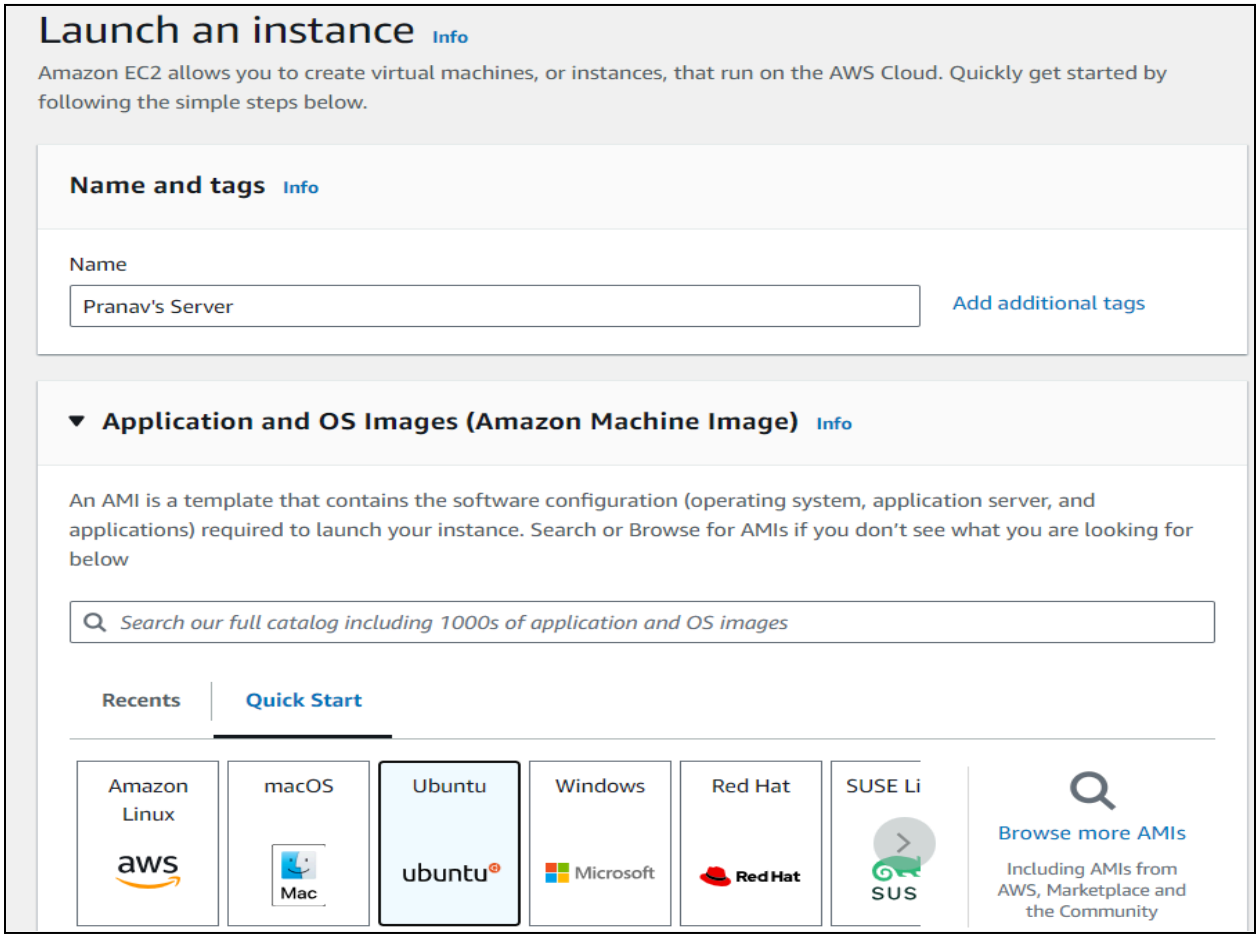
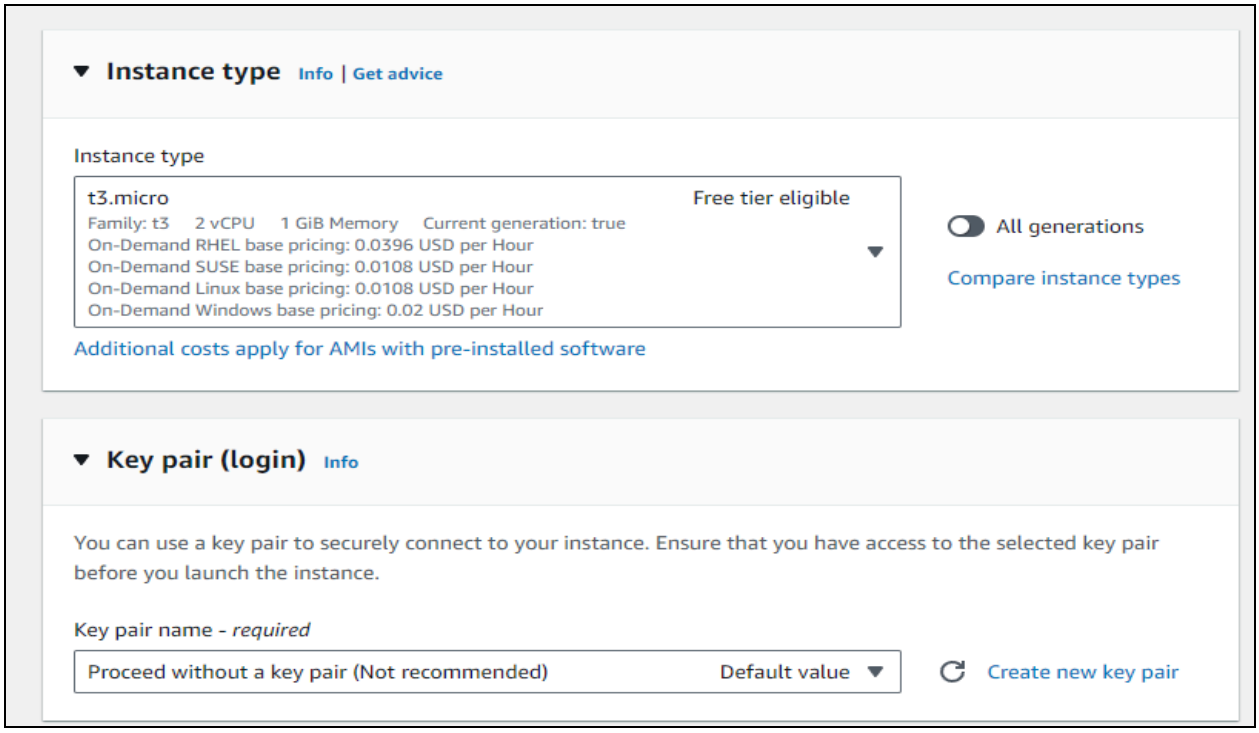
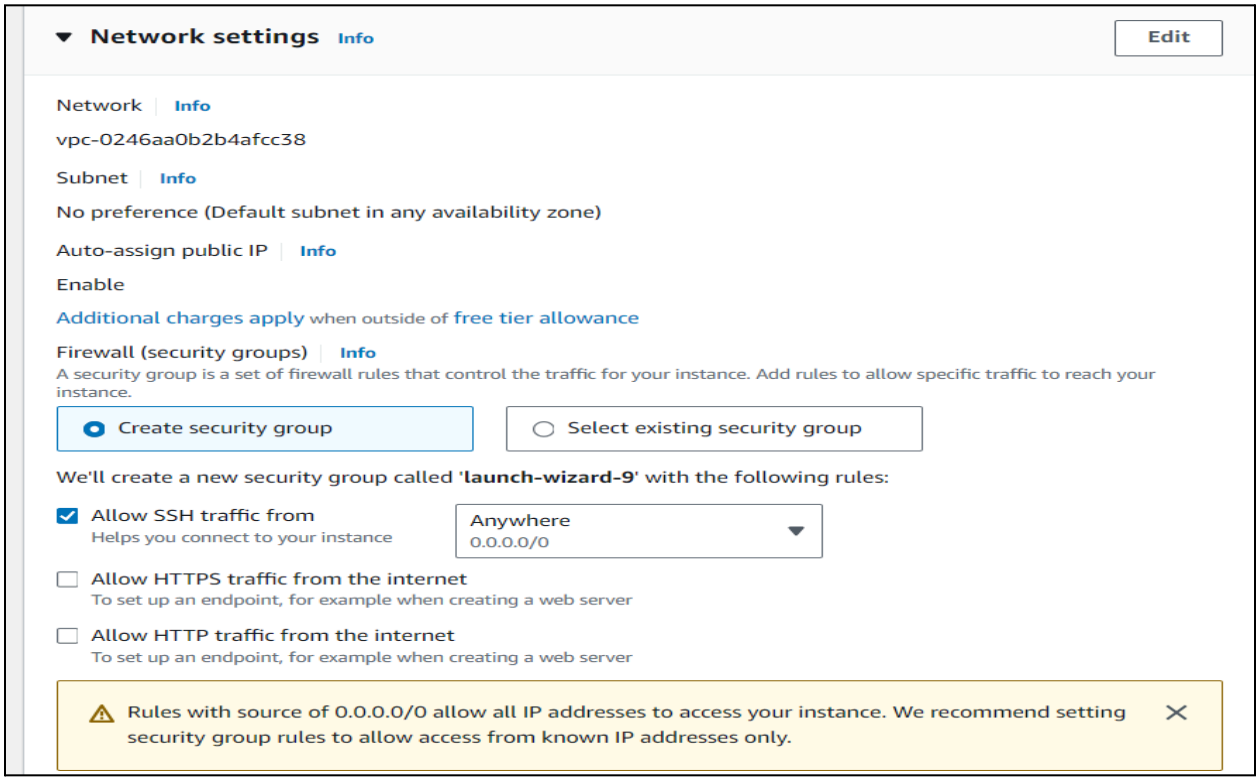
ADVANCE DEVOPS EXP-1

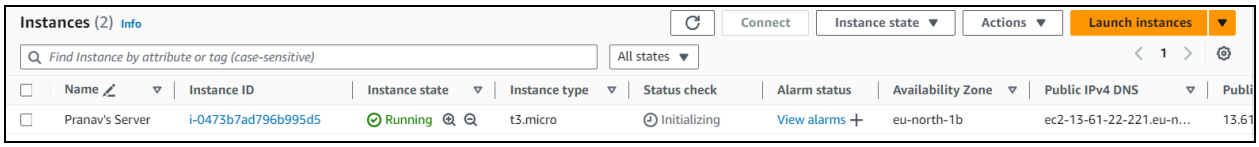
Pranav Titambe **D15A/62**

Aim: To understand the benefits of Cloud infrastructure and Setup AWS Cloud9 IDE, Launch AWS Cloud9 IDE and and Perform Collaboration Demonstration.




Instance type Info | Get advice
Instance type
t3.micro
Free tier eligible
Family: t3 2 vCPU 1 GiB Memory Current generation: true
On-Demand RHEL base pricing: 0.0396 USD per Hour
On-Demand SUSE base pricing: 0.0108 USD per Hour
On-Demand Linux base pricing: 0.0108 USD per Hour
On-Demand Windows base pricing: 0.02 USD per Hour
All generations
Compare instance types
Additional costs apply for AMIs with pre-installed software
Key pair (login) Info
You can use a key pair to securely connect to your instance. Ensure that you have access to the selected key pair before you launch the instance.
Key pair name - required
Proceed without a key pair (Not recommended)
Default value
Create new key pair


Network settings Info
Edit
Network Info
vpc-0246aa0b2b4afcc38
Subnet Info
No preference (Default subnet in any availability zone)
Auto-assign public IP Info
Enable
Additional charges apply when outside of free tier allowance
Firewall (security groups) Info
A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.
Create security group
Select existing security group
We'll create a new security group called 'launch-wizard-9' with the following rules:
Allow SSH traffic from
Helps you connect to your instance
Anywhere
0.0.0.0/0
Allow HTTPS traffic from the internet
To set up an endpoint, for example when creating a web server
Allow HTTP traffic from the internet
To set up an endpoint, for example when creating a web server
Rules with source of 0.0.0.0/0 allow all IP addresses to access your instance. We recommend setting security group rules to allow access from known IP addresses only.

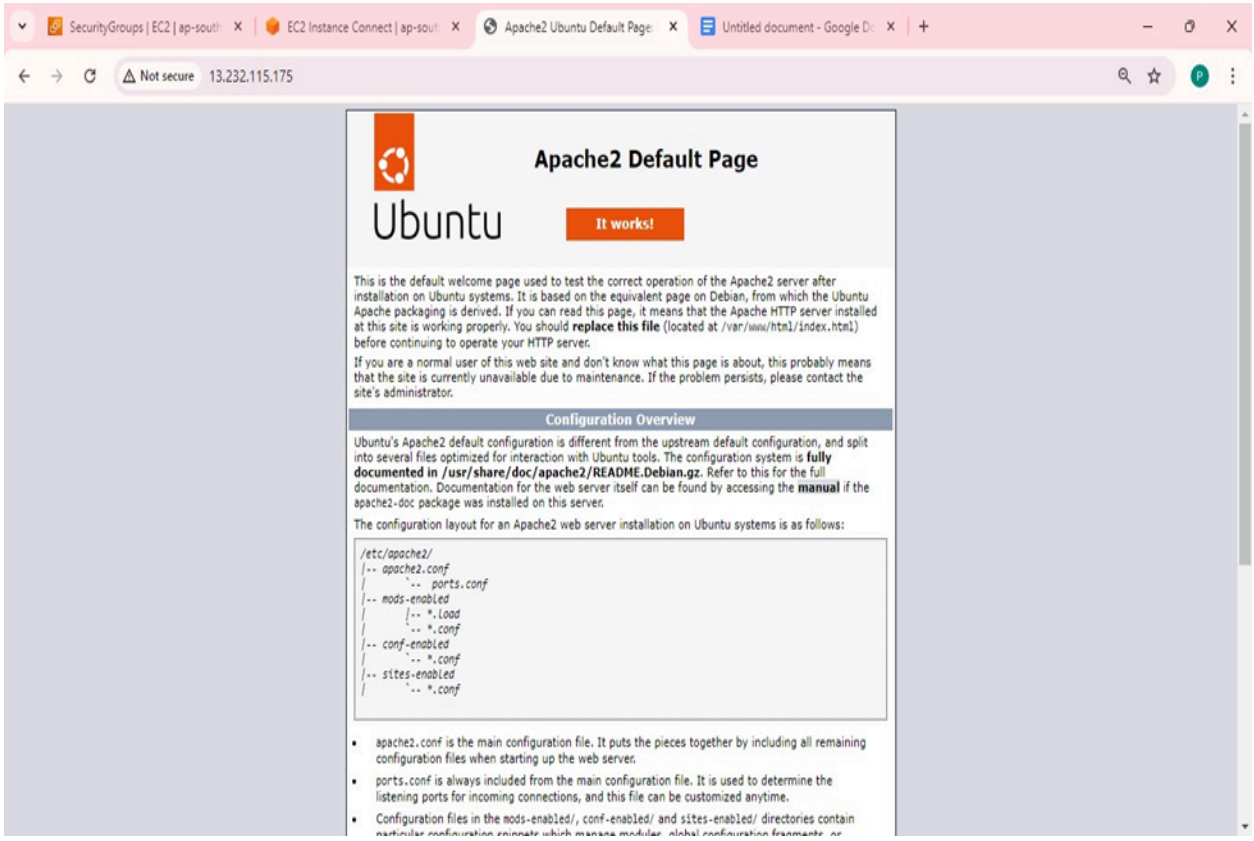



```
login as: ubuntu
Authenticating with public key "test-key"
Welcome to Ubuntu 24.04 LTS (GNU/Linux 6.8.0-1009-aws x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/pro

 System information as of Mon Jul 29 17:28:10 UTC 2024

  System load:  0.02               Processes:             109
  Usage of /:   10.5% of 14.46GB   Users logged in:       0
  Memory usage: 20%                IPv4 address for enX0: 172.31.43.87
  Swap usage:   0%

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status


The list of available updates is more than a week old.
```

```
root@ip-172-31-43-87:/# apt install apache2
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
apache2 is already the newest version (2.4.58-1ubuntu8.4).
0 upgraded, 0 newly installed, 0 to remove and 26 not upgraded.
root@ip-172-31-43-87:/#
```

```
root@ip-172-31-43-87:/# systemctl status apache2
● apache2.service - The Apache HTTP Server
     Loaded: loaded (/usr/lib/systemd/system/apache2.service; enabled; preset: enabled)
     Active: active (running) since Tue 2024-07-30 05:02:47 UTC; 12min ago
       Docs: https://httpd.apache.org/docs/2.4/
    Process: 494 ExecStart=/usr/sbin/apachectl start (code=exited, status=0/SUCCESS)
   Main PID: 527 (apache2)
      Tasks: 55 (limit: 1130)
     Memory: 8.3M (peak: 8.5M)
        CPU: 94ms
     CGroup: /system.slice/apache2.service
             ├─527 /usr/sbin/apache2 -k start
             ├─532 /usr/sbin/apache2 -k start
             └─533 /usr/sbin/apache2 -k start

Jul 30 05:02:45 ip-172-31-43-87 systemd[1]: Starting apache2.service - The Apache HTTP Server...
Jul 30 05:02:47 ip-172-31-43-87 systemd[1]: Started apache2.service - The Apache HTTP Server.
root@ip-172-31-43-87:/#
```

```
root@ip-172-31-43-87:/# cd /var/www/html/
root@ip-172-31-43-87:/var/www/html#
```

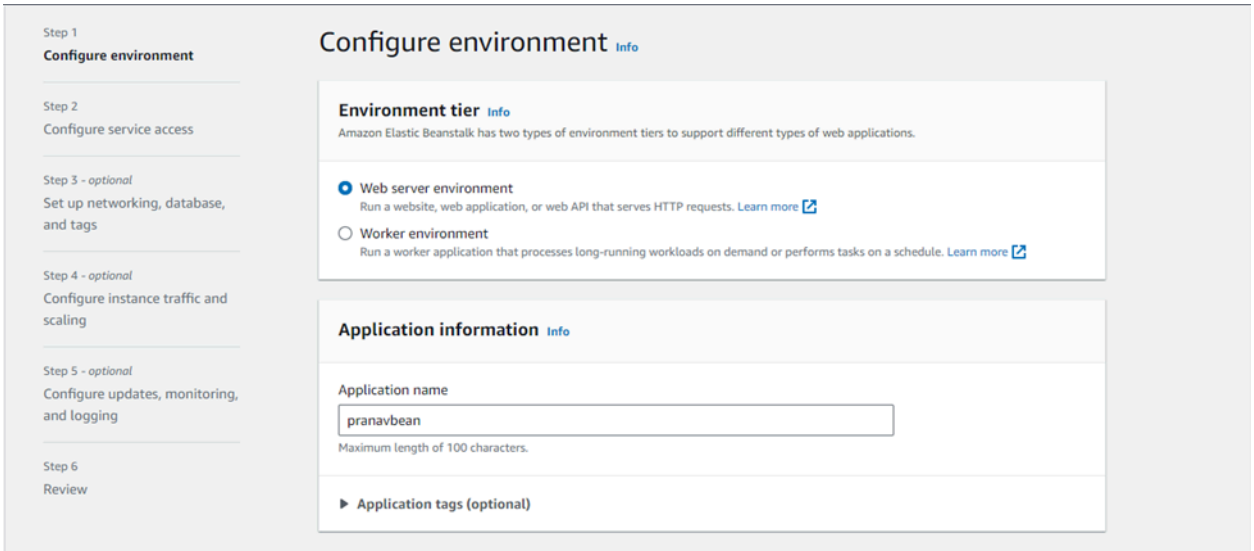


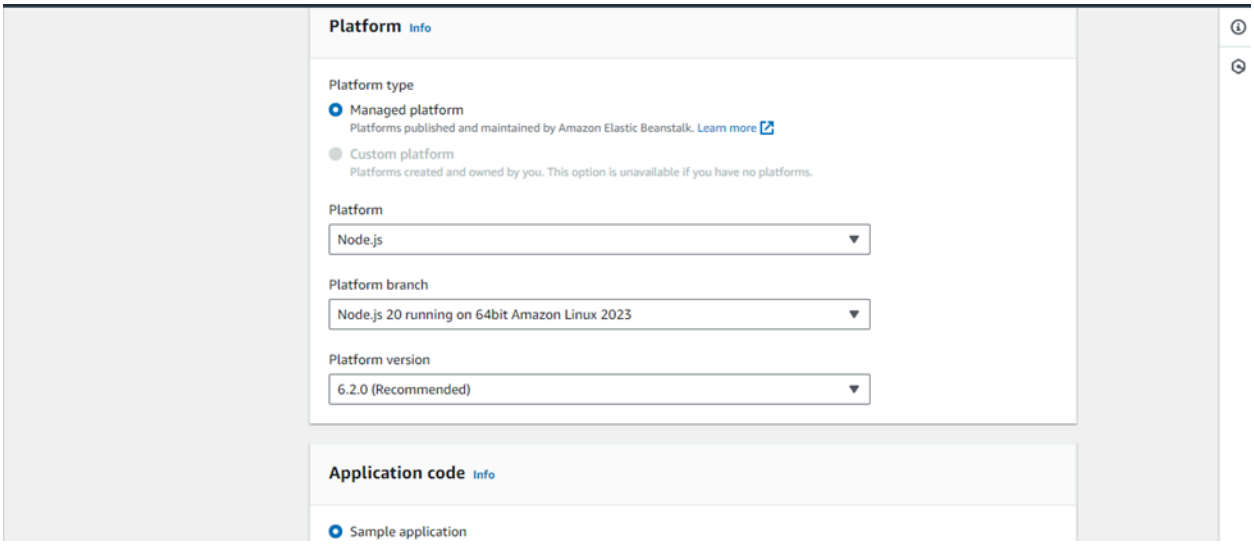
ADVANCE DEVOPS EXP-2

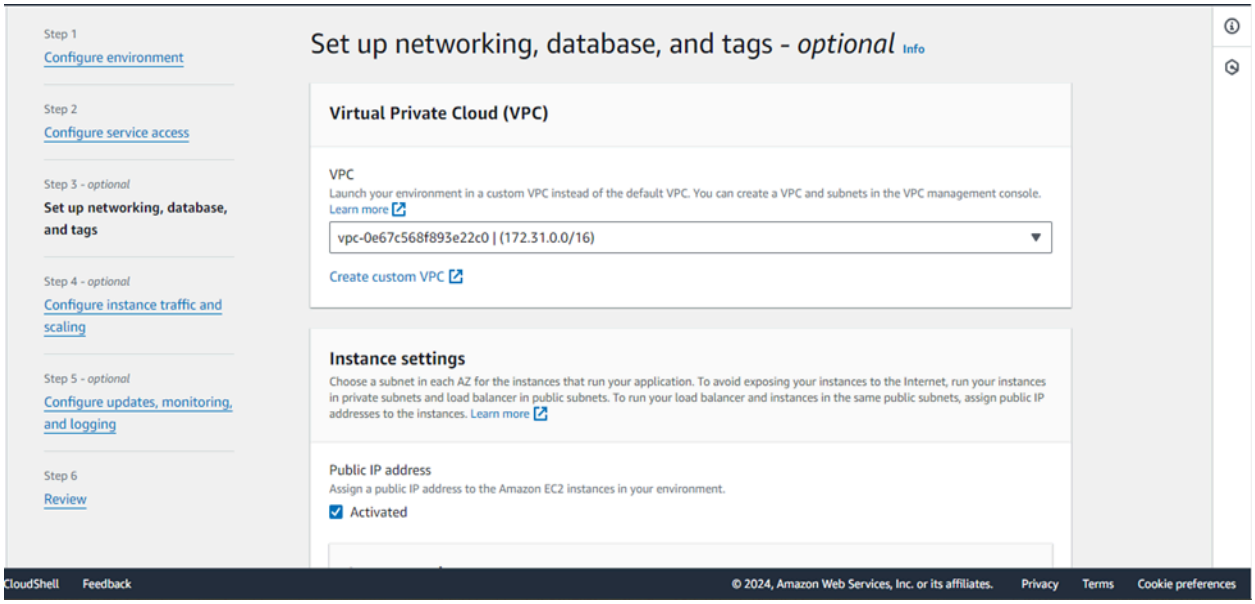
Pranav Titambe D15A/62

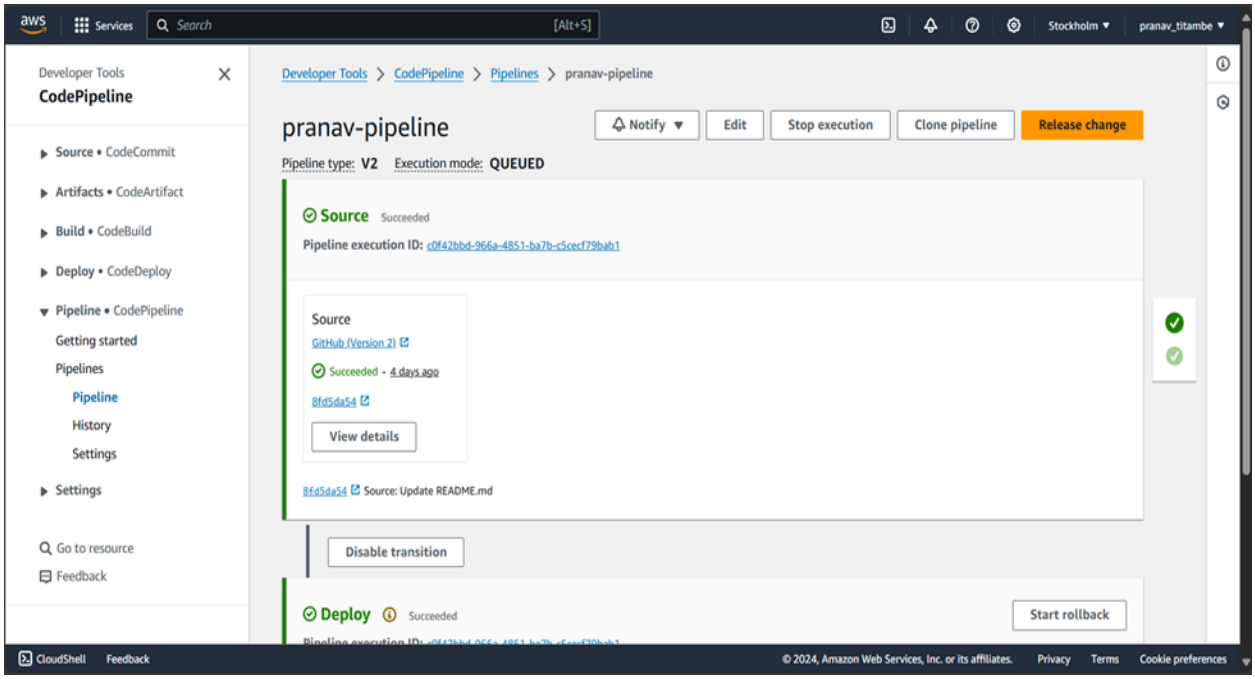
Aim: To build your application using AWS Codebuild and deploy on S3 using AWS CodePipeline deploy sample application on EC2 instance using AWS codedeploy.

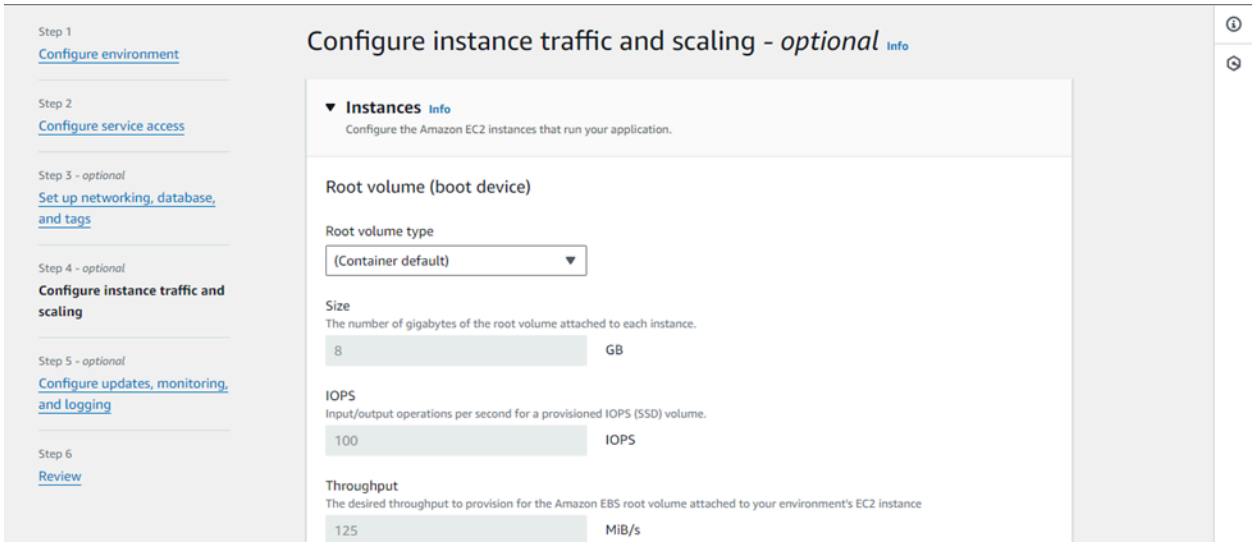
Code and Output :
Using elastic beanstalk:

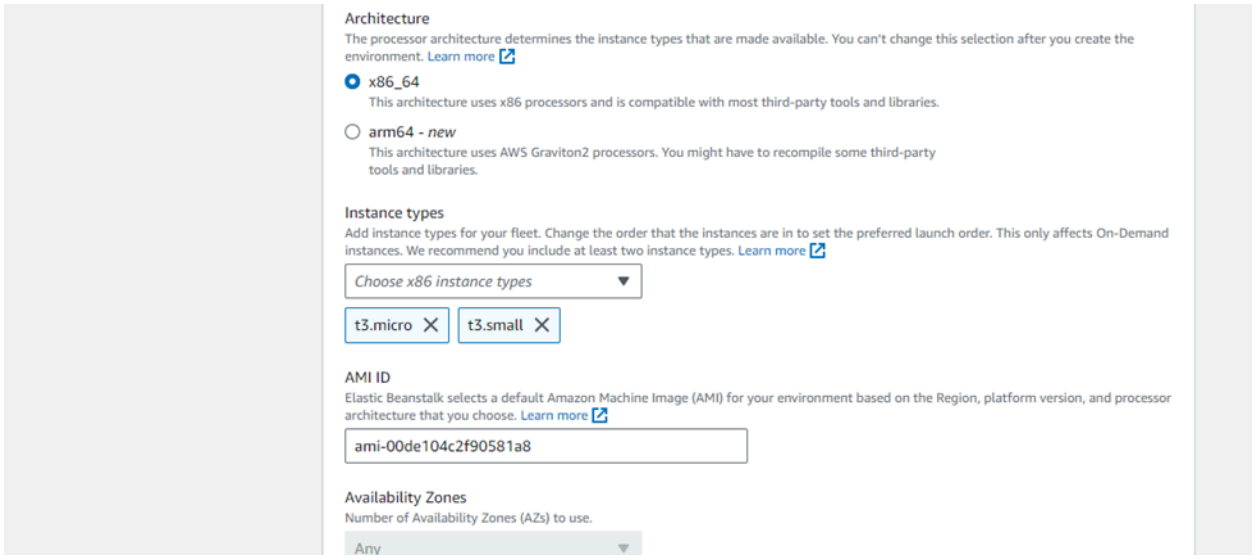


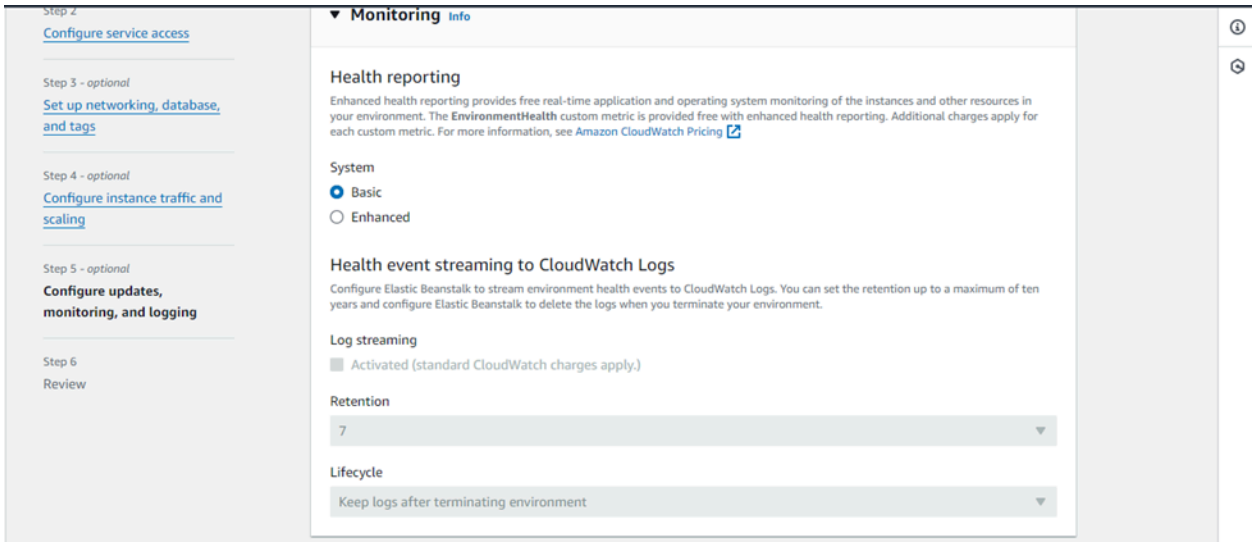


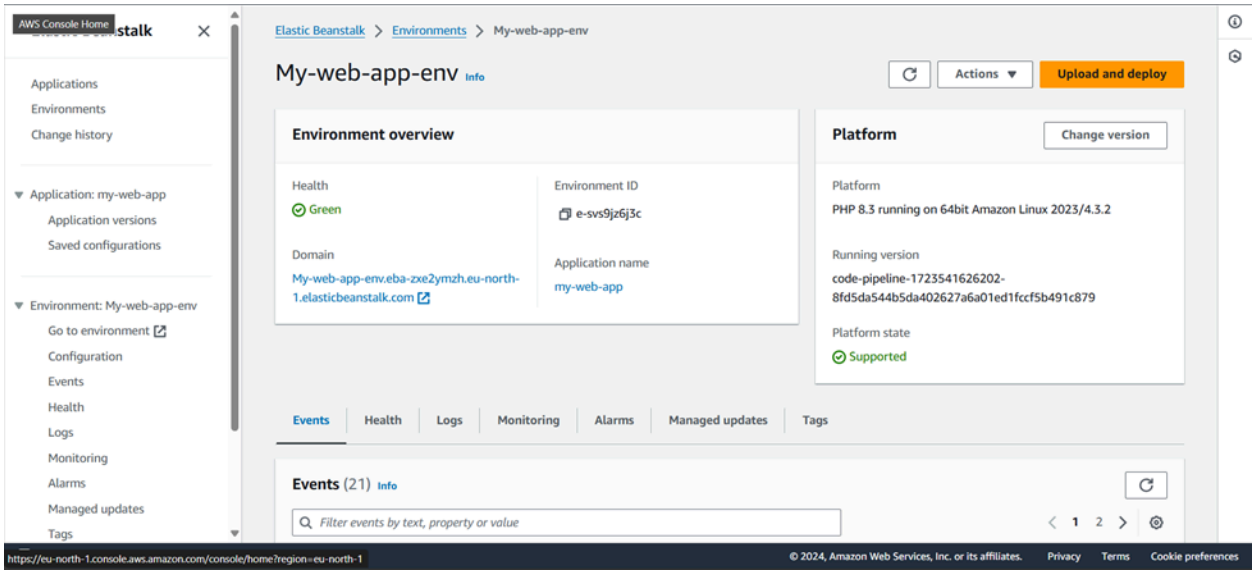

Step 1
Configure environment
Step 2
Configure service access
Step 3 - optional
Set up networking, database, and tags
Step 4 - optional
Configure instance traffic and scaling
Step 5 - optional
Configure updates, monitoring, and logging
Step 6
Review
Set up networking, database, and tags - optional Info
Virtual Private Cloud (VPC)
VPC
Launch your environment in a custom VPC instead of the default VPC. You can create a VPC and subnets in the VPC management console.
Learn more
vpc-0e67c568f893e22c0 | (172.31.0.0/16)
Create custom VPC
Instance settings
Choose a subnet in each AZ for the instances that run your application. To avoid exposing your instances to the Internet, run your instances in private subnets and load balancer in public subnets. To run your load balancer and instances in the same public subnets, assign public IP addresses to the instances. Learn more
Public IP address
Assign a public IP address to the Amazon EC2 instances in your environment.
Activated
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

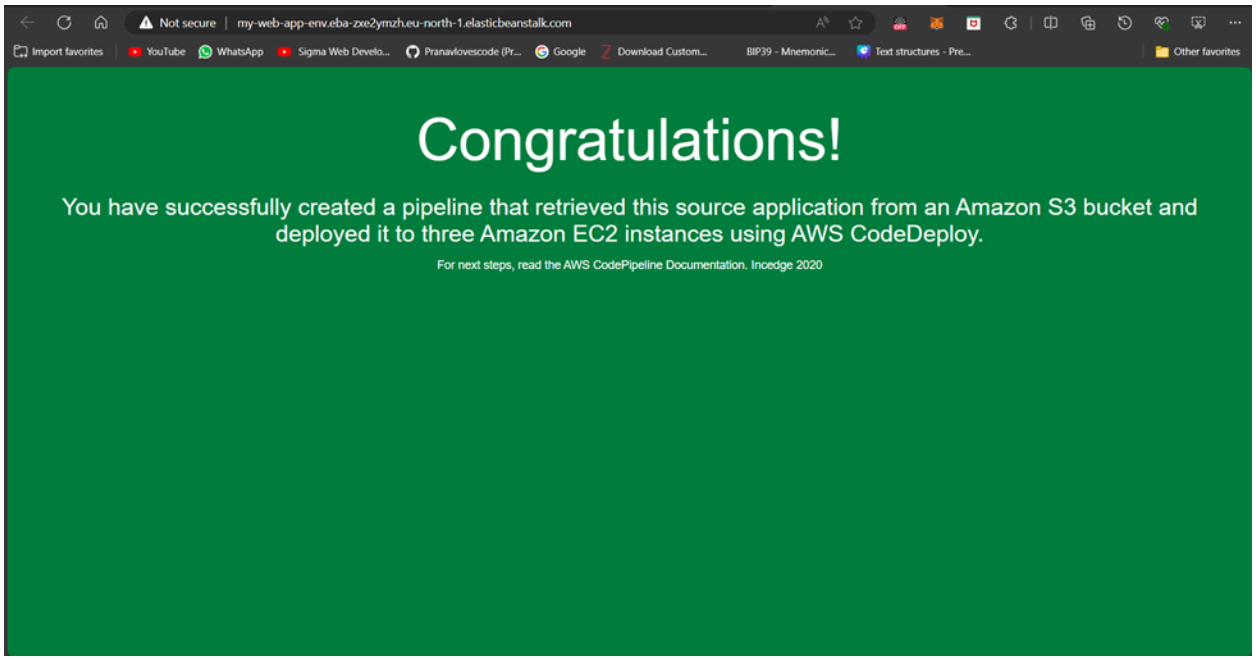

aws
Services
Search
[Alt+S]
Stockholm
pranav_titambe
Developer Tools
CodePipeline
Source • CodeCommit
Artifacts • CodeArtifact
Build • CodeBuild
Deploy • CodeDeploy
Pipeline • CodePipeline
Getting started
Pipelines
Pipeline
History
Settings
Settings
Go to resource
Feedback
Developer Tools > CodePipeline > Pipelines > pranav-pipeline
pranav-pipeline
Notify
Edit
Stop execution
Clone pipeline
Release change
Pipeline type: V2
Execution mode: QUEUED
Source
Succeeded
Pipeline execution ID: c0f42bbd-966a-4851-ba7b-c5cecf79bab1
Source
GitHub (Version 2)
Succeeded - 4 days ago
8fd5da54
View details
8fd5da54 Source: Update README.md
Disable transition
Deploy
Succeeded
Start rollback
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences


Step 1
Configure environment
Step 2
Configure service access
Step 3 - optional
Set up networking, database, and tags
Step 4 - optional
Configure instance traffic and scaling
Step 5 - optional
Configure updates, monitoring, and logging
Step 6
Review
Configure instance traffic and scaling - optional Info
Instances Info
Configure the Amazon EC2 instances that run your application.
Root volume (boot device)
Root volume type
(Container default)
Size
The number of gigabytes of the root volume attached to each instance.
8
GB
IOPS
Input/output operations per second for a provisioned IOPS (SSD) volume.
100
IOPS
Throughput
The desired throughput to provision for the Amazon EBS root volume attached to your environment's EC2 instance
125
MiB/s

