.20230601165947-6ce0bf390ce3
ubuntu@ip-172-31-95-31:~$
```

i-08445b544fbb946f9 (slave1)

```
Processing triggers for man-db (2.9.1-1) ...
kubelet, kubeadm & kubectl are successfully installed
kubelet set on hold.
kubeadm set on hold.
kubectl set on hold.
ubuntu@ip-172-31-86-123:~$    docker -v
Docker version 25.0.3, build 4debf41
ubuntu@ip-172-31-86-123:~$    cri-dockerd --version
cri-dockerd 0.3.4 (e88b1605)
ubuntu@ip-172-31-86-123:~$    kubeadm version -o short
v1.29.1
ubuntu@ip-172-31-86-123:~$    kubelet --version
Kubernetes v1.29.1
ubuntu@ip-172-31-86-123:~$    kubectl version --client
```

i-020eb95e295f6dd50 (slave2)

```
ubuntu@ip-172-31-80-58:~$ kubectl get node
NAME              STATUS   ROLES           AGE   VERSION
ip-172-31-80-58   Ready    control-plane   9m    v1.29.1
ubuntu@ip-172-31-80-58:~$ kubeadm token create --print-join-command
kubeadm join 172.31.80.58:6443 --token ym0g11.6qttb1b9rbadu8m5 --discovery-token-ca-cert-hash sha256:ba609d3179035de1a78938fdef7ff281c5db3ba7461a6833387232f3c6e4c19b
ubuntu@ip-172-31-80-58:~$
```

i-030e886ce62109bb6 (Master)

```
ubuntu@ip-172-31-95-31:~$ sudo kubeadm join 172.31.80.58:6443 --token ym0g11.6qttb1b9rbadu8m5 --discovery-token-ca-cert-hash sha256:ba609d3179035de1a78938fdef7ff281c5
