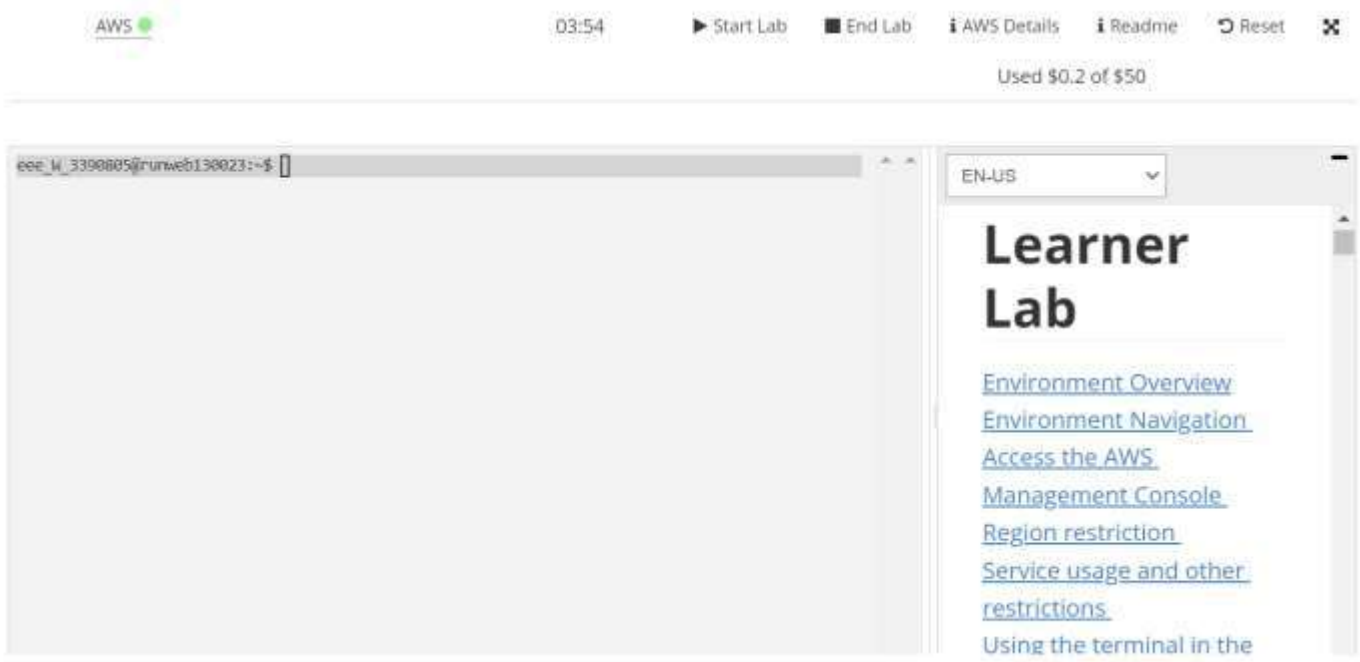
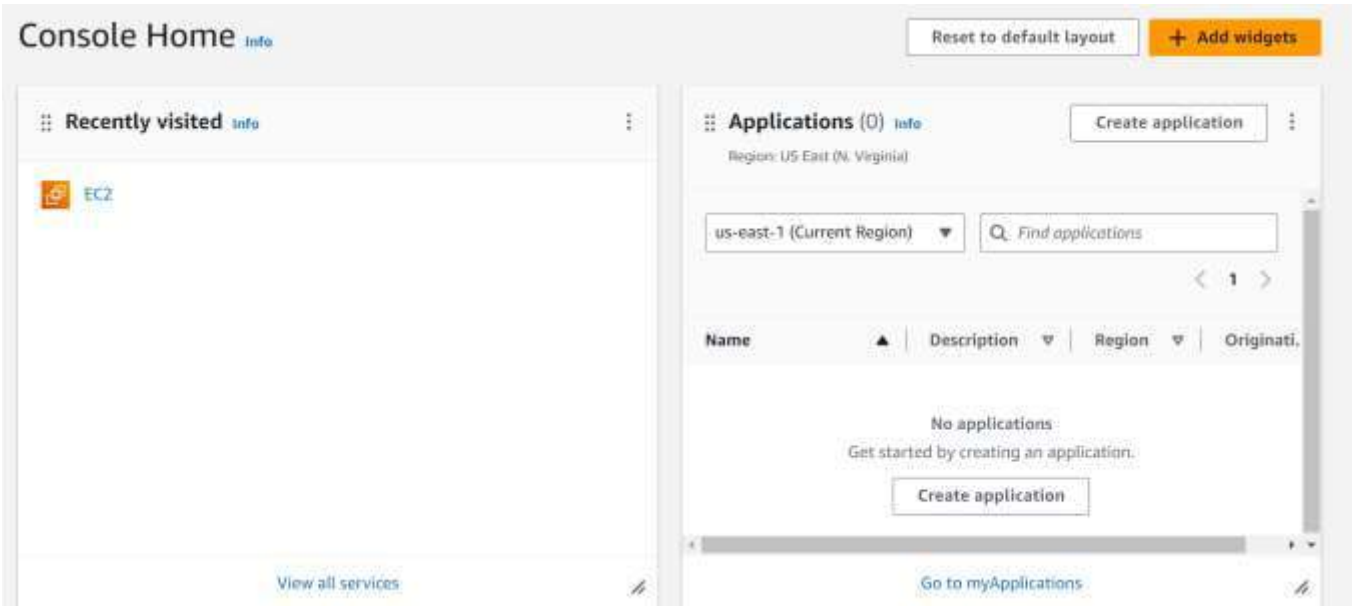