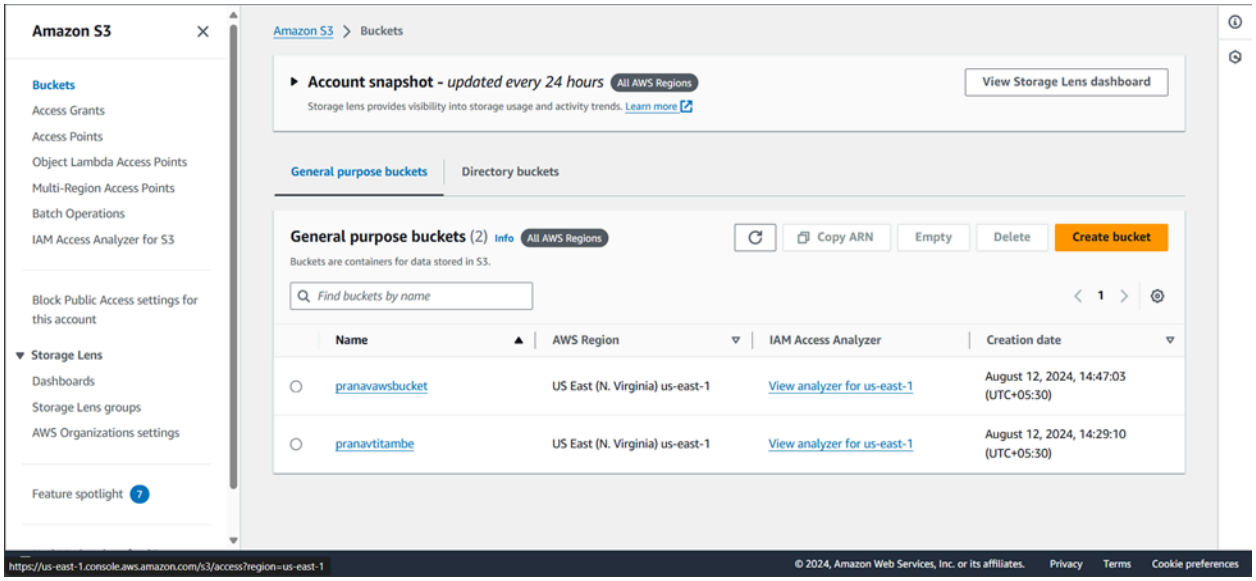

Architecture
The processor architecture determines the instance types that are made available. You can't change this selection after you create the environment. Learn more
x86_64
This architecture uses x86 processors and is compatible with most third-party tools and libraries.
arm64 - new
This architecture uses AWS Graviton2 processors. You might have to recompile some third-party tools and libraries.
Instance types
Add instance types for your fleet. Change the order that the instances are in to set the preferred launch order. This only affects On-Demand instances. We recommend you include at least two instance types. Learn more
Choose x86 instance types
t3.micro
t3.small
AMI ID
Elastic Beanstalk selects a default Amazon Machine Image (AMI) for your environment based on the Region, platform version, and processor architecture that you choose. Learn more
ami-00de104c2f90581a8
Availability Zones
Number of Availability Zones (AZs) to use.
Any

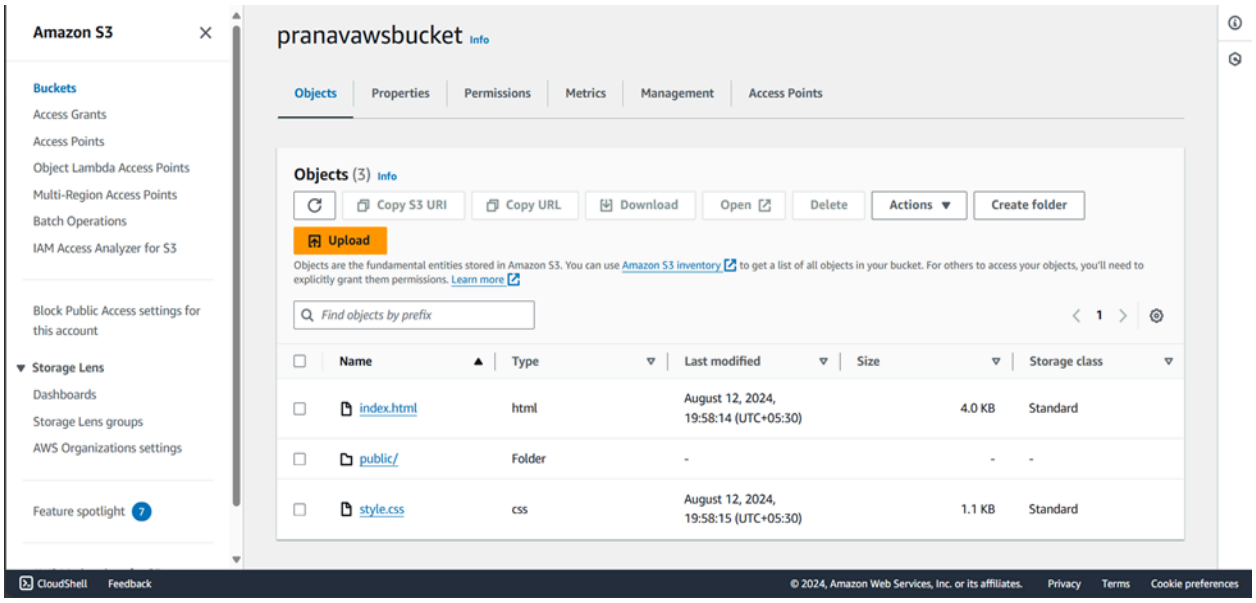

Step 2
Configure service access
Step 3 - optional
Set up networking, database, and tags
Step 4 - optional
Configure instance traffic and scaling
Step 5 - optional
Configure updates, monitoring, and logging
Step 6
Review
Monitoring Info
Health reporting
Enhanced health reporting provides free real-time application and operating system monitoring of the instances and other resources in your environment. The EnvironmentHealth custom metric is provided free with enhanced health reporting. Additional charges apply for each custom metric. For more information, see Amazon CloudWatch Pricing
System
Basic
Enhanced
Health event streaming to CloudWatch Logs
Configure Elastic Beanstalk to stream environment health events to CloudWatch Logs. You can set the retention up to a maximum of ten years and configure Elastic Beanstalk to delete the logs when you terminate your environment.
Log streaming
Activated (standard CloudWatch charges apply.)
Retention
7
Lifecycle
Keep logs after terminating environment

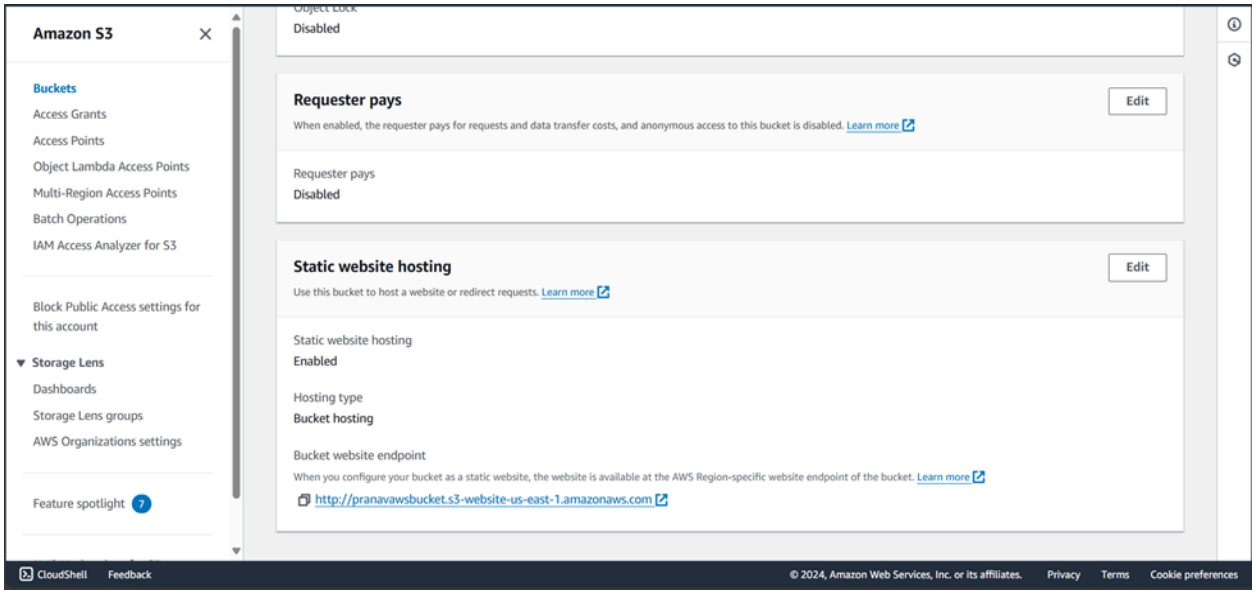

AWS Console Home
Applications
Environments
Change history
Application: my-web-app
Application versions
Saved configurations
Environment: My-web-app-env
Go to environment
Configuration
Events
Health
Logs
Monitoring
Alarms
Managed updates
Tags
Elastic Beanstalk > Environments > My-web-app-env
My-web-app-env Info
Actions
Upload and deploy
Environment overview
Health
Green
Environment ID
e-svs9jz6j3c
Domain
My-web-app-env.eba-zxe2ymzh.eu-north-1.elasticbeanstalk.com
Application name
my-web-app
Platform
Change version
Platform
PHP 8.3 running on 64bit Amazon Linux 2023/4.3.2
Running version
code-pipeline-1723541626202-8fd5da544b5da402627a6a01ed1fccf5b491c879
Platform state
Supported
Events
Health
Logs
Monitoring
Alarms
Managed updates
Tags
Events (21) Info
Filter events by text, property or value
https://eu-north-1.console.aws.amazon.com/console/home?region=eu-north-1
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

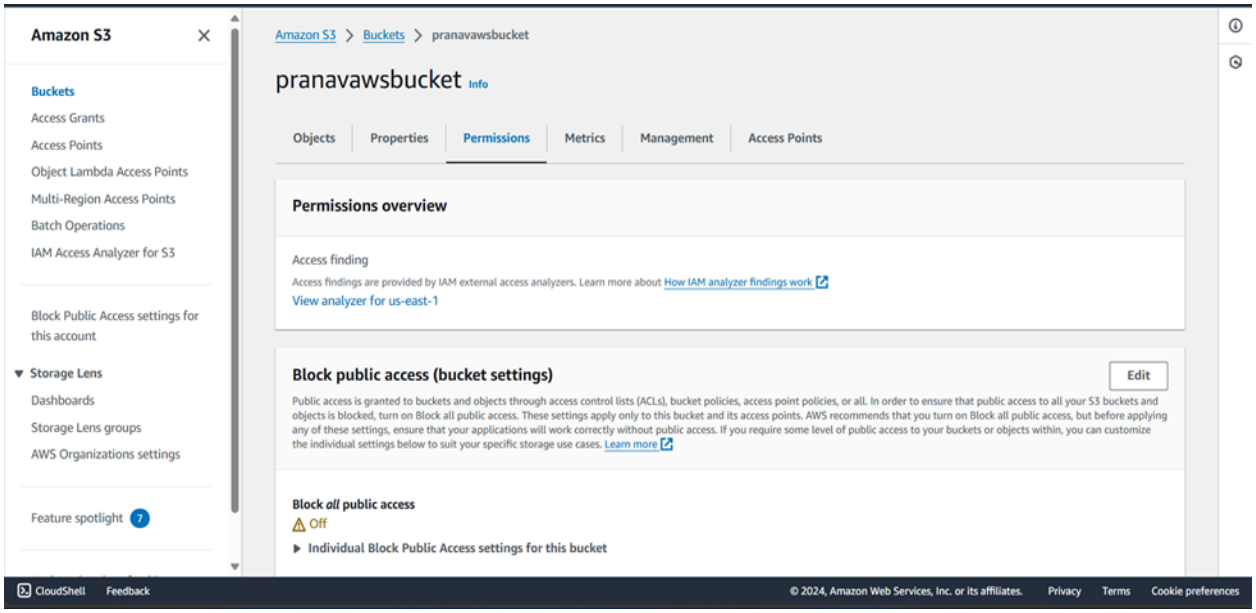

Not secure | my-web-app-env.eba-zxe2ymzh.eu-north-1.elasticbeanstalk.com
Import favorites
YouTube
WhatsApp
Sigma Web Develo...
Pranavlovescode (Pr...
Google
Download Custom...
BIP39 - Mnemonic...
Text structures - Pre...
Other favorites
Congratulations!
You have successfully created a pipeline that retrieved this source application from an Amazon S3 bucket and deployed it to three Amazon EC2 instances using AWS CodeDeploy.
For next steps, read the AWS CodePipeline Documentation. Incedge 2020

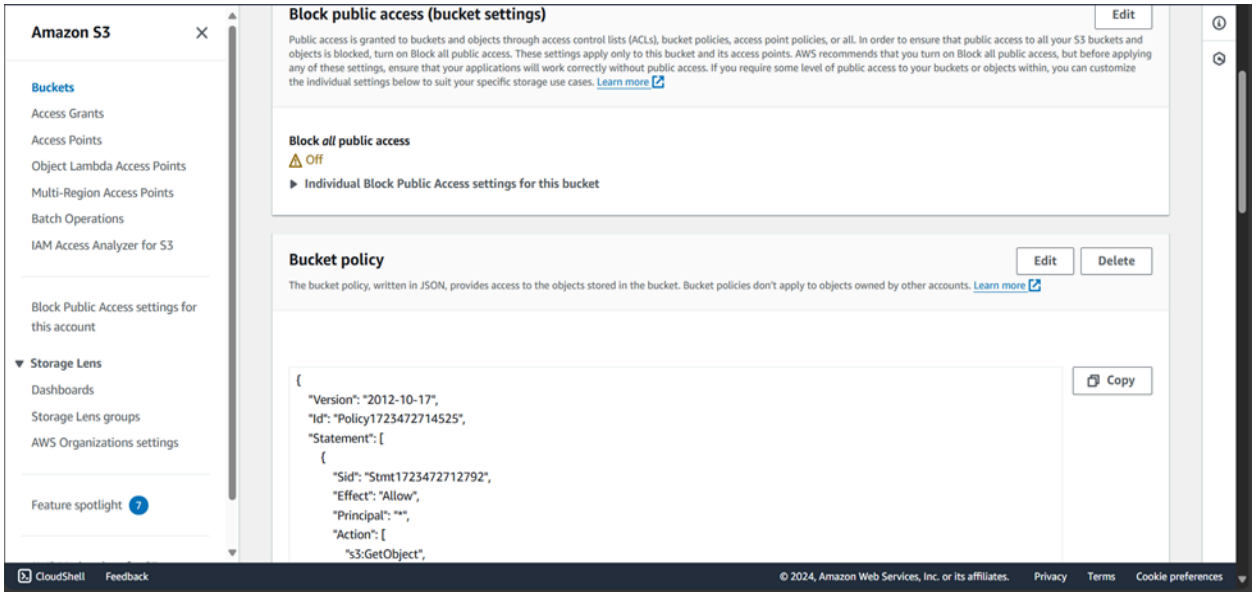
Using S3 Bucket:

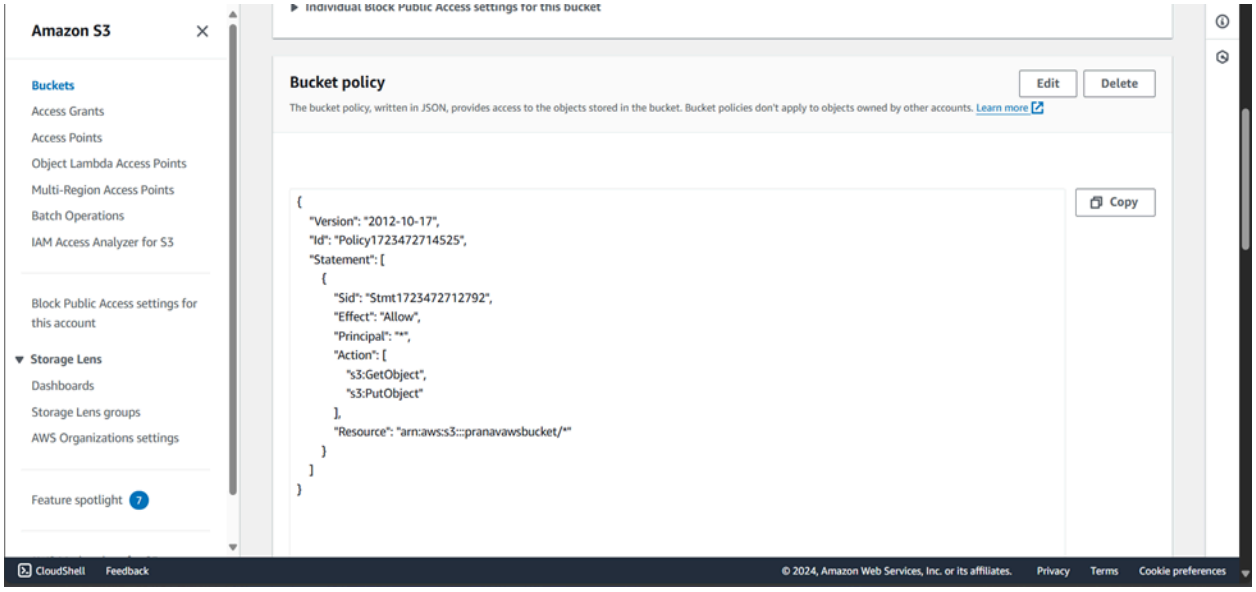


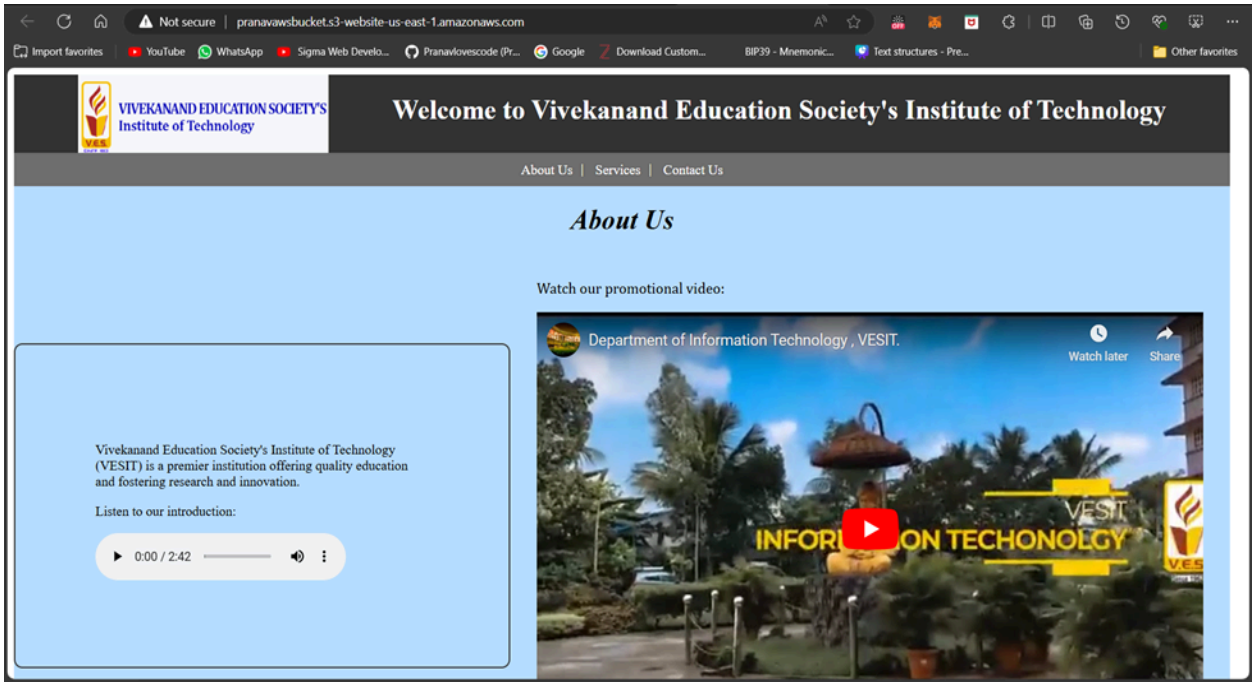



Amazon S3
Buckets
Access Grants
Access Points
Object Lambda Access Points
Multi-Region Access Points
Batch Operations
IAM Access Analyzer for S3
Block Public Access settings for this account
Storage Lens
Dashboards
Storage Lens groups
AWS Organizations settings
Feature spotlight 7
Disabled
Requester pays
When enabled, the requester pays for requests and data transfer costs, and anonymous access to this bucket is disabled. Learn more
Edit
Requester pays
Disabled
Static website hosting
Use this bucket to host a website or redirect requests. Learn more
Edit
Static website hosting
Enabled
Hosting type
Bucket hosting
Bucket website endpoint
When you configure your bucket as a static website, the website is available at the AWS Region-specific website endpoint of the bucket. Learn more
http://pranavawsbucket.s3-website-us-east-1.amazonaws.com
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

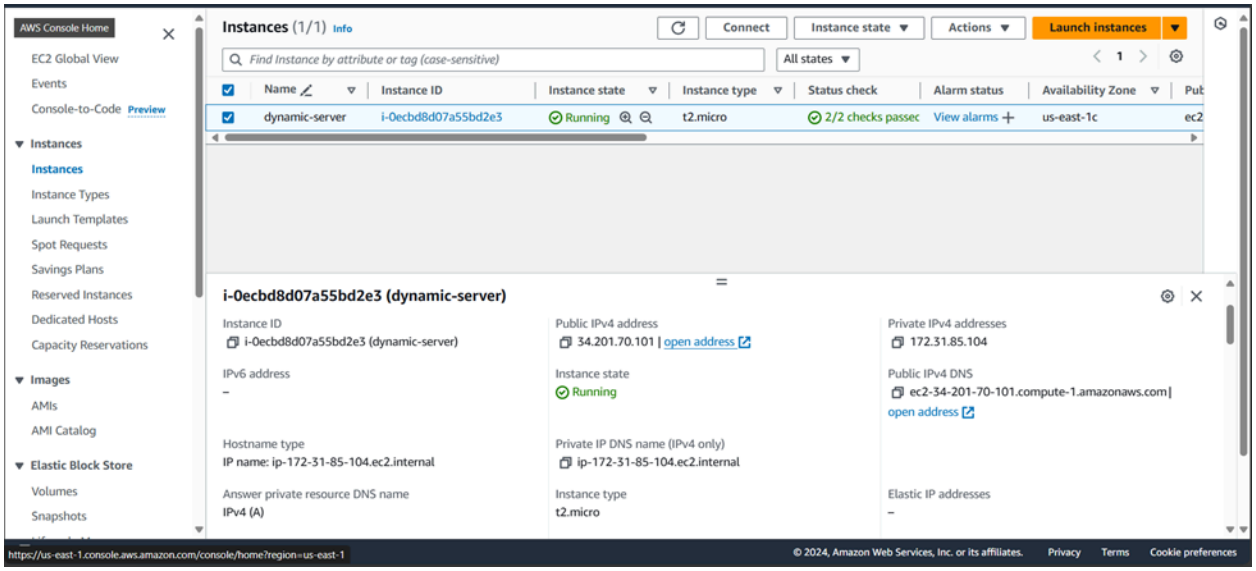

Amazon S3
Buckets
Access Grants
Access Points
Object Lambda Access Points
Multi-Region Access Points
Batch Operations
IAM Access Analyzer for S3
Block Public Access settings for this account
Storage Lens
Dashboards
Storage Lens groups
AWS Organizations settings
Feature spotlight 7
Amazon S3 > Buckets > pranavawsbucket
pranavawsbucket Info
Objects
Properties
Permissions
Metrics
Management
Access Points
Permissions overview
Access finding
Access findings are provided by IAM external access analyzers. Learn more about How IAM analyzer findings work
View analyzer for us-east-1
Block public access (bucket settings)
Edit
Public access is granted to buckets and objects through access control lists (ACLs), bucket policies, access point policies, or all. In order to ensure that public access to all your S3 buckets and objects is blocked, turn on Block all public access. These settings apply only to this bucket and its access points. AWS recommends that you turn on Block all public access, but before applying any of these settings, ensure that your applications will work correctly without public access. If you require some level of public access to your buckets or objects within, you can customize the individual settings below to suit your specific storage use cases. Learn more
Block all public access
Off
Individual Block Public Access settings for this bucket
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

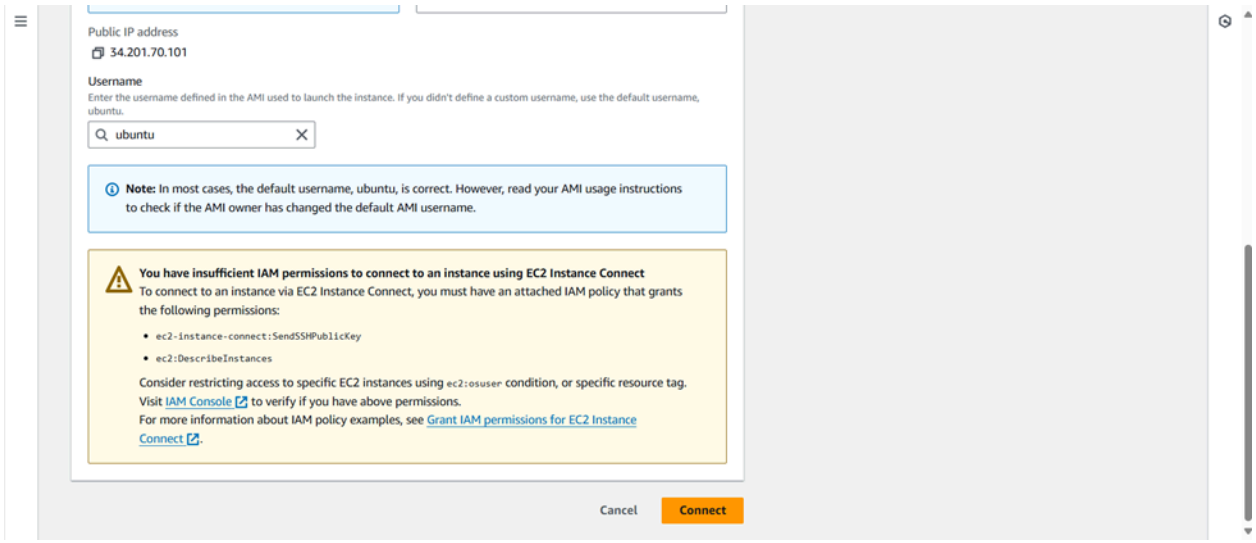

Amazon S3
Buckets
Access Grants
Access Points
Object Lambda Access Points
Multi-Region Access Points
Batch Operations
IAM Access Analyzer for S3
Block Public Access settings for this account
Storage Lens
Dashboards
Storage Lens groups
AWS Organizations settings
Feature spotlight 7
Block public access (bucket settings)
Edit
Public access is granted to buckets and objects through access control lists (ACLs), bucket policies, access point policies, or all. In order to ensure that public access to all your S3 buckets and objects is blocked, turn on Block all public access. These settings apply only to this bucket and its access points. AWS recommends that you turn on Block all public access, but before applying any of these settings, ensure that your applications will work correctly without public access. If you require some level of public access to your buckets or objects within, you can customize the individual settings below to suit your specific storage use cases. Learn more
Block all public access
Off
Individual Block Public Access settings for this bucket
Bucket policy
Edit
Delete
The bucket policy, written in JSON, provides access to the objects stored in the bucket. Bucket policies don't apply to objects owned by other accounts. Learn more
Copy
{
"Version": "2012-10-17",
"Id": "Policy1723472714525",
"Statement": [
{
"Sid": "Stmt1723472712792",
"Effect": "Allow",
"Principal": "*",
"Action": [
"s3:GetObject",
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences


Amazon S3
Buckets
Access Grants
Access Points
Object Lambda Access Points
Multi-Region Access Points
Batch Operations
IAM Access Analyzer for S3
Block Public Access settings for this account
Storage Lens
Dashboards
Storage Lens groups
AWS Organizations settings
Feature spotlight 7
Individual Block Public Access settings for this bucket
Bucket policy
Edit
Delete
The bucket policy, written in JSON, provides access to the objects stored in the bucket. Bucket policies don't apply to objects owned by other accounts. Learn more
Copy
{
"Version": "2012-10-17",
"Id": "Policy1723472714525",
"Statement": [
{
"Sid": "Stmt1723472712792",
"Effect": "Allow",
"Principal": "*",
"Action": [
"s3:GetObject",
"s3:PutObject"
],
"Resource": "arn:aws:s3:::pranavawsbucket/*"
}
]
}
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences


Not secure | pranavawsbucket.s3-website-us-east-1.amazonaws.com
Import favorites
YouTube
WhatsApp
Sigma Web Develo...
Pranavlovescode (Pr...
Google
Download Custom...
BIP39 - Mnemonic...
Text structures - Pre...
Other favorites
VIVEKANAND EDUCATION SOCIETY'S
Institute of Technology
Welcome to Vivekanand Education Society's Institute of Technology
About Us | Services | Contact Us
About Us
Watch our promotional video:
Vivekanand Education Society's Institute of Technology (VESIT) is a premier institution offering quality education and fostering research and innovation.
Listen to our introduction:
0:00 / 2:42
Department of Information Technology , VESIT.
Watch later
Share
VESIT
ON TECHONOLGY

Using EC2:

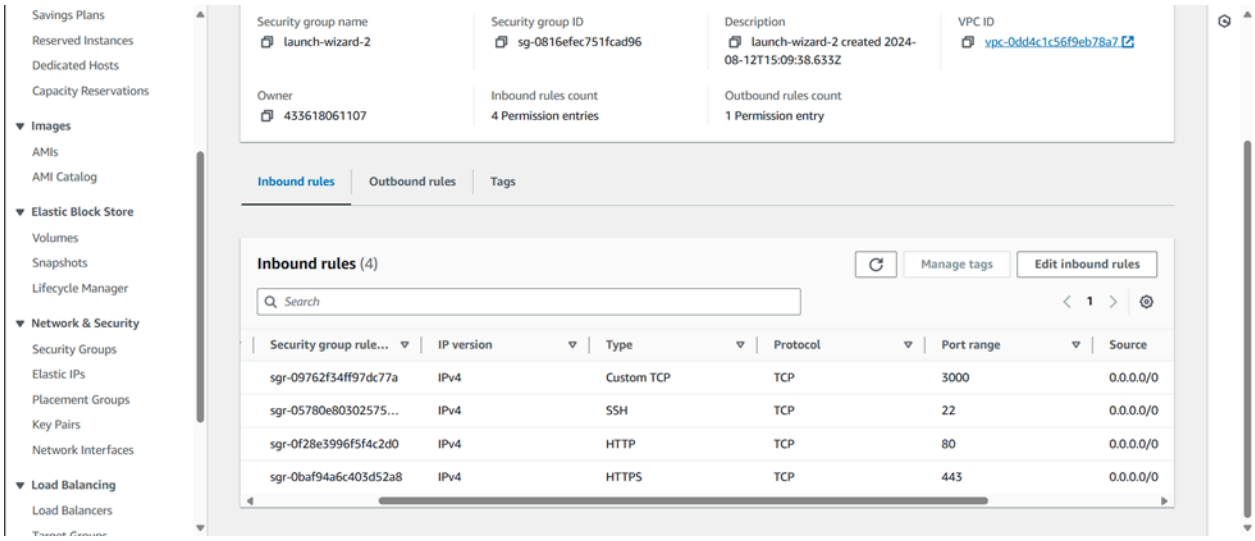


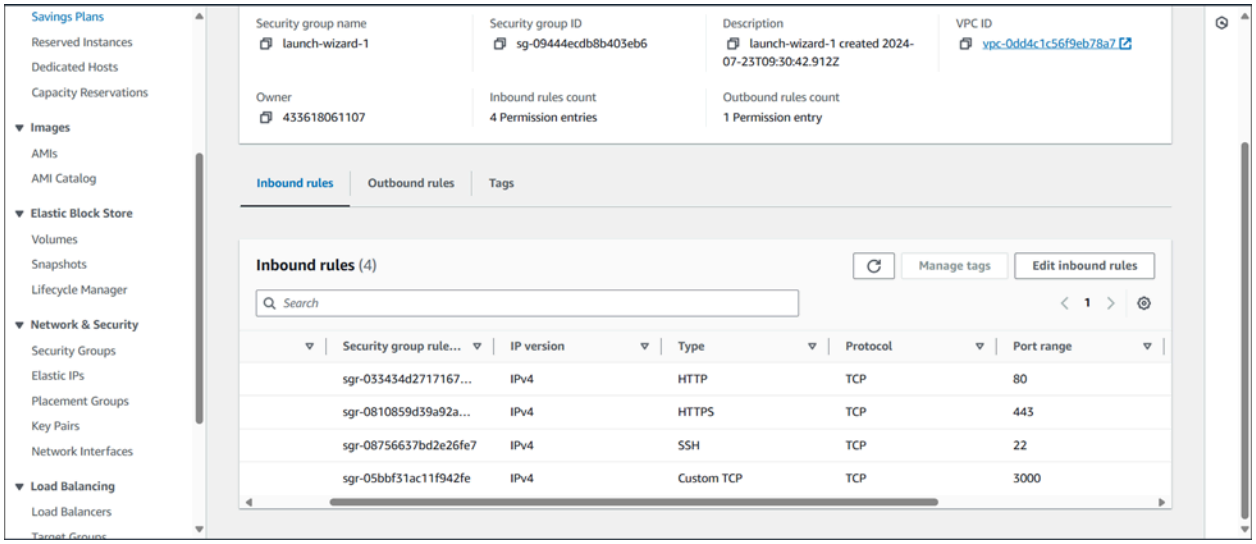



```
ubuntu@ip-172-31-85-104:~$ pwd
/home/ubuntu
ubuntu@ip-172-31-85-104:~$ mkdir pranav
ubuntu@ip-172-31-85-104:~$ cd pranav
ubuntu@ip-172-31-85-104:~/pranav$ git clone https://github.com/Pranavlovescode/Dynamic-website-hosting-sample.git
Cloning into 'Dynamic-website-hosting-sample'...
remote: Enumerating objects: 6, done.
remote: Counting objects: 100% (6/6), done.
remote: Compressing objects: 100% (5/5), done.
remote: Total 6 (delta 0), reused 6 (delta 0), pack-reused 0
Receiving objects: 100% (6/6), 11.16 KiB | 5.58 MiB/s, done.
```

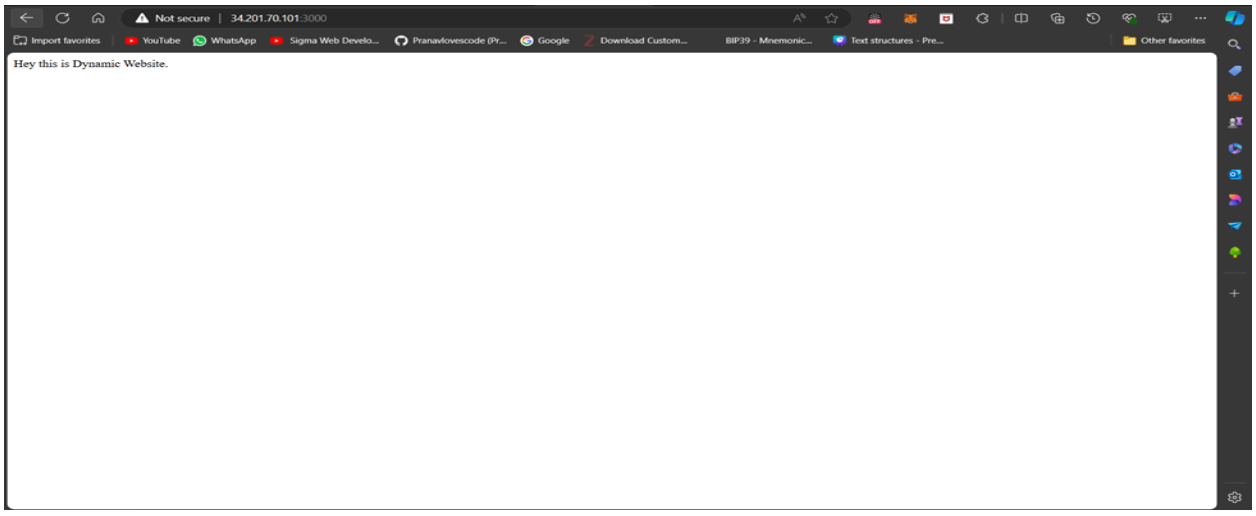
```
ubuntu@ip-172-31-85-104:~/pranav$ ls
Dynamic-website-hosting-sample
ubuntu@ip-172-31-85-104:~/pranav$ cd Dynamic-website-hosting-sample/
ubuntu@ip-172-31-85-104:~/pranav/Dynamic-website-hosting-sample$ ls
index.js  package-lock.json  package.json
ubuntu@ip-172-31-85-104:~/pranav/Dynamic-website-hosting-sample$ npm i

added 93 packages, and audited 94 packages in 3s

16 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
npm notice
npm notice New patch version of npm available! 10.8.1 -> 10.8.2
npm notice Changelog: https://github.com/npm/cli/releases/tag/v10.8.2
npm notice To update run: npm install -g npm@10.8.2
npm notice
```

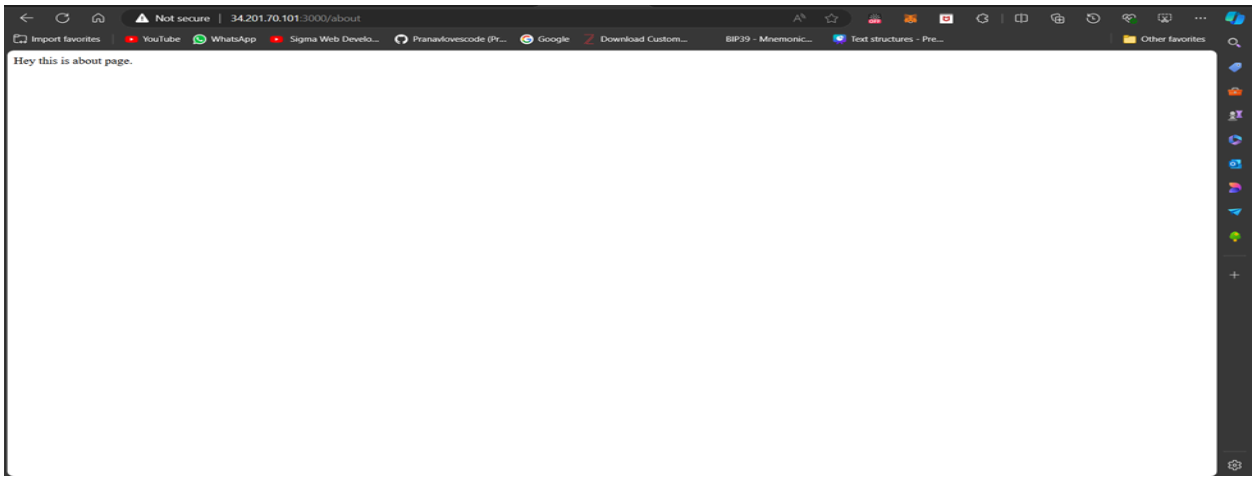
```
ubuntu@ip-172-31-85-104:~/pranav/Dynamic-website-hosting-sample$ npm start

> hosting-dynamic-website@1.0.0 start
> nodemon index.js

[nodemon] 3.1.4
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): *.*
[nodemon] watching extensions: js,mjs,cjs,json
[nodemon] starting `node index.js`
Server is running on port 3000
```



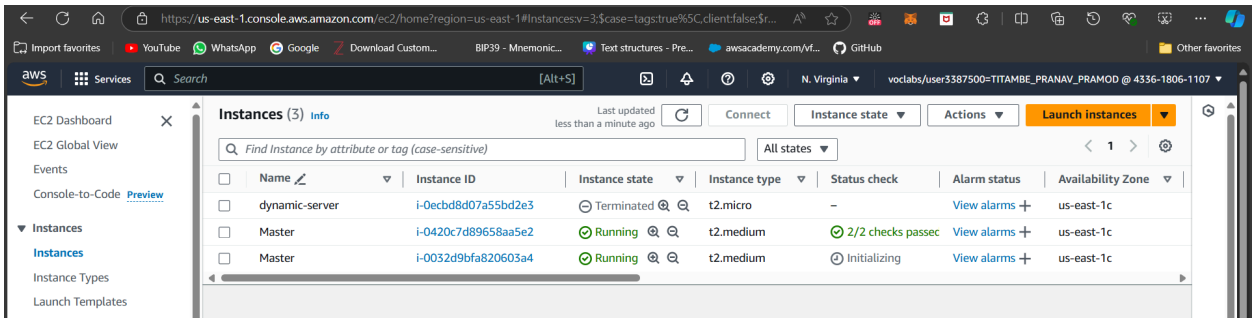

Savings Plans
Reserved Instances
Dedicated Hosts
Capacity Reservations
Images
AMIs
AMI Catalog
Elastic Block Store
Volumes
Snapshots
Lifecycle Manager
Network & Security
Security Groups
Elastic IPs
Placement Groups
Key Pairs
Network Interfaces
Load Balancing
Load Balancers
Target Groups
Security group name
launch-wizard-1
Security group ID
sg-09444ecdb8b403eb6
Description
launch-wizard-1 created 2024-07-23T09:30:42.912Z
VPC ID
vpc-0dd4c1c56f9eb78a7
Owner
433618061107
Inbound rules count
4 Permission entries
Outbound rules count
1 Permission entry
Inbound rules
Outbound rules
Tags
Inbound rules (4)
Manage tags
Edit inbound rules
Search
1
Security group rule...
IP version
Type
Protocol
Port range
sgr-033434d2717167...
IPv4
HTTP
TCP
80
sgr-0810859d39a92a...
IPv4
HTTPS
TCP
443
sgr-08756637bd2e26fe7
IPv4
SSH
TCP
22
sgr-05bbf31ac11f942fe
IPv4
Custom TCP
TCP
3000

Hosting:

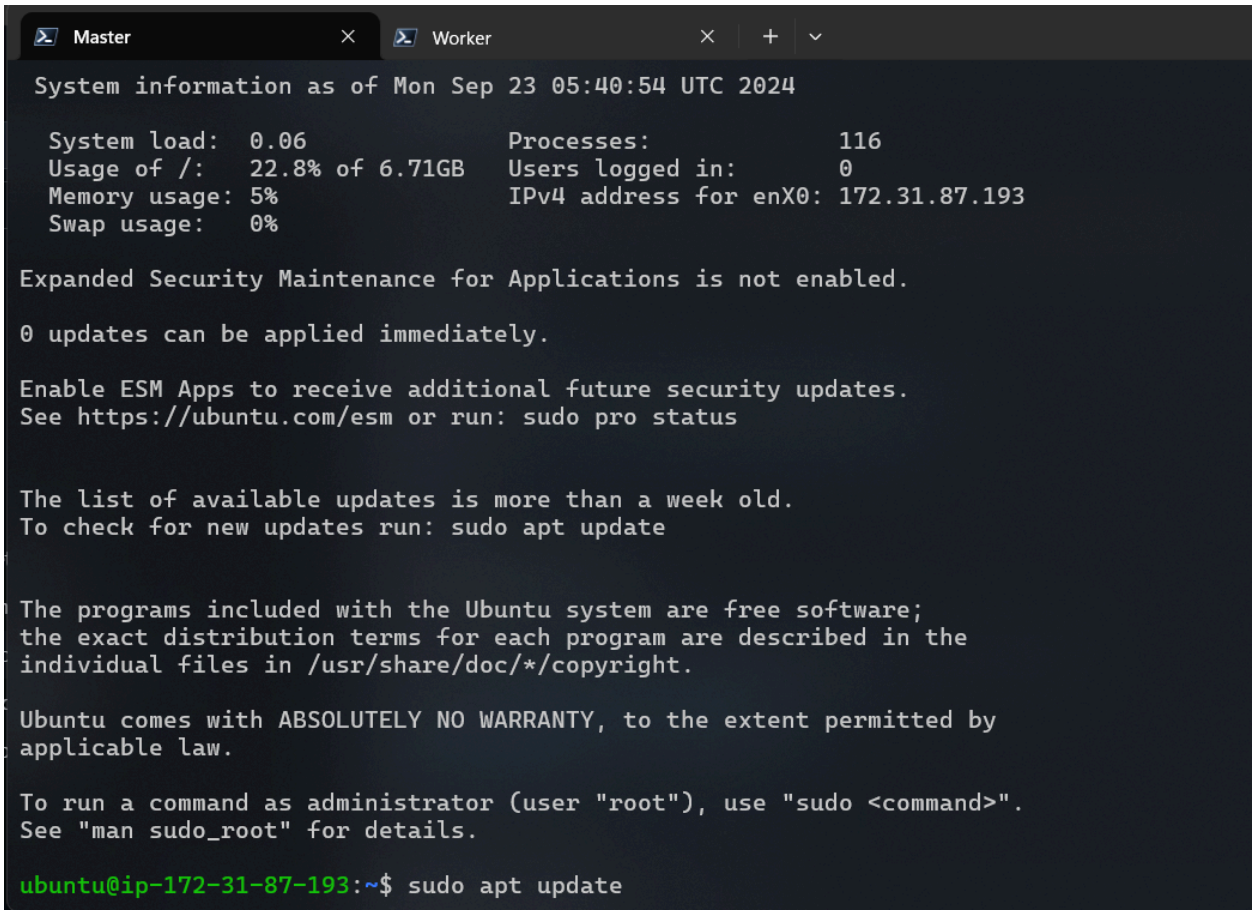

Not secure | 34.201.70.101:3000
Hey this is Dynamic Website.