db3ba7461a6833387232f3c6e4c19b --cri-socket unix:///var/run/cri-dockerd.sock
```

i-08445b544fbb946f9 (slave1)

```
ubuntu@ip-172-31-86-123:~$  sudo kubeadm join 172.31.80.58:6443 --token ym0g11.6qttb1b9rbadu8m5 --discovery-token-ca-cert-hash sha256:ba609d3179035de1a78938fdef7ff281
c5db3ba7461a6833387232f3c6e4c19b --cri-socket unix:///var/run/cri-dockerd.sock
[preflight] Running pre-flight checks
[preflight] Reading configuration from the cluster...
[preflight] FYI: You can look at this config file with 'kubectl -n kube-system get cm kubeadm-config -o yaml'
[kubelet-start] Writing kubelet configuration to file "/var/lib/kubelet/config.yaml"
[kubelet-start] Writing kubelet environment file with flags to file "/var/lib/kubelet/kubeadm-flags.env"
[kubelet-start] Starting the kubelet
[kubelet-start] Waiting for the kubelet to perform the TLS Bootstrap...

This node has joined the cluster:
* Certificate signing request was sent to apiserver and a response was received.
* The Kubelet was informed of the new secure connection details.

Run 'kubectl get nodes' on the control-plane to see this node join the cluster.