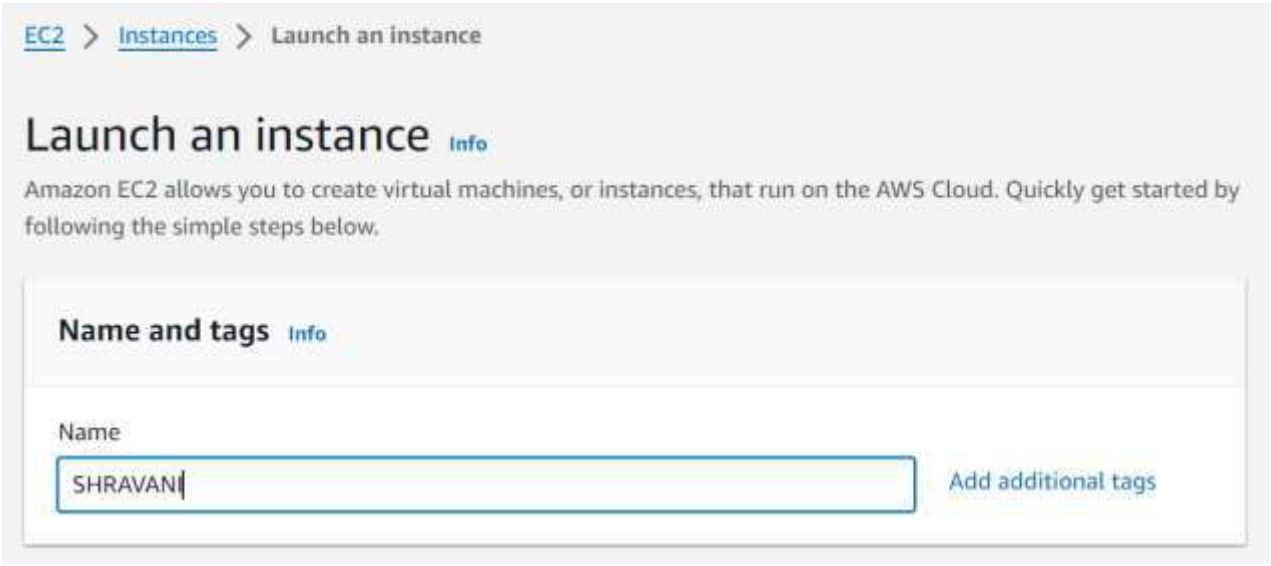
ADVANCE DEVOPS EXPERIMENT 1

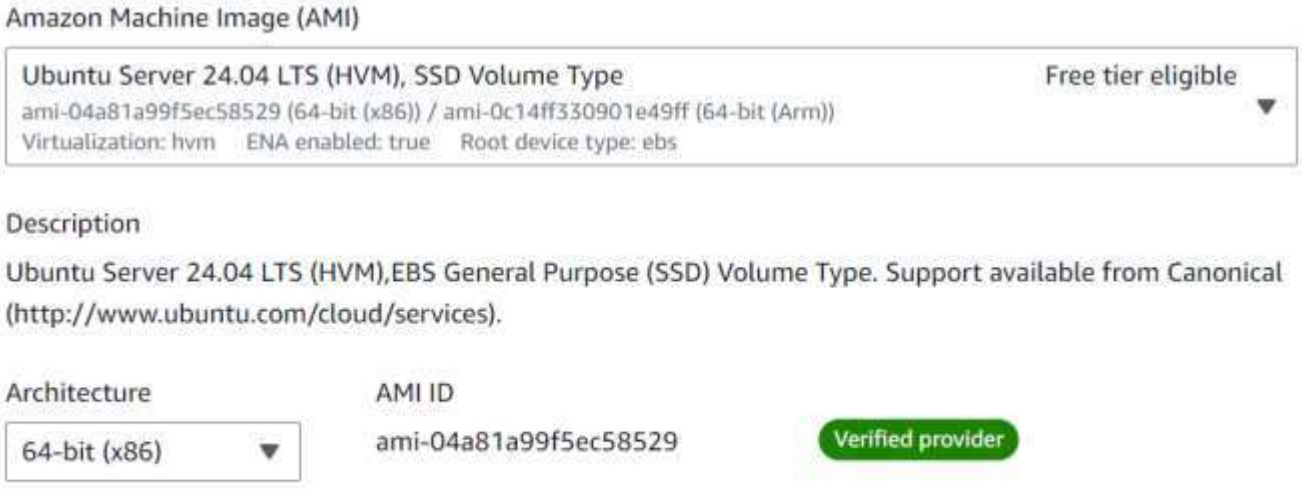