Not secure | 34.201.70.101:3000/about
Hey this is about page.

Name- Pranav Titambe Rollno. - 62 Class - D15A



Master node



Worker node

```
 System information as of Mon Sep 23 05:42:31 UTC 2024

  System load:  0.0               Processes:             116
  Usage of /:   22.8% of 6.71GB   Users logged in:       0
  Memory usage: 5%                IPv4 address for enX0: 172.31.85.68
  Swap usage:   0%

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status


The list of available updates is more than a week old.
To check for new updates run: sudo apt update


The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

ubuntu@ip-172-31-85-68:~$ sudo apt update
```

Operations on both the nodes

```
ubuntu@ip-172-31-87-193:~$ sudo swapoff -a
ubuntu@ip-172-31-87-193:~$ cat <<EOF | sudo tee /etc/modules-load.d/k8s.conf
overlay
br_netfilter
EOF
overlay
br_netfilter
ubuntu@ip-172-31-87-193:~$
```

```
ubuntu@ip-172-31-85-68:~$ sudo modprobe overlay
sudo modprobe br_netfilter
ubuntu@ip-172-31-85-68:~$ cat <<EOF | sudo tee /etc/sysctl.d/k8s.conf
net.bridge.bridge-nf-call-iptables  = 1
net.bridge.bridge-nf-call-ip6tables = 1
net.ipv4.ip_forward                 = 1
EOF
net.bridge.bridge-nf-call-iptables  = 1
net.bridge.bridge-nf-call-ip6tables = 1
net.ipv4.ip_forward                 = 1
ubuntu@ip-172-31-85-68:~$
```

```
ubuntu@ip-172-31-87-193:~$ sudo curl -fsSL https://pkgs.k8s.io/addons:/cri-o:/prerelease:/main/deb/Release.key | sudo gp
g --dearmor -o /etc/apt/keyrings/cri-o-apt-keyring.gpg
echo "deb [signed-by=/etc/apt/keyrings/cri-o-apt-keyring.gpg] https://pkgs.k8s.io/addons:/cri-o:/prerelease:/main/deb/ /
" | sudo tee /etc/apt/sources.list.d/cri-o.list
deb [signed-by=/etc/apt/keyrings/cri-o-apt-keyring.gpg] https://pkgs.k8s.io/addons:/cri-o:/prerelease:/main/deb/ /
ubuntu@ip-172-31-87-193:~$
```

```
ubuntu@ip-172-31-87-193:~$ sudo systemctl daemon-reload
sudo systemctl enable crio --now
sudo systemctl start crio.service
ubuntu@ip-172-31-87-193:~$
```

```
ubuntu@ip-172-31-87-193:~$ sudo systemctl daemon-reload
sudo systemctl enable crio --now
sudo systemctl start crio.service
ubuntu@ip-172-31-87-193:~$ echo "CRI runtime installed successfully"
CRI runtime installed successfully
ubuntu@ip-172-31-87-193:~$
```

```
ubuntu@ip-172-31-87-193:~$ sudo apt-get update -y
sudo apt-get install -y kubelet="1.29.0-*" kubectl="1.29.0-*" kubeadm="1.29.0-*"
sudo apt-get update -y
sudo apt-get install -y jq
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Hit:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease
Hit:4 http://security.ubuntu.com/ubuntu noble-security InRelease
Hit:5 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/addons:/cri-o:/prerelease:/main/deb  InRelease
Get:6 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.29/deb  InRelease [1189 B]
Get:7 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.29/deb  Packages [14.0 kB]
Fetched 15.1 kB in 0s (32.4 kB/s)
Reading package lists... Done
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Selected version '1.29.0-1.1' (isv:kubernetes:core:stable:v1.29:pkgs.k8s.io [amd64]) for 'kubelet'
Selected version '1.29.0-1.1' (isv:kubernetes:core:stable:v1.29:pkgs.k8s.io [amd64]) for 'kubectl'
Selected version '1.29.0-1.1' (isv:kubernetes:core:stable:v1.29:pkgs.k8s.io [amd64]) for 'kubeadm'
```

```
ubuntu@ip-172-31-87-193:~$ sudo systemctl enable --now kubelet
sudo systemctl start kubelet
ubuntu@ip-172-31-87-193:~$
```

Some operations on Master node only

```
ubuntu@ip-172-31-87-193:~$ sudo kubeadm config images pull
I0923 18:19:32.389246    3210 version.go:256] remote version is much newer: v1.31.0; falling back to: stable-1.29
[config/images] Pulled registry.k8s.io/kube-apiserver:v1.29.9
[config/images] Pulled registry.k8s.io/kube-controller-manager:v1.29.9
[config/images] Pulled registry.k8s.io/kube-scheduler:v1.29.9
[config/images] Pulled registry.k8s.io/kube-proxy:v1.29.9
[config/images] Pulled registry.k8s.io/coredns/coredns:v1.11.1
[config/images] Pulled registry.k8s.io/pause:3.9
[config/images] Pulled registry.k8s.io/etcd:3.5.10-0
ubuntu@ip-172-31-87-193:~$
```

```
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 172.31.87.193:6443 --token ps609h.u9595jvtvtmdxdde \
        --discovery-token-ca-cert-hash sha256:b999d95c92f13fe95df32236ae5cb5c50e2a4e5dc9c00cf44d65a4c64869ec53
ubuntu@ip-172-31-87-193:~$
```

```
ubuntu@ip-172-31-87-193:~$ mkdir -p "$HOME"/.kube
sudo cp -i /etc/kubernetes/admin.conf "$HOME"/.kube/config
sudo chown "$(id -u)":"$(id -g)" "$HOME"/.kube/config
ubuntu@ip-172-31-87-193:~$ kubectl apply -f https://raw.githubusercontent.com/projectcalico/calico/v3.26.0/manifests/cal
ico.yaml
poddisruptionbudget.policy/calico-kube-controllers created
serviceaccount/calico-kube-controllers created
serviceaccount/calico-node created
serviceaccount/calico-cni-plugin created
configmap/calico-config created
customresourcedefinition.apiextensions.k8s.io/bgpconfigurations.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/bgpfilters.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/bgppeers.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/blockaffinities.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/caliconodestatuses.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/clusterinformations.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/felixconfigurations.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/globalnetworkpolicies.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/globalnetworksets.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/hostendpoints.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/ipamblocks.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/ipamconfigs.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/ipamhandles.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/ippools.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/ipreservations.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/kubecontrollersconfigurations.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/networkpolicies.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/networksets.crd.projectcalico.org created
clusterrole.rbac.authorization.k8s.io/calico-kube-controllers created
clusterrole.rbac.authorization.k8s.io/calico-node created
```

```
ubuntu@ip-172-31-85-68:~$ sudo kubeadm join 172.31.87.193:6443 --token e4qbqx.6fmmeowzhklspzk8 --discovery-token-ca-cert
-hash sha256:b999d95c92f13fe95df32236ae5cb5c50e2a4e5dc9c00cf44d65a4c64869ec53 --v=5
I0923 19:03:35.769113    2876 join.go:413] [preflight] found NodeName empty; using OS hostname as NodeName
I0923 19:03:35.769235    2876 initconfiguration.go:122] detected and using CRI socket: unix:///var/run/crio/crio.sock
[preflight] Running pre-flight checks
I0923 19:03:35.769292    2876 preflight.go:93] [preflight] Running general checks
I0923 19:03:35.769322    2876 checks.go:280] validating the existence of file /etc/kubernetes/kubelet.conf
I0923 19:03:35.769329    2876 checks.go:280] validating the existence of file /etc/kubernetes/bootstrap-kubelet.conf
I0923 19:03:35.769337    2876 checks.go:104] validating the container runtime
I0923 19:03:35.789662    2876 checks.go:639] validating whether swap is enabled or not
I0923 19:03:35.790675    2876 checks.go:370] validating the presence of executable crictl
I0923 19:03:35.790703    2876 checks.go:370] validating the presence of executable conntrack
I0923 19:03:35.790719    2876 checks.go:370] validating the presence of executable ip
I0923 19:03:35.790736    2876 checks.go:370] validating the presence of executable iptables
I0923 19:03:35.790750    2876 checks.go:370] validating the presence of executable mount
I0923 19:03:35.790767    2876 checks.go:370] validating the presence of executable nsenter
I0923 19:03:35.790779    2876 checks.go:370] validating the presence of executable ebtables
I0923 19:03:35.790791    2876 checks.go:370] validating the presence of executable ethtool
I0923 19:03:35.790801    2876 checks.go:370] validating the presence of executable socat
I0923 19:03:35.790813    2876 checks.go:370] validating the presence of executable tc
I0923 19:03:35.790823    2876 checks.go:370] validating the presence of executable touch
I0923 19:03:35.790836    2876 checks.go:516] running all checks
I0923 19:03:35.802590    2876 checks.go:401] checking whether the given node name is valid and reachable using net.Looku
pHost
I0923 19:03:35.803633    2876 checks.go:605] validating kubelet version
I0923 19:03:35.857173    2876 checks.go:130] validating if the "kubelet" service is enabled and active
I0923 19:03:35.867430    2876 checks.go:203] validating availability of port 10250
```

```
ubuntu@ip-172-31-87-193:~$ kubectl get nodes
NAME                STATUS   ROLES           AGE     VERSION
ip-172-31-85-68     Ready    <none>          7m35s   v1.29.0
ip-172-31-87-193    Ready    control-plane   43m     v1.29.0
ubuntu@ip-172-31-87-193:~$
```

EXPERIMENT NO. 4

NAME : PRANAV TITAMBE CLASS : D15A ROLL NO. : 62

Aim : To install Kubectl and execute Kubectl commands to manage the Kubernetes cluster and deploy Your First Kubernetes Application.

Theory:

What is kubectl?

kubectl is the command-line interface (CLI) used to interact with a Kubernetes cluster. It allows users to manage cluster resources, deploy applications, inspect and manage cluster components, and much more. Using kubectl, you can communicate with the Kubernetes API server to issue commands and queries.

Common kubectl commands:

- kubectl get: View information about resources.
- kubectl describe: Detailed description of resources.
- kubectl create/apply: Create or update resources.
- kubectl delete: Delete resources.

kubectl plays a crucial role in the day-to-day operation of a Kubernetes cluster.

Basic Concepts in Kubernetes

Before diving into the application deployment process, it's important to understand a few key Kubernetes objects:

1. **Pods**: The smallest deployable unit in Kubernetes. A pod encapsulates one or more containers (usually a single container) that share the same network namespace and storage.
2. **Deployments**: A Kubernetes resource that defines how to create and manage pods. It ensures the specified number of pod replicas are running at any given time and handles updates and rollbacks.
3. **Services**: An abstraction that defines how to access the pods. A service allows you to expose your pods to internal or external clients.
4. **ReplicaSets**: Ensures that a specified number of pod replicas are running at all times. It is managed by a Deployment, but can also be used independently.

Step 1: Install Kubectl on Ubuntu

1.1 Add Kubernetes APT repository

First, add the Kubernetes repository to your system.

1. **Install prerequisites:**

sudo apt-get update

sudo apt-get install -y apt-transport-https ca-certificates curl

```
ubuntu@ip-172-31-44-131:~$ sudo apt-get update
Hit:1 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Hit:2 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:3 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease
Hit:4 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/addons:/cri-o:/prerelease:/main/deb  InRelease
Hit:5 http://security.ubuntu.com/ubuntu noble-security InRelease
Hit:6 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.29/deb  InRelease
Reading package lists... Done
```

```
ubuntu@ip-172-31-44-131:~$ sudo apt-get install -y apt-transport-https ca-certificates curl
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
apt-transport-https is already the newest version (2.7.14build2).
ca-certificates is already the newest version (20240203).
curl is already the newest version (8.5.0-2ubuntu10.4).
0 upgraded, 0 newly installed, 0 to remove and 10 not upgraded.
```

2. **Add the GPG key for Kubernetes:**

sudo curl -fsSLo /usr/share/keyrings/kubernetes-archive-keyring.gpg https://packages.cloud.google.com/apt/doc/apt-key.gpg

3. **Add the Kubernetes repository:**

echo "deb [signed-by=/usr/share/keyrings/kubernetes-archive-keyring.gpg] https://apt.kubernetes.io/ kubernetes-focal main" | sudo tee

/etc/apt/sources.list.d/kubernetes.list

```
ubuntu@ip-172-31-44-131:~$ echo "deb [signed-by=/usr/share/keyrings/kubernetes-archive-keyring.gpg] https://apt.kubernetes.io/ kubernetes-focal main" | sudo
 tee /etc/apt/sources.list.d/kubernetes.list
deb [signed-by=/usr/share/keyrings/kubernetes-archive-keyring.gpg] https://apt.kubernetes.io/ kubernetes-focal main
```

1.2 Install kubectl Now install kubectl: sudo apt-get update

sudo apt-get install -y kubectl

```
ubuntu@ip-172-31-44-131:~$ sudo apt-get update
Hit:1 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Hit:2 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:3 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease
Hit:4 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/addons:/cri-o:/prerelease:/main/deb  InRelease
Hit:6 http://security.ubuntu.com/ubuntu noble-security InRelease
Ign:5 https://packages.cloud.google.com/apt kubernetes-focal InRelease
Err:7 https://packages.cloud.google.com/apt kubernetes-focal Release
  404  Not Found [IP: 142.250.76.206 443]
Reading package lists... Done
E: The repository 'https://apt.kubernetes.io kubernetes-focal Release' does not have a Release file.
N: Updating from such a repository can't be done securely, and is therefore disabled by default.
N: See apt-secure(8) manpage for repository creation and user configuration details.
```

```
ubuntu@ip-172-31-44-131:~$ sudo apt-get install -y kubectl
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
kubectl is already the newest version (1.29.0-1.1).
0 upgraded, 0 newly installed, 0 to remove and 7 not upgraded.
ubuntu@ip-172-31-44-131:~$
```

Verify the installation(extra): kubectlversion --client

```
ubuntu@ip-172-31-44-131:~$ kubectl version --client
Client Version: v1.29.0
Kustomize Version: v5.0.4-0.20230601165947-6ce0bf390ce3
```

Step 2: Deploying Your Application on Kubernetes

2.1 Set up Kubernetes Cluster

1. If you haven't already set up a Kubernetes cluster (e.g., with kubeadm), use minikube or any managed Kubernetes service (like EKS, GKE, etc.) to get a cluster running.

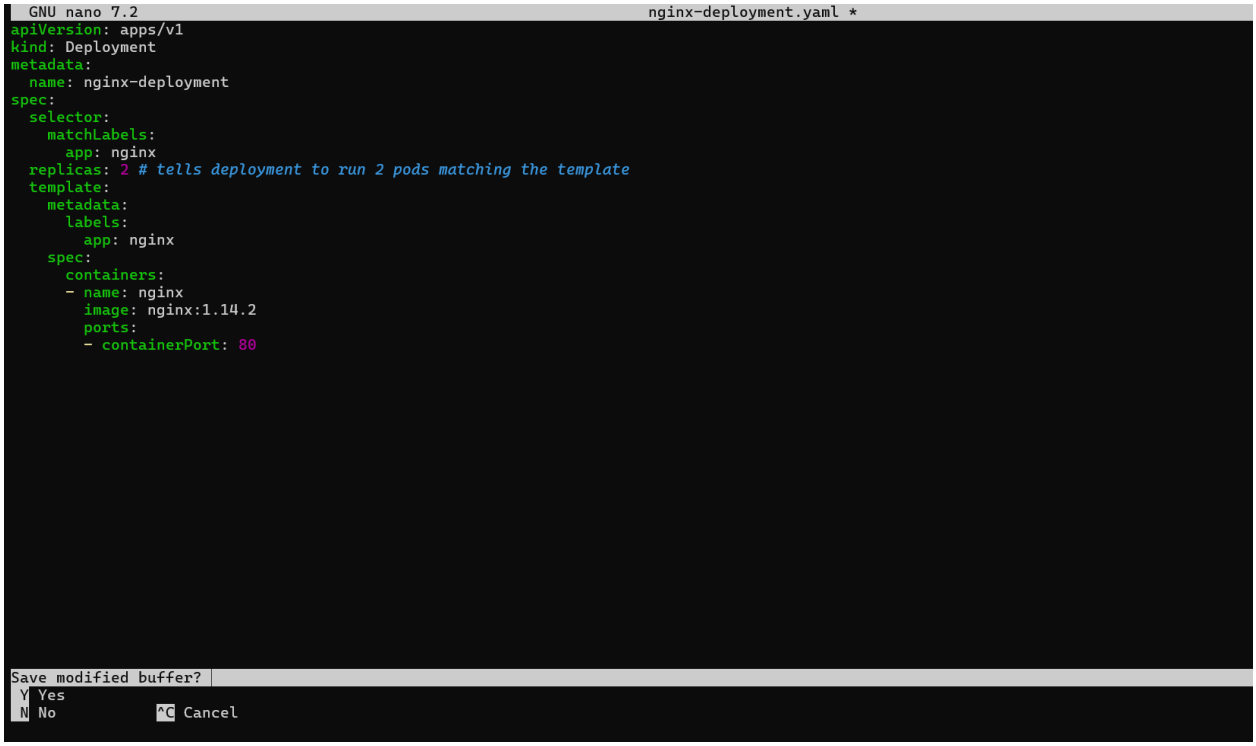
2. Once your cluster is ready, verify the nodes:

kubectl get nodes

```
ubuntu@ip-172-31-44-131:~$ kubectl get nodes
NAME               STATUS   ROLES           AGE     VERSION
ip-172-31-40-114   Ready    <none>          9m55s   v1.29.0
ip-172-31-44-131   Ready    control-plane   33m     v1.29.0
```

Step 3:**Create the Deployment YAML file**

a)Create the YAML file: Use a text editor to create a file named nginx-deployment.yaml
Add the Deployment Configuration: Copy and paste the following YAML content into the file. Save and exit the editor (Press Ctrl+X, then Y, and Enter).



Step 4:Create the Service YAML File

a) Create the YAML File: Create another file named nginx-service.yaml Add the Service Configuration: Copy and paste the following YAML content into the file given below.

Step 5:Apply the YAML Files

a) Deploy the Application: Use kubectl to create the Deployment and Service from the YAML files.

Verify the Deployment: Check the status of your Deployment,Pods and Services.

Describe the deployment(Extra)

```
ubuntu@ip-172-31-44-131:~$ ubuntu@ip-172-31-44-131:~$ kubectl apply -f nginx-deployment.yaml
deployment.apps/nginx-deployment created
ubuntu@ip-172-31-44-131:~$ kubectl apply -f nginx-service.yaml
service/my-nginx created
```

Step 6:Ensure Service is Running

6.1 **Verify Service**: Run the following command to check the services running in your cluster:

Kubectl get deployment

Kubectl get pods

kubectl get service

```
ubuntu@ip-172-31-44-131:~$ kubectl get deployments
NAME               READY   UP-TO-DATE   AVAILABLE   AGE
nginx-deployment   2/2     2            2           74s
ubuntu@ip-172-31-44-131:~$ kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
nginx-deployment-86dcfdf4c6-8d7rx   1/1     Running   0          81s
nginx-deployment-86dcfdf4c6-bdbcm   1/1     Running   0          81s
ubuntu@ip-172-31-44-131:~$ kubectl get services
NAME         TYPE        CLUSTER-IP       EXTERNAL-IP   PORT(S)   AGE
kubernetes   ClusterIP   10.96.0.1        <none>        443/TCP   48m
my-nginx     ClusterIP   10.111.168.255   <none>        80/TCP    55s
```

Step 7:Forward the Service Port to Your Local Machine

kubectl port-forward allows you to forward a port from your local machine to a port on a service running in the Kubernetes cluster.

1. **Forward the Service Port**: Use the following command to forward a local port to the service's target port.

kubectl port-forward service/<service-name> <local-port>:<service-port>

This command will forward local port 8080 on your machine to port 80 of the service nginx-service running inside the cluster.

```
ubuntu@ip-172-31-44-131:~$ kubectl describe deployments
Name:                   nginx-deployment
Namespace:              default
CreationTimestamp:      Tue, 17 Sep 2024 17:00:22 +0000
Labels:                 <none>
Annotations:            deployment.kubernetes.io/revision: 1
Selector:               app=nginx
Replicas:               2 desired | 2 updated | 2 total | 2 available | 0 unavailable
StrategyType:           RollingUpdate
MinReadySeconds:        0
RollingUpdateStrategy:  25% max unavailable, 25% max surge
Pod Template:
  Labels:  app=nginx
  Containers:
   nginx:
    Image:        nginx:1.14.2
    Port:         80/TCP
    Host Port:    0/TCP
    Environment:  <none>
    Mounts:       <none>
  Volumes:        <none>
Conditions:
  Type           Status  Reason
  ----           ------  ------
  Available      True    MinimumReplicasAvailable
  Progressing    True    NewReplicaSetAvailable
OldReplicaSets:  <none>
NewReplicaSet:   nginx-deployment-86dcfdf4c6 (2/2 replicas created)
Events:
  Type    Reason             Age   From                   Message
  ----    ------             ----  ----                   -------
  Normal  ScalingReplicaSet  2m9s  deployment-controller  Scaled up replica set nginx-deployment-86dcfdf4c6 to 2
```

2. This means port forwarding is now active, and any traffic to localhost:8080 will be routed to the nginx-service on port 80.

```
ubuntu@ip-172-31-44-131:~$ kubectl get services
NAME          TYPE        CLUSTER-IP       EXTERNAL-IP   PORT(S)   AGE
kubernetes    ClusterIP   10.96.0.1        <none>        443/TCP   49m
my-nginx      ClusterIP   10.111.168.255   <none>        80/TCP    2m9s
ubuntu@ip-172-31-44-131:~$
```

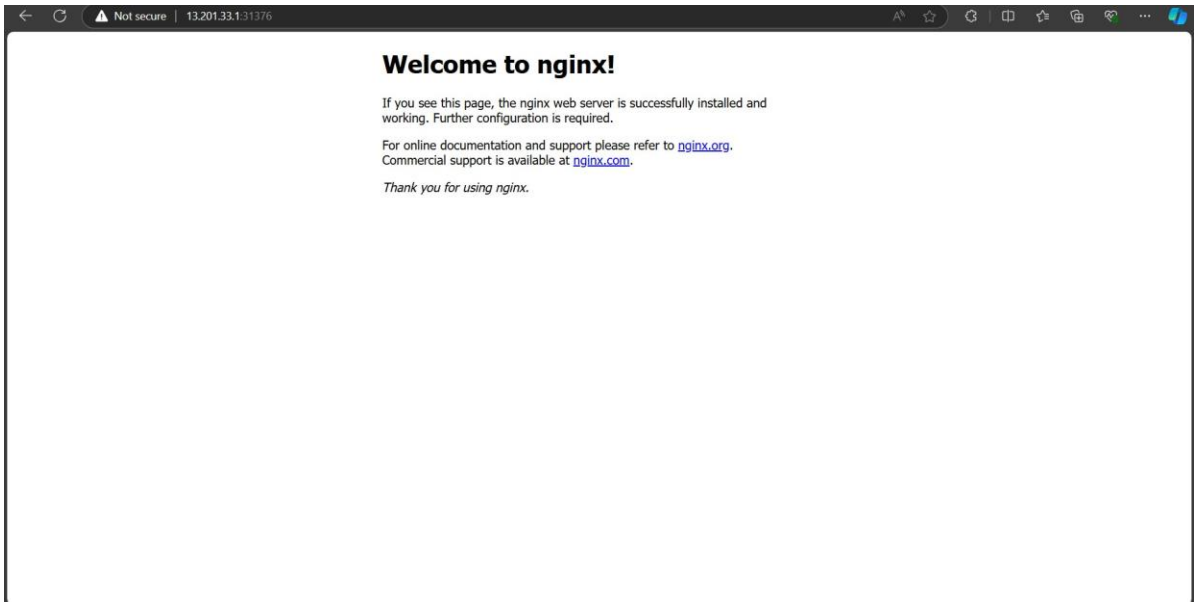
```
ubuntu@ip-172-31-44-131:~$ nano nginx-services.yaml
ubuntu@ip-172-31-44-131:~$ nano nginx-service.yaml
ubuntu@ip-172-31-44-131:~$ kubectl apply -f nginx-service.yaml
service/nginx-service created
ubuntu@ip-172-31-44-131:~$ kubectl get services
NAME            TYPE           CLUSTER-IP       EXTERNAL-IP   PORT(S)        AGE
kubernetes      ClusterIP      10.96.0.1        <none>        443/TCP        71m
my-nginx        ClusterIP      10.111.168.255   <none>        80/TCP         23m
nginx-service   LoadBalancer   10.105.174.168   <pending>     80:31376/TCP   10s
ubuntu@ip-172-31-44-131:~$ kubectl port-forward service/nginx-service 8088:80
Forwarding from 127.0.0.1:8088 -> 80
Forwarding from [::1]:8088 -> 80
^Cubuntu@ip-172-31-44-131:~kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
nginx-deployment-86dcfdf4c6-8d7rx   1/1     Running   0          26m
nginx-deployment-86dcfdf4c6-bdbcm   1/1     Running   0          26m
```

Step 8: Access the Application Locally

1. **Open a Web Browser**: Now open your web browser and go to the following URL:

http://localhost:8080
You should see the application (in this case, Nginx) that you have deployed running in the Kubernetes cluster, served locally via port 8080.

In case the port 8080 is unavailable, try using a different port like 8081



ADVANCED DEVOPS EXP-5

NAME- PRANAV TITAMBE CLASS-D15A ROLLNO. - 62

Aim : To understand terraform lifecycle, core concepts/terminologies and install it on a Linux Machine and Windows.

Theory :

Terraform is an open-source Infrastructure as Code (IaC) tool developed by HashiCorp. It allows users to define and provision infrastructure using a high-level configuration language known as HashiCorp Configuration Language (HCL) or JSON. Terraform supports a wide range of cloud providers, such as AWS, Azure, Google Cloud, and
on-premises solutions, enabling users to manage infrastructure across multiple environments consistently.

Core Concepts and Terminologies

1. **Providers:**

   Providers are plugins that allow Terraform to interact with various APIs of cloud providers, SaaS providers, and other services. Each provider requires configuration and manages resources for that specific service.

2. **Resources:**

   Resources are the most fundamental elements in Terraform. They represent components of your infrastructure, such as virtual machines, databases, networks, and more.

3. **Modules:**

   Modules are containers for multiple resources that are used together. A module can call other modules, creating a hierarchical structure. This makes it easier to organize and reuse code.

4. **State:**

   Terraform maintains a state file that keeps track of the infrastructure managed by Terraform. The state file is crucial as it provides a mapping between the real-world resources and the configuration defined in Terraform.

5. **Variables:**

   Variables in Terraform are used to make configurations dynamic and reusable. They can be defined in the configuration files and assigned values at runtime.

6. **Outputs:**

   Outputs are used to extract information from the Terraform-managed infrastructure and display it after the execution of a Terraform plan or apply.

Terraform Lifecycle

1. **Write:**

   Write the configuration file (typically with .tf extension) using HCL to describe the desired infrastructure.

2. **Initialize (terraform init):**

   Initialize the working directory containing the configuration files. This command downloads the necessary provider plugins and sets up the environment.

3. **Plan (terraform plan):**

   Terraform creates an execution plan based on the configuration files. It compares the current state with the desired state and shows the changes that will be made.

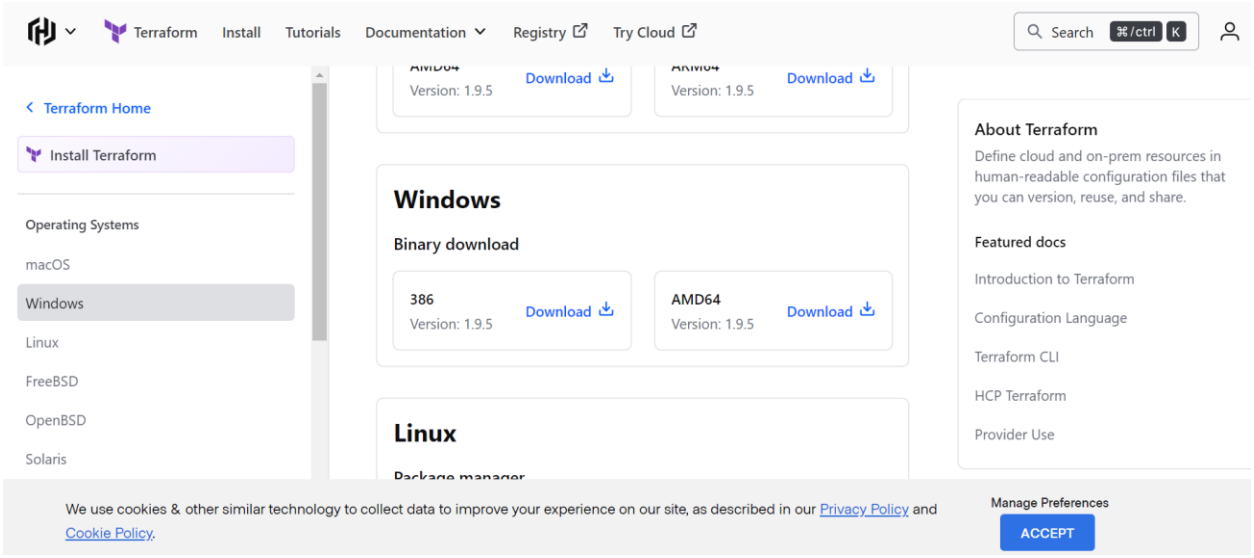
4. **Apply (terraform apply):**

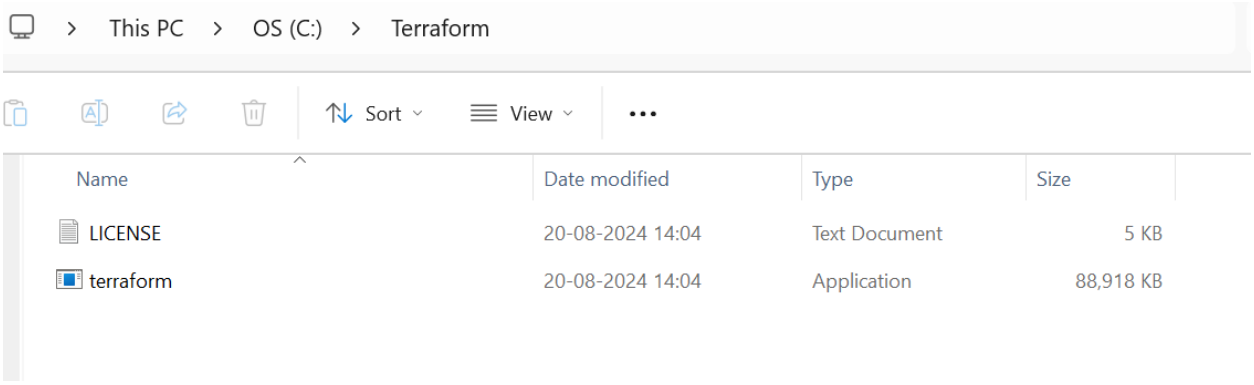
   Apply the changes required to reach the desired state of the configuration. Terraform will prompt for confirmation before making any changes.

5. **Destroy (terraform destroy):**

   Destroy the infrastructure managed by Terraform. This command is used to remove all resources defined in the configuration files.

Implementation:-


Terraform
Install
Tutorials
Documentation
Registry
Try Cloud
Search ⌘/ctrl K
Terraform Home
Install Terraform
Operating Systems
macOS
Windows
Linux
FreeBSD
OpenBSD
Solaris
Version: 1.9.5
Download
Version: 1.9.5
Download
Windows
Binary download
386
Version: 1.9.5
Download
AMD64
Version: 1.9.5
Download
Linux
About Terraform
Define cloud and on-prem resources in human-readable configuration files that you can version, reuse, and share.
Featured docs
Introduction to Terraform
Configuration Language
Terraform CLI
HCP Terraform
Provider Use
We use cookies & other similar technology to collect data to improve your experience on our site, as described in our Privacy Policy and Cookie Policy.
Manage Preferences
ACCEPT


This PC
OS (C:)
Terraform
Sort
View
Name
Date modified
Type
Size
LICENSE
20-08-2024 14:04
Text Document
5 KB
terraform
20-08-2024 14:04
Application
88,918 KB

```
PS C:\Terraform> terraform
Usage: terraform [global options] <subcommand> [args]

The available commands for execution are listed below.
The primary workflow commands are given first, followed by
less common or more advanced commands.

Main commands:
  init          Prepare your working directory for other commands
  validate      Check whether the configuration is valid
  plan          Show changes required by the current configuration
  apply         Create or update infrastructure
  destroy       Destroy previously-created infrastructure

All other commands:
  console       Try Terraform expressions at an interactive command prompt
  fmt           Reformat your configuration in the standard style
  force-unlock  Release a stuck lock on the current workspace
  get           Install or upgrade remote Terraform modules
  graph         Generate a Graphviz graph of the steps in an operation
  import        Associate existing infrastructure with a Terraform resource
  login         Obtain and save credentials for a remote host
  logout        Remove locally-stored credentials for a remote host
  metadata      Metadata related commands
  output        Show output values from your root module
  providers     Show the providers required for this configuration
  refresh       Update the state to match remote systems
  show          Show the current state or a saved plan
  state         Advanced state management
  taint         Mark a resource instance as not fully functional
```

```
Microsoft Windows [Version 10.0.22631.3958]
(c) Microsoft Corporation. All rights reserved.

C:\Users\[redacted]>terraform --version
Terraform v1.9.4
on windows_amd64

Your version of Terraform is out of date! The latest version
is 1.9.5. You can update by downloading from https://www.terraform.io/downloads.html
```

```
PS C:\Terraform> docker --version
Docker version 27.0.3, build 7d4bcd8
PS C:\Terraform>
```

EXPERIMENT NO. 6

NAME : PRANAV TITAMBE CLASS : D15A ROLL NO. : 62

Aim : To Build, change, and destroy AWS infrastructure Using Terraform (S3 bucket or Docker) .

Theory :

Terraform is an open-source tool that enables developers and operations teams to define, provision, and manage cloud infrastructure through code. It uses a declarative language to specify the desired state of infrastructure, which can include servers, storage, networking components, and more. With Terraform, infrastructure changes can be automated, versioned, and tracked efficiently.

Building Infrastructure

When you build infrastructure using Terraform, you define the desired state of your infrastructure in configuration files. For example, you may want to create an S3 bucket or deploy a Docker container on an EC2 instance. Terraform reads these configuration files and, using the specified cloud provider (such as AWS), it provisions the necessary resources to match the desired state.

- **S3 Buckets:** Terraform can create and manage S3 buckets, which are used to store and retrieve data objects in the cloud. You can define the properties of the bucket, such as its name, region, access permissions, and versioning.
- **Docker on AWS:** Terraform can deploy Docker containers on AWS infrastructure. This often involves setting up an EC2 instance and configuring it to run Docker containers, which encapsulate applications and their dependencies.

Changing Infrastructure

As your needs evolve, you may need to modify the existing infrastructure. Terraform makes it easy to implement changes by updating the configuration files to reflect the new desired state. For instance, you might want to change the storage settings of an S3 bucket, add new security policies, or modify the Docker container's configuration.

Terraform's "plan" command helps you preview the changes that will be made to your infrastructure before applying them. This step ensures that you understand the impact of your changes and can avoid unintended consequences.

Destroying Infrastructure

When certain resources are no longer needed, Terraform allows you to destroy them in a controlled manner. This might involve deleting an S3 bucket or terminating an EC2 instance running Docker containers. By running the "destroy" command, Terraform ensures that all associated resources are properly de-provisioned and removed.

Destroying infrastructure with Terraform is beneficial because it helps avoid unnecessary costs associated with unused resources and ensures that the environment remains clean and free of clutter.

Benefits of Using Terraform for AWS Infrastructure

1. **Consistency:** Terraform ensures that infrastructure is consistent across environments by applying the same configuration files.
2. **Automation:** Manual processes are reduced, and infrastructure is provisioned, updated, and destroyed automatically based on code.
3. **Version Control:** Infrastructure configurations can be stored in version control systems (like Git), allowing teams to track changes, collaborate, and roll back if necessary.
4. **Scalability:** Terraform can manage complex infrastructures, scaling them up or down as needed, whether for small projects or large-scale applications.
5. **Modularity:** Terraform configurations can be broken down into reusable modules, making it easier to manage and scale infrastructure.