ubuntu@ip-172-31-86-123:~$
```

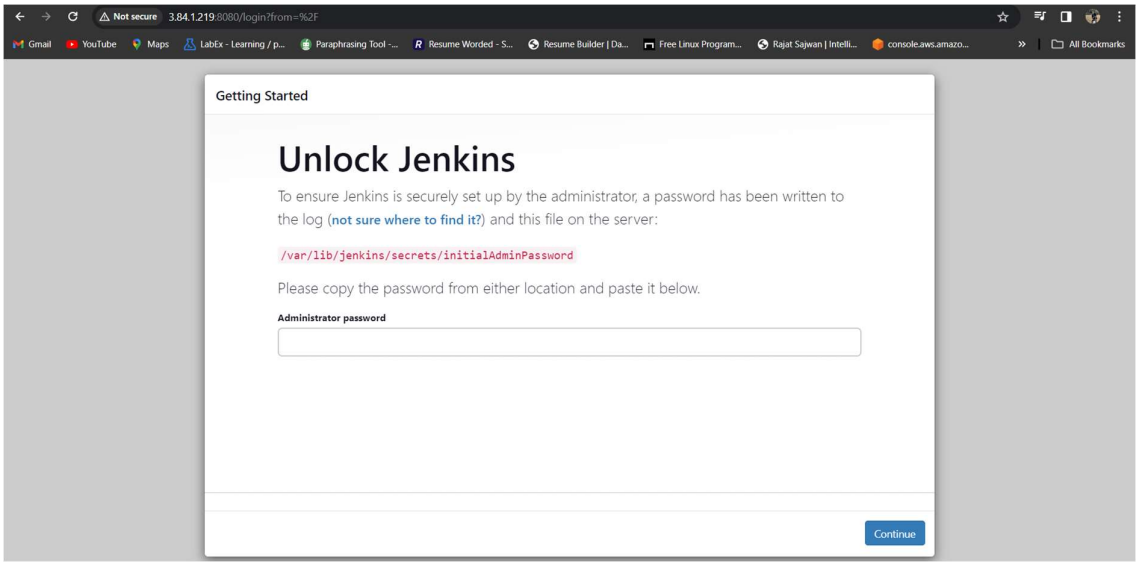
i-020eb95e295f6dd50 (slave2)

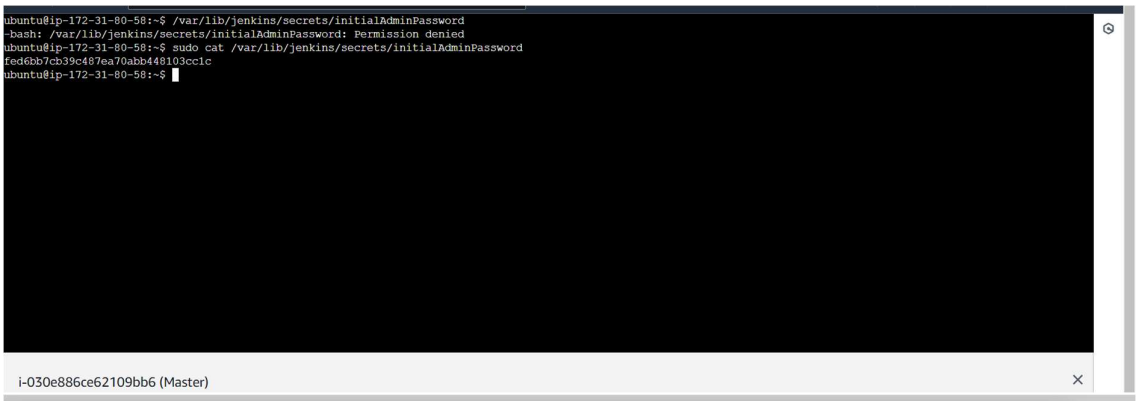
```
ubuntu@ip-172-31-80-58:~$ kubectl get node
NAME              STATUS   ROLES           AGE   VERSION
ip-172-31-80-58   Ready    control-plane   9m    v1.29.1
ubuntu@ip-172-31-80-58:~$ kubeadm token create --print-join-command
kubeadm join 172.31.80.58:6443 --token ym0g11.6qttb1b9rbadu8m5 --discovery-token-ca-cert-hash sha256:ba609d3179035de1a78938fdef7ff281c5db3ba7461a6833387232f3c6e4c19b
ubuntu@ip-172-31-80-58:~$ kubectl get node
NAME              STATUS   ROLES           AGE   VERSION
ip-172-31-80-58   Ready    control-plane   24m   v1.29.1
ip-172-31-86-123  Ready    <none>          73s   v1.29.1
ip-172-31-95-31   Ready    <none>          76s   v1.29.1
ubuntu@ip-172-31-80-58:~$
```

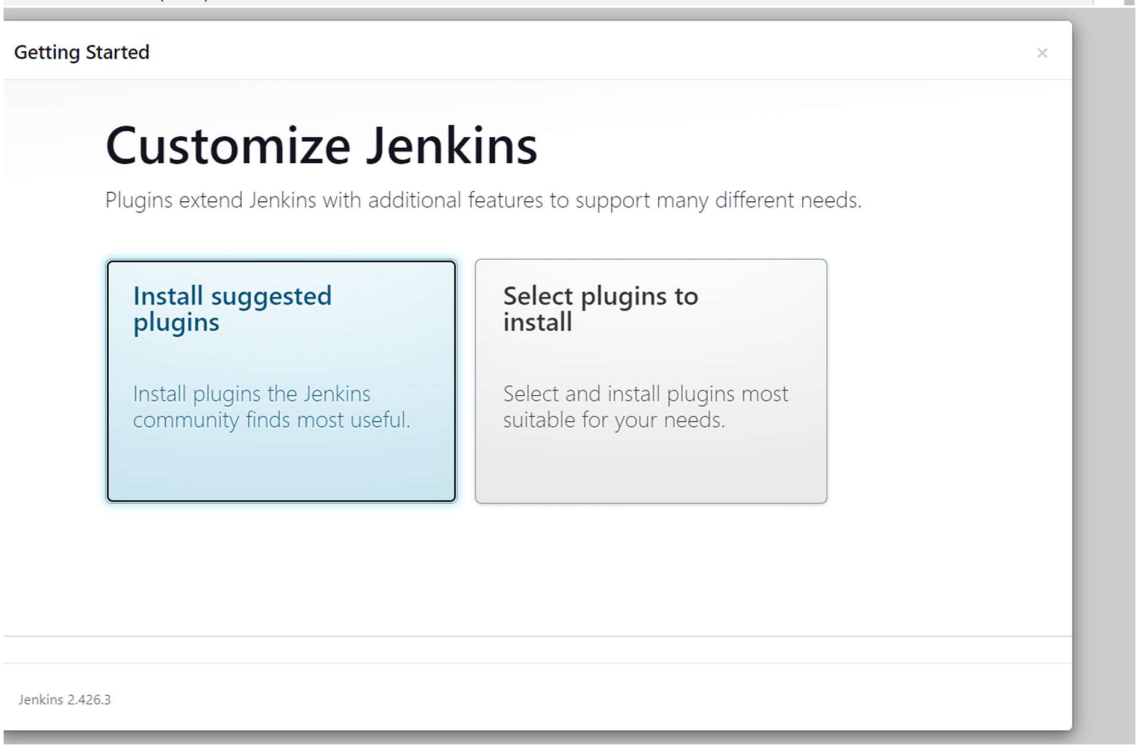
i-030e886ce62109bb6 (Master)

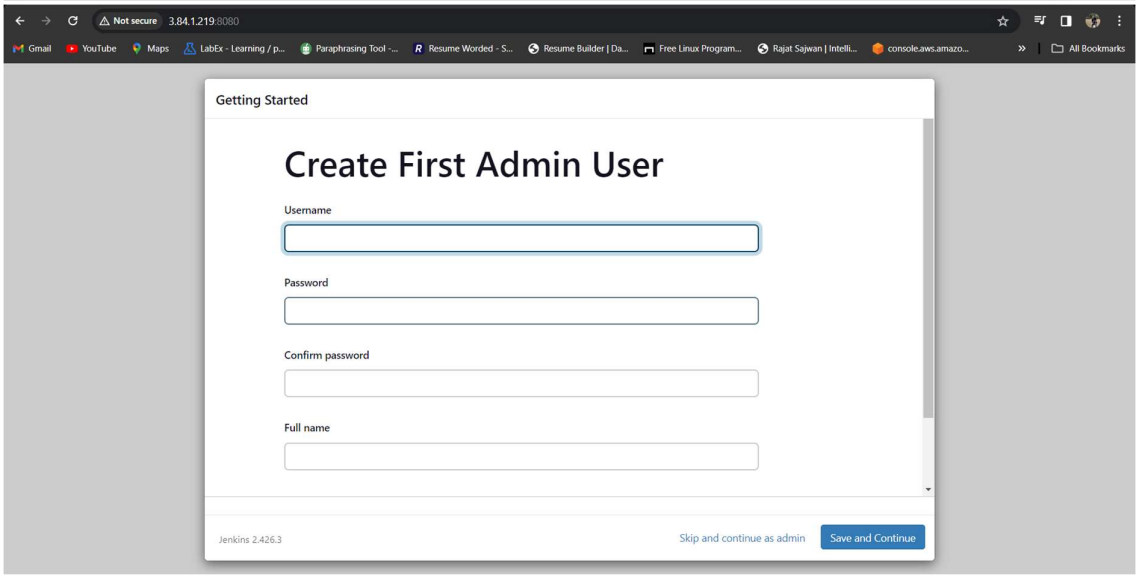
```
   40  sudo wget -O /usr/share/keyrings/jenkins-keyring.asc https://pkg.jenkins.io/debian-stable/jenkins.io-2023.key
   41  echo deb [signed-by=/usr/share/keyrings/jenkins-keyring.asc] https://pkg.jenkins.io/debian-stable binary/ | sudo tee /etc/apt/sources.list.d/jenkins.list > /de