EC2 INSTANCE CREATION

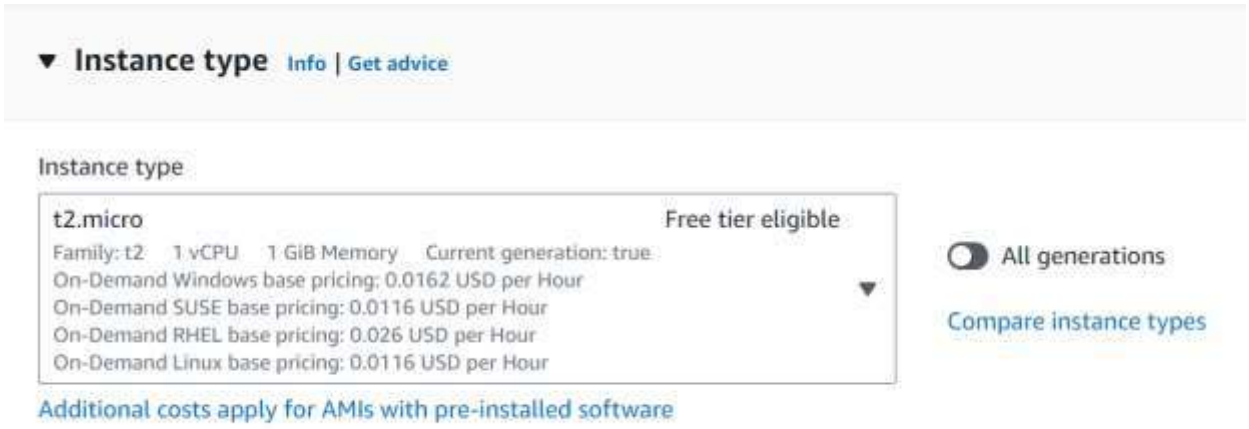


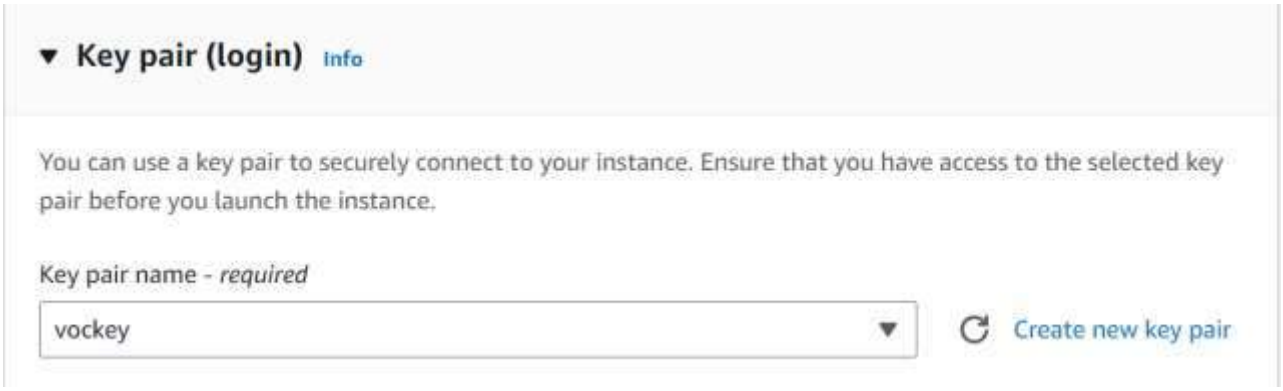