Implementation :

Terraform and Docker -

Step 1 : check docker installation and version

```
PS C:\Users\prana\Desktop\Adv-devOps\terraform-exp-6> docker -v
Docker version 27.1.1, build 6312585
PS C:\Users\prana\Desktop\Adv-devOps\terraform-exp-6>
```

Step 2 : create docker.tf file and write following code for terraform and docker

Code -

```
terraform {
  required_providers {
  docker = {
      source = "kreuzwerker/docker"
      version = "~> 3.0.1"
   }
  }
}

provider "docker" {
  host    = "npipe:////.//pipe//docker_engine"
}

resource "docker_image" "nginx" {
  name         = "nginx:latest"
  keep_locally = false
}
resource "docker_container" "nginx" {
  image = docker_image.nginx.image_id
  name  = "tutorial"
  ports {
    internal = 80
    external = 8000
```

}
}


docker.tf > terraform > required_providers > docker
terraform {
required_providers {
docker = {
source = "kreuzwerker/docker"
version = "~> 3.0.1"
}
}
}
provider "docker" {
host ="npipe:////.//pipe//docker_engine"
}
resource "docker_image" "nginx" {
name = "nginx:latest"
keep_locally = false
}
resource "docker_container" "nginx" {
image = docker_image.nginx.image_id
name ="tutorial"
ports {
internal = 80
external = 8000
}
}

Step 3 : Type terraform init command to initialize terraform backend

```
PS C:\Users\prana\Desktop\Adv-devOps\terraform-exp-6> terraform init
Initializing the backend...
Initializing provider plugins...
- Finding kreuzwerker/docker versions matching "~> 3.0.1"...
- Installing kreuzwerker/docker v3.0.2...
- Installed kreuzwerker/docker v3.0.2 (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
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.
```

Step 4(EXTRA) : type terraform fmt and validate commands .
The two Terraform commands – terraform validate and terraform fmt – are used to maintain a clean, error-free, and well-structured Terraform codebase.

```
PS C:\Users\prana\Desktop\Adv-devOps\terraform-exp-6> terraform fmt
docker.tf
PS C:\Users\prana\Desktop\Adv-devOps\terraform-exp-6> terraform validate
Success! The configuration is valid.
```

Step 5 : Type Terraform plan command to create execution plan .

```
PS C:\Users\prana\Desktop\Adv-devOps\terraform-exp-6> terraform plan

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

Terraform will perform the following actions:

  # docker_container.nginx will be created
  + resource "docker_container" "nginx" {
      + attach                                      = false
      + bridge                                      = (known after apply)
      + command                                     = (known after apply)
      + container_logs                              = (known after apply)
      + container_read_refresh_timeout_milliseconds = 15000
      + entrypoint                                  = (known after apply)
      + env                                         = (known after apply)
      + exit_code                                   = (known after apply)
      + hostname                                    = (known after apply)
      + id                                          = (known after apply)
      + image                                       = (known after apply)
      + init                                        = (known after apply)
      + ipc_mode                                    = (known after apply)
      + log_driver                                  = (known after apply)
      + logs                                        = false
      + must_run                                    = true
      + name                                        = "tutorial"
      + network_data                                = (known after apply)
      + read_only                                   = false
      + remove_volumes                              = true
      + restart                                     = "no"
      + rm                                          = false
      + runtime                                     = (known after apply)
      + security_opts                               = (known after apply)
      + shm_size                                    = (known after apply)
      + start                                       = true
      + stdin_open                                  = false
```

```
      + stdin_open                                  = false
      + stop_signal                                 = (known after apply)
      + stop_timeout                                = (known after apply)
      + tty                                         = false
      + wait                                        = false
      + wait_timeout                                = 60

      + healthcheck (known after apply)

      + labels (known after apply)

      + ports {
          + external = 8000
          + internal = 80
          + ip       = "0.0.0.0"
          + protocol = "tcp"
        }
    }

  # docker_image.nginx will be created
  + resource "docker_image" "nginx" {
      + id           = (known after apply)
      + image_id     = (known after apply)
      + keep_locally = false
      + name         = "nginx:latest"
      + repo_digest  = (known after apply)
    }

Plan: 2 to add, 0 to change, 0 to destroy.
```

Step 6 : Type terraform apply to apply changes .

```
PS C:\Users\prana\Desktop\Adv-devOps\terraform-exp-6> terraform apply

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

Terraform will perform the following actions:

  # docker_container.nginx will be created
  + resource "docker_container" "nginx" {
      + attach                                      = false
      + bridge                                      = (known after apply)
      + command                                     = (known after apply)
      + container_logs                              = (known after apply)
      + container_read_refresh_timeout_milliseconds = 15000
      + entrypoint                                  = (known after apply)
      + env                                         = (known after apply)
      + exit_code                                   = (known after apply)
      + hostname                                    = (known after apply)
      + id                                          = (known after apply)
      + image                                       = (known after apply)
      + init                                        = (known after apply)
      + ipc_mode                                    = (known after apply)
      + log_driver                                  = (known after apply)
      + logs                                        = false
      + must_run                                    = true
      + name                                        = "tutorial"
      + network_data                                = (known after apply)
      + read_only                                   = false
      + remove_volumes                              = true
      + restart                                     = "no"
      + rm                                          = false
      + runtime                                     = (known after apply)
      + security_opts                               = (known after apply)
      + shm_size                                    = (known after apply)
      + start                                       = true
      + stdin_open                                  = false
```

```
      + healthcheck (known after apply)

      + labels (known after apply)

      + ports {
          + external = 8000
          + internal = 80
          + ip       = "0.0.0.0"
          + protocol = "tcp"
        }
    }

  # docker_image.nginx will be created
  + resource "docker_image" "nginx" {
      + id          = (known after apply)
      + image_id    = (known after apply)
      + keep_locally = false
      + name        = "nginx:latest"
      + repo_digest  = (known after apply)
    }

Plan: 2 to add, 0 to change, 0 to destroy.

Do you want to perform these actions?
  Terraform will perform the actions described above.

  Enter a value: yes

docker_image.nginx: Creating...
docker_image.nginx: Still creating... [10s elapsed]
docker_image.nginx: Still creating... [20s elapsed]
docker_image.nginx: Creation complete after 24s [id=sha256:39286ab8a5e14aeaf5fdd6e2fac76e0c8d31a0c07224f0ee5e6be502f12e93f3nginx:latest]
docker_container.nginx: Creating...
docker_container.nginx: Creation complete after 1s [id=b51c3ca78d8fa2bf52386d8b0423fbc364ff83106c404a8efb1fa8f05095532e]

Apply complete! Resources: 2 added, 0 changed, 0 destroyed.
```

Step 7 : Docker container before and after step 6 execution

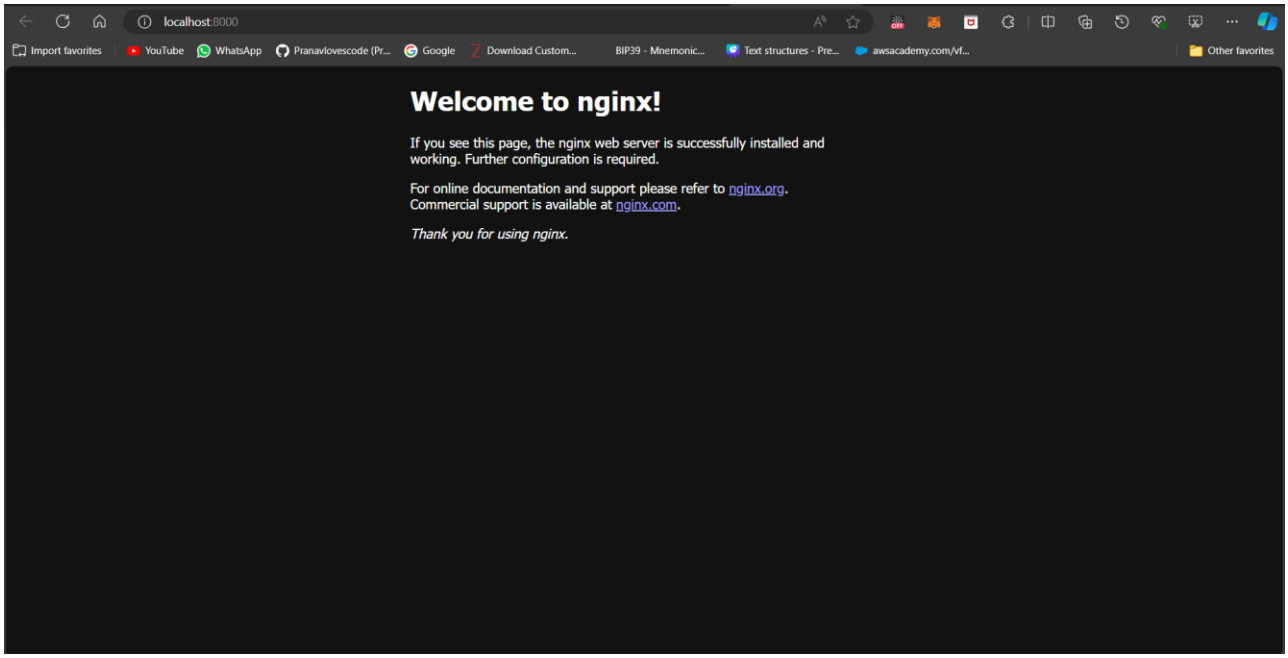
BEFORE –

```
PS C:\Users\prana\Desktop\Adv-devOps\terraform-exp-6> docker ps
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
PS C:\Users\prana\Desktop\Adv-devOps\terraform-exp-6>
```

AFTER -

```
PS C:\Users\prana\Desktop\Adv-devOps\terraform-exp-6> docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED         STATUS         PORTS                    NAMES
b51c3ca78d8f   39286ab8a5e1   "/docker-entrypoint.…"   9 minutes ago   Up 8 seconds   0.0.0.0:8000->80/tcp   tutorial
PS C:\Users\prana\Desktop\Adv-devOps\terraform-exp-6>
```

OUTPUT-



Step 8 (EXTRA) : Execution of change .

docker.tf

```
terraform {
  required_providers {
    docker = {
      source  = "kreuzwerker/docker"
      version = "~> 3.0.1"
    }
  }
}

provider "docker" {
  host = "npipe:////.//pipe//docker_engine"
}

resource "docker_image" "nginx" {
  name         = "nginx:latest"
  keep_locally = false
}

resource "docker_container" "nginx" {
  image = docker_image.nginx.image_id
  name  = "tutorial"
  ports {
    internal = 80
    external = 8080
  }
}

```

```
      + publish_all_ports                               = (known after apply)
      + read_only                                       = (known after apply)
      + remove_volumes                                  = (known after apply)
      + restart                                         = (known after apply)
      + rm                                              = (known after apply)
      + runtime                                         = (known after apply)
      + security_opts                                   = (known after apply)
      + shm_size                                        = (known after apply)
      + start                                           = (known after apply)
      + stdin_open                                      = (known after apply)
      + stop_signal                                     = (known after apply)
      + stop_timeout                                    = (known after apply)
      + storage_opts                                    = (known after apply)
      + sysctls                                         = (known after apply)
      + tmpfs                                           = (known after apply)
      + tty                                             = (known after apply)
      + user                                            = (known after apply)
      + userns_mode                                     = (known after apply)
      + wait                                            = (known after apply)
      + wait_timeout                                    = (known after apply)
      + working_dir                                     = (known after apply)
    } -> (known after apply)

  ~ ports {
      ~ external = 8000 -> 8080 # forces replacement
        # (3 unchanged attributes hidden)
    }
}

Plan: 1 to add, 0 to change, 1 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.nginx: Destroying... [id=c25805e44841645209l2c50ac3080526c9926219c98c673021078772eb484357]
docker_container.nginx: Destruction complete after 1s
docker_container.nginx: Creating...
```

Step 9 : terraform destroy to destroy infrastructure.

```
Apply complete! Resources: 1 added, 0 changed, 0 destroyed.
(base) PS C:\Users\sbpol\Documents\terraform_scripts\docker> terraform destroy
docker_image.nginx: Refreshing state... [id=sha256:5ef79149e0ec84a7a9f9284c3f91aa3c20608f8391f5445eabe92ef07dbda03cnginx:latest]
docker_container.nginx: Refreshing state... [id=c648cc3dd8129abf9acb7cb06dfdd0aa9bafb0c7973f16695cd06a7ad447c631]

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:

  # docker_container.nginx will be destroyed
  - resource "docker_container" "nginx" {
      - attach                                      = false -> null
      - command                                     = [
          - "nginx",
          - "-g",
          - "daemon off;",
        ] -> null
      - container_read_refresh_timeout_milliseconds = 15000 -> null
      - cpu_shares                                  = 0 -> null
      - dns                                         = [] -> null
      - dns_opts                                    = [] -> null
      - dns_search                                  = [] -> null
      - entrypoint                                  = [
          - "/docker-entrypoint.sh",
        ] -> null
      - env                                         = [] -> null
      - group_add                                   = [] -> null
      - hostname                                    = "c648cc3dd812" -> null
      - id                                          = "c648cc3dd8129abf9acb7cb06dfdd0aa9bafb0c7973f16695cd06a7ad447c631" -> null
      - image                                       = "sha256:5ef79149e0ec84a7a9f9284c3f91aa3c20608f8391f5445eabe92ef07dbda03c" -> null
      - init                                        = false -> null
      - ipc_mode                                    = "private" -> null
      - log_driver                                  = "json-file" -> null
      - log_opts                                    = {} -> null
      - logs                                        = false -> null
      - max_retry_count                             = 0 -> null
      - memory                                      = 0 -> null
      - memory_swap                                 = 0 -> null
      - must_run                                    = true -> null
```

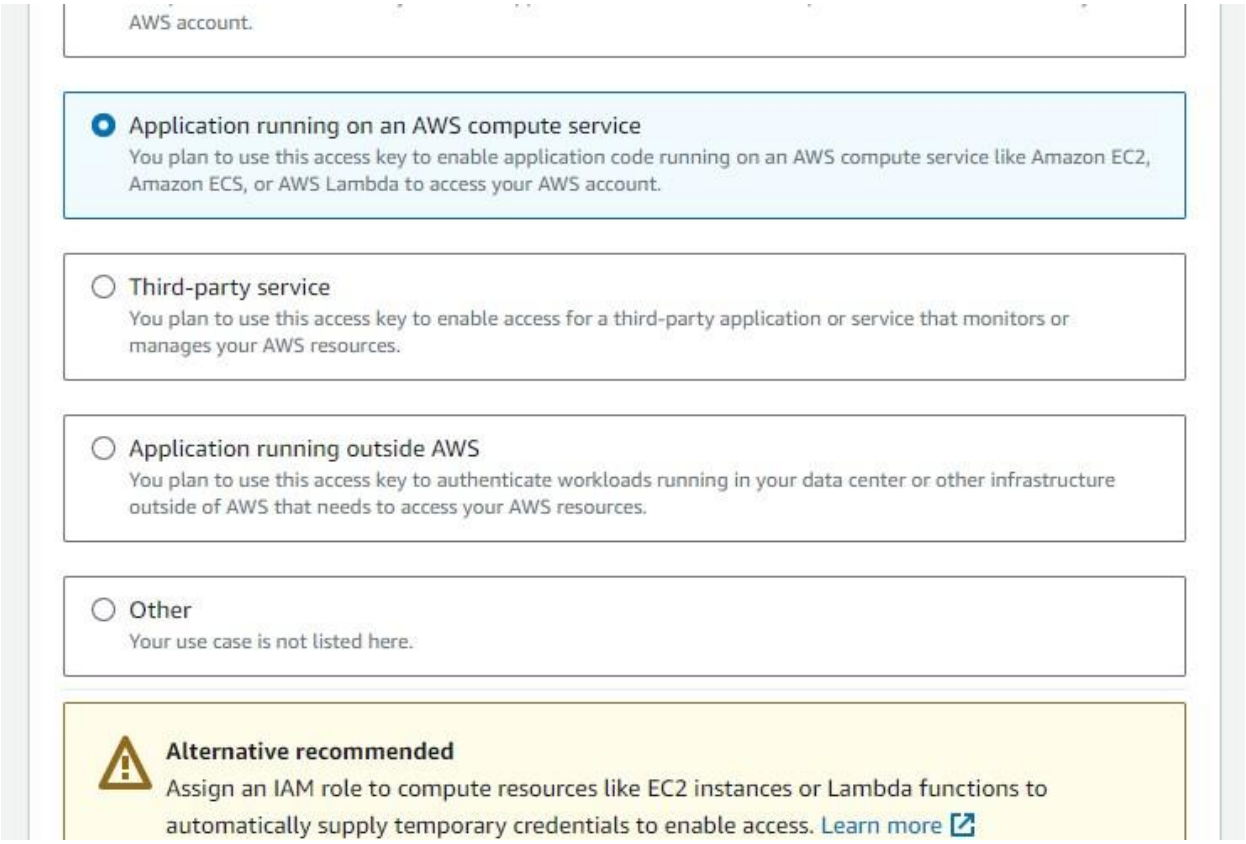
```
      - stop_timeout                              = 0 -> null
      - storage_opts                              = {} -> null
      - sysctls                                   = {} -> null
      - tmpfs                                     = {} -> null
      - tty                                       = false -> null
      - wait                                      = false -> null
      - wait_timeout                              = 60 -> null
        # (7 unchanged attributes hidden)

      - ports {
          - external = 8000 -> null
          - internal = 80 -> null
          - ip       = "0.0.0.0" -> null
          - protocol = "tcp" -> null
        }
    }

  # docker_image.nginx will be destroyed
  - resource "docker_image" "nginx" {
      - id           = "sha256:5ef79149e0ec84a7a9f9284c3f91aa3c20608f8391f5445eabe92ef07dbda03cnginx:latest" -> null
      - image_id     = "sha256:5ef79149e0ec84a7a9f9284c3f91aa3c20608f8391f5445eabe92ef07dbda03c" -> null
      - keep_locally = false -> null
      - name         = "nginx:latest" -> null
      - repo_digest  = "nginx@sha256:447a8665cc1dab95b1ca778e162215839ccbb9189104c79d7ec3a81e14577add" -> 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: yes

docker_container.nginx: Destroying... [id=c648cc3dd8129abf9acb7cb06dfdd0aa9bafb0c7973f16695cd06a7ad447c631]
docker_container.nginx: Destruction complete after 1s
docker_image.nginx: Destroying... [id=sha256:5ef79149e0ec84a7a9f9284c3f91aa3c20608f8391f5445eabe92ef07dbda03cnginx:latest]
docker_image.nginx: Destruction complete after 0s

Destroy complete! Resources: 2 destroyed.
(base) PS C:\Users\sbpol\Documents\terraform_scripts\docker>
```

Step 10 : Docker after destroy command.

```
Destroy complete! Resources: 2 destroyed.
(base) PS C:\Users\sbpol\Documents\terraform_scripts\docker> docker container list
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
(base) PS C:\Users\sbpol\Documents\terraform_scripts\docker> docker images
REPOSITORY   TAG       IMAGE ID   CREATED   SIZE
(base) PS C:\Users\sbpol\Documents\terraform_scripts\docker>
```

Terraform and S3 -

Step 1: Create access keys and secret key for IAM user



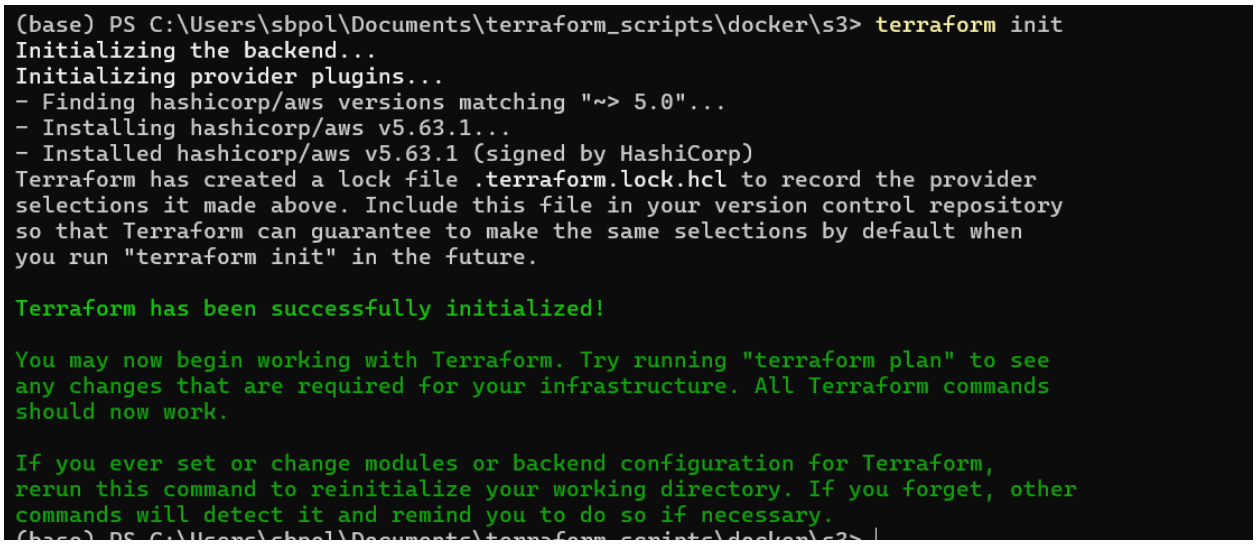
Step 2 : Type below code in main.tf in editor for aws and terraform connection and environment creation .

Code -

```
terraform {
  required_providers {
  aws = {
      source = "hashicorp/aws"
      version = "~> 5.0"
    }
  }
}
```

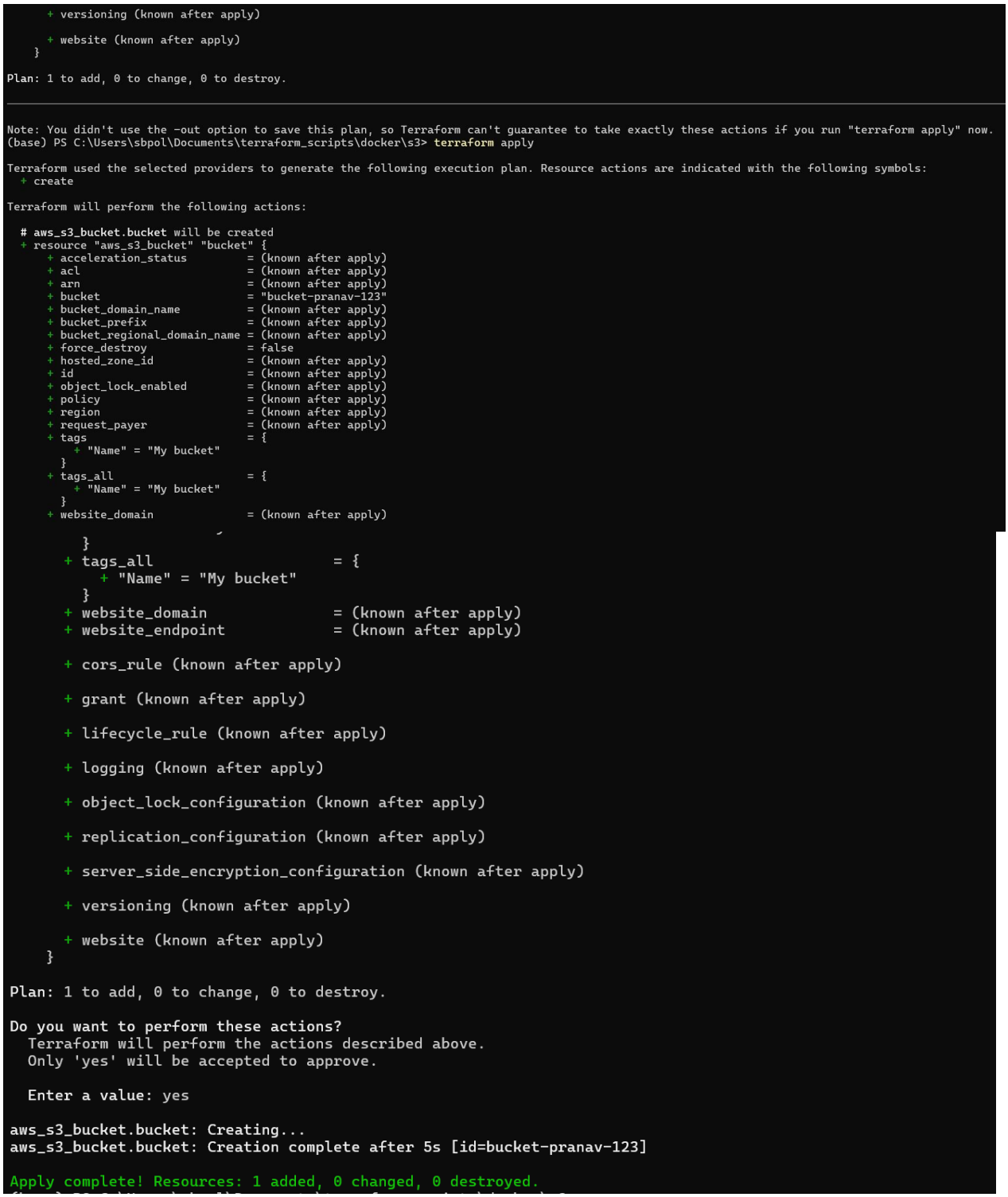
```
# Configure the AWS Provider
provider "aws" {
  region = "us-east-1"
  access_key = ""
  secret_key = ""
}
resource "aws_s3_bucket" "bucket" {
  bucket = "bucket-pranav-123"

  tags = {
    Name = "My bucket"

  }
}
```

Step 3 : Type terraform init command in powershell.



Step 4 : Type terraform plan command in powershell.

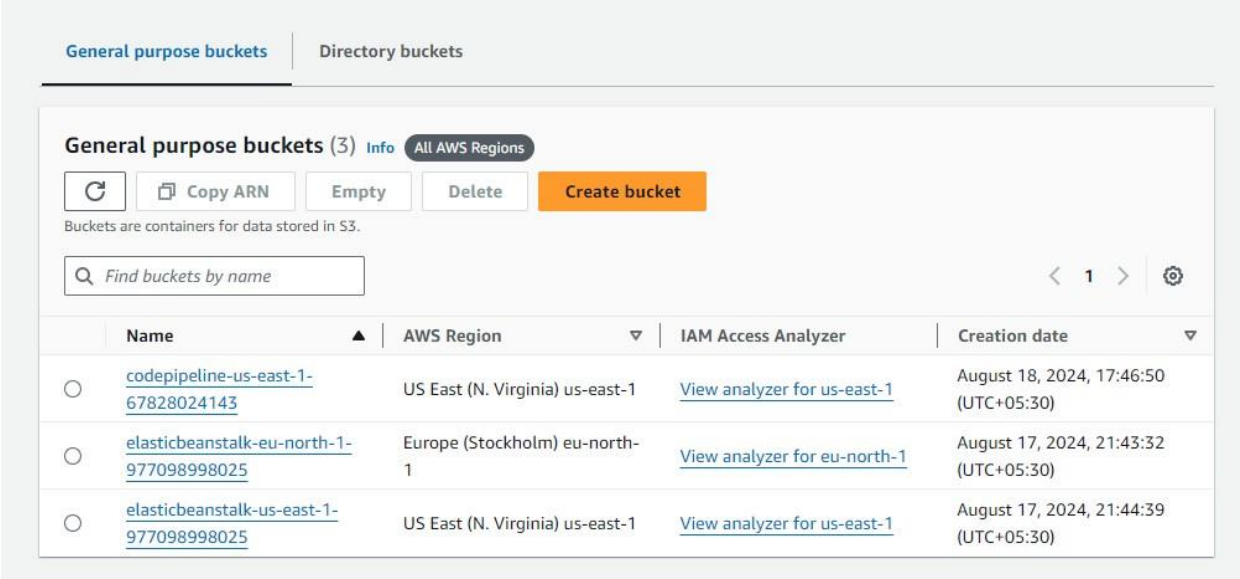
Step 5 : Type terraform apply command in powershell.



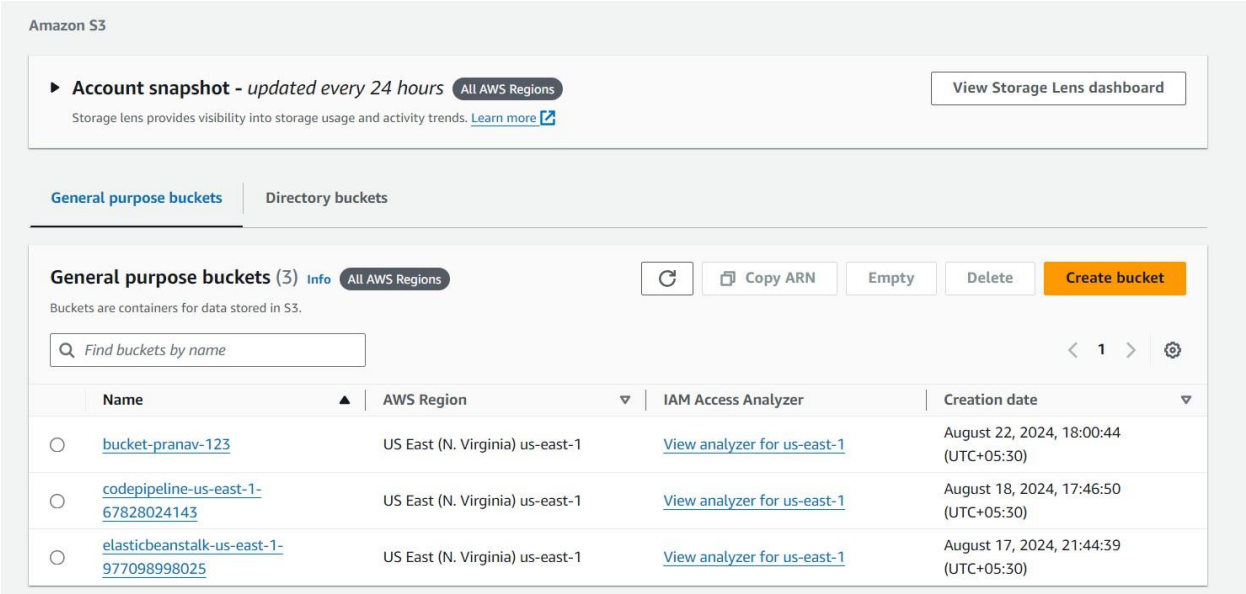
```
      + versioning (known after apply)

      + website (known after apply)
    }

Plan: 1 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.
(base) PS C:\Users\sbpol\Documents\terraform_scripts\docker\s3> terraform apply

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

Terraform will perform the following actions:

  # aws_s3_bucket.bucket will be created
  + resource "aws_s3_bucket" "bucket" {
      + acceleration_status         = (known after apply)
      + acl                         = (known after apply)
      + arn                         = (known after apply)
      + bucket                      = "bucket-pranav-123"
      + bucket_domain_name          = (known after apply)
      + bucket_prefix               = (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)
      + request_payer               = (known after apply)
      + tags                        = {
          + "Name" = "My bucket"
        }
      + tags_all                    = {
          + "Name" = "My bucket"
        }
      + website_domain              = (known after apply)
          }
        + tags_all                      = {
            + "Name" = "My bucket"
          }
        + website_domain                = (known after apply)
        + website_endpoint              = (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(EXTRA) : 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(EXTRA) : Terraform plan and apply command to apply the changes for file .

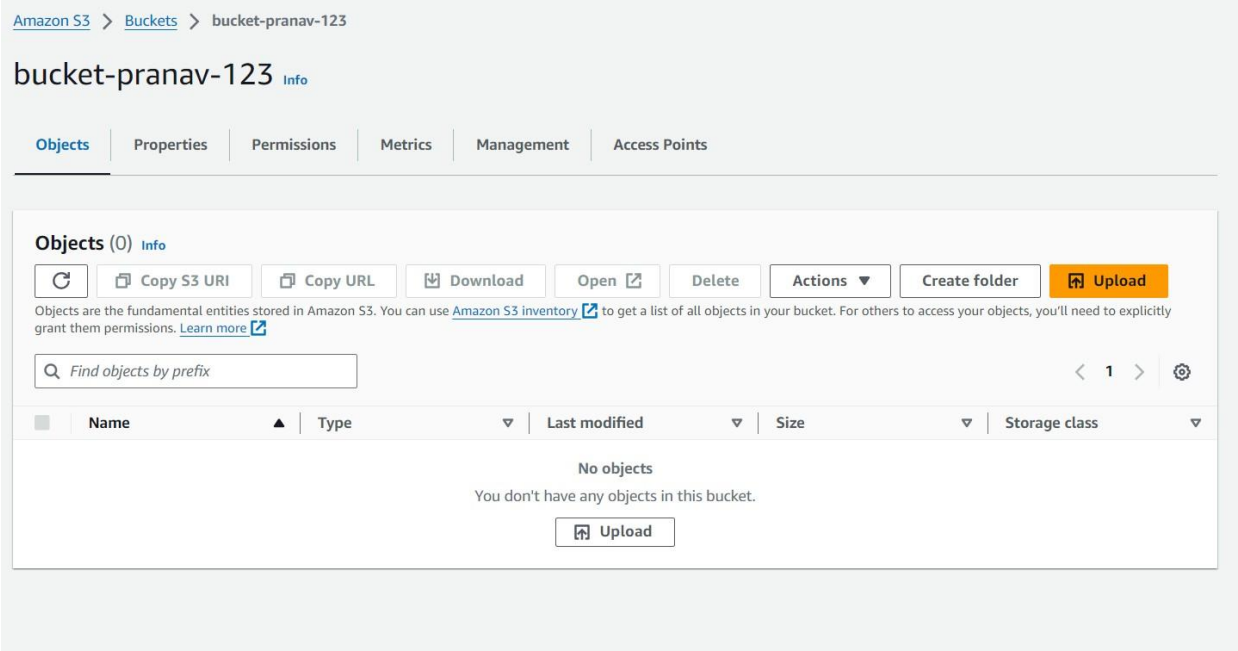
```
(base) PS C:\Users\sbpol\Documents\terraform_scripts\docker\s3> terraform plan
aws_s3_bucket.bucket: Refreshing state... [id=bucket-pranav-123]

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following
  + create

Terraform will perform the following actions:

  # aws_s3_bucket_object.file will be created
  + resource "aws_s3_bucket_object" "file" {
      + acl                    = "private"
      + arn                    = (known after apply)
      + bucket                 = "bucket-pranav-123"
      + bucket_key_enabled     = (known after apply)
      + content_type           = (known after apply)
      + etag                   = (known after apply)
      + force_destroy          = false
      + id                     = (known after apply)
      + key                    = "hello.txt"
      + kms_key_id             = (known after apply)
      + server_side_encryption = (known after apply)
      + source                 = "C:/Users/sbpol/Documents/terraform_scripts/docker/s3/hello.txt"
      + storage_class          = (known after apply)
      + tags_all               = (known after apply)
      + version_id             = (known after apply)
    }

Plan: 1 to add, 0 to change, 0 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

  (and one more similar warning elsewhere)
```

```
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 C:\Users\sbpol\Documents\terraform_scripts\docker\s3> 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:
  + create

Terraform will perform the following actions:

  # aws_s3_bucket_object.file will be created
  + resource "aws_s3_bucket_object" "file" {
      + acl                    = "private"
      + arn                    = (known after apply)
      + bucket                 = "bucket-pranav-123"
      + bucket_key_enabled     = (known after apply)
      + content_type           = (known after apply)
      + etag                   = (known after apply)
      + force_destroy          = false
      + id                     = (known after apply)
      + key                    = "hello.txt"
      + kms_key_id             = (known after apply)
      + server_side_encryption = (known after apply)
      + source                 = "C:/Users/sbpol/Documents/terraform_scripts/docker/s3/hello.txt"
      + storage_class          = (known after apply)
      + tags_all               = (known after apply)
      + version_id             = (known after apply)
    }

Plan: 1 to add, 0 to change, 0 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

  (and one more similar warning elsewhere)
```

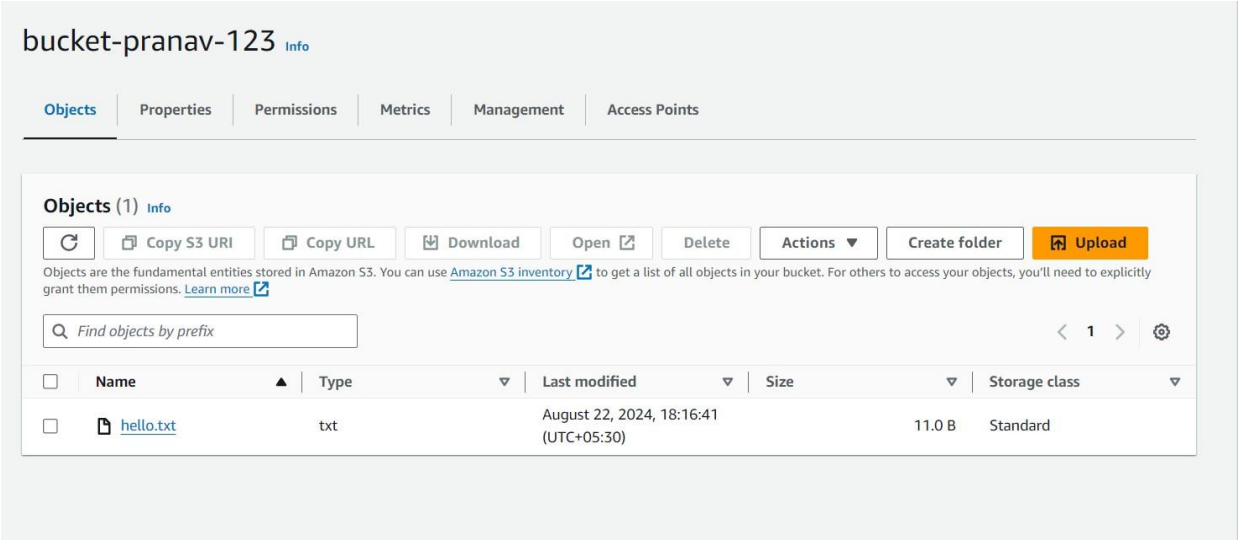
```
Do you want to perform these actions?
  Terraform will perform the actions described above.
  Only 'yes' will be accepted to approve.