v/null
   42  sudo apt-get update
   43  sudo apt-get install jenkins
   44  clear
   45  history
ubuntu@ip-172-31-80-58:~$
```

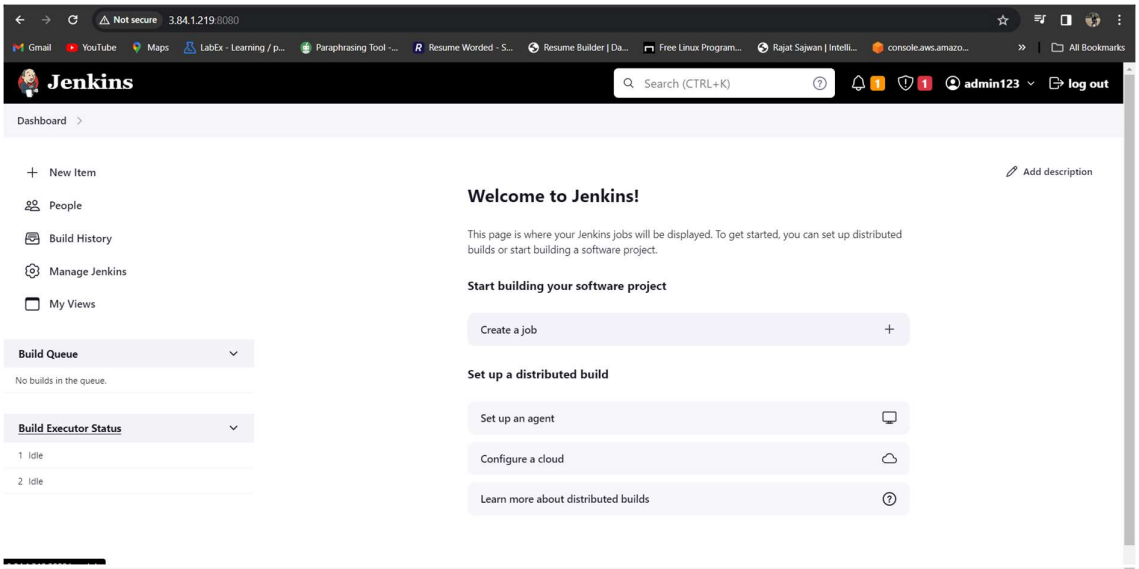
i-030e886ce62109bb6 (Master)

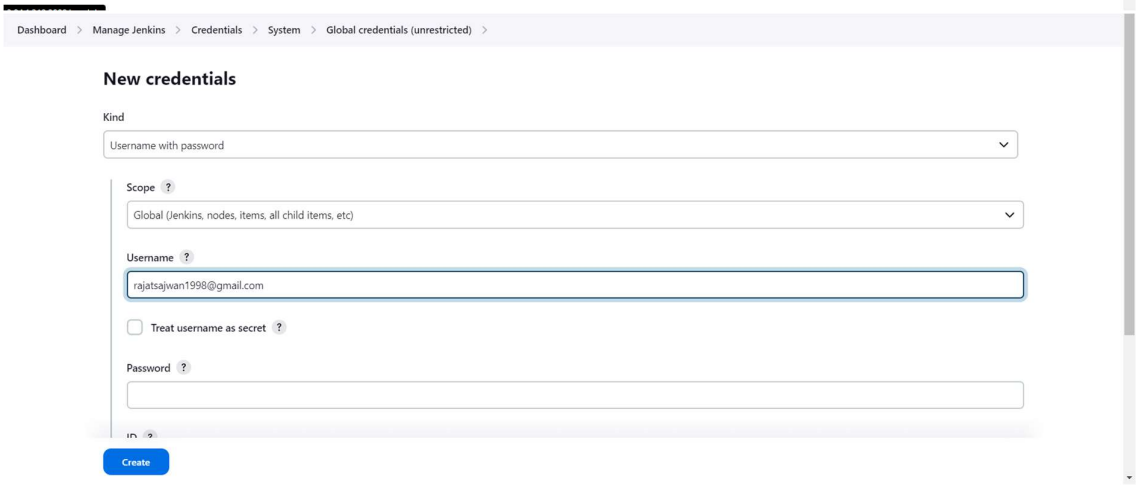


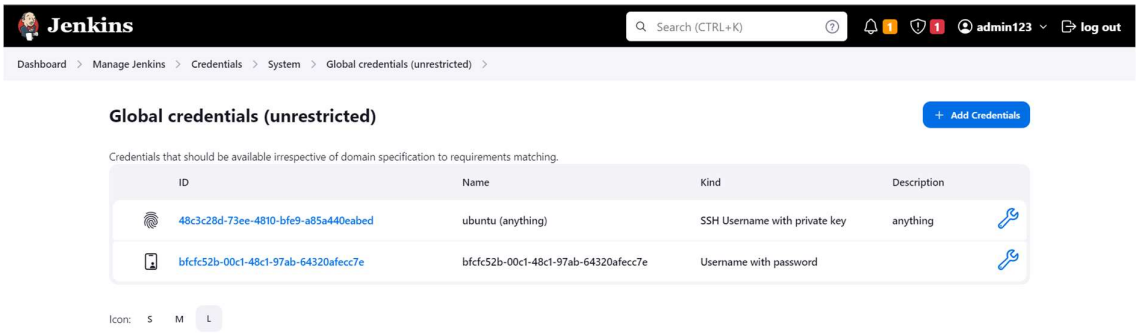

ubuntu@ip-172-31-80-58:~$ /var/lib/jenkins/secrets/initialAdminPassword
-bash: /var/lib/jenkins/secrets/initialAdminPassword: Permission denied
ubuntu@ip-172-31-80-58:~$ sudo cat /var/lib/jenkins/secrets/initialAdminPassword
fed6bb7cb39c487ea70abb448103cc1c
ubuntu@ip-172-31-80-58:~$
i-030e886ce62109bb6 (Master)

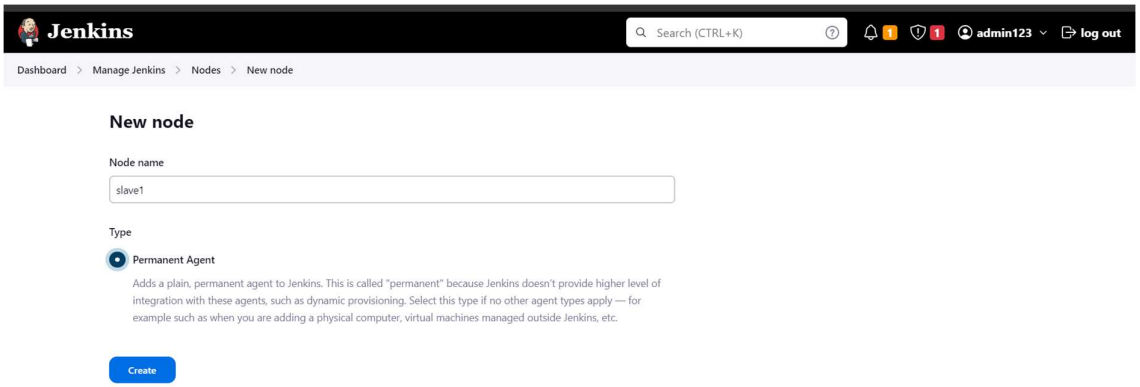

Getting Started
Customize Jenkins
Plugins extend Jenkins with additional features to support many different needs.
Install suggested plugins
Install plugins the Jenkins community finds most useful.
Select plugins to install
Select and install plugins most suitable for your needs.
Jenkins 2.426.3

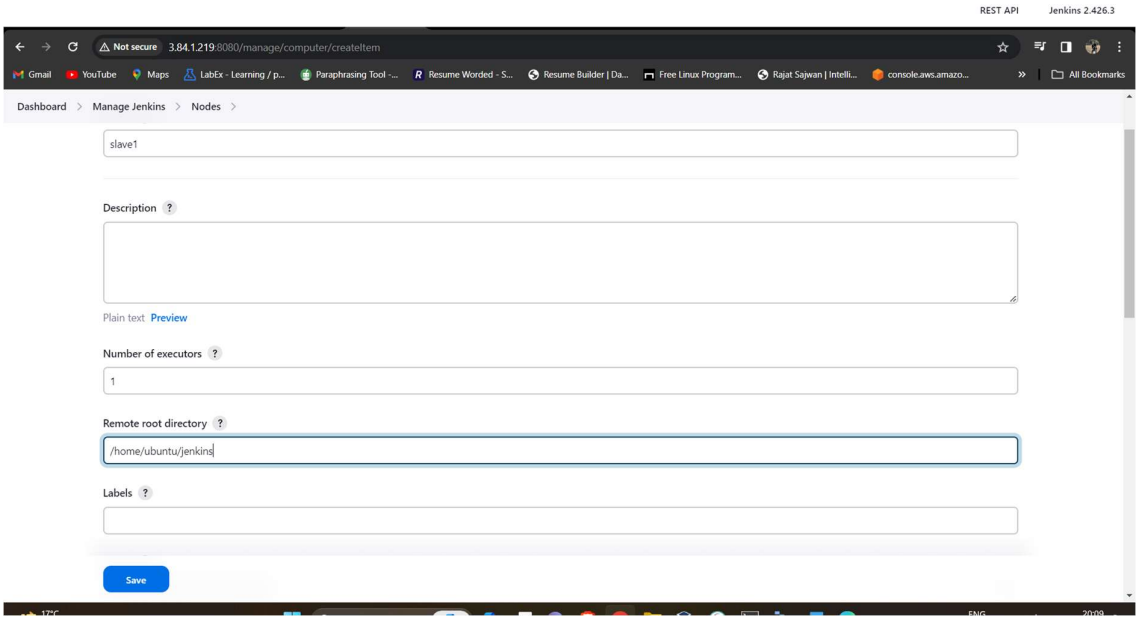

Not secure 3.84.1.219:8080
Getting Started
Create First Admin User
Username
Password
Confirm password
Full name
Jenkins 2.426.3
Skip and continue as admin
Save and Continue

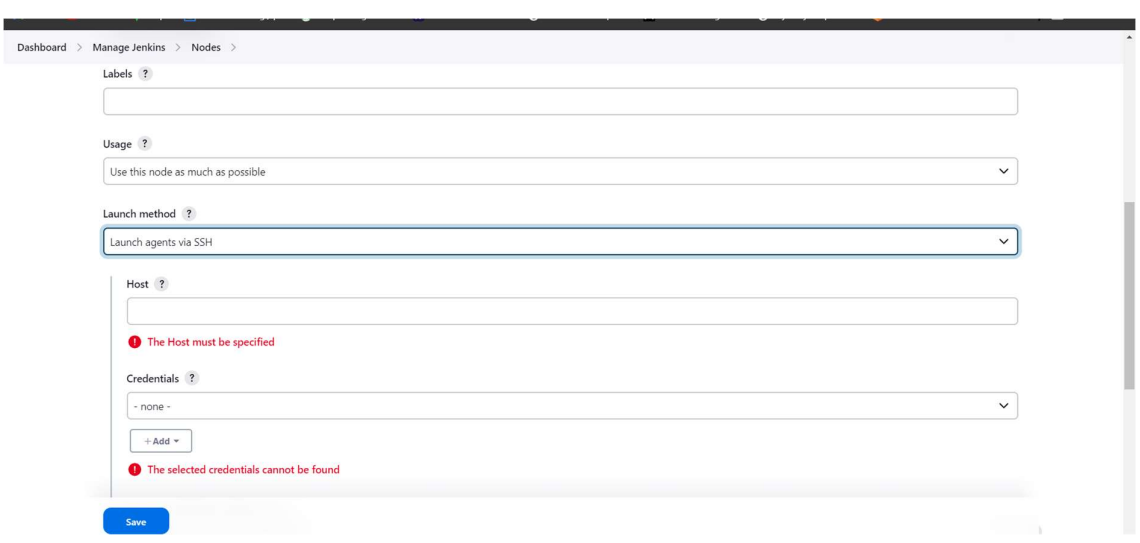

Not secure 3.84.1.219:8080
Jenkins
Search (CTRL+K)
admin123
log out
Dashboard
New Item
People
Build History
Manage Jenkins
My Views
Add description
Welcome to Jenkins!
This page is where your Jenkins jobs will be displayed. To get started, you can set up distributed builds or start building a software project.
Start building your software project
Create a job
Set up a distributed build
Set up an agent
Configure a cloud
Learn more about distributed builds
Build Queue
No builds in the queue.
Build Executor Status
1 Idle
2 Idle

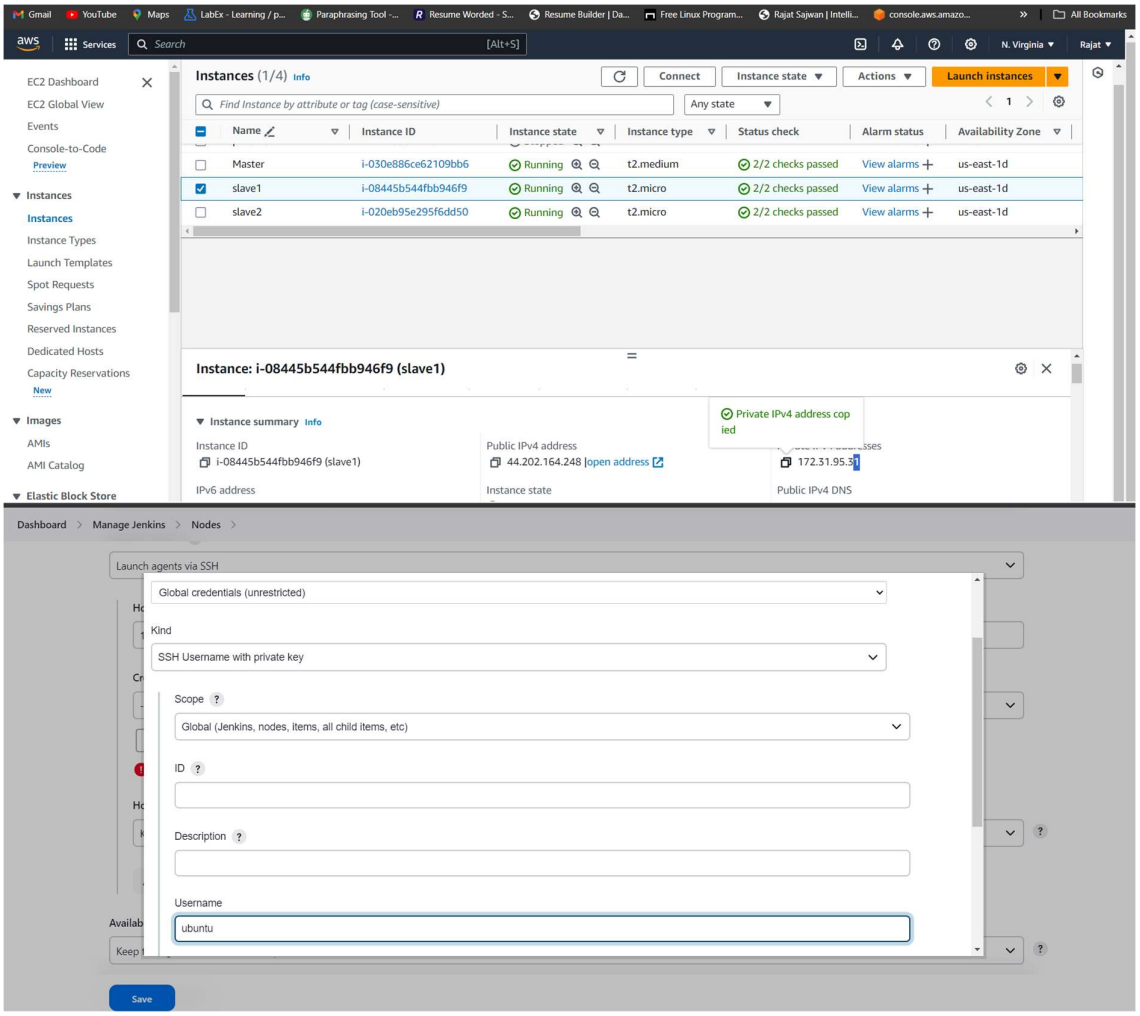

Dashboard > Manage Jenkins > Credentials > System > Global credentials (unrestricted)
New credentials
Kind
Username with password
Scope
Global (Jenkins, nodes, items, all child items, etc)
Username
rajatsajwan1998@gmail.com
Treat username as secret
Password
Create


Jenkins
Search (CTRL+K)
admin123
log out
Dashboard > Manage Jenkins > Credentials > System > Global credentials (unrestricted)
Global credentials (unrestricted)
Add Credentials
Credentials that should be available irrespective of domain specification to requirements matching.