  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" {

  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(EXTRA) : s3 bucket before and after execution of upload

BEFORE -



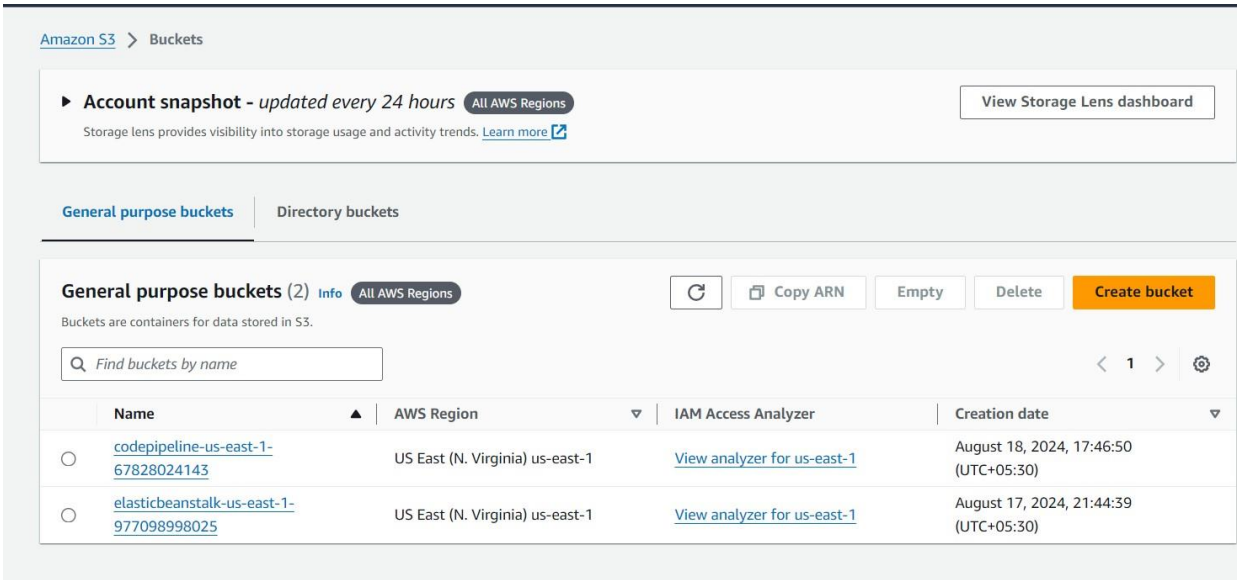
AFTER -



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                         = "arn:aws:s3:::bucket-pranav-123" -> null
      - bucket                      = "bucket-pranav-123" -> null
      - bucket_domain_name          = "bucket-pranav-123.s3.amazonaws.com" -> null
      - bucket_regional_domain_name = "bucket-pranav-123.s3.us-east-1.amazonaws.com" -> null
      - force_destroy               = false -> null
      - hosted_zone_id              = "Z3AQBSTGFYJSTF" -> null
      - id                          = "bucket-pranav-123" -> null
      - object_lock_enabled         = false -> null
      - region                      = "us-east-1" -> null
      - request_payer               = "BucketOwner" -> null
      - tags                        = {
          - "Name" = "My bucket"
        } -> null
      - tags_all                    = {
          - "Name" = "My bucket"
        } -> null
        # (3 unchanged attributes hidden)

      - grant {
          - id          = "10def03d73e09d8adda11bfe68e632f70a83a37758b74ea6e933dafd0250c850" -> null
          - permissions = [
              - "FULL_CONTROL",
            ] -> null
          - type        = "CanonicalUser" -> null
            # (1 unchanged attribute hidden)
        }

      - 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>
```

Step 11: s3 after the destroy command execution .



Hosting Website on s3 using Terraform (EXTRA) -

Step 1 : create main.tf and write following code

Code -

```
terraform {
  required_providers {
    aws = {
      source  = "hashicorp/aws"
      version = "5.64.0"
    }
    random = {
      source  = "hashicorp/random"
      version = "3.6.2"
    }
  }
}

resource "random_id" "rand_id" {
  byte_length = 8
}
resource "aws_s3_bucket" "mywebappp-bucket" {
  bucket = "mywebappp-bucket-${random_id.rand_id.hex}"
}

resource "aws_s3_object" "index_html" {
  bucket       = aws_s3_bucket.mywebappp-bucket.bucket
  source       = "./index.html"
  key          = "index.html"
  content_type = "text/html"
}

resource "aws_s3_object" "styles_css" {
  bucket       = aws_s3_bucket.mywebappp-bucket.bucket
  source       = "./styles.css"
  key          = "styles.css"
  content_type = "text/css"
}

resource "aws_s3_bucket_public_access_block" "example" {
  bucket                  = aws_s3_bucket.mywebappp-bucket.id
```

```
  block_public_acls    =    false
  block_public_policy   =   false
  ignore_public_acls    =   false
  restrict_public_buckets = false
}

resource "aws_s3_bucket_policy" "mywebappp" {
  bucket = aws_s3_bucket.mywebappp-bucket.id

  policy = jsonencode({
    Version = "2012-10-17",
    Statement = [
      {
        Sid = "PublicReadGetObject",
        Effect = "Allow",
        Principal = "*",
        Action = "s3:GetObject",
        Resource = "arn:aws:s3:::${aws_s3_bucket.mywebappp-bucket.id}/*"
      }
    ]
  })
}


resource "aws_s3_bucket_website_configuration" "example" {
  bucket = aws_s3_bucket.mywebappp-bucket.id

  index_document {
    suffix = "index.html"
  }
}

output "website_endpoint" {
  value = aws_s3_bucket_website_configuration.example.website_endpoint
}
```

Step 2 : Create Provider.tf and write following code

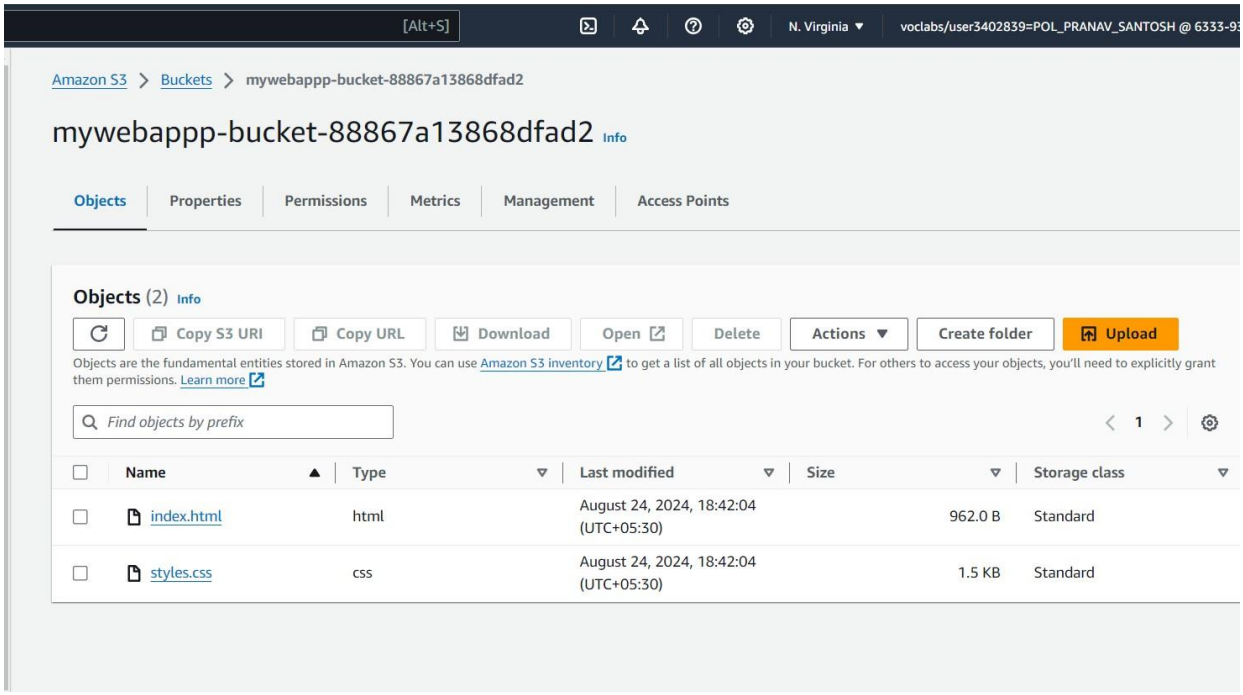
Code -

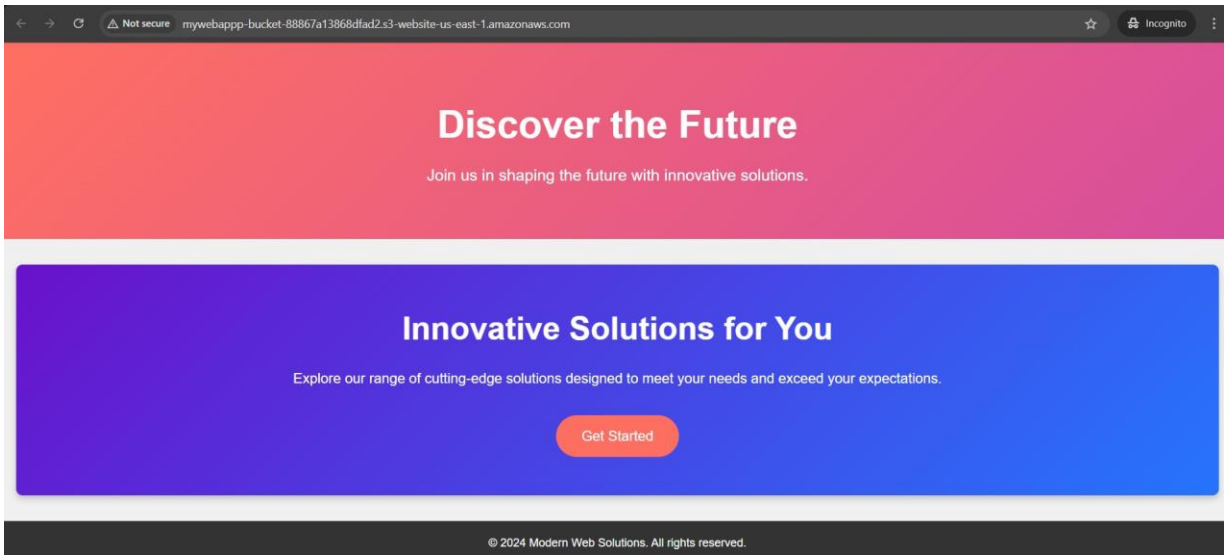
```
provider "aws" {
  access_key="ASIAZG6JVYHRLQ7XABVF"
  secret_key="FV+B+/JDLgRHpPs2bLr9jB+835PQ4cyz7HQ4LAzR"

token="IQoJb3JpZ2luX2VjELT/////////wEaCXVzLXdlc3QtMiJGMEQCIGM45rz6GOsZBjB
cMcCWfAJetwP1F2qgToQCSoJbLE+HAiB2t1XfLcQY0BFOSBsbvJwCmQQ1vQ6/5m4YmzBC1rRel
Cq1Agi9/////////8BEAIaDDYzMzM5Mzc1ODY5MCIM3vgTOnS9B6JyQQmeKokCJkhMaeK5NcX
azpFuqObvIOQpIjKOVtHR/NwxdQCrfqPa2qbn+VsG9i7tF0pvxniO/OQmqxXXaNlRjnq2Qomyd
Ate/91VXJ1cqT7R7k/06ISBc2AVcSAJfgAYEIB7kKVF2UkY01VJ845VjTPnER7O4enKd5jYyHa
kuOkj29olSph1sjrq6VFYBo0foLgLJcDsL/QbipTk8HXX7XT8f/Gh8jGKfUjy2CUvJfuAAX3zv
sTFjSsGEb69J1pZd0sQfoBGi6Mv0vezW+ljWX+dLdpnzDEJrnk0x7g6po1uXrCjDF6+pB+5QwP
hI78D2lF/tcLahLbr5El6ri2DXv0eQ0woOaL6u0xsKDPvwzDCkqe2BjqeAYi5Fs7WB0Ei5FiAq
HdJEzXcQZI18JX5H59W3p+v71sN7sGLxJYrXoMmFLH7amaZxQ7r5xkn9/is6Ge3ZcuxROIy5GO
LuqoHVsNRxCRQ83ZoIewd32TRN8h3uRLQnE7ZMf6ggljBvqvT1e2IlA+YcdeWrkeM/fCXJ0g7k
KEcnkNgBMv+W9LXi2P8DMsm0AnP6jhFK5R6Ch16JI+ePiL1"
  region="us-east-1"
}
```

Step 3: Execute Terraform init , terraform plan and terraform apply command.

```
Terraform will perform the following actions:

  # aws_s3_bucket_policy.mywebappp will be created
  + resource "aws_s3_bucket_policy" "mywebappp" {
      + bucket = "mywebappp-bucket-88867a13868dfad2"
      + id     = (known after apply)
      + policy = jsonencode(
            {
              + Statement = [
                  + {
                      + Action    = "s3:GetObject"
                      + Effect    = "Allow"
                      + Principal = "*"
                      + Resource  = "arn:aws:s3:::mywebappp-bucket-88867a13868dfad2/*"
                      + Sid       = "PublicReadGetObject"
                    },
                ]
              + Version   = "2012-10-17"
            }
        )
    }

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_policy.mywebappp: Creating...
aws_s3_bucket_policy.mywebappp: Creation complete after 2s [id=mywebappp-bucket-88867a13868dfad2]

Apply complete! Resources: 1 added, 0 changed, 0 destroyed.

Outputs:

website_endpoint = "mywebappp-bucket-88867a13868dfad2.s3-website-us-east-1.amazonaws.com"
```

Step 4 : check bucket for if files are uploaded and if the site is hosted correctly at the website_endpoint given in cmd Outputs





Step 5 : terraform destroy to destroy the bucket

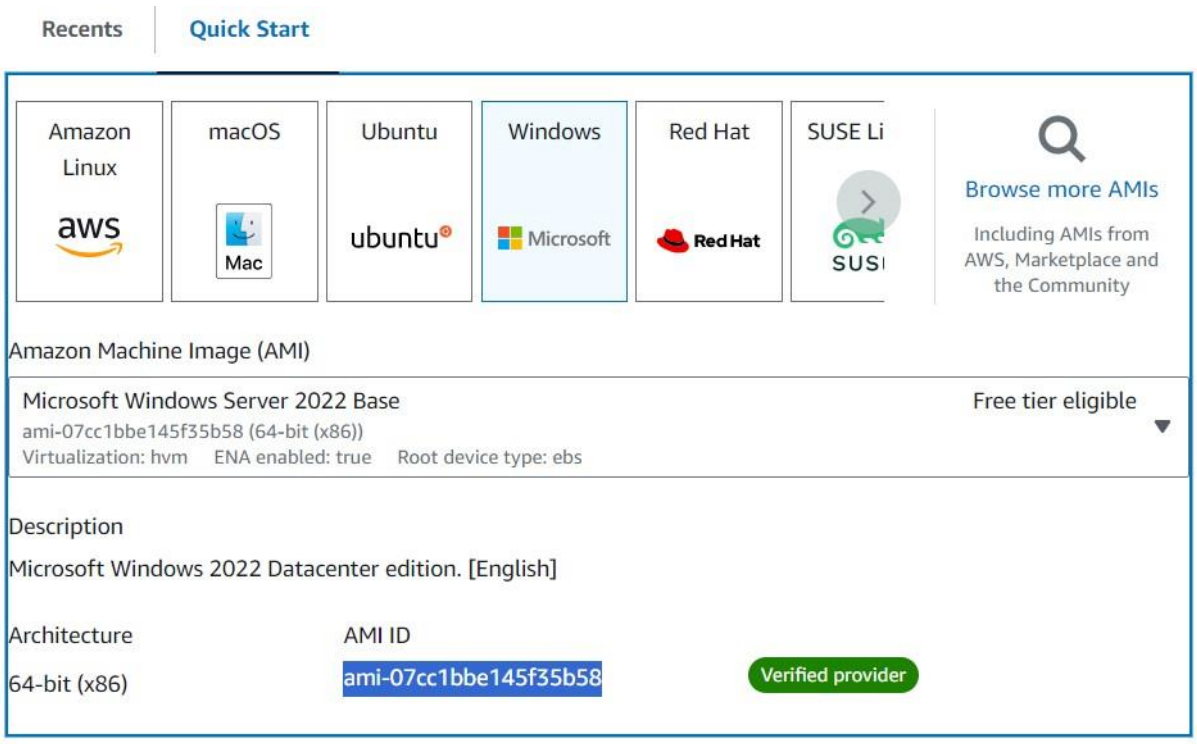
```
  # random_id.rand_id will be destroyed
  - resource "random_id" "rand_id" {
      - b64_std     = "iIZ6E4aN+tI=" -> null
      - b64_url     = "iIZ6E4aN-tI" -> null
      - byte_length = 8 -> null
      - dec         = "9837684660317846226" -> null
      - hex         = "88867a13868dfad2" -> null
      - id          = "iIZ6E4aN-tI" -> null
    }

Plan: 0 to add, 0 to change, 7 to destroy.

Changes to Outputs:
  - website_endpoint = "mywebappp-bucket-88867a13868dfad2.s3-website-us-east-1.amazonaws.com" -> null

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_policy.mywebappp: Destroying... [id=mywebappp-bucket-88867a13868dfad2]
aws_s3_bucket_public_access_block.example: Destroying... [id=mywebappp-bucket-88867a13868dfad2]
aws_s3_bucket_website_configuration.example: Destroying... [id=mywebappp-bucket-88867a13868dfad2]
aws_s3_object.index_html: Destroying... [id=index.html]
aws_s3_object.styles_css: Destroying... [id=styles.css]
aws_s3_object.index_html: Destruction complete after 1s
aws_s3_object.styles_css: Destruction complete after 1s
aws_s3_bucket_website_configuration.example: Destruction complete after 1s
aws_s3_bucket_public_access_block.example: Destruction complete after 1s
aws_s3_bucket_policy.mywebappp: Destruction complete after 2s
aws_s3_bucket.mywebappp-bucket: Destroying... [id=mywebappp-bucket-88867a13868dfad2]
aws_s3_bucket.mywebappp-bucket: Destruction complete after 0s
random_id.rand_id: Destroying... [id=iIZ6E4aN-tI]
random_id.rand_id: Destruction complete after 0s

Destroy complete! Resources: 7 destroyed.

C:\Users\chpel\Documents\terraform_scripts\docker\siteHosting>
```

Creating EC2 instance using Terraform (EXTRA) -

Step 1 : connect the aws academy and terraform using the credentials

```
eee_W_3413358@runweb131733:~$ ^V
bash: $'\026': command not found
eee_W_3413358@runweb131733:~$ export AWS_ACCESS_KEY_ID="ASIAZG6JVYHRLQ7XABVF"
eee_W_3413358@runweb131733:~$ export AWS_SECRET_ACCESS_KEY="FV+B+/JDLgRHpPs2bLr9jB+835PQ4cyz7HQ4LAzR"
eee_W_3413358@runweb131733:~$ export AWS_SESSION_TOKEN= "IQoJb3JpZ2luX2VjELT//////////wEaCXVzLXd1c3QtMiJGMEQCIGM45rz6G
OsZBjBcMcCWfAJetwP1F2qgToQCSoJbLE+HAiB2t1XfLcQY0BFOSBsbvJwCmQQ1vQ6/5m4YmzBC1rRelCq1Agi9//////////8BEAIaDDYzMzM5Mzc1ODY
5MCIM3vgTOnS9B6JyQQmeKokCJkhMaeK5NcXazpFuqObvIOQpIjKOVtHR/NwxdQCrfqPa2qbn+VsG9i7tF0pvxniO/OQmqxXXaNlRjnq2QomydAte/91VX
J1cqT7R7k/06ISBc2AVcSAJfgAYEIB7kKVF2UkY01VJ845VjTPnER7O4enKd5jYyHakuOkj29olSph1sjrq6VFYBo0foLgLJcDsL/QbipTk8HXX7XT8f/G
h8jGKfUjy2CUvJfuAAX3zvsTFjSsGEb69J1pZd0sQfoBGi6Mv0vezW+ljWX+dLdpnzDEJrnk0x7g6po1uXrCjDF6+pB+5QwPhI78D2lF/tcLahLbr5El6r
i2DXv0eQ0woOaL6u0xsKDPvwzDCkqe2BjqeAYi5Fs7WB0Ei5FiAqHdJEzXcQZI18JX5H59W3p+v71sN7sGLxJYrXoMmFLH7amaZxQ7r5xkn9/is6Ge3Zcu
xROIy5GOLuqoHVsNRxCRQ83ZoIewd32TRN8h3uRLQnE7ZMf6ggljBvqvT1e2IlA+YcdeWrkeM/fCXJ0g7kKEcnkNgBMv+W9LXi2P8DMsm0AnP6jhFK5R6C
h16JI+ePiL1"
```

Step 2 : copy the AMI ID from the EC2



Step 3 : Create the main.tf and provider.tf

provider.tf

```
provider "aws" {
  access_key="ASIAZG6JVYHRLQ7XABVF"
  secret_key="FV+B+/JDLgRHpPs2bLr9jB+835PQ4cyz7HQ4LAzR"
  token="IQoJb3JpZ2luX2VjELT/////////wEaCXVzLXdlc3QtMiJGMEQCIGM45rz6GOsZBjBcMcCWfAJetwP1F2qgToQCSoJbLE+HAiB2t1XfLcQY0BFOSBsbvJwCmQQ1vQ6/5m4
  region="us-east-1"
}
```

main.tf

```
terraform {
  required_providers {
    aws = {
      source  = "hashicorp/aws"
      version = "~> 5.0"
    }
  }
}


resource "aws_instance" "myServer" {
  ami = "ami-07cc1bbe145f35b58"
  instance_type = "t2.micro"
    tags = {
        Name = "my Server"
    }
}
```

Step 4 : Execute terraform init , terraform plan and terraform apply command

```
C:\Users\sbpol\Documents\terraform_scripts\docker\ec2>terraform init
Initializing the backend...
Initializing provider plugins...
- Finding hashicorp/aws versions matching "~> 5.0"...
- Installing hashicorp/aws v5.64.0...
- Installed hashicorp/aws v5.64.0 (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.

C:\Users\sbpol\Documents\terraform_scripts\docker\ec2>
```

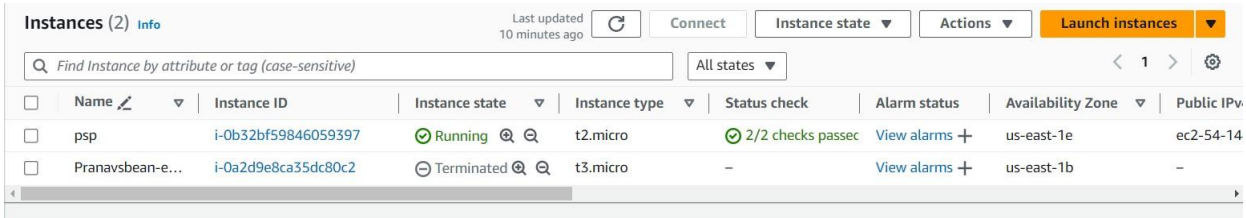
```
C:\Users\sbpol\Documents\terraform_scripts\docker\ec2>terraform plan

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

Terraform will perform the following actions:

  # aws_instance.myServer will be created
  + resource "aws_instance" "myServer" {
      + ami                                  = "ami-07cc1bbe145f35b58"
      + arn                                  = (known after apply)
      + associate_public_ip_address          = (known after apply)
      + availability_zone                    = (known after apply)
      + cpu_core_count                       = (known after apply)
      + cpu_threads_per_core                 = (known after apply)
      + disable_api_stop                     = (known after apply)
      + disable_api_termination              = (known after apply)
      + ebs_optimized                        = (known after apply)
      + get_password_data                    = false
      + host_id                              = (known after apply)
      + host_resource_group_arn              = (known after apply)
      + iam_instance_profile                 = (known after apply)
      + id                                   = (known after apply)
      + instance_initiated_shutdown_behavior = (known after apply)
      + instance_lifecycle                   = (known after apply)
      + instance_state                       = (known after apply)
      + instance_type                        = "t2.micro"
      + ipv6_address_count                   = (known after apply)
      + ipv6_addresses                       = (known after apply)
      + key_name                             = (known after apply)
      + monitoring                           = (known after apply)
      + outpost_arn                          = (known after apply)
      + password_data                        = (known after apply)
      + placement_group                      = (known after apply)
      + placement_partition_number           = (known after apply)
      + primary_network_interface_id         = (known after apply)
      + private_dns                          = (known after apply)
```

```
C:\Users\sbpol\Documents\terraform_scripts\docker\ec2>terraform apply

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the follow
  + create

Terraform will perform the following actions:

  # aws_instance.myServer will be created
  + resource "aws_instance" "myServer" {
      + ami                                  = "ami-07cc1bbe145f35b58"
      + arn                                  = (known after apply)
      + associate_public_ip_address          = (known after apply)
      + availability_zone                    = (known after apply)
      + cpu_core_count                       = (known after apply)
      + cpu_threads_per_core                 = (known after apply)
      + disable_api_stop                     = (known after apply)
      + disable_api_termination              = (known after apply)
      + ebs_optimized                        = (known after apply)
      + get_password_data                    = false
      + host_id                              = (known after apply)
      + host_resource_group_arn              = (known after apply)
      + iam_instance_profile                 = (known after apply)
      + id                                   = (known after apply)
      + instance_initiated_shutdown_behavior = (known after apply)
      + instance_lifecycle                   = (known after apply)
      + instance_state                       = (known after apply)
      + instance_type                        = "t2.micro"
      + ipv6_address_count                   = (known after apply)
      + ipv6_addresses                       = (known after apply)
      + key_name                             = (known after apply)
      + monitoring                           = (known after apply)
      + outpost_arn                          = (known after apply)
      + password_data                        = (known after apply)
      + placement_group                      = (known after apply)
      + placement_partition_number           = (known after apply)
      + primary_network_interface_id         = (known after apply)
      + private_dns                          = (known after apply)
      + private_ip                           = (known after apply)
```

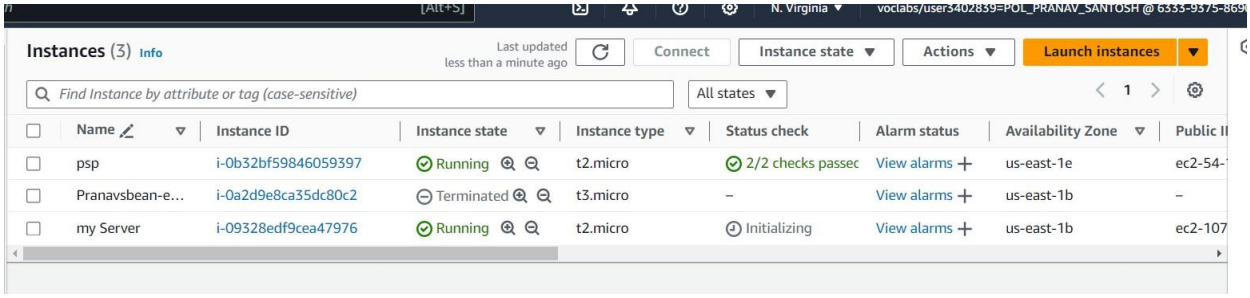
```
Plan: 1 to add, 0 to change, 0 to destroy.

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_instance.myServer: Creating...
aws_instance.myServer: Still creating... [10s elapsed]
aws_instance.myServer: Creation complete after 18s [id=i-09328edf9cea47976]

Apply complete! Resources: 1 added, 0 changed, 0 destroyed.
```

Step 5 : Ec2 before and after instance creation .

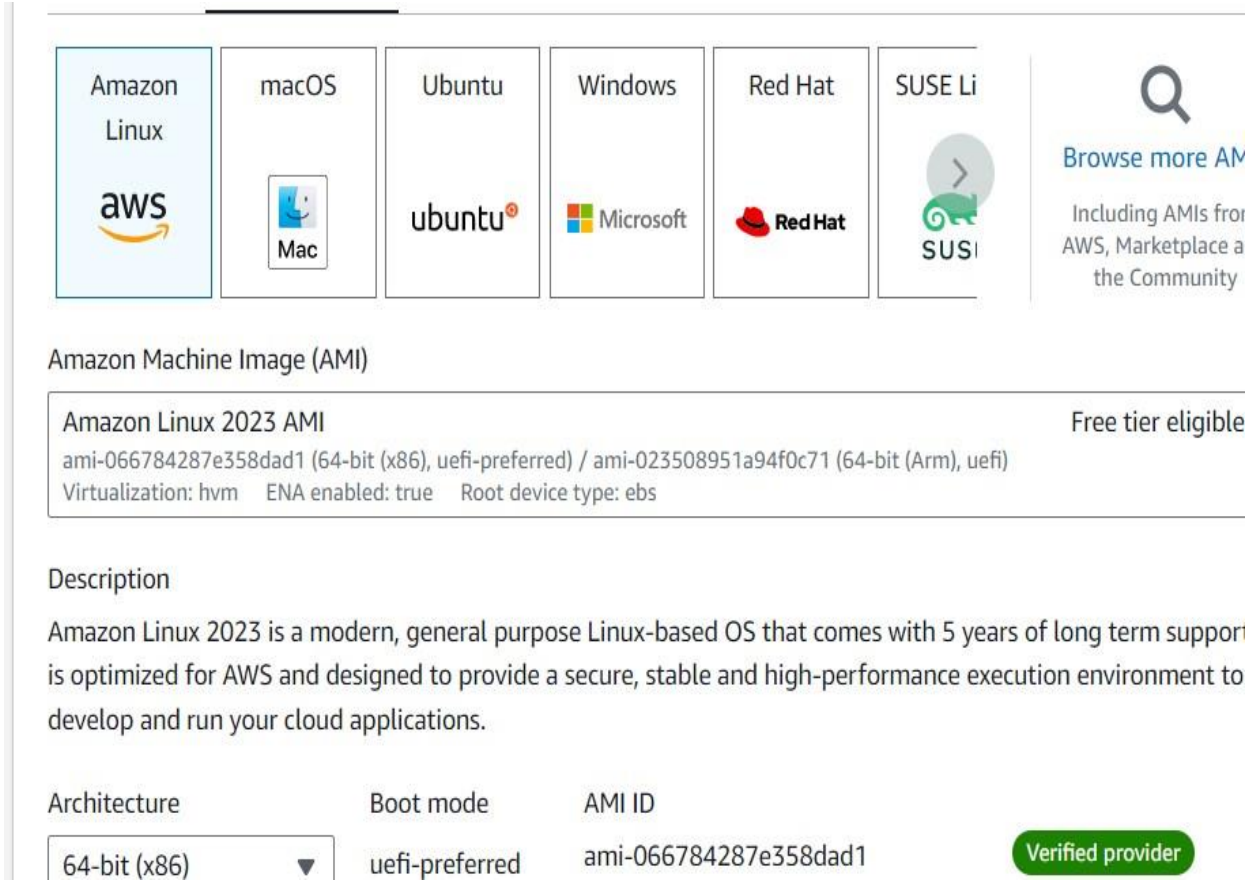
BEFORE -



AFTER -



Step 6 : Copy AWS AMI ID and change it in code

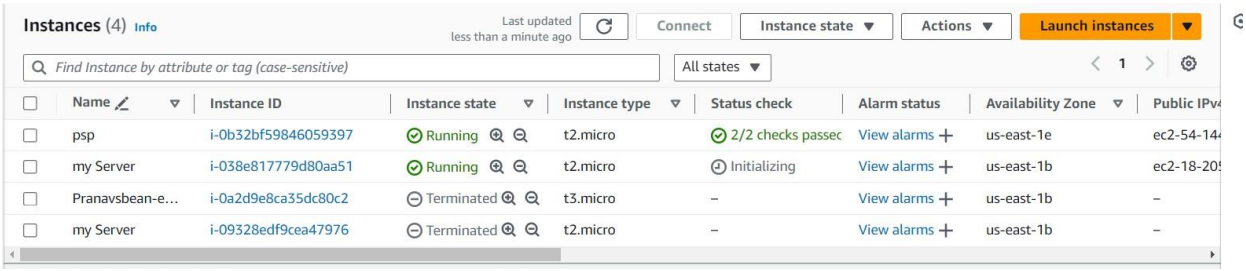


Step 7 : Type terraform plan and terraform apply command.

```
      + password_data                      = (known after apply)
      + placement_group                    = (known after apply)
      + placement_partition_number         = (known after apply)
      + primary_network_interface_id       = (known after apply)
      + private_dns                        = (known after apply)
      + private_ip                         = (known after apply)
      + public_dns                         = (known after apply)
      + public_ip                          = (known after apply)
      + secondary_private_ips              = (known after apply)
      + security_groups                    = (known after apply)
      + source_dest_check                  = (known after apply)
      + spot_instance_request_id           = (known after apply)
      + subnet_id                          = (known after apply)
      + tags                               = (known after apply)
      + tags_all                           = (known after apply)
      + tenancy                            = (known after apply)
      + user_data                          = (known after apply)
      + user_data_base64                   = (known after apply)
      + user_data_replace_on_change        = (known after apply)
      + volume_tags                        = (known after apply)
      + vpc_security_group_ids             = (known after apply)
    } -> (known after apply)
}

Plan: 1 to add, 0 to change, 1 to destroy.
```

```
      + public_ip                          = (known after apply)
      + secondary_private_ips              = (known after apply)
      + security_groups                    = (known after apply)
      + source_dest_check                  = (known after apply)
      + spot_instance_request_id           = (known after apply)
      + subnet_id                          = (known after apply)
      + tags                               = (known after apply)
      + tags_all                           = (known after apply)
      + tenancy                            = (known after apply)
      + user_data                          = (known after apply)
      + user_data_base64                   = (known after apply)
      + user_data_replace_on_change        = (known after apply)
      + volume_tags                        = (known after apply)
      + vpc_security_group_ids             = (known after apply)
    } -> (known after apply)
}

Plan: 1 to add, 0 to change, 1 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_instance.myServer: Destroying... [id=i-09328edf9cea47976]
aws_instance.myServer: Still destroying... [id=i-09328edf9cea47976, 10s elapsed]
aws_instance.myServer: Still destroying... [id=i-09328edf9cea47976, 20s elapsed]
aws_instance.myServer: Still destroying... [id=i-09328edf9cea47976, 30s elapsed]
aws_instance.myServer: Destruction complete after 33s
aws_instance.myServer: Creating...
aws_instance.myServer: Still creating... [10s elapsed]
aws_instance.myServer: Still creating... [20s elapsed]
aws_instance.myServer: Still creating... [30s elapsed]
aws_instance.myServer: Creation complete after 35s [id=i-038e817779d80aa51]

Apply complete! Resources: 1 added, 0 changed, 1 destroyed.
```

Step 8 : Instances after deleting window instance and creating AWS instance



Instances (4) Info

Last updated less than a minute ago | Connect | Instance state | Actions | Launch instances

Find Instance by attribute or tag (case-sensitive) | All states | 1

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4
psp	i-0b32bf59846059397	Running	t2.micro	2/2 checks passed	View alarms	us-east-1e	ec2-54-144
my Server	i-038e817779d80aa51	Running	t2.micro	Initializing	View alarms	us-east-1b	ec2-18-205
Pranavsbean-e...	i-0a2d9e8ca35dc80c2	Terminated	t3.micro	–	View alarms	us-east-1b	–
my Server	i-09328edf9cea47976	Terminated	t2.micro	–	View alarms	us-east-1b	–

Step 9 : Destroy the instance using terraform destroy

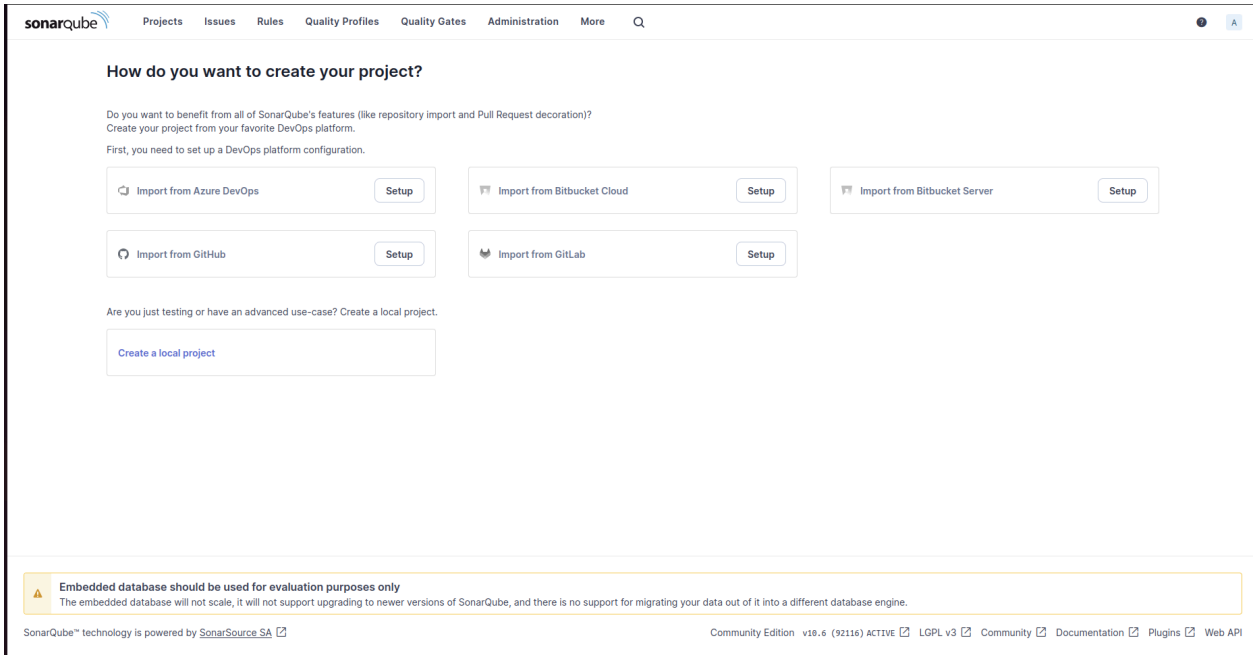
```
C:\Users\sbpol\Documents\terraform_scripts\docker\ec2>terraform destroy
aws_instance.myServer: Refreshing state... [id=i-038e817779d80aa51]

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with
  - destroy

Terraform will perform the following actions:

  # aws_instance.myServer will be destroyed
  - resource "aws_instance" "myServer" {
      - ami                                  = "ami-066784287e358dad1" -> null
      - arn                                  = "arn:aws:ec2:us-east-1:633393758690:instance/i-038e817779d80aa51" -
      - associate_public_ip_address          = true -> null
      - availability_zone                    = "us-east-1b" -> null
      - cpu_core_count                       = 1 -> null
      - cpu_threads_per_core                 = 1 -> null
      - disable_api_stop                     = false -> null
      - disable_api_termination              = false -> null
      - ebs_optimized                        = false -> null
      - get_password_data                    = false -> null
      - hibernation                          = false -> null
      - id                                   = "i-038e817779d80aa51" -> null
      - instance_initiated_shutdown_behavior = "stop" -> null
      - instance_state                       = "running" -> null
      - instance_type                        = "t2.micro" -> null
      - ipv6_address_count                   = 0 -> null
      - ipv6_addresses                       = [] -> null
      - monitoring                           = false -> null
      - placement_partition_number           = 0 -> null
      - primary_network_interface_id         = "eni-0c93e7a6f650aaacb" -> null
      - private_dns                          = "ip-172-31-84-36.ec2.internal" -> null
      - private_ip                           = "172.31.84.36" -> null
      - public_dns                           = "ec2-18-205-116-164.compute-1.amazonaws.com" -> null
      - public_ip                            = "18.205.116.164" -> null
      - secondary_private_ips                = [] -> null
      - security_groups                      = [
          - "default",
        ] -> null
      - source_dest_check                    = true -> null
    }

Plan: 0 to add, 0 to change, 1 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: yes

aws_instance.myServer: Destroying... [id=i-038e817779d80aa51]
aws_instance.myServer: Still destroying... [id=i-038e817779d80aa51, 10s elapsed]
aws_instance.myServer: Still destroying... [id=i-038e817779d80aa51, 20s elapsed]
aws_instance.myServer: Still destroying... [id=i-038e817779d80aa51, 30s elapsed]
aws_instance.myServer: Still destroying... [id=i-038e817779d80aa51, 40s elapsed]
aws_instance.myServer: Still destroying... [id=i-038e817779d80aa51, 50s elapsed]
aws_instance.myServer: Destruction complete after 53s

Destroy complete! Resources: 1 destroyed.
```

ADVANCE DEVOPS - 7

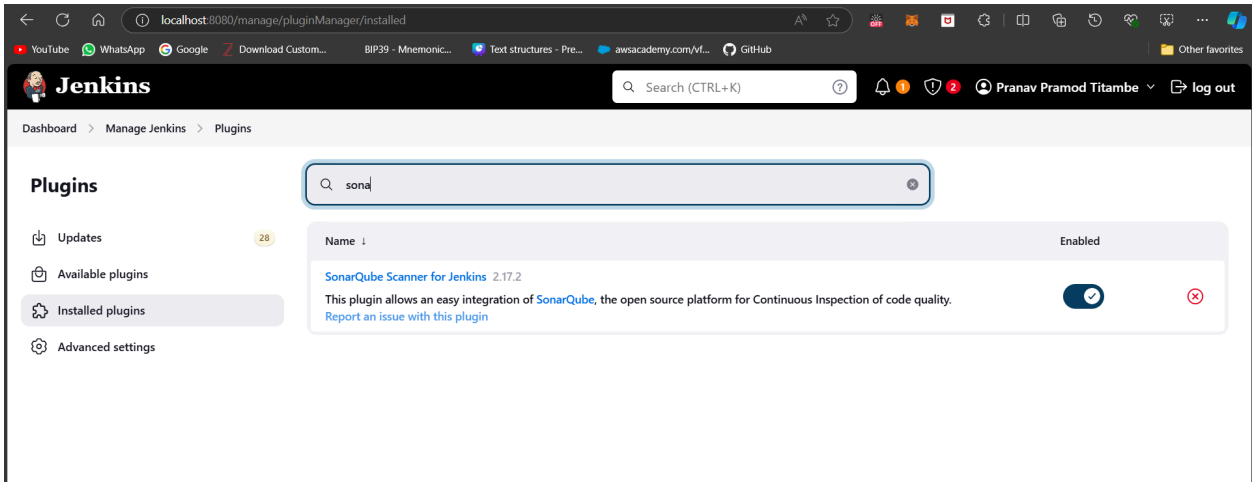
NAME - PRANAV TITAMBE D15A/62

Install SonarQube docker image from the docker registry.
SonarQube will run on http://localhost:9000

```
pranavtitambe@pranavtitambe:~$ sudo docker run -d --name sonarqube -e SONAR_ES_BOOTSTRAP_CHECKS_DISABLE=true -p 9000:9000 sonarqube:latest

Unable to find image 'sonarqube:latest' locally
latest: Pulling from library/sonarqube
7478e0ac0f23: Pull complete
90a925ab929a: Pull complete
7d9a34308537: Pull complete
80338217a4ab: Pull complete
1a5fd5c7e184: Pull complete
7b87d6fa783d: Downloading [======>                                            ]  91.37MB/738.4MB
bd819c9b5ead: Download complete
4f4fb700ef54: Download complete
```

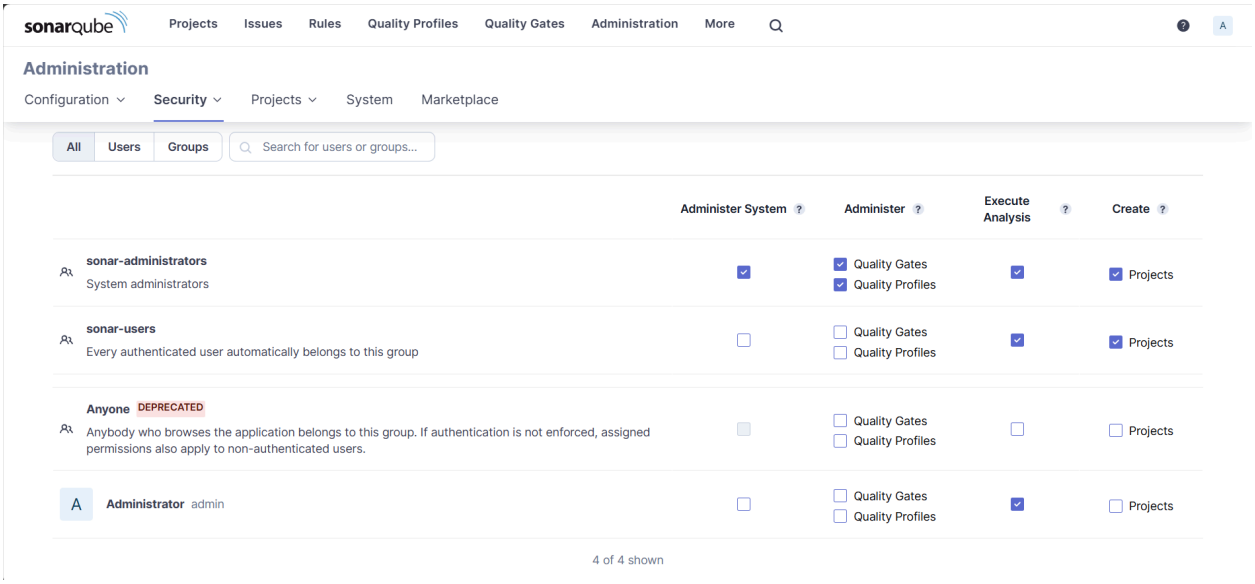


Step 1 - Login to Jenkins dashboard.

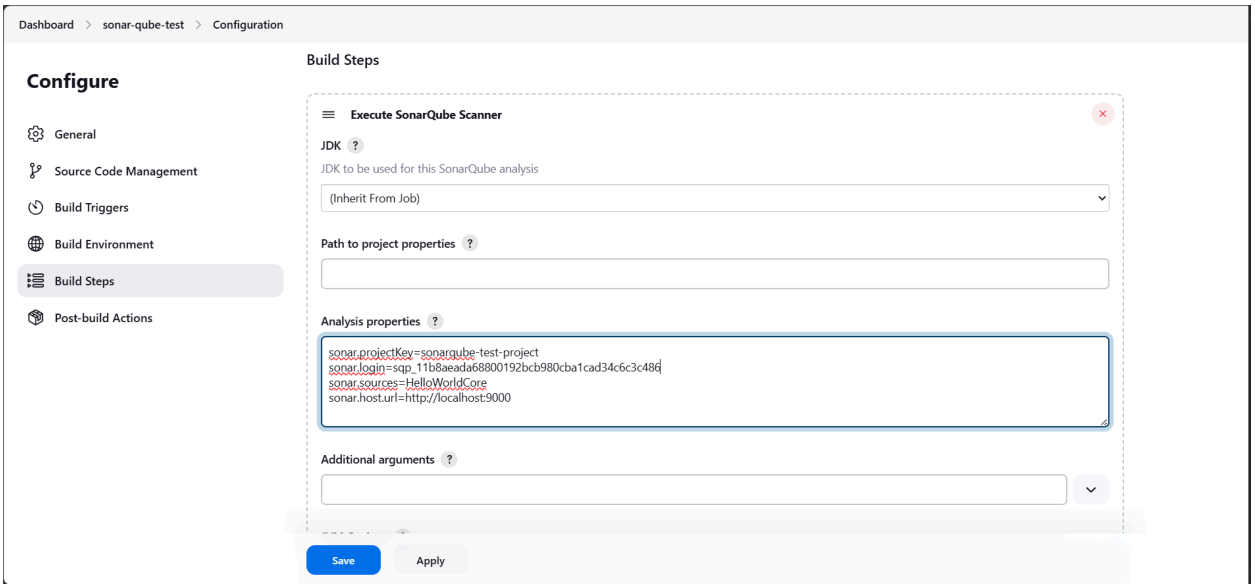


Step 2 - Add the SonarQube Server to the Jenkins configuration in Configure System option.

Step 3 - Login to SonarQube dashboard which is by default installed on http://localhost:9000



Step 4 - Create a new free style project and add the github link in the github section of this following new item and add these properties in the build steps.

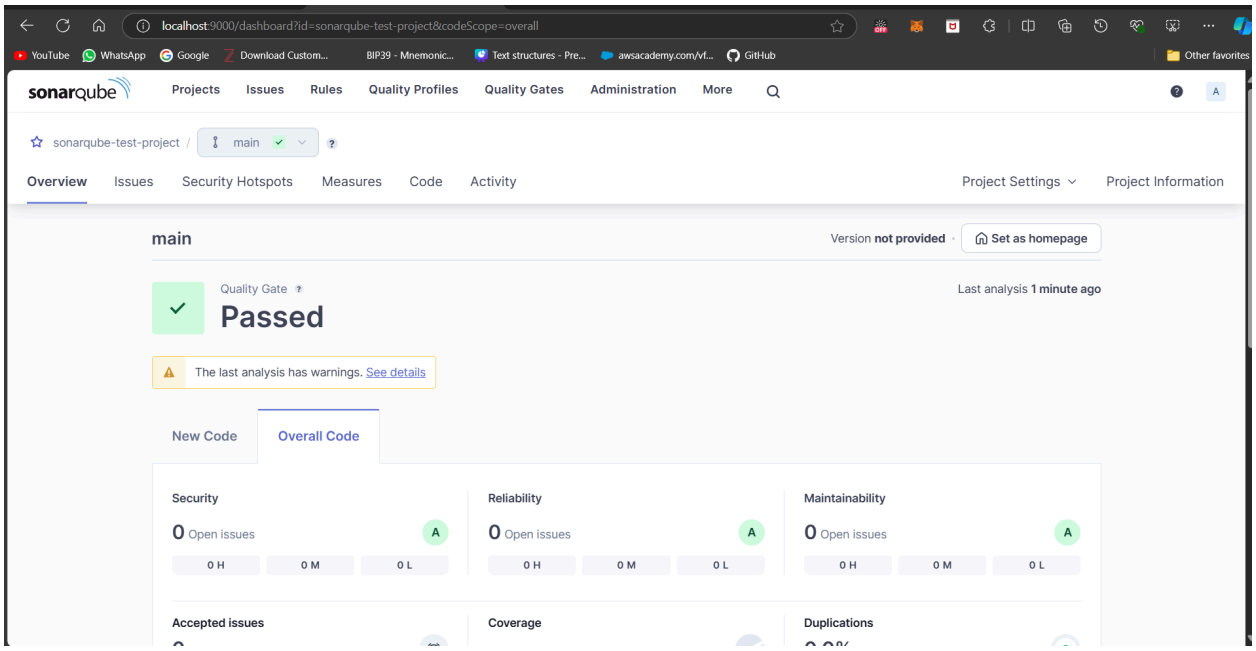


Step 5 - Click on Bulid now option to build your free style project

Console Output