ID
Name
Kind
Description
48c3c28d-73ee-4810-bfe9-a85a440eabed
ubuntu (anything)
SSH Username with private key
anything
bfcfc52b-00c1-48c1-97ab-64320afecc7e
bfcfc52b-00c1-48c1-97ab-64320afecc7e
Username with password
Icon: S M L

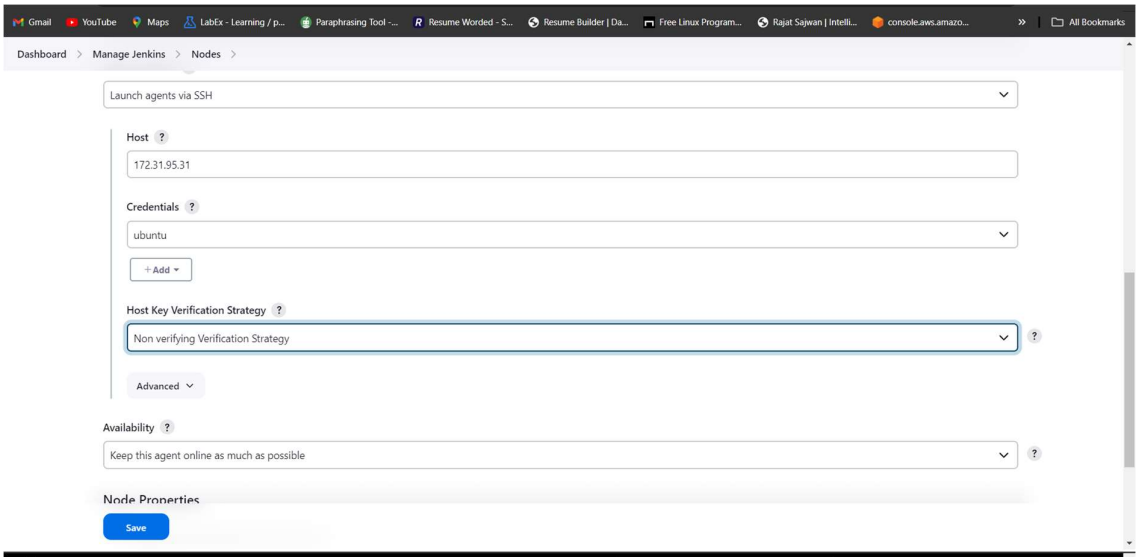

Jenkins
Search (CTRL+K)
admin123
log out
Dashboard > Manage Jenkins > Nodes > New node
New node
Node name
slave1
Type
Permanent Agent
Adds a plain, permanent agent to Jenkins. This is called "permanent" because Jenkins doesn't provide higher level of integration with these agents, such as dynamic provisioning. Select this type if no other agent types apply — for example such as when you are adding a physical computer, virtual machines managed outside Jenkins, etc.
Create

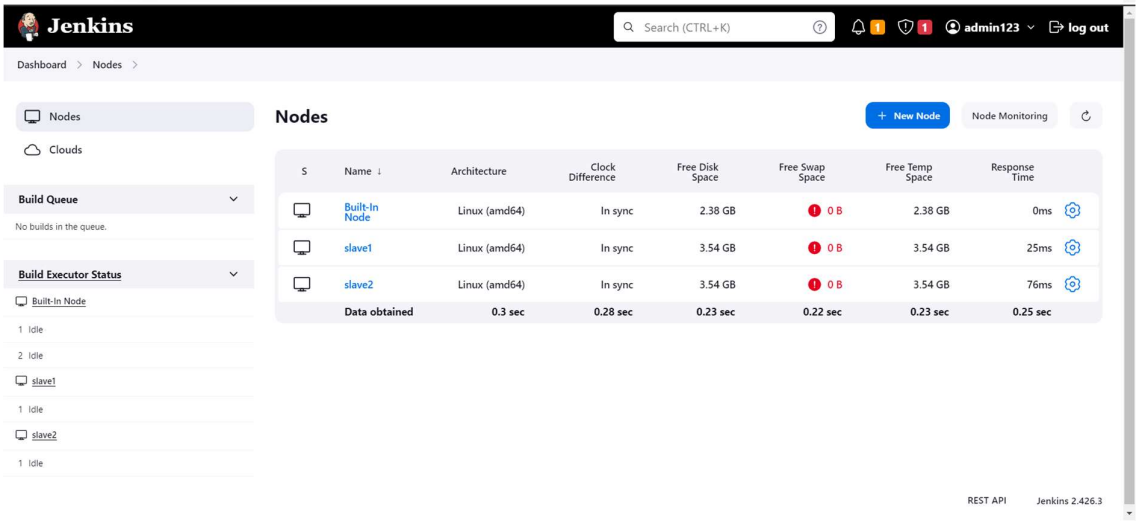

REST API
Jenkins 2.426.3
Not secure 3.84.1.219:8080/manage/computer/createItem
Dashboard > Manage Jenkins > Nodes >
slave1
Description
Plain text Preview
Number of executors
1
Remote root directory
/home/ubuntu/jenkins
Labels
Save

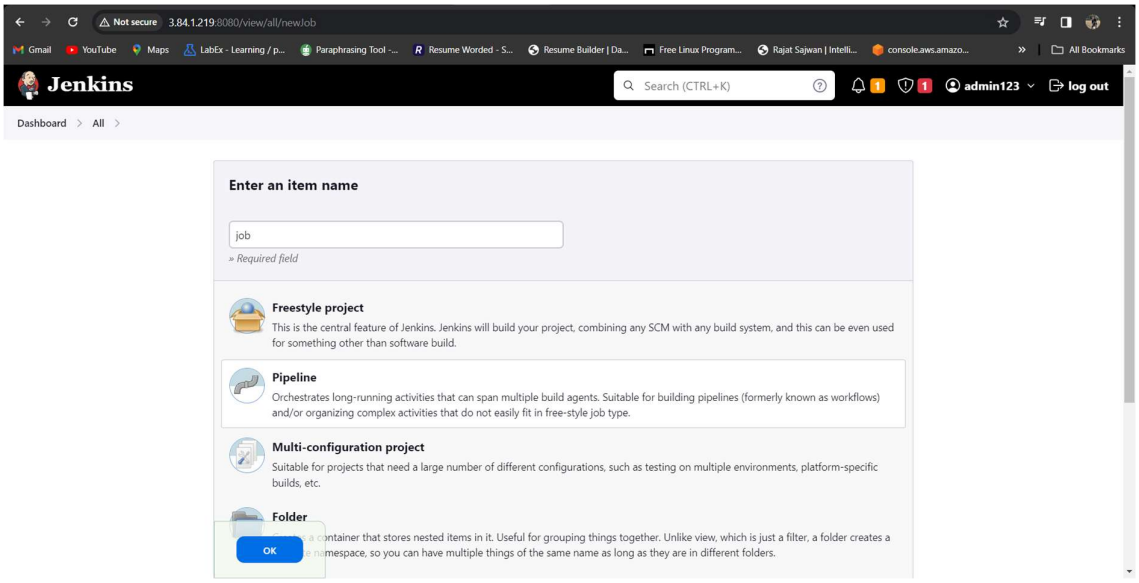

Dashboard > Manage Jenkins > Nodes >
Labels
Usage
Use this node as much as possible
Launch method
Launch agents via SSH
Host
The Host must be specified
Credentials
- none -
+ Add
The selected credentials cannot be found
Save

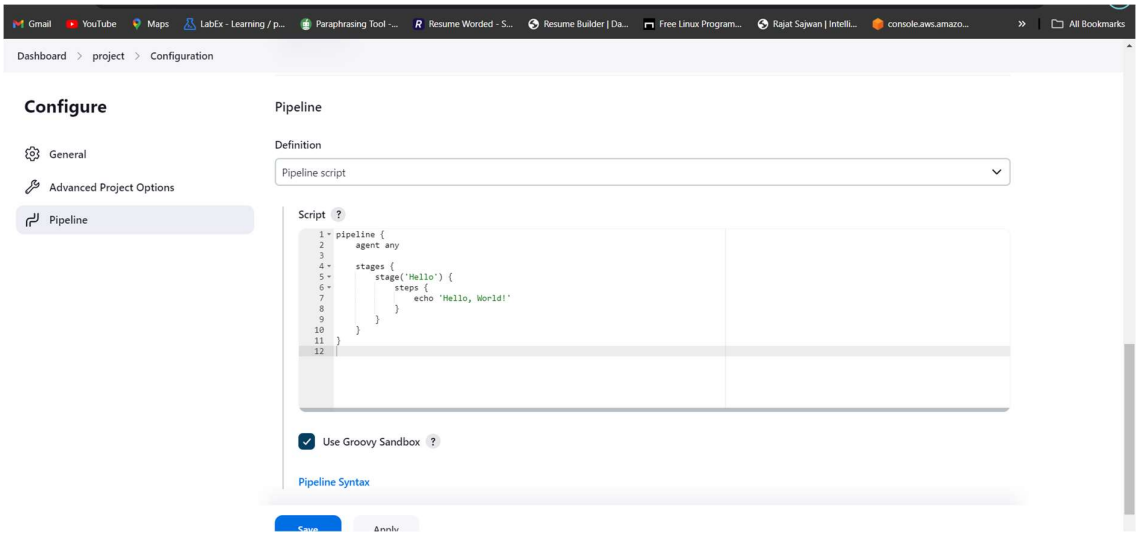

Gmail
YouTube
Maps
LabEx - Learning / p...
Paraphrasing Tool -...
Resume Worded - S...
Resume Builder | Da...
Free Linux Program...
Rajat Sajwan | Intelli...
console.aws.amazo...
All Bookmarks
aws
Services
Search
[Alt+S]
N. Virginia
Rajat
EC2 Dashboard
EC2 Global View
Events
Console-to-Code
Preview
Instances
Instances
Instance Types
Launch Templates
Spot Requests
Savings Plans
Reserved Instances
Dedicated Hosts
Capacity Reservations
New
Images
AMIs
AMI Catalog
Elastic Block Store
Instances (1/4) Info
Connect
Instance state
Actions
Launch instances
Find Instance by attribute or tag (case-sensitive)
Any state
Name
Instance ID
Instance state
Instance type
Status check
Alarm status
Availability Zone
Master
i-030e886ce62109bb6
Running
t2.medium
2/2 checks passed
View alarms
us-east-1d
slave1
i-08445b544fbb946f9
Running
t2.micro
2/2 checks passed
View alarms
us-east-1d
slave2
i-020eb95e295f6dd50
Running
t2.micro
2/2 checks passed
View alarms
us-east-1d
Instance: i-08445b544fbb946f9 (slave1)
Instance summary Info
Private IPv4 address copied
Instance ID
i-08445b544fbb946f9 (slave1)
Public IPv4 address
44.202.164.248 |open address
172.31.95.31
IPv6 address
Instance state
Public IPv4 DNS

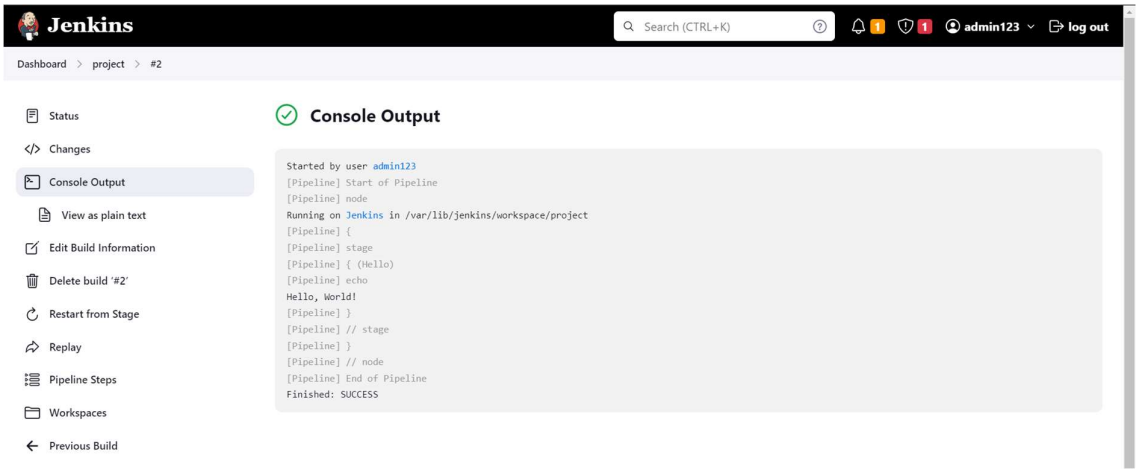
Dashboard
Manage Jenkins
Nodes
Launch agents via SSH
Global credentials (unrestricted)
Kind
SSH Username with private key
Scope
Global (Jenkins, nodes, items, all child items, etc)
ID
Description
Username
ubuntu
Save


Gmail YouTube Maps LabEx - Learning / p... Paraphrasing Tool -... Resume Worded - S... Resume Builder | Da... Free Linux Program... Rajat Sajwan | Intelli... console.aws.amazo... All Bookmarks
Dashboard > Manage Jenkins > Nodes >
Launch agents via SSH
Host
172.31.95.31
Credentials
ubuntu
+ Add
Host Key Verification Strategy
Non verifying Verification Strategy
Advanced
Availability
Keep this agent online as much as possible
Node Properties
Save


Jenkins
Search (CTRL+K)
admin123
log out
Dashboard > Nodes >
Nodes
Clouds
Build Queue
No builds in the queue.
Build Executor Status
Built-In Node
1 Idle
2 Idle
slave1
1 Idle
slave2
1 Idle
Nodes
New Node
Node Monitoring
S
Name
Architecture
Clock Difference
Free Disk Space
Free Swap Space
Free Temp Space
Response Time
Built-In Node Linux (amd64) In sync 2.38 GB 0 B 2.38 GB 0ms
slave1 Linux (amd64) In sync 3.54 GB 0 B 3.54 GB 25ms
slave2 Linux (amd64) In sync 3.54 GB 0 B 3.54 GB 76ms
Data obtained 0.3 sec 0.28 sec 0.23 sec 0.22 sec 0.23 sec 0.25 sec
REST API
Jenkins 2.426.3


Not secure 3.84.1.219:8080/view/all/newJob
Gmail YouTube Maps LabEx - Learning / p... Paraphrasing Tool -... Resume Worded - S... Resume Builder | Da... Free Linux Program... Rajat Sajwan | Intelli... console.aws.amazo... All Bookmarks
Jenkins
Search (CTRL+K)
admin123
log out
Dashboard > All >
Enter an item name
job
» Required field
Freestyle project
This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.
Pipeline
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.
Multi-configuration project
Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.
Folder
container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a namespace, so you can have multiple things of the same name as long as they are in different folders.
OK





Script :

```
pipeline{
        agent none
        environment {
           DOCKERHUB_CREDENTIALS= credentials('9faeb173-e2a4-446d-a272-2a3c0ccc8906')
        }
        stages{
          stage('Hello'){
                agent{
                  label 'Kub-master'
                }
                steps{
```

```
                echo 'Hello World'
            }
        }
        stage('git'){
            agent{
                label 'Kub-master'
            }
            steps{
                git'https://github.com/intellipaat2/website.git'
            }
        }
        stage('docker'){
            agent{
                label 'Kub-master'
            }
            steps{
                sh 'sudo docker build /home/ubuntu/jenkins/workspace/pipeline -t
rrajatsajwan/capstoneproj2'
                sh 'echo $DOCKERHUB_CREDENTIALS_PSW | sudo docker login -u
$DOCKERHUB_CREDENTIALS_USR --password-stdin'
                sh 'sudo docker push rrajatsajwan/capstoneproj2'
            }
        }
        stage('kubernetes'){
            agent{
                label 'Kub-master'
            }
            steps{
                sh 'kubectl delete deploy nginx-deployment'
                sh 'kubectl apply -f deploy.yml'
                sh 'kubectl delete svc my-nginx'
                sh 'kubectl create -f svc.yml'
            }
        }
    }
```

}

	Hello	git	docker	Kubernetes
tage times: time: ~10s)	378ms	1s	4s	907ms
	209ms	456ms	4s	1s
	252ms	436ms	4s	1s
	229ms	512ms	4s	1s

Hello world!!!!!!!!!!!!!

GitHub