```
Started by user Pranav Pramod Titambe
Running as SYSTEM
Building on the built-in node in workspace C:\ProgramData\Jenkins\.jenkins\workspace\sonar-qube-test
The recommended git tool is: NONE
No credentials specified
 > git.exe rev-parse --resolve-git-dir C:\ProgramData\Jenkins\.jenkins\workspace\sonar-qube-test\.git # timeout=10
Fetching changes from the remote Git repository
 > git.exe config remote.origin.url https://github.com/shazforiot/MSBuild_firstproject.git # timeout=10
Fetching upstream changes from https://github.com/shazforiot/MSBuild_firstproject.git
 > git.exe --version # timeout=10
 > git --version # 'git version 2.45.1.windows.1'
 > git.exe fetch --tags --force --progress -- https://github.com/shazforiot/MSBuild_firstproject.git +refs/heads/*:refs/remotes/origin/* #
timeout=10
 > git.exe rev-parse "refs/remotes/origin/master^{commit}" # timeout=10
Checking out Revision f2bc042c04c6e72427c380bcaee6d6fee7b49adf (refs/remotes/origin/master)
 > git.exe config core.sparsecheckout # timeout=10
 > git.exe checkout -f f2bc042c04c6e72427c380bcaee6d6fee7b49adf # timeout=10
Commit message: "updated"
 > git.exe rev-list --no-walk f2bc042c04c6e72427c380bcaee6d6fee7b49adf # timeout=10
Unpacking https://repo1.maven.org/maven2/org/sonarsource/scanner/cli/sonar-scanner-cli/6.2.0.4584/sonar-scanner-cli-6.2.0.4584.zip to
C:\ProgramData\Jenkins\.jenkins\tools\hudson.plugins.sonar.SonarRunnerInstallation\SonarQube on Jenkins
[sonar-qube-test] $ C:\ProgramData\Jenkins\.jenkins\tools\hudson.plugins.sonar.SonarRunnerInstallation\SonarQube\bin\sonar-scanner.bat -
Dsonar.host.url=http://localhost:9000 -Dsonar.projectKey=sonarqube-test-project -Dsonar.login=sqp_11b8aeada68800192bcb980cba1cad34c6c3c486 -
Dsonar.host.url=http://localhost:9000 -Dsonar.sources=HelloWorldCore -Dsonar.projectBaseDir=C:\ProgramData\Jenkins\.jenkins\workspace\sonar-qube-
test
10:50:31.017 WARN  Property 'sonar.host.url' with value 'http://localhost:9000' is overridden with value 'http://localhost:9000'
```

Step 6 - Check your SonarQube dashboard and you will get the following analysis.

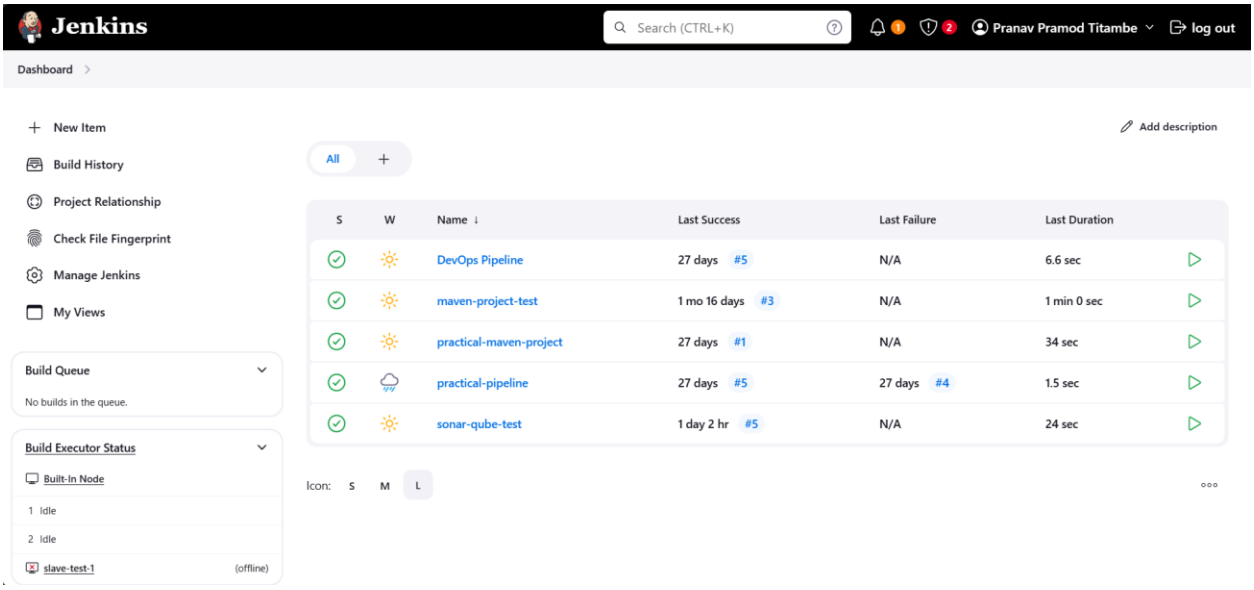


ADVANCE DEVOPS EXP-8

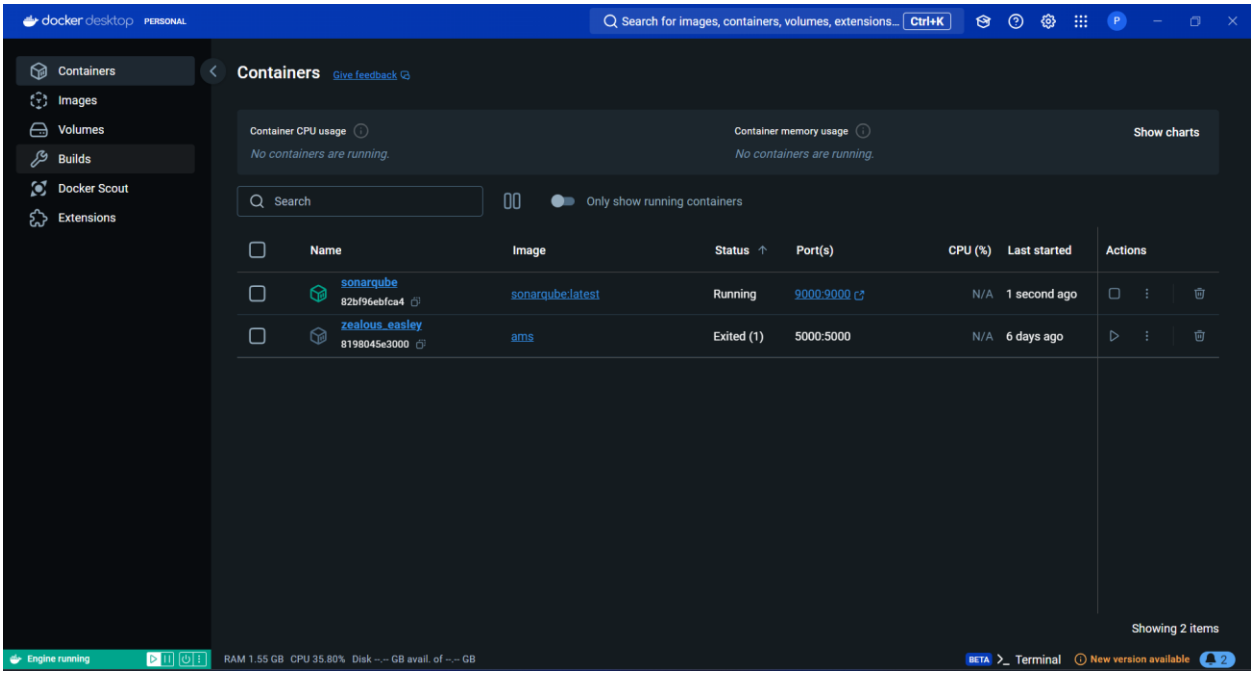
PRANAV TITAMBE D15A/62

Aim: To understand Static Analysis SAST process and learn to integrate Jenkins SAST to SonarQube/GitLab.

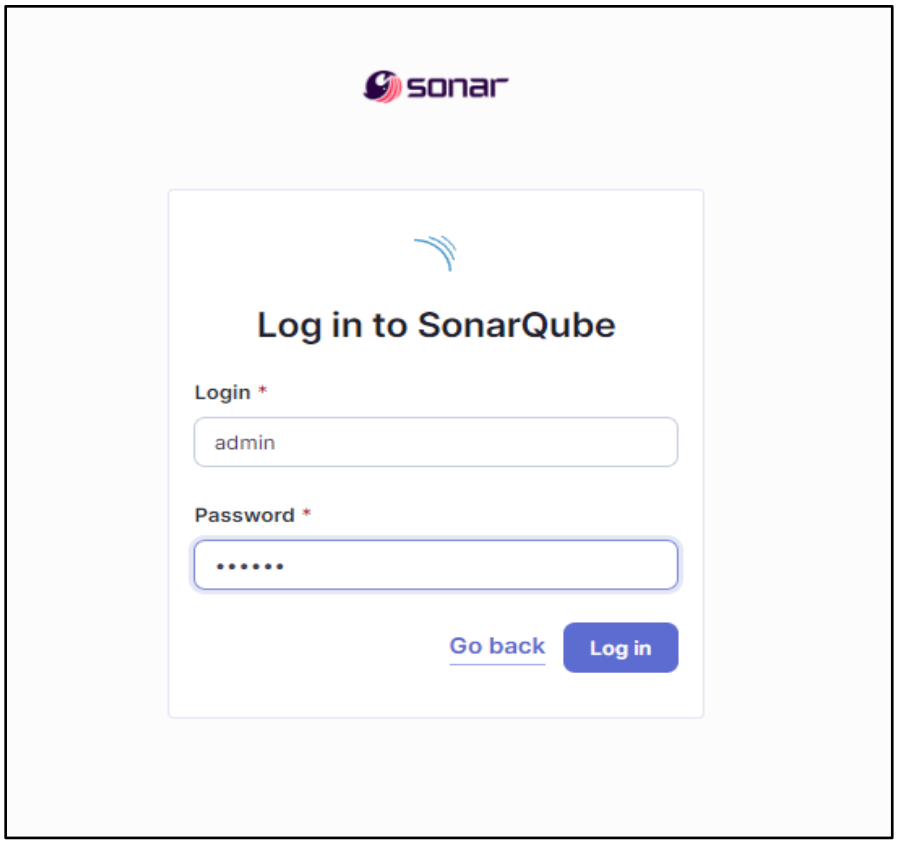
Step-1: Open up Jenkins Dashboard on localhost, port 8080 or whichever port it is at for you.



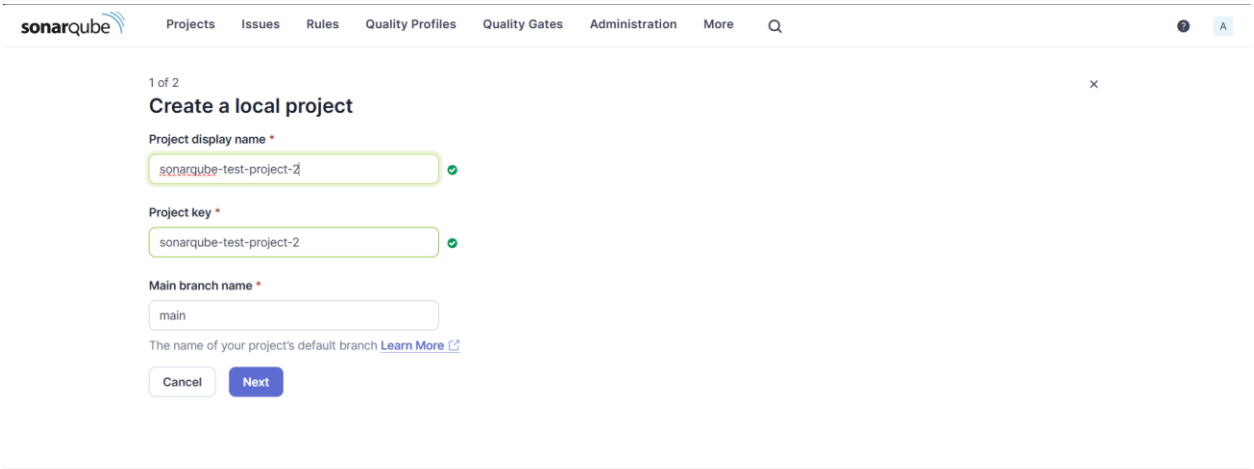
Step-2: Run SonarQube in a Docker container using this command :- a]docker -v
b] docker run -d --name sonarqube -e SONAR_ES_BOOTSTRAP_CHECKS_DISABLE=true -p 9000:9000 sonarqube:latest



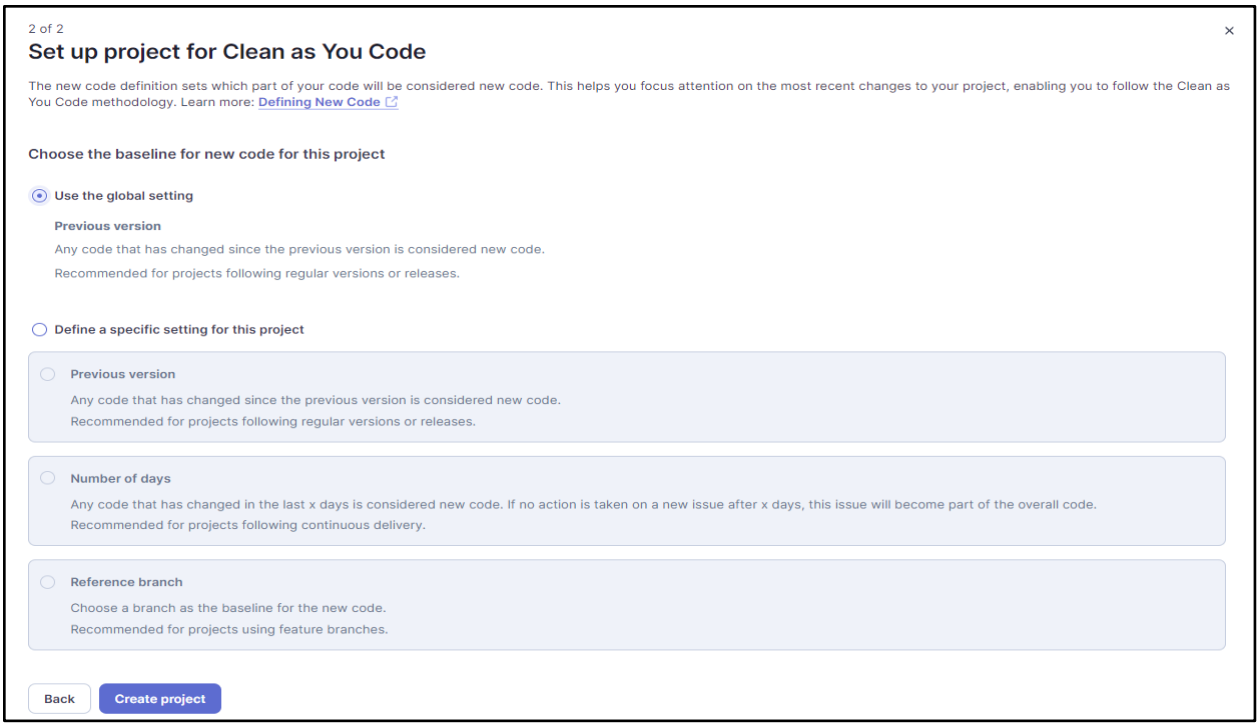
Step-3: Once the container is up and running, you can check the status of SonarQube at localhost port 9000. The login id is “admin” and the password is also “pranav”.



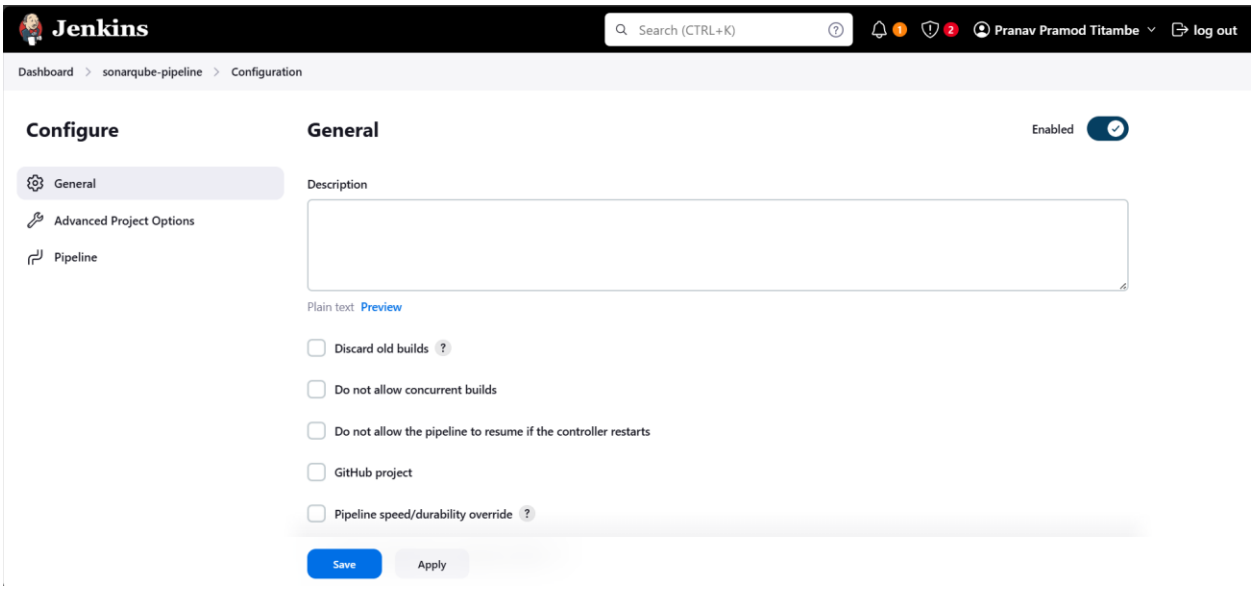
Step-4: Create a local project in SonarQube with the name sonarqube-test.



Step-5: Setup the project and come back to Jenkins Dashboard.



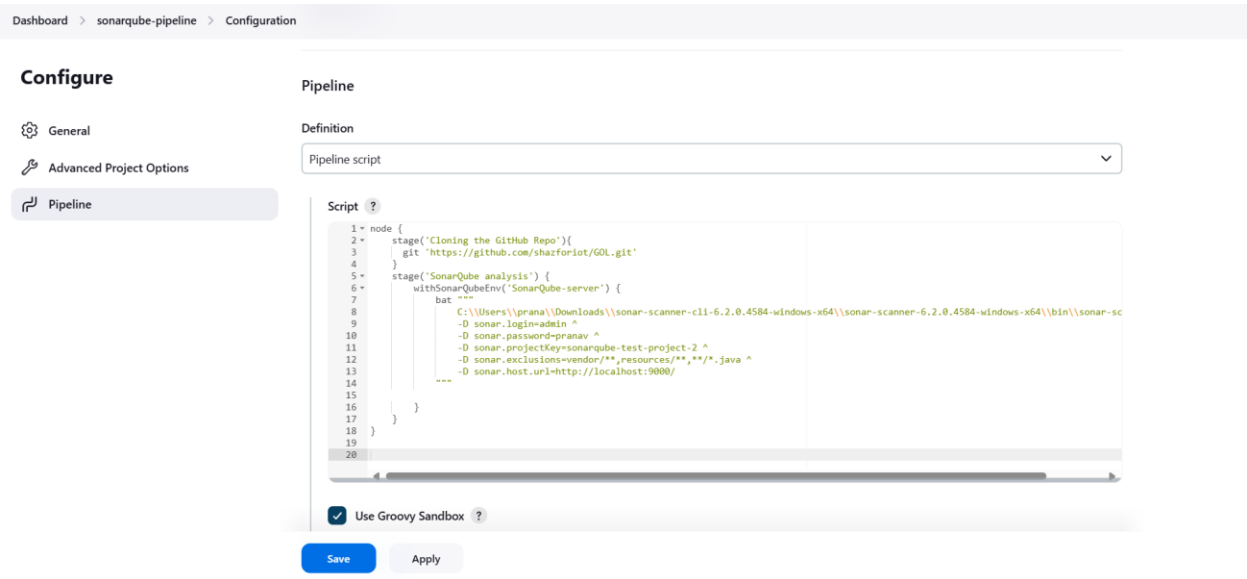
Step-6: Create a New Item in Jenkins, choose Pipeline.



Step-7: Under Pipeline Script, enter the following -

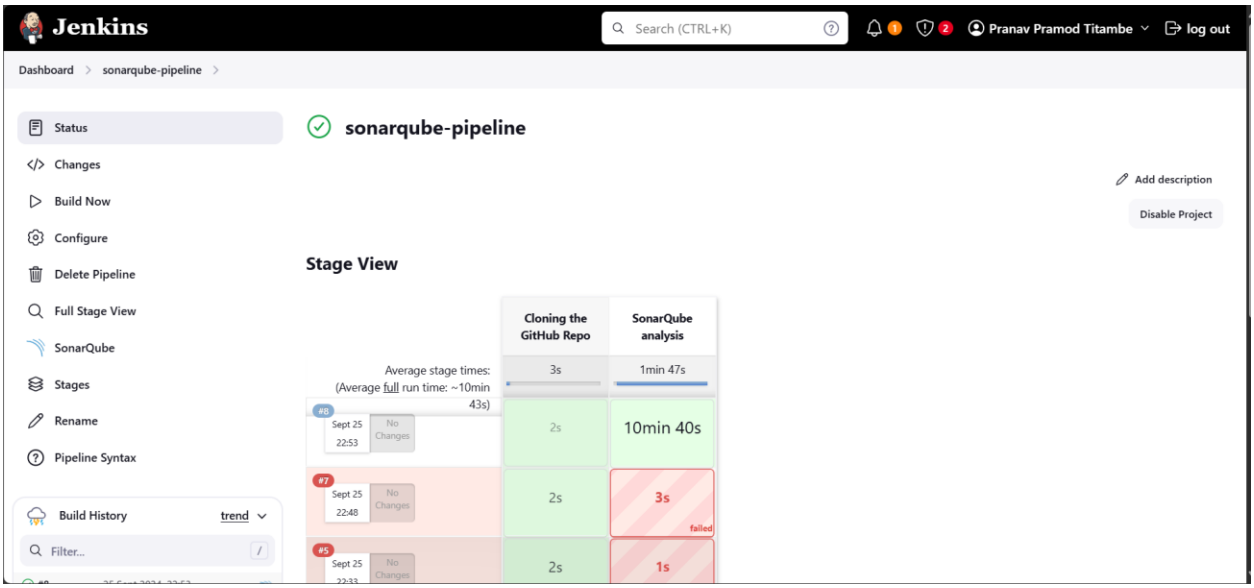
```
node {
   stage('Cloning the GitHub Repo'){
     git 'https://github.com/shazforiot/GOL.git'
   }
   stage('SonarQube analysis') {
```

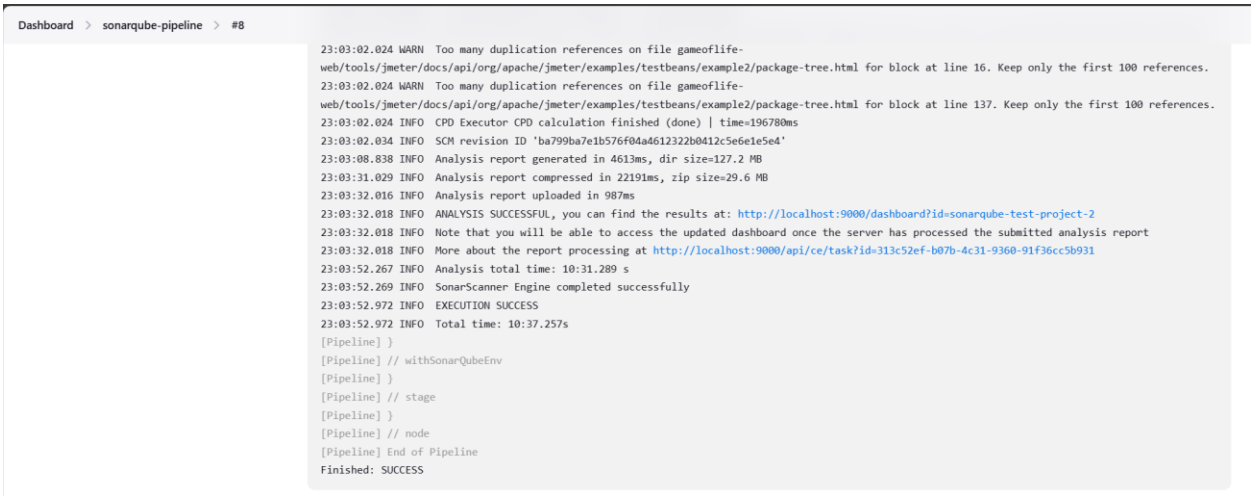
```
        withSonarQubeEnv('SonarQube-server') {
          bat """
            C:\\Users\\prana\\Downloads\\sonar-scanner-cli-6.2.0.4584-windows-x64\\sonar-scanner-6.2.0.4584-windows-x64\\bin\\sonar-scanner.bat ^
            -D sonar.login=admin ^
            -D sonar.password=pranav ^
            -D sonar.projectKey=sonarqube-test-project-2 ^
            -D sonar.exclusions=vendor/**,resources/**,**/*.java ^
            -D sonar.host.url=http://localhost:9000/
          """

        }
    }
}
```



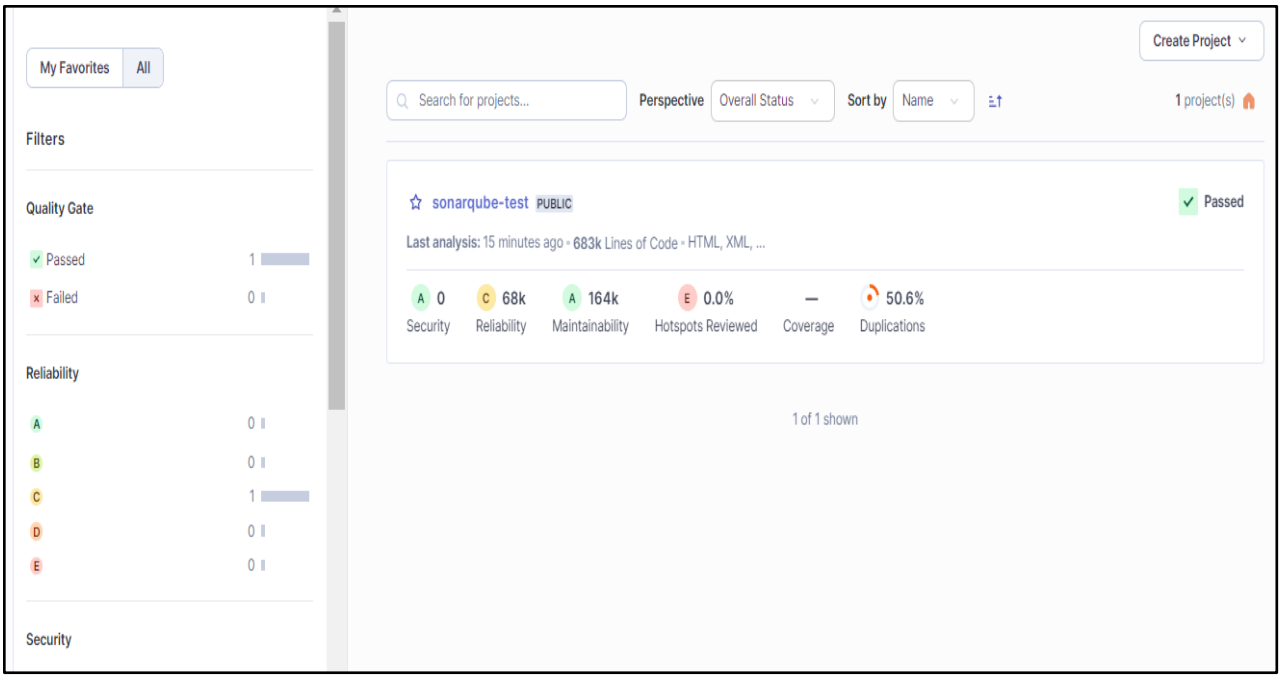
It is a java sample project which has a lot of repetitions and issues that will be detected by SonarQube.

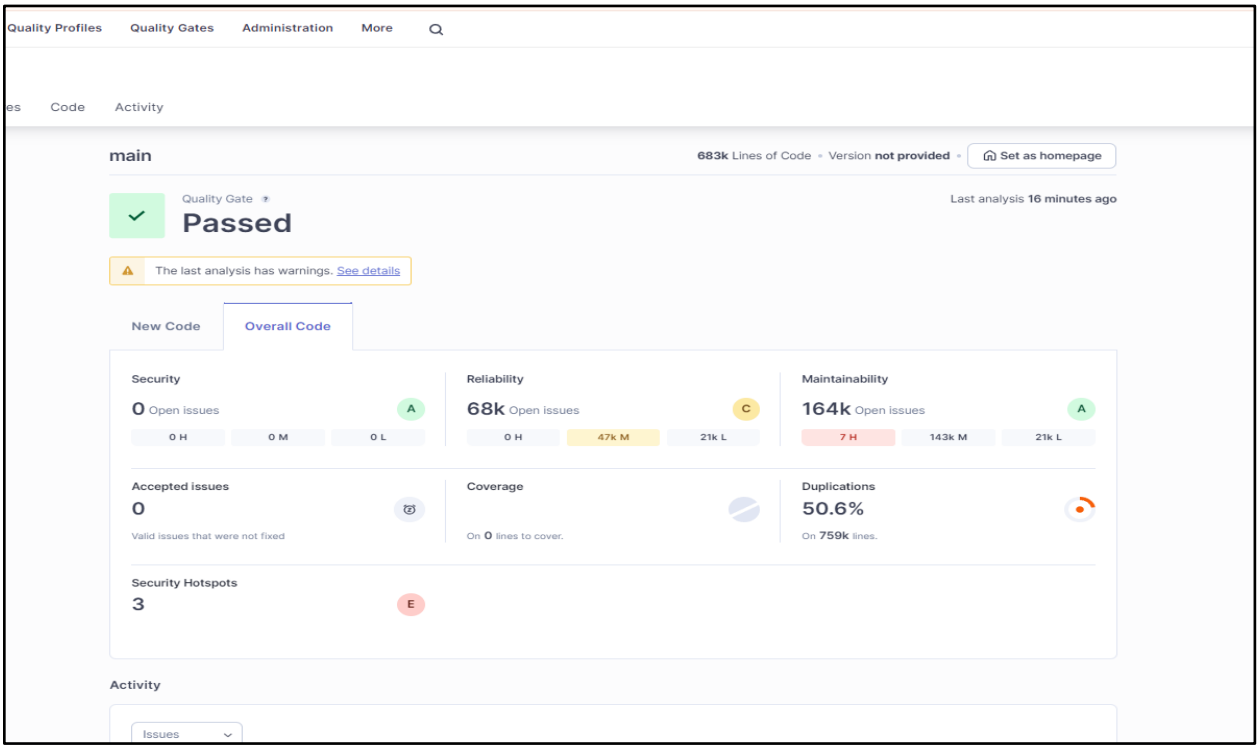
Step-8: Run The Build and check the console output:





Step-9: After that, check the project in SonarQube.

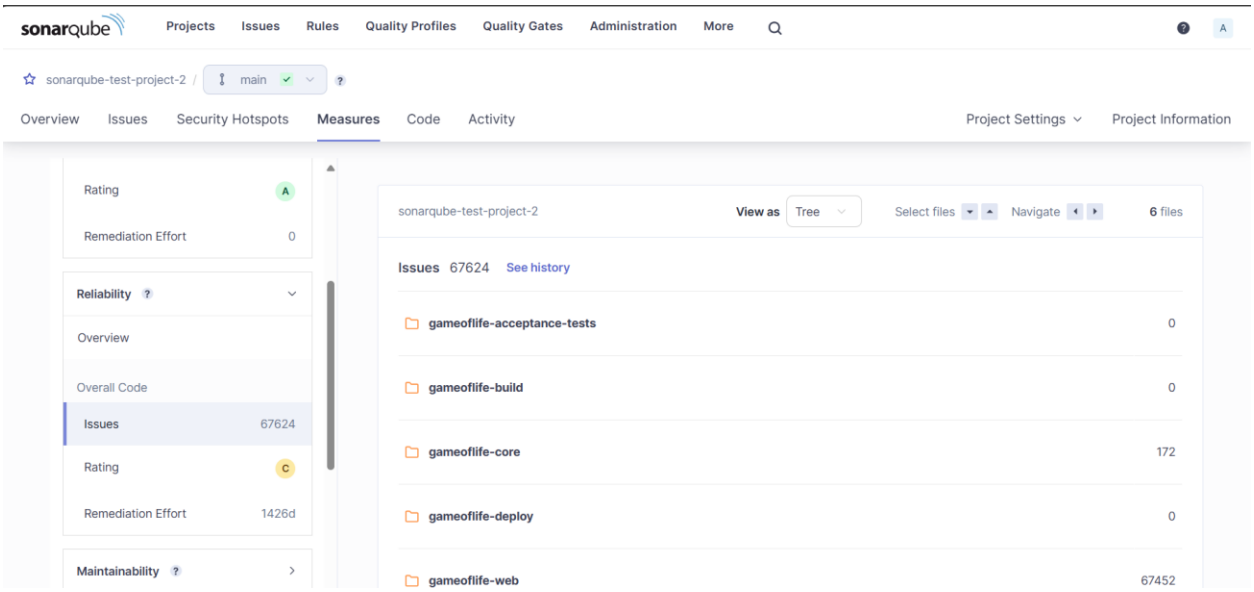




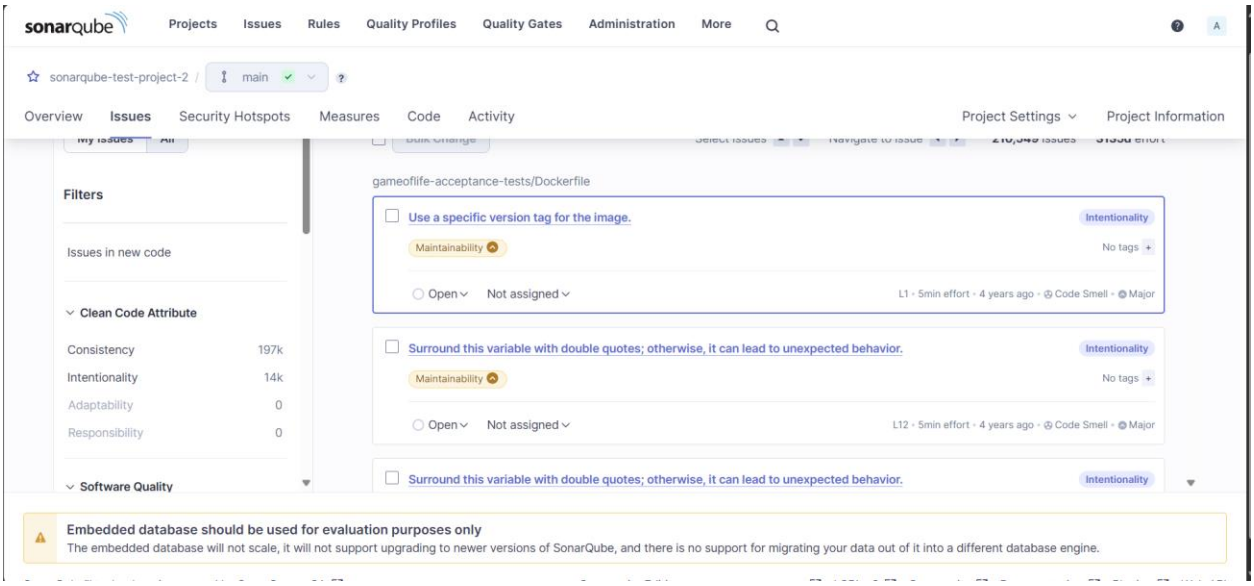
Step-10: Under different tabs, check all different issues with the code.

Code Problems

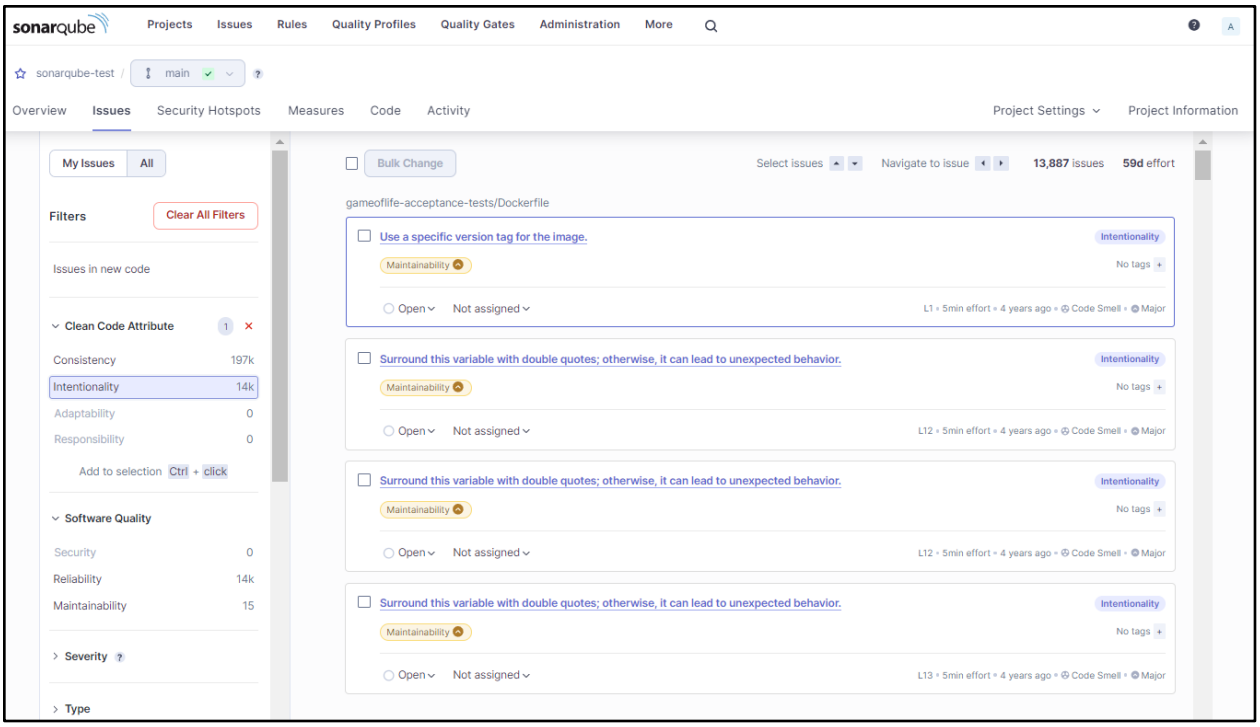
Code issues:



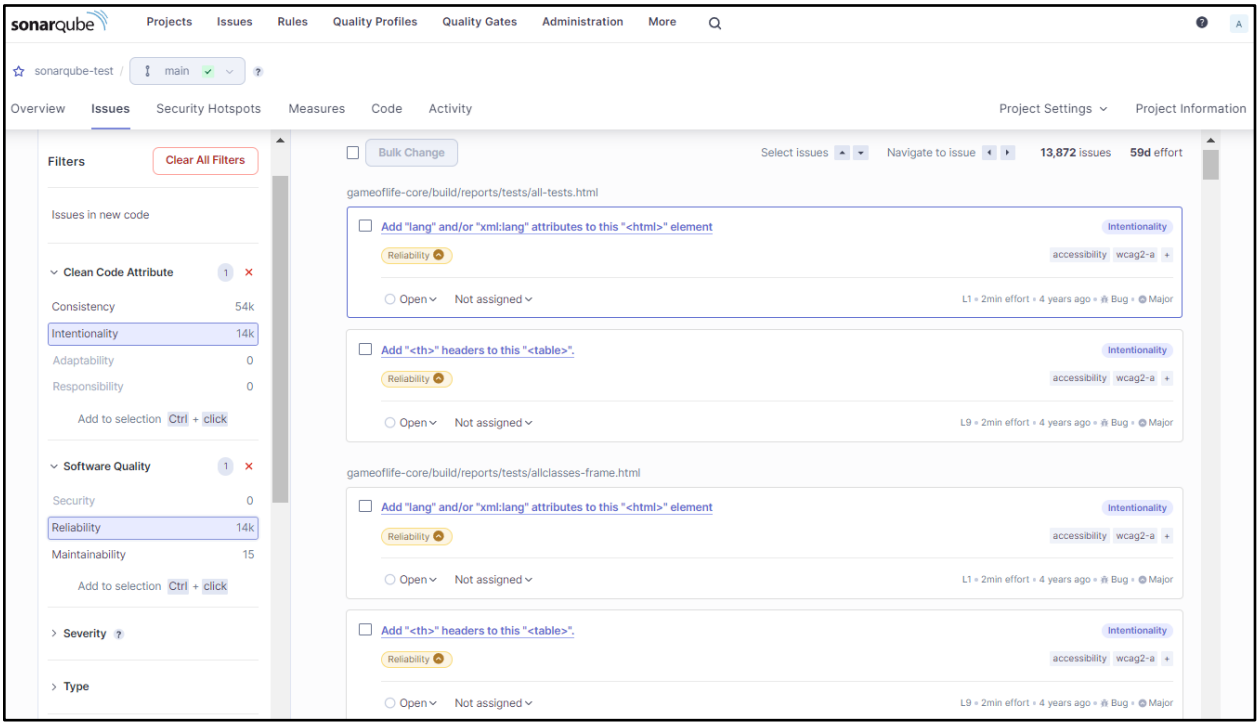
Consistency:



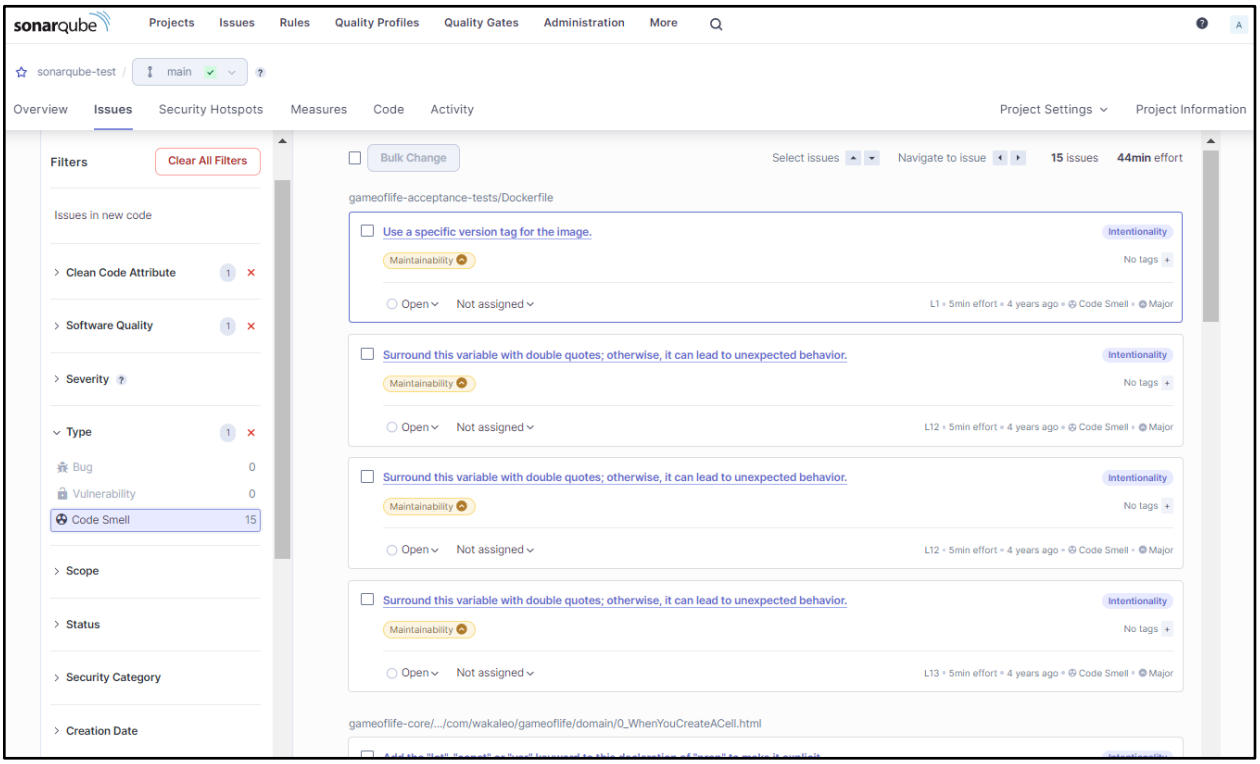
Intentionally:



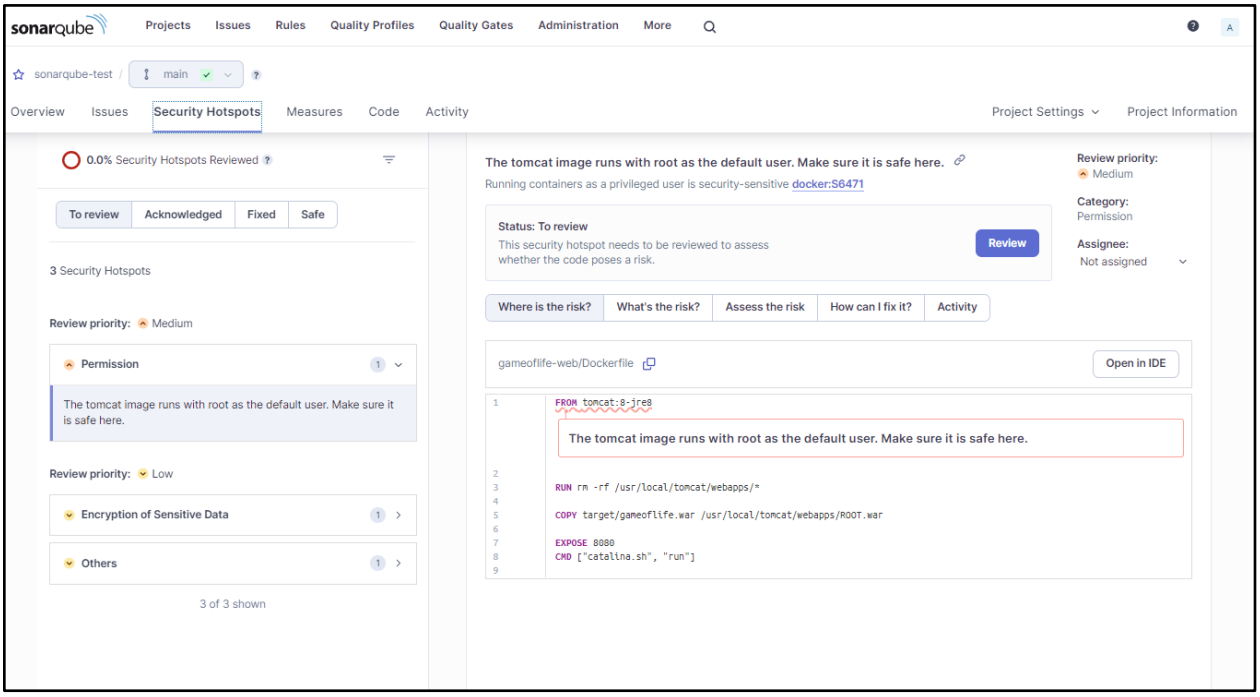
Reliability:



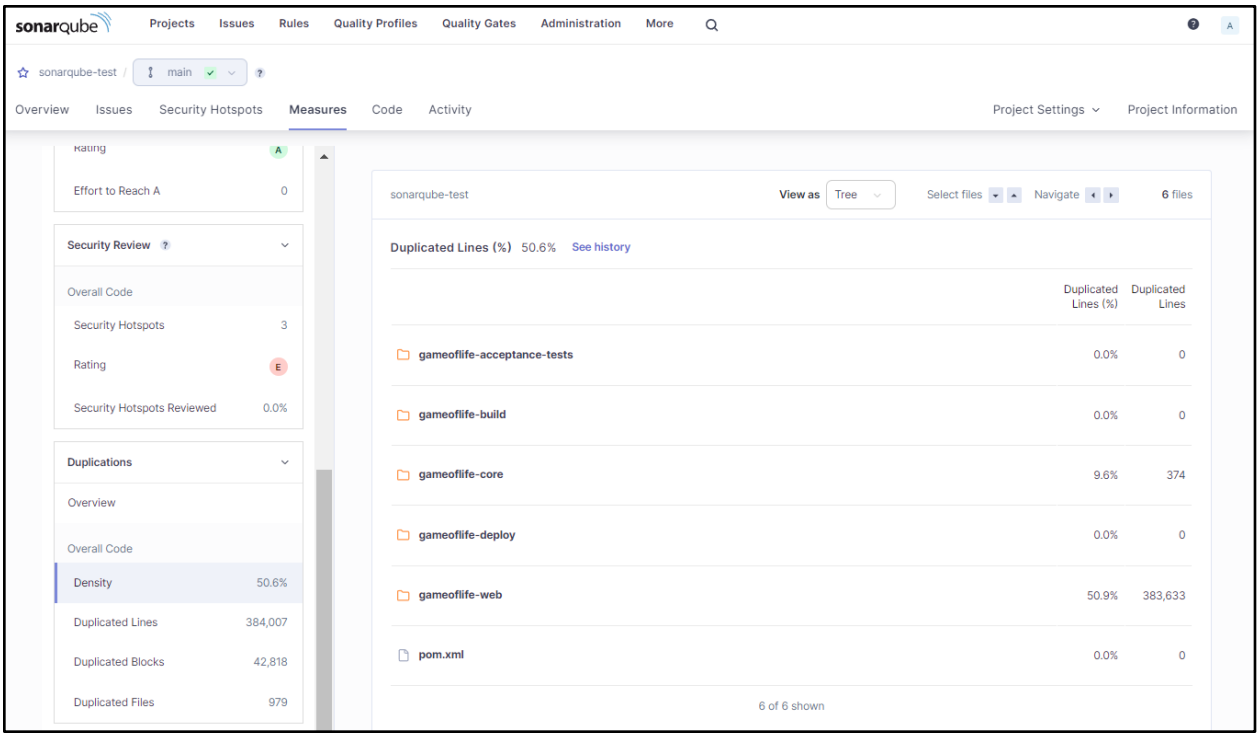
Code smells:



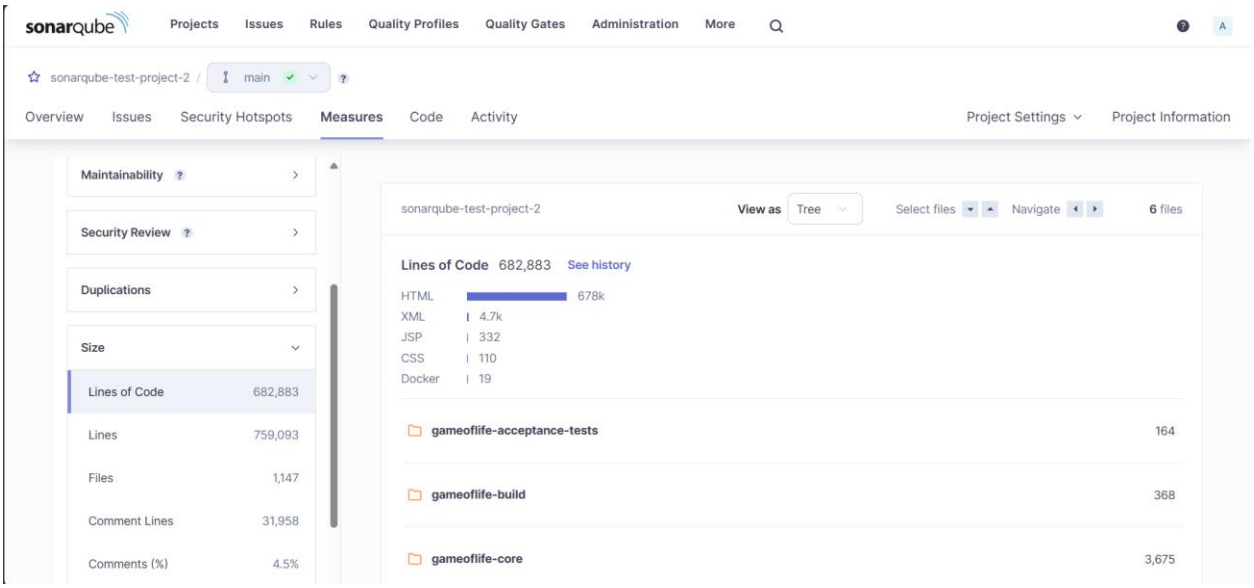
Security hotspot:



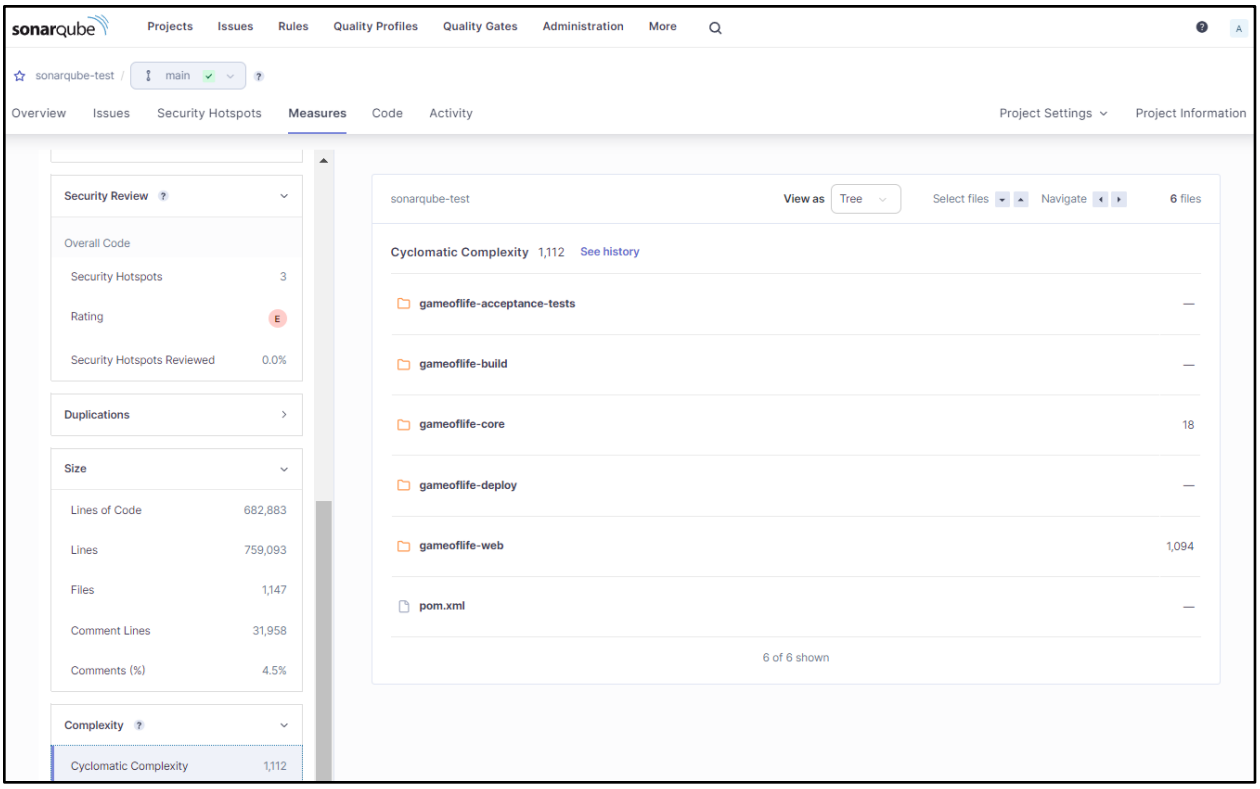
Duplicates:



Size:



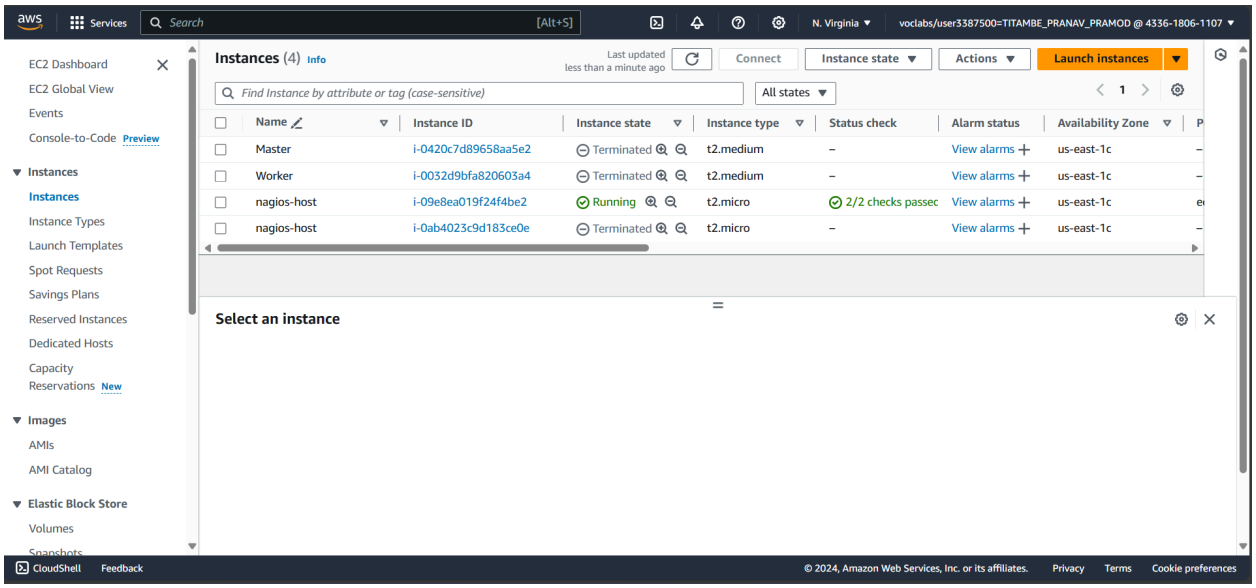
Complexity:


sonarqube
Projects
Issues
Rules
Quality Profiles
Quality Gates
Administration
More
sonarqube-test
main
Overview
Issues
Security Hotspots
Measures
Code
Activity
Project Settings
Project Information
Security Review
Overall Code
Security Hotspots 3
Rating E
Security Hotspots Reviewed 0.0%
Duplications
Size
Lines of Code 682,883
Lines 759,093
Files 1,147
Comment Lines 31,958
Comments (%) 4.5%
Complexity
Cyclomatic Complexity 1,112
sonarqube-test
View as Tree
Select files
Navigate
6 files
Cyclomatic Complexity 1,112 See history
gameoflife-acceptance-tests —
gameoflife-build —
gameoflife-core 18
gameoflife-deploy —
gameoflife-web 1,094
pom.xml —
6 of 6 shown

ADVANCE DEVOPS - 9

NAME - PRANAV TITAMBE 62/D15A

Create an EC2 instance at AWS



After this install following commands:-

```
sudo apt install apache2 libapache2-mod-php php php-gd libgd-dev gcc make autoconf libssl-dev wget unzip bc gawk dc build-essential snmp libnet-snmp-perl gettext -y
```

```
 System information as of Sat Sep 28 12:51:58 UTC 2024

  System load:  0.16              Processes:             106
  Usage of /:   10.6% of 14.46GB  Users logged in:       0
  Memory usage: 21%               IPv4 address for enX0: 172.31.89.161
  Swap usage:   0%

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status


The list of available updates is more than a week old.
To check for new updates run: sudo apt update


The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

ubuntu@ip-172-31-89-161:~$ sudo apt update
```

```
ubuntu@ip-172-31-89-161:~$ sudo apt install apache2 php
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  apache2-bin apache2-data apache2-utils libapache2-mod-php8.3 libapr1t64 libaprutil1-dbd-sqlite3 libaprutil1-ldap
  libaprutil1t64 liblua5.4-0 php-common php8.3 php8.3-cli php8.3-common php8.3-opcache php8.3-readline ssl-cert
Suggested packages:
  apache2-doc apache2-suexec-pristine | apache2-suexec-custom www-browser php-pear
The following NEW packages will be installed:
  apache2 apache2-bin apache2-data apache2-utils libapache2-mod-php8.3 libapr1t64 libaprutil1-dbd-sqlite3
  libaprutil1-ldap libaprutil1t64 liblua5.4-0 php php-common php8.3 php8.3-cli php8.3-common php8.3-opcache
  php8.3-readline ssl-cert
0 upgraded, 18 newly installed, 0 to remove and 143 not upgraded.
Need to get 6998 kB of archives.
After this operation, 30.5 MB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 libapr1t64 amd64 1.7.2-3.1ubuntu0.1 [108 k
B]
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 libaprutil1t64 amd64 1.6.3-1.1ubuntu7 [91.9 kB]
Get:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 libaprutil1-dbd-sqlite3 amd64 1.6.3-1.1ubuntu7 [11
.2 kB]
Get:4 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 libaprutil1-ldap amd64 1.6.3-1.1ubuntu7 [9116 B]
Get:5 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 liblua5.4-0 amd64 5.4.6-3build2 [166 kB]
Get:6 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 apache2-bin amd64 2.4.58-1ubuntu8.4 [1329
kB]
Get:7 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 apache2-data all 2.4.58-1ubuntu8.4 [163 kB
]
Get:8 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 apache2-utils amd64 2.4.58-1ubuntu8.4 [97.
1 kB]
```

```
ubuntu@ip-172-31-89-161:~$ sudo apt install gcc build-essential
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  binutils binutils-common binutils-x86-64-linux-gnu bzip2 cpp cpp-13 cpp-13-x86-64-linux-gnu cpp-x86-64-linux-gnu
  dpkg dpkg-dev fakeroot fontconfig-config fonts-dejavu-core fonts-dejavu-mono g++ g++-13 g++-13-x86-64-linux-gnu
  g++-x86-64-linux-gnu gcc-13 gcc-13-base gcc-13-x86-64-linux-gnu gcc-x86-64-linux-gnu libalgorithm-diff-perl
  libalgorithm-diff-xs-perl libalgorithm-merge-perl libaom3 libasan8 libatomic1 libbinutils libbz2-1.0 libc-bin
  libc-dev-bin libc-devtools libc6 libc6-dev libcc1-0 libcrypt-dev libctf-nobfd0 libctf0 libde265-0 libdeflate0
  libdpkg-perl libfakeroot libfile-fcntllock-perl libfontconfig1 libgcc-13-dev libgd3 libgomp1 libgprofng0
  libheif-plugin-aomdec libheif-plugin-aomenc libheif-plugin-libde265 libheif1 libhwasan0 libisl23 libitm1 libjbig0
  libjpeg-turbo8 libjpeg8 liblerc4 liblsan0 libmpc3 libquadmath0 libsframe1 libsharpyuv0 libstdc++-13-dev libtiff6
  libtsan2 libubsan1 libwebp7 libxpm4 linux-libc-dev linux-tools-common locales lto-disabled-list make manpages-dev
  rpcsvc-proto
Suggested packages:
  binutils-doc gprofng-gui bzip2-doc cpp-doc gcc-13-locales cpp-13-doc debsig-verify debian-keyring g++-multilib
  g++-13-multilib gcc-13-doc gcc-multilib autoconf automake libtool flex bison gdb gcc-doc gcc-13-multilib
  gdb-x86-64-linux-gnu glibc-doc libnss-nis libnss-nisplus bzr libgd-tools libheif-plugin-x265
  libheif-plugin-ffmpegdec libheif-plugin-jpegdec libheif-plugin-jpegenc libheif-plugin-j2kdec libheif-plugin-j2kenc
  libheif-plugin-rav1e libheif-plugin-svtenc libstdc++-13-doc make-doc
The following NEW packages will be installed:
  binutils binutils-common binutils-x86-64-linux-gnu build-essential bzip2 cpp cpp-13 cpp-13-x86-64-linux-gnu
  cpp-x86-64-linux-gnu dpkg-dev fakeroot fontconfig-config fonts-dejavu-core fonts-dejavu-mono g++ g++-13
  g++-13-x86-64-linux-gnu g++-x86-64-linux-gnu gcc gcc-13 gcc-13-base gcc-13-x86-64-linux-gnu gcc-x86-64-linux-gnu
  libalgorithm-diff-perl libalgorithm-diff-xs-perl libalgorithm-merge-perl libaom3 libasan8 libatomic1 libbinutils
  libc-dev-bin libc-devtools libc6-dev libcc1-0 libcrypt-dev libctf-nobfd0 libctf0 libde265-0 libdeflate0 libdpkg-perl
  libfakeroot libfile-fcntllock-perl libfontconfig1 libgcc-13-dev libgd3 libgomp1 libgprofng0 libheif-plugin-aomdec
  libheif-plugin-aomenc libheif-plugin-libde265 libheif1 libhwasan0 libisl23 libitm1 libjbig0 libjpeg-turbo8 libjpeg8
  liblerc4 liblsan0 libmpc3 libquadmath0 libsframe1 libsharpyuv0 libstdc++-13-dev libtiff6 libtsan2 libubsan1 libwebp7
```

```
ubuntu@ip-172-31-89-161:~$ sudo apt install libgd-dev
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
libgd-dev is already the newest version (2.3.3-9ubuntu5).
0 upgraded, 0 newly installed, 0 to remove and 119 not upgraded.
ubuntu@ip-172-31-89-161:~$
```

Use this command to setup a new user called as nagios and adding it to nagcmd user group

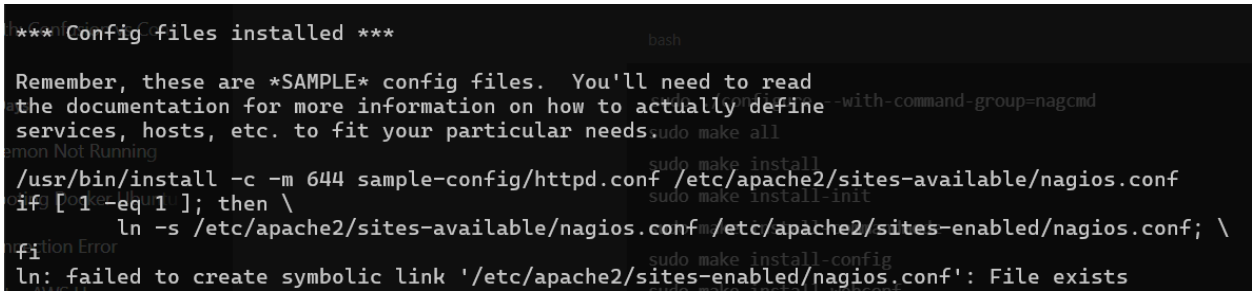
```
sudo useradd nagios
sudo groupadd nagcmd
sudo usermod -aG nagcmd nagios
sudo usermod -aG nagcmd www-data
```

Now download and install nagios core

```
cd /tmp
wget
https://assets.nagios.com/downloads/nagioscore/releases/nagios-4.4.6.
tar.gz
tar -zxvf nagios-*.tar.gz
cd nagios-4.4.6
```

Now compile and install nagios

```
sudo ./configure --with-command-group=nagcmd
sudo make all
sudo make install
sudo make install-init
sudo make install-commandmode
sudo make install-config
sudo make install-webconf
```



Now install Nagios plugins.

```
cd /tmp
wget https://nagios-plugins.org/download/nagios-plugins-2.3.3.tar.gz
tar -zxvf nagios-plugins-*.tar.gz
cd nagios-plugins-2.3.3
sudo ./configure --with-nagios-user=nagios --with-nagios-group=nagios
sudo make
sudo make install
```

```
  GNU nano 7.2                          /usr/local/nagios/etc/objects/contacts.cfg *

# Just one contact defined by default - the Nagios admin (that's you)
# This contact definition inherits a lot of default values from the
# 'generic-contact' template which is defined elsewhere.

define contact {

    contact_name            nagiosadmin             ; Short name of user
    use                     generic-contact         ; Inherit default values from generic-contact template (defined abo>
    alias                   Nagios Admin            ; Full name of user
    email                   pranavtitambe04@gmail.com ; <<***** CHANGE THIS TO YOUR EMAIL ADDRESS ******
}



###############################################################################
#
# CONTACT GROUPS
#
###############################################################################

# We only have one contact in this simple configuration file, so there is
# no need to create more than one contact group.

define contactgroup {

^G Help        ^O Write Out   ^W Where Is    ^K Cut         ^T Execute     ^C Location    M-U Undo       M-A Set Mark
^X Exit        ^R Read File   ^\ Replace     ^U Paste       ^J Justify     ^/ Go To Line  M-E Redo       M-6 Copy
```

Here by executing this command you need to set the password for ‘nagiosadmin’

```
sudo htpasswd -c /usr/local/nagios/etc/htpasswd.users nagiosadmin
```

```
ubuntu@ip-172-31-89-161:/tmp/nagios-plugins-2.3.3$ sudo htpasswd -c /usr/local/nagios/etc/htpasswd.users nagiosadmin
New password:
Re-type new password:
Adding password for user nagiosadmin
ubuntu@ip-172-31-89-161:/tmp/nagios-plugins-2.3.3$ sudo a2enmod rewrite
sudo a2enmod cgi
sudo systemctl restart apache2
Module rewrite already enabled
Module cgi already enabled
ubuntu@ip-172-31-89-161:/tmp/nagios-plugins-2.3.3$
```

Set proper permissions for the configuration:

```
sudo a2enmod rewrite
sudo a2enmod cgi
sudo systemctl restart apache2
```

```
ubuntu@ip-172-31-89-161:~$ sudo a2enmod rewrite
sudo a2enmod cgi
sudo systemctl restart apache2
Module rewrite already enabled
Module cgi already enabled
ubuntu@ip-172-31-89-161:~$
```

Ensure your firewall allows HTTP traffic (port 80):

```
sudo ufw allow Apache
sudo ufw reload
```

```
ubuntu@ip-172-31-89-161:~$ sudo ufw allow Apache
sudo ufw reload
Skipping adding existing rule
Skipping adding existing rule (v6)
Firewall not enabled (skipping reload)
ubuntu@ip-172-31-89-161:~$
```

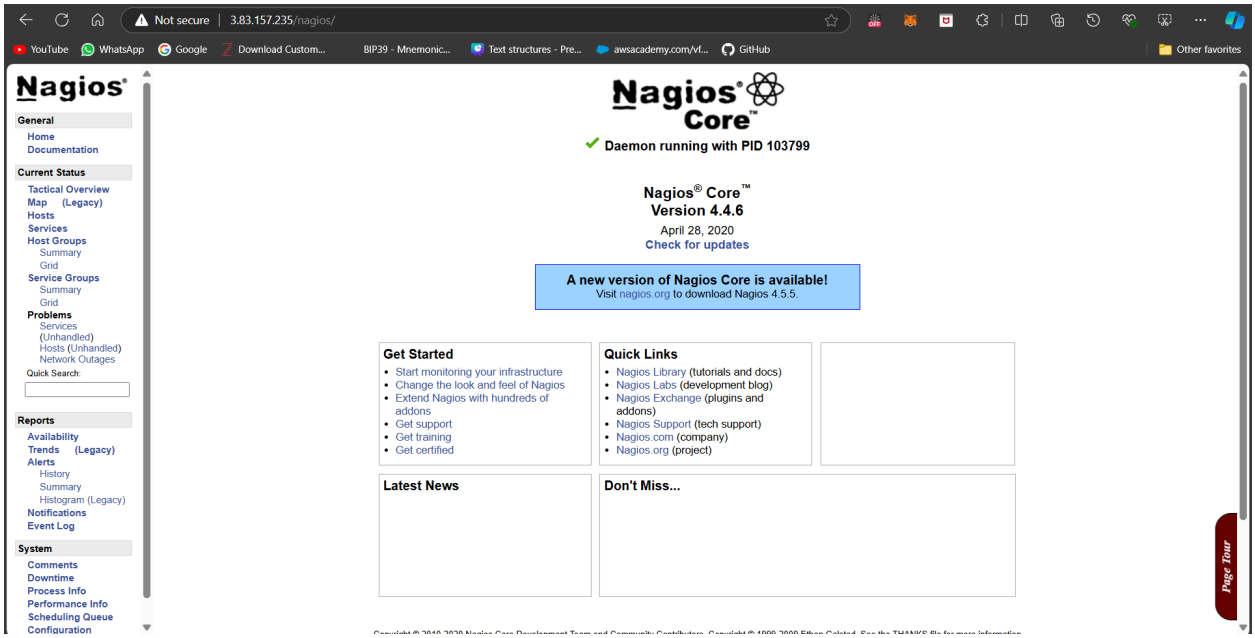
Starting the nagios services

```
sudo systemctl enable nagios
sudo systemctl start nagios
sudo systemctl status nagios
```

```
ubuntu@ip-172-31-89-161:~$ sudo systemctl enable nagios
sudo systemctl start nagios
ubuntu@ip-172-31-89-161:~$ sudo systemctl status nagios
● nagios.service - Nagios Core 4.4.6
     Loaded: loaded (/usr/lib/systemd/system/nagios.service; enabled; preset: enabled)
     Active: active (running) since Sat 2024-09-28 13:59:55 UTC; 19min ago
       Docs: https://www.nagios.org/documentation
   Main PID: 103799 (nagios)
      Tasks: 6 (limit: 1130)
     Memory: 2.3M (peak: 4.4M)
        CPU: 341ms
     CGroup: /system.slice/nagios.service
             ├─103799 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg
             ├─103800 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─103801 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─103802 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─103803 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             └─103805 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg

Sep 28 14:04:17 ip-172-31-89-161 nagios[103799]: SERVICE ALERT: localhost;Swap Usage;CRITICAL;SOFT;1;SWAP CRITICAL - 0%>
Sep 28 14:05:17 ip-172-31-89-161 nagios[103799]: SERVICE ALERT: localhost;Swap Usage;CRITICAL;SOFT;2;SWAP CRITICAL - 0%>
Sep 28 14:06:17 ip-172-31-89-161 nagios[103799]: SERVICE ALERT: localhost;Swap Usage;CRITICAL;SOFT;3;SWAP CRITICAL - 0%>
Sep 28 14:07:17 ip-172-31-89-161 nagios[103799]: SERVICE NOTIFICATION: nagiosadmin;localhost;Swap Usage;CRITICAL;notify>
Sep 28 14:07:17 ip-172-31-89-161 nagios[103799]: SERVICE ALERT: localhost;Swap Usage;CRITICAL;HARD;4;SWAP CRITICAL - 0%>
Sep 28 14:07:17 ip-172-31-89-161 nagios[103799]: wproc: NOTIFY job 4 from worker Core Worker 103800 is a non-check help>
Sep 28 14:07:17 ip-172-31-89-161 nagios[103799]: wproc:   host=localhost; service=Swap Usage; contact=nagiosadmin
Sep 28 14:07:17 ip-172-31-89-161 nagios[103799]: wproc:   early_timeout=0; exited_ok=1; wait_status=32512; error_code=0;
Sep 28 14:07:17 ip-172-31-89-161 nagios[103799]: wproc:   stderr line 01: /bin/sh: 1: /bin/mail: not found
Sep 28 14:07:17 ip-172-31-89-161 nagios[103799]: wproc:   stderr line 02: /usr/bin/printf: write error: Broken pipe
lines 1-26/26 (END)
```

Here it will prompt to enter the username and password provide that we set earlier.
Go to your AWS EC2 dashboard and and copy the public ip address. The url should look like this

‘`http://<public-ip-address>/nagios`’


Not secure | 3.83.157.235/nagios/
YouTube
WhatsApp
Google
Download Custom...
BIP39 - Mnemonic...
Text structures - Pre...
awsacademy.com/vf...
GitHub
Other favorites
Nagios
General
Home
Documentation
Current Status
Tactical Overview
Map (Legacy)
Hosts
Services
Host Groups
Summary
Grid
Service Groups
Summary
Grid
Problems
Services (Unhandled)
Hosts (Unhandled)
Network Outages
Quick Search:
Reports
Availability
Trends (Legacy)
Alerts
History
Summary
Histogram (Legacy)
Notifications
Event Log
System
Comments
Downtime
Process Info
Performance Info
Scheduling Queue
Configuration
Nagios Core
Daemon running with PID 103799
Nagios® Core™
Version 4.4.6
April 28, 2020
Check for updates
A new version of Nagios Core is available!
Visit nagios.org to download Nagios 4.5.5.
Get Started
Start monitoring your infrastructure
Change the look and feel of Nagios
Extend Nagios with hundreds of addons
Get support
Get training
Get certified
Quick Links
Nagios Library (tutorials and docs)
Nagios Labs (development blog)
Nagios Exchange (plugins and addons)
Nagios Support (tech support)
Nagios.com (company)
Nagios.org (project)
Latest News
Don't Miss...
Page Tour

ADVANCE DEVOPS - 10

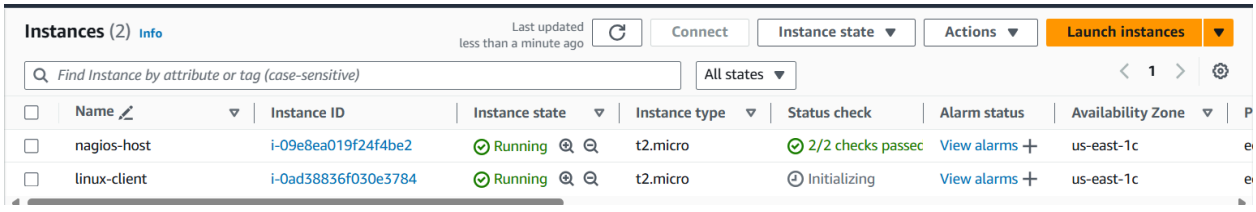
NAME - PRANAV TITAMBE 62/D15A

Check if the nagios service is running by executing following command

```
sudo systemctl status nagios
```

```
ubuntu@ip-172-31-89-161:~$ sudo systemctl status nagios
● nagios.service - Nagios Core 4.4.6
     Loaded: loaded (/usr/lib/systemd/system/nagios.service; enabled; preset: enabled)
     Active: active (running) since Sat 2024-09-28 16:08:58 UTC; 1min 2s ago
       Docs: https://www.nagios.org/documentation
    Process: 15743 ExecStartPre=/usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg (code=exited, status=0/SUCCESS)
    Process: 15753 ExecStart=/usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg (code=exited, status=0/SUCCESS)
   Main PID: 15764 (nagios)
      Tasks: 6 (limit: 1130)
     Memory: 2.4M (peak: 3.2M)
        CPU: 29ms
     CGroup: /system.slice/nagios.service
             ├─15764 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg
             ├─15765 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─15766 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─15767 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─15768 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             └─15769 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg

Sep 28 16:08:58 ip-172-31-89-161 nagios[15764]: qh: Socket '/usr/local/nagios/var/rw/nagios.qh' successfully initialized
Sep 28 16:08:58 ip-172-31-89-161 nagios[15764]: qh: core query handler registered
Sep 28 16:08:58 ip-172-31-89-161 nagios[15764]: qh: echo service query handler registered
Sep 28 16:08:58 ip-172-31-89-161 nagios[15764]: qh: help for the query handler registered
Sep 28 16:08:58 ip-172-31-89-161 nagios[15764]: wproc: Successfully registered manager as @wproc with query handler
Sep 28 16:08:58 ip-172-31-89-161 nagios[15764]: wproc: Registry request: name=Core Worker 15765;pid=15765
Sep 28 16:08:58 ip-172-31-89-161 nagios[15764]: wproc: Registry request: name=Core Worker 15766;pid=15766
Sep 28 16:08:58 ip-172-31-89-161 nagios[15764]: wproc: Registry request: name=Core Worker 15767;pid=15767
```

Now, create a new EC2 instance on AWS



Now perform the following commands on nagios-host EC2 instance.

On the server, run this command

```
ps -ef | grep nagios
```

```
ubuntu@ip-172-31-89-161:~$ ps -ef | grep nagios
nagios     15764       1  0 16:08 ?        00:00:00 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg
nagios     15765   15764  0 16:08 ?        00:00:00 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
nagios     15766   15764  0 16:08 ?        00:00:00 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
nagios     15767   15764  0 16:08 ?        00:00:00 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
nagios     15768   15764  0 16:08 ?        00:00:00 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
nagios     15769   15764  0 16:08 ?        00:00:00 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg
ubuntu     15957    1342  0 16:13 pts/0    00:00:00 grep --color=auto nagios
ubuntu@ip-172-31-89-161:~$
```

Become a root user and create 2 folders

```
sudo su
mkdir /usr/local/nagios/etc/objects/monitorhosts
mkdir /usr/local/nagios/etc/objects/monitorhosts/linuxhosts
```

```
ubuntu@ip-172-31-89-161:~$ sudo su
mkdir /usr/local/nagios/etc/objects/monitorhosts
mkdir /usr/local/nagios/etc/objects/monitorhosts/linuxhosts
root@ip-172-31-89-161:/home/ubuntu#
```

Copy localhost.cfg file to the mentioned location

`cp /usr/local/nagios/etc/objects/localhost.cfg /usr/local/nagios/etc/objects/monitorhosts/linuxhosts`

```
root@ip-172-31-89-161:/usr/local/nagios/etc/objects# cp /usr/local/nagios/etc/objects/localhost.cfg /usr/local/nagios/etc/objects/monitorhosts/linuxhosts
cp: cannot create regular file '/usr/local/nagios/etc/objects/monitorhosts/linuxhosts': No such file or directory
root@ip-172-31-89-161:/usr/local/nagios/etc/objects# sudo mkdir -p /usr/local/nagios/etc/objects/monitorhosts/linuxhosts
root@ip-172-31-89-161:/usr/local/nagios/etc/objects# cp /usr/local/nagios/etc/objects/localhost.cfg /usr/local/nagios/etc/objects/monitorhosts/linuxhosts
root@ip-172-31-89-161:/usr/local/nagios/etc/objects#
```

Open the nano editor for localhost.cfg file and make these changes. Add the Ip address of the linux-client for the address field.

`nano /usr/local/nagios/etc/objects/monitorhosts/linuxhosts/localhost.cfg`

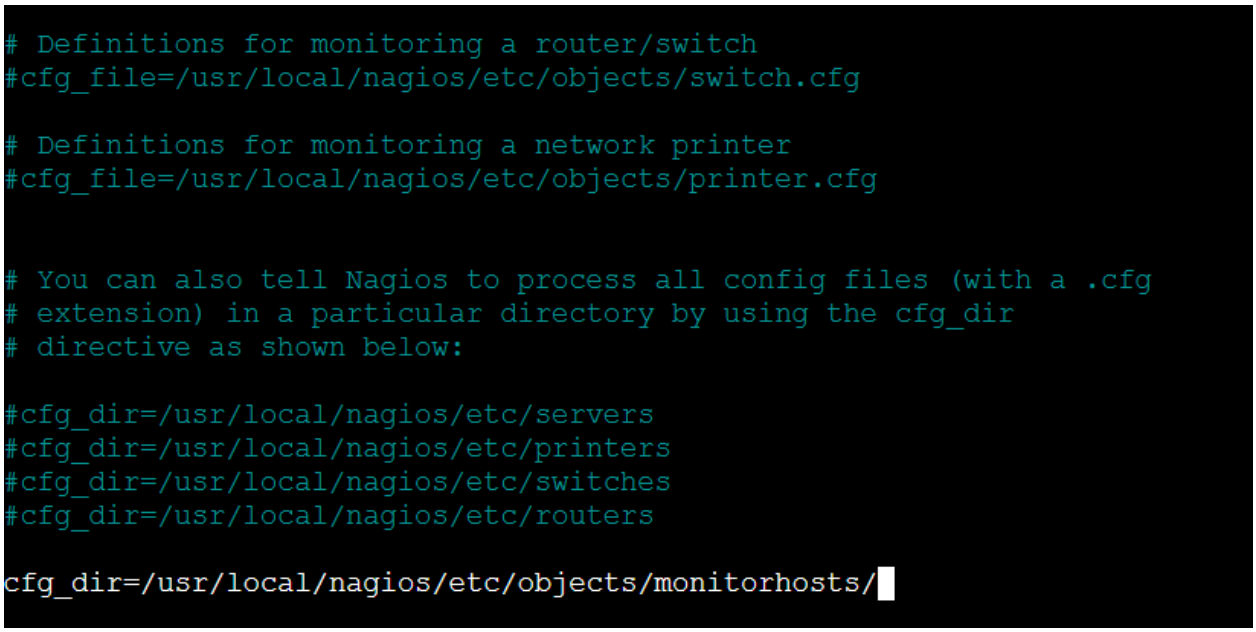
```
  GNU nano 7.2                                        /usr/local/nagios/et
##############################################################################
#
# HOST DEFINITION
#
##############################################################################

# Define a host for the local machine

define host {

    use                     linux-server            ; Name of host te
                                                    ; This host defin
                                                    ; in (or inherite
    host_name               linuxserver
    alias                   linuxserver
    address                 52.207.253.18
}



##############################################################################
#
# HOST GROUP DEFINITION

^G Help          ^O Write Out     ^W Where Is      ^K Cut           ^T Ex
^X Exit          ^R Read File     ^\ Replace       ^U Paste         ^J Ju
```

Note - Here replace hostname with linuxserver

```
nano /usr/local/nagios/etc/nagios.cfg
```

Add the following line to the nagios.cfg file

```
cfg_dir=/usr/local/nagios/etc/objects/monitorhosts/
```



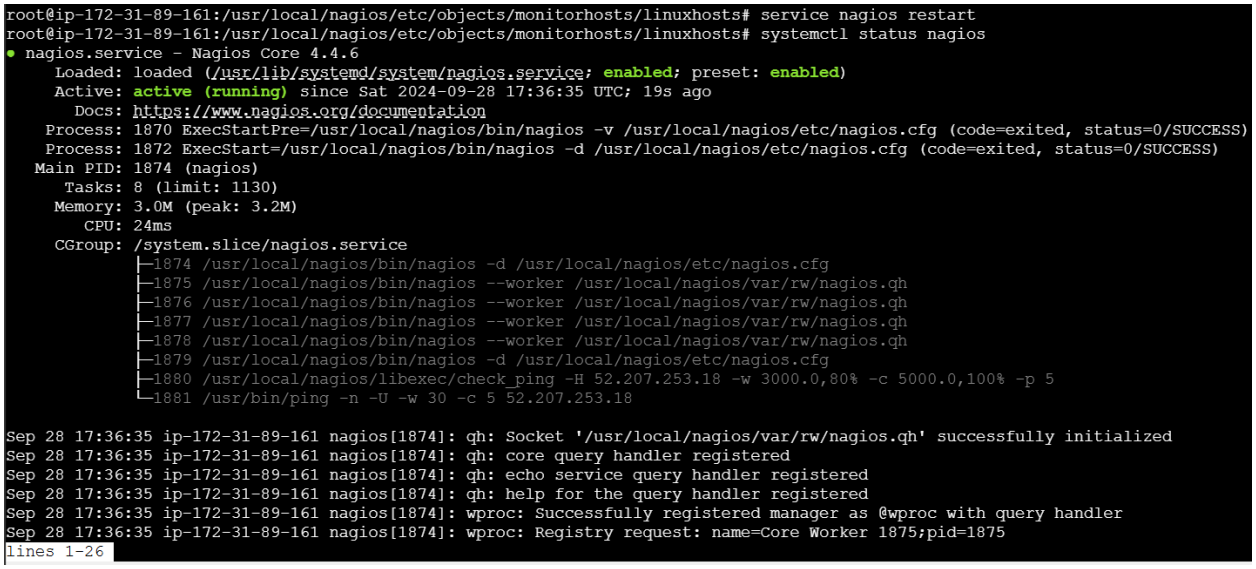
After making the changes in nagios.cfg file now check validate the file by typing the following command in the terminal.

```
/usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg
```

```
License: GPL

Website: https://www.nagios.org
Reading configuration data...
   Read main config file okay...
   Read object config files okay...

Running pre-flight check on configuration data...

Checking objects...
        Checked 16 services.
        Checked 2 hosts.
        Checked 2 host groups.
        Checked 0 service groups.
        Checked 1 contacts.
        Checked 1 contact groups.
        Checked 24 commands.
        Checked 5 time periods.
        Checked 0 host escalations.
        Checked 0 service escalations.
Checking for circular paths...
        Checked 2 hosts
        Checked 0 service dependencies
        Checked 0 host dependencies
        Checked 5 timeperiods
Checking global event handlers...
Checking obsessive compulsive processor commands...
Checking misc settings...

Total Warnings: 0
Total Errors:   0

Things look okay - No serious problems were detected during the pre-flight check
root@ip-172-31-89-161:/usr/local/nagios/etc/objects/monitorhosts/linuxhosts#
```

Now restart the service by using this command

```
service nagios restart
```



Now using this command update the apt repository of ubuntu (linux-client), install gcc, nagios-nrpe-server and nagios-plugin

```
sudo apt update -y
sudo apt install gcc -y
sudo apt install -y nagios-nrpe-server nagios-plugins
```

Now open nrpe.cfg file and add the ip address of the nagios host as shown. To open the nrpe.cfg file copy this command.

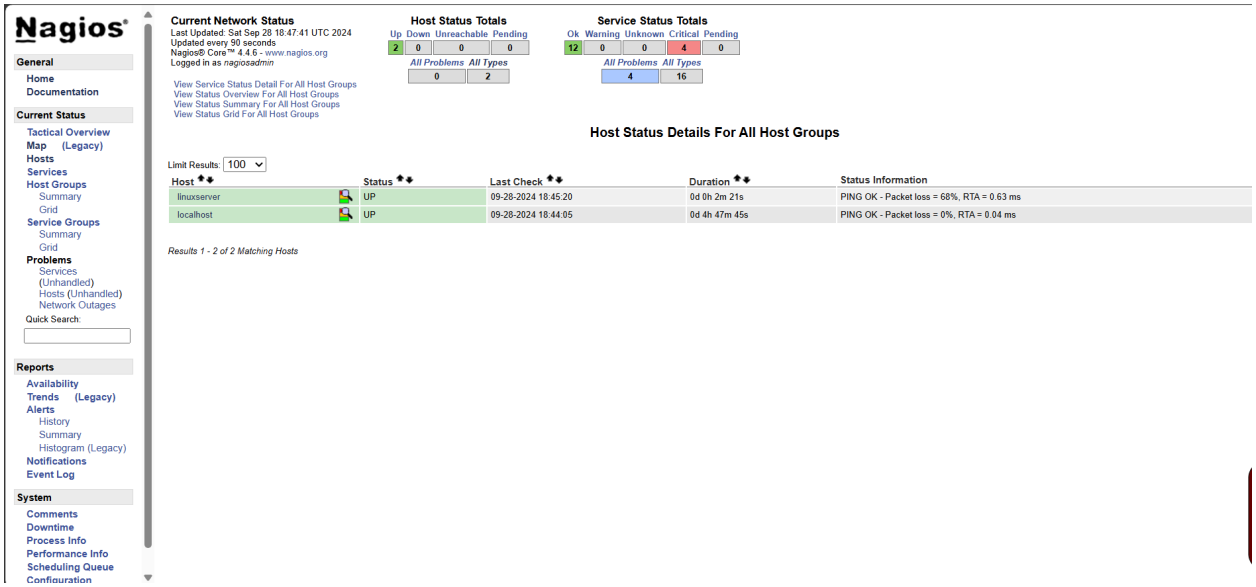
```
sudo nano /etc/nagios/nrpe.cfg
```

```
# Supported.
#
# Note: The daemon only does rudimentary checkin
# address.  I would highly recommend adding entr
# file to allow only the specified host to conne
# you are running this daemon on.
#
# NOTE: This option is ignored if NRPE is runnin

allowed_hosts=127.0.0.1,54.167.169.0


# COMMAND ARGUMENT PROCESSING
# This option determines whether or not the NRPE
# to specify arguments to commands that are exec
# if the daemon was configured with the --enable-
# option.
```

Now restart nrpe server by using this command

```
sudo systemctl restart nagios-nrpe-server
```

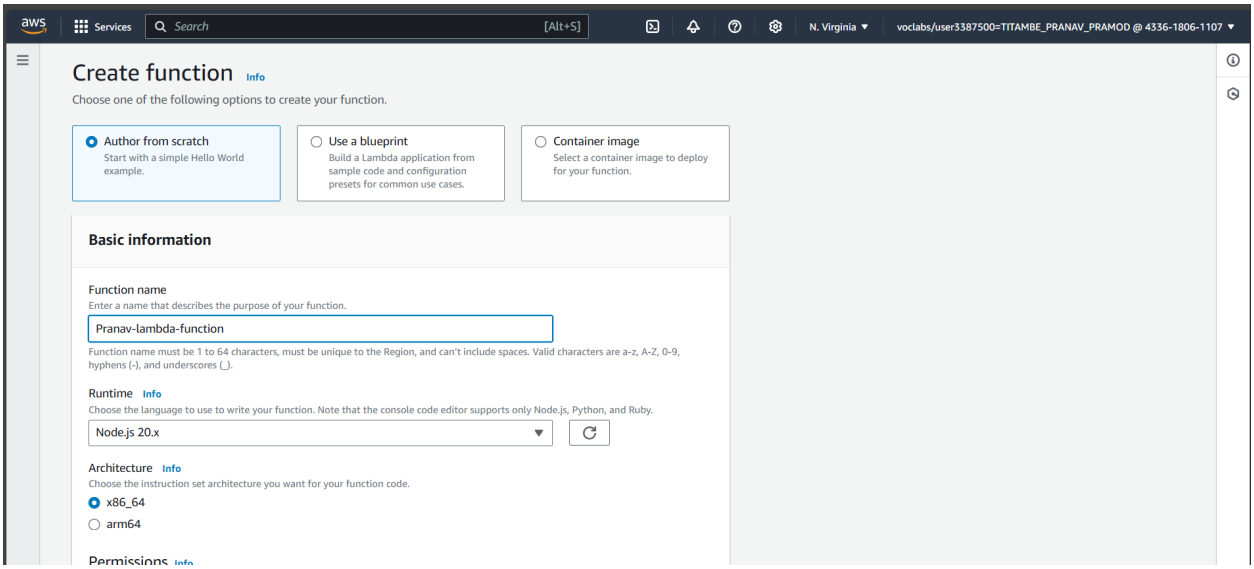
Now, check nagios dashboard, you should see linuxserver up and running, if not check security groups of the EC2 instances.



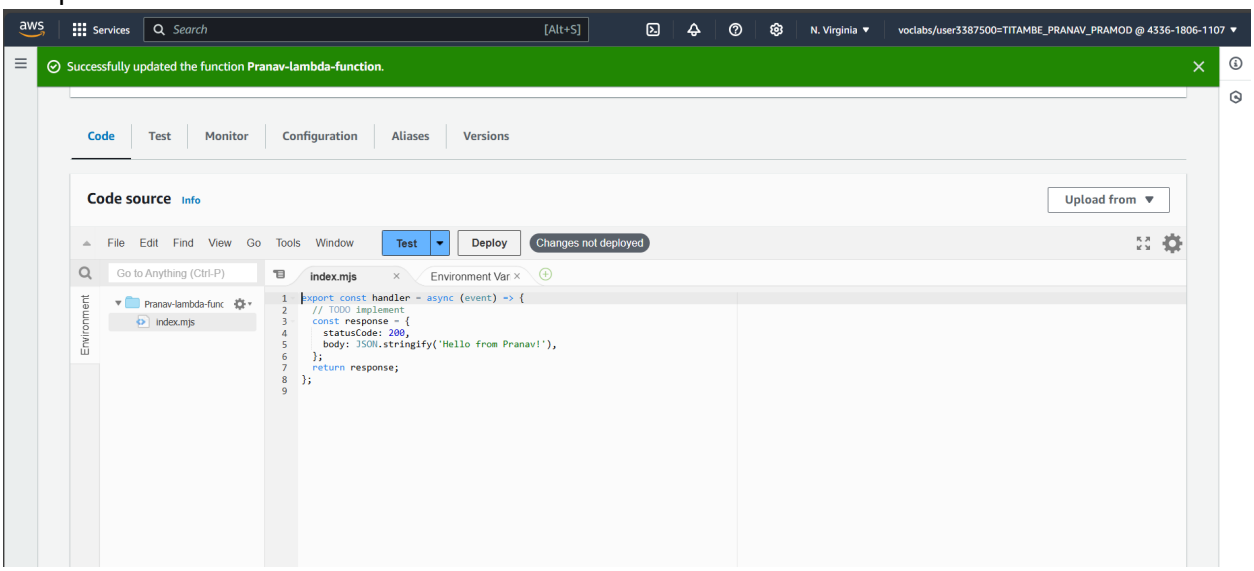
ADVANCE DEVOPS EXP-11

NAME-PRANAV TITAMBE D15A/62

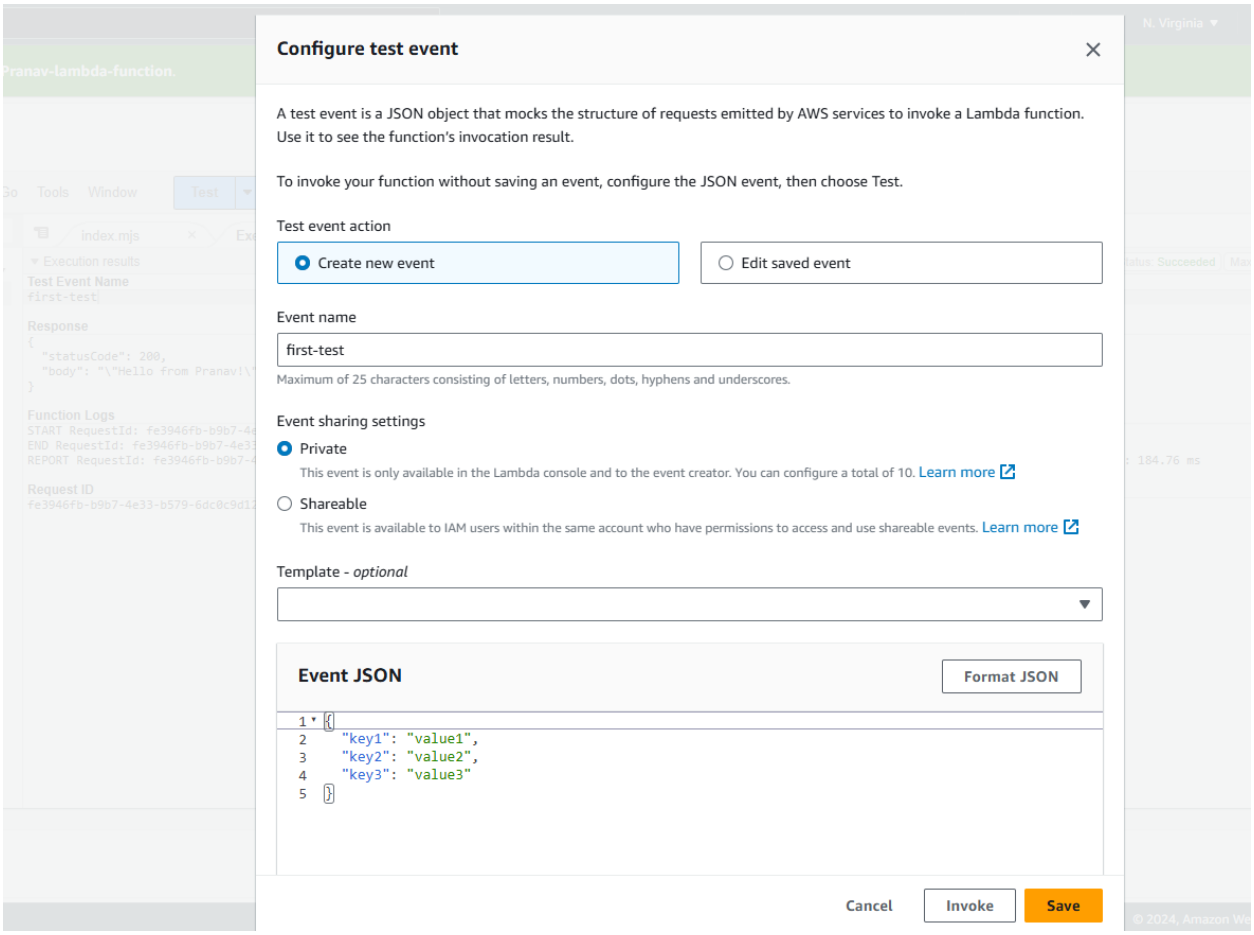
Step 1: Create a lambda function on AWS.



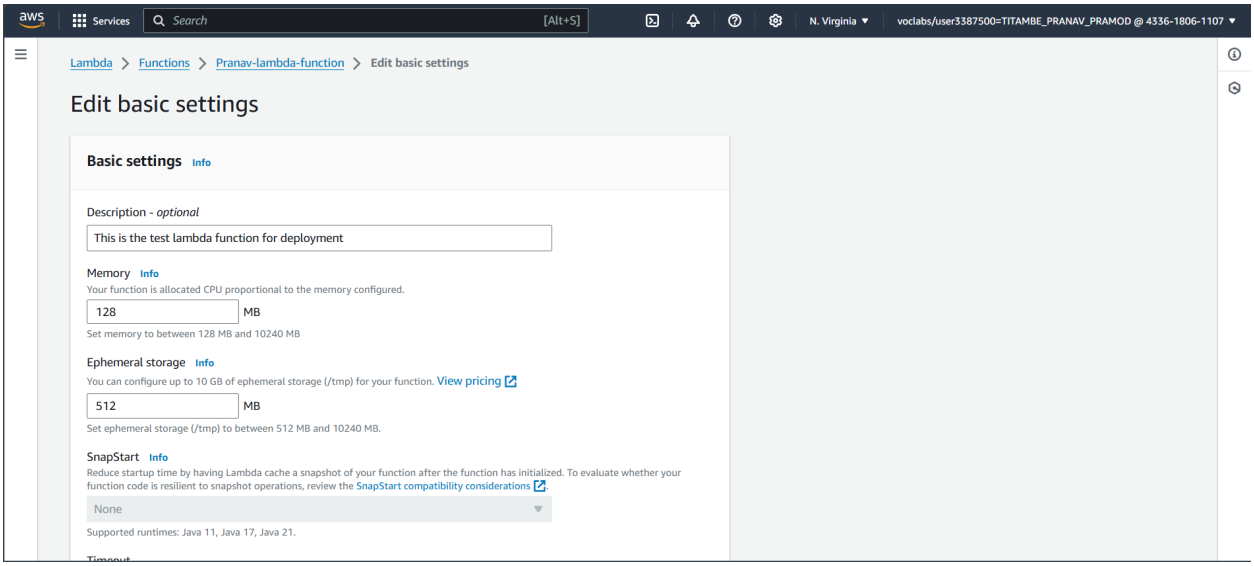
Step 2: There are blueprints mentioned while creating a lambda function choose hello world blueprint.

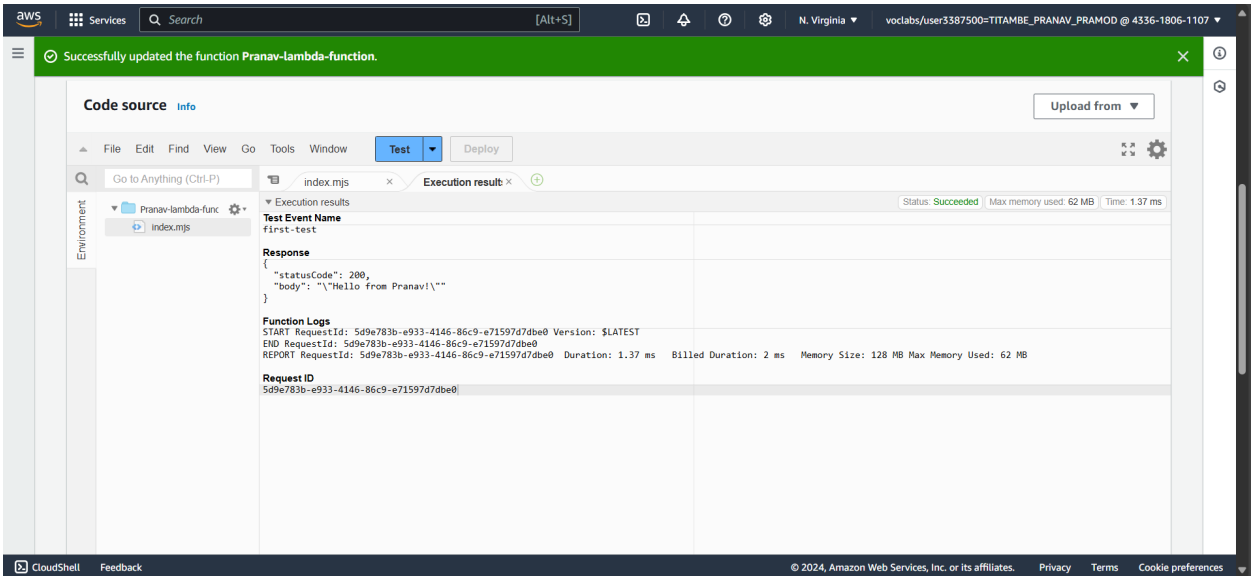


Step 3: Then deploy the function and click on test to test the function. You need to create the test function, ensure the event is in proper json format.



Step 4: Once the test event is created then click on test and you will get the results.

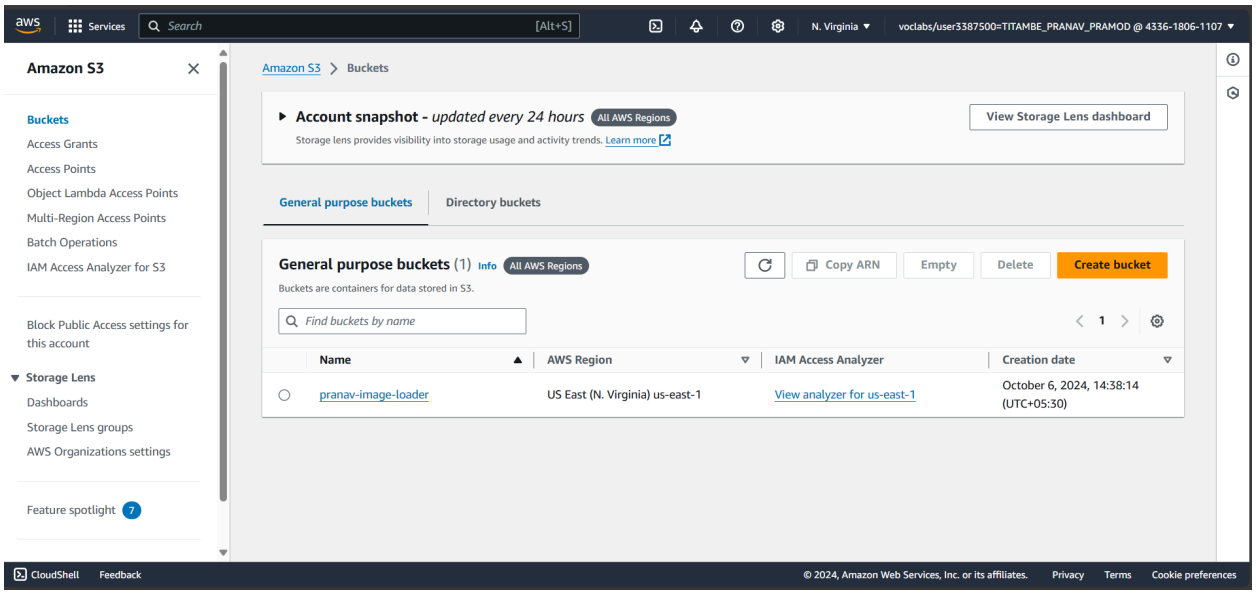



Services
Search
[Alt+S]
N. Virginia
voclabs/user3387500=TITAMBE_PRANAV_PRAMOD @ 4336-1806-1107
Successfully updated the function Pranav-lambda-function.
Code source Info
Upload from
File Edit Find View Go Tools Window
Test
Deploy
Go to Anything (Ctrl-P)
index.mjs
Execution result
Environment
Pranav-lambda-func
index.mjs
Execution results
Status: Succeeded
Max memory used: 62 MB
Time: 1.37 ms
Test Event Name
first-test
Response
{
"statusCode": 200,
"body": "\"Hello from Pranav!\""
}
Function Logs
START RequestId: 5d9e783b-e933-4146-86c9-e71597d7dbe0 Version: $LATEST
END RequestId: 5d9e783b-e933-4146-86c9-e71597d7dbe0
REPORT RequestId: 5d9e783b-e933-4146-86c9-e71597d7dbe0 Duration: 1.37 ms Billed Duration: 2 ms Memory Size: 128 MB Max Memory Used: 62 MB
Request ID
5d9e783b-e933-4146-86c9-e71597d7dbe0
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

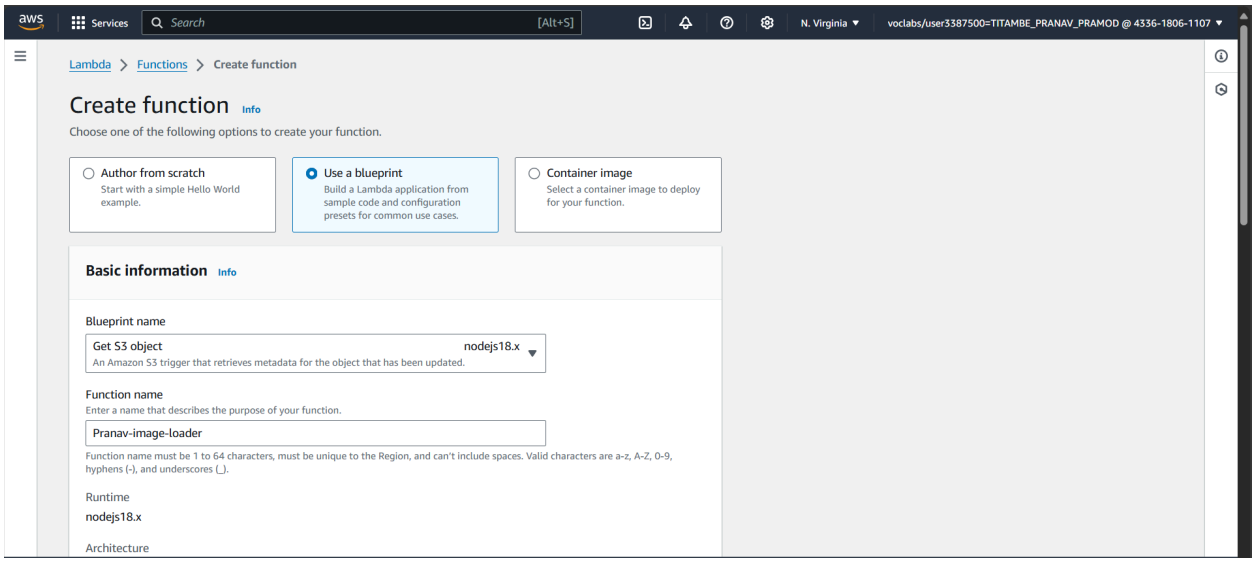
ADVANCE DEVOPS EXP-12

NAME - PRANAV TITAMBE D15A/62

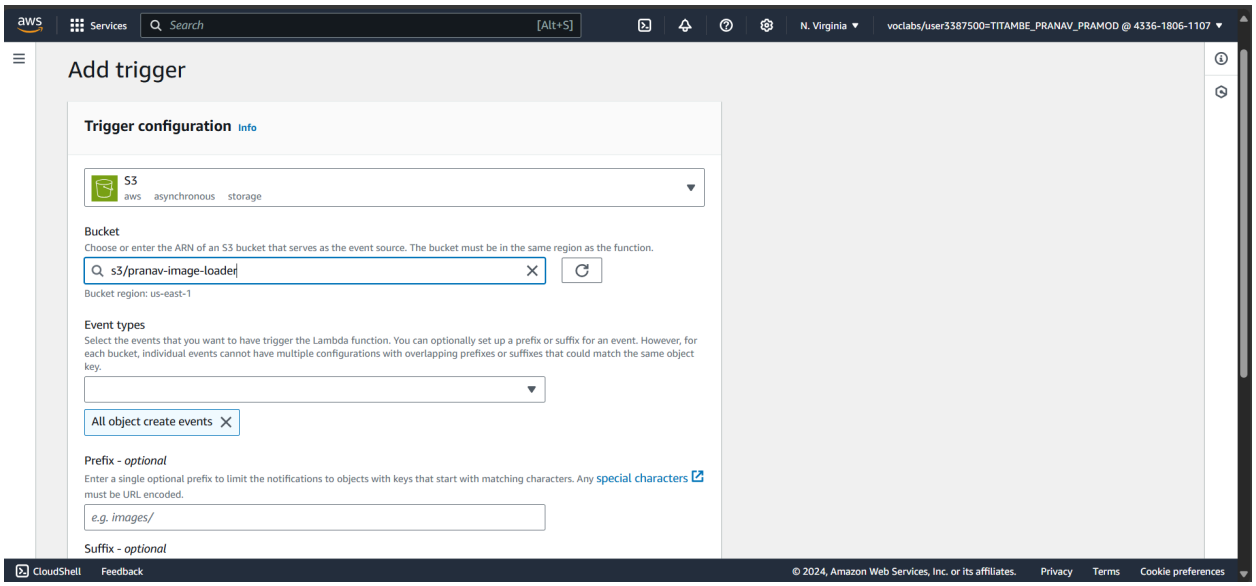
Step 1: Create a Lambda function on AWS.

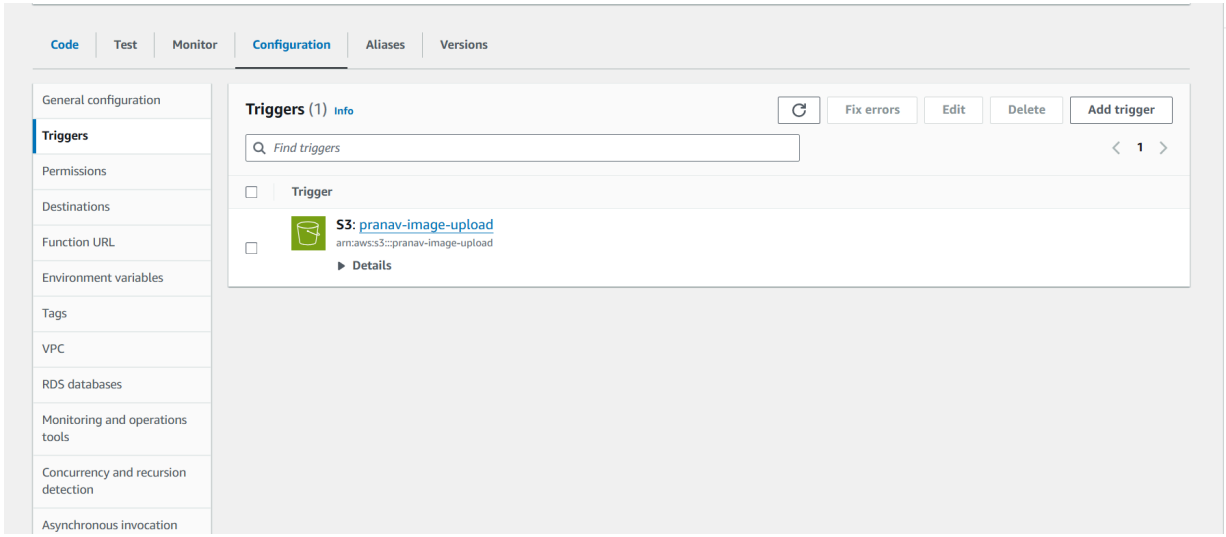


Here choose in which language you want the function to be, and select ‘Use a Blueprint’ in that choose ‘Get Objects from S3’.



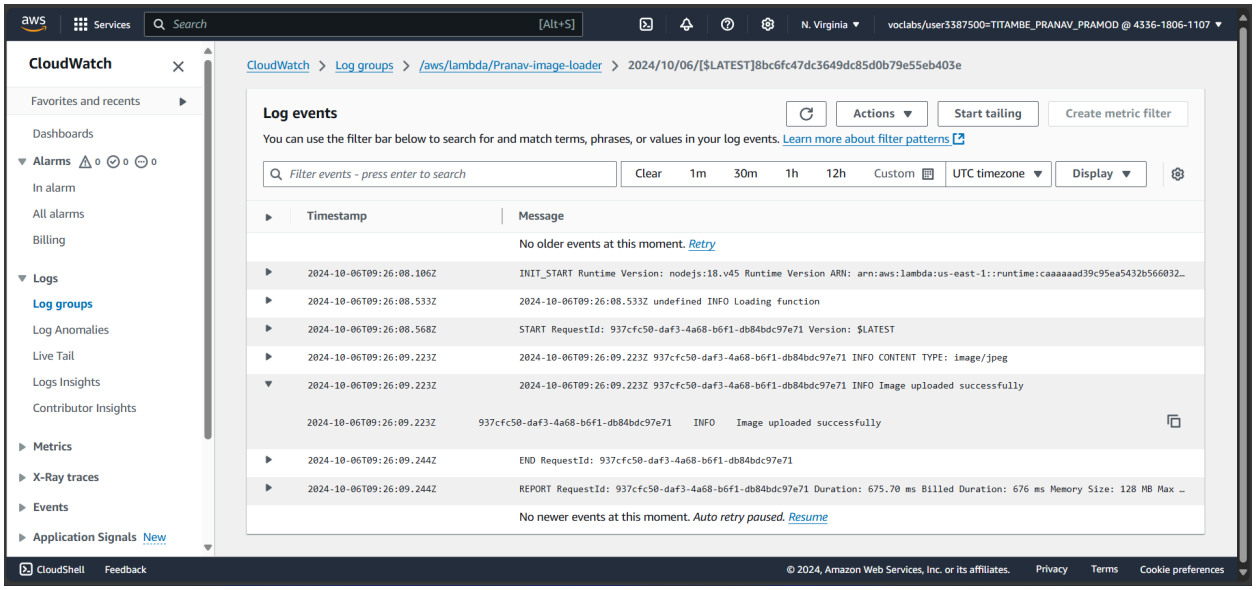
Step 2: After selecting S3 bucket now create a trigger.





Step 3: After creating the trigger you will see the the trigger applied to your Lambda function. Now add image in S3 bucket.

Step 4: After adding image to the bucket go to AWS CloudWatch then go to S3 events log, click on the latest even there you will see the logs of the image which is being added by you on S3 bucket.


aws
Services
Search
[Alt+S]
N. Virginia
voclabs/user3387500=TITAMBE_PRANAV_PRAMOD @ 4336-1806-1107
CloudWatch
Favorites and recents
Dashboards
Alarms 0 0 0
In alarm
All alarms
Billing
Logs
Log groups
Log Anomalies
Live Tail
Logs Insights
Contributor Insights
Metrics
X-Ray traces
Events
Application Signals New
CloudWatch > Log groups > /aws/lambda/Pranav-image-loader > 2024/10/06/[$LATEST]8bc6fc47dc3649dc85d0b79e55eb403e
Log events
Actions
Start tailing
Create metric filter
You can use the filter bar below to search for and match terms, phrases, or values in your log events. Learn more about filter patterns
Filter events - press enter to search
Clear
1m
30m
1h
12h
Custom
UTC timezone
Display
Timestamp
Message
No older events at this moment. Retry
2024-10-06T09:26:08.106Z INIT_START Runtime Version: nodejs:18.v45 Runtime Version ARN: arn:aws:lambda:us-east-1::runtime:caaaaaad39c95ea5432b566032…
2024-10-06T09:26:08.533Z 2024-10-06T09:26:08.533Z undefined INFO Loading function
2024-10-06T09:26:08.568Z START RequestId: 937cfc50-daf3-4a68-b6f1-db84bdc97e71 Version: $LATEST
2024-10-06T09:26:09.223Z 2024-10-06T09:26:09.223Z 937cfc50-daf3-4a68-b6f1-db84bdc97e71 INFO CONTENT TYPE: image/jpeg
2024-10-06T09:26:09.223Z 2024-10-06T09:26:09.223Z 937cfc50-daf3-4a68-b6f1-db84bdc97e71 INFO Image uploaded successfully
2024-10-06T09:26:09.223Z 937cfc50-daf3-4a68-b6f1-db84bdc97e71 INFO Image uploaded successfully
2024-10-06T09:26:09.244Z END RequestId: 937cfc50-daf3-4a68-b6f1-db84bdc97e71
2024-10-06T09:26:09.244Z REPORT RequestId: 937cfc50-daf3-4a68-b6f1-db84bdc97e71 Duration: 675.70 ms Billed Duration: 676 ms Memory Size: 128 MB Max …
No newer events at this moment. Auto retry paused. Resume
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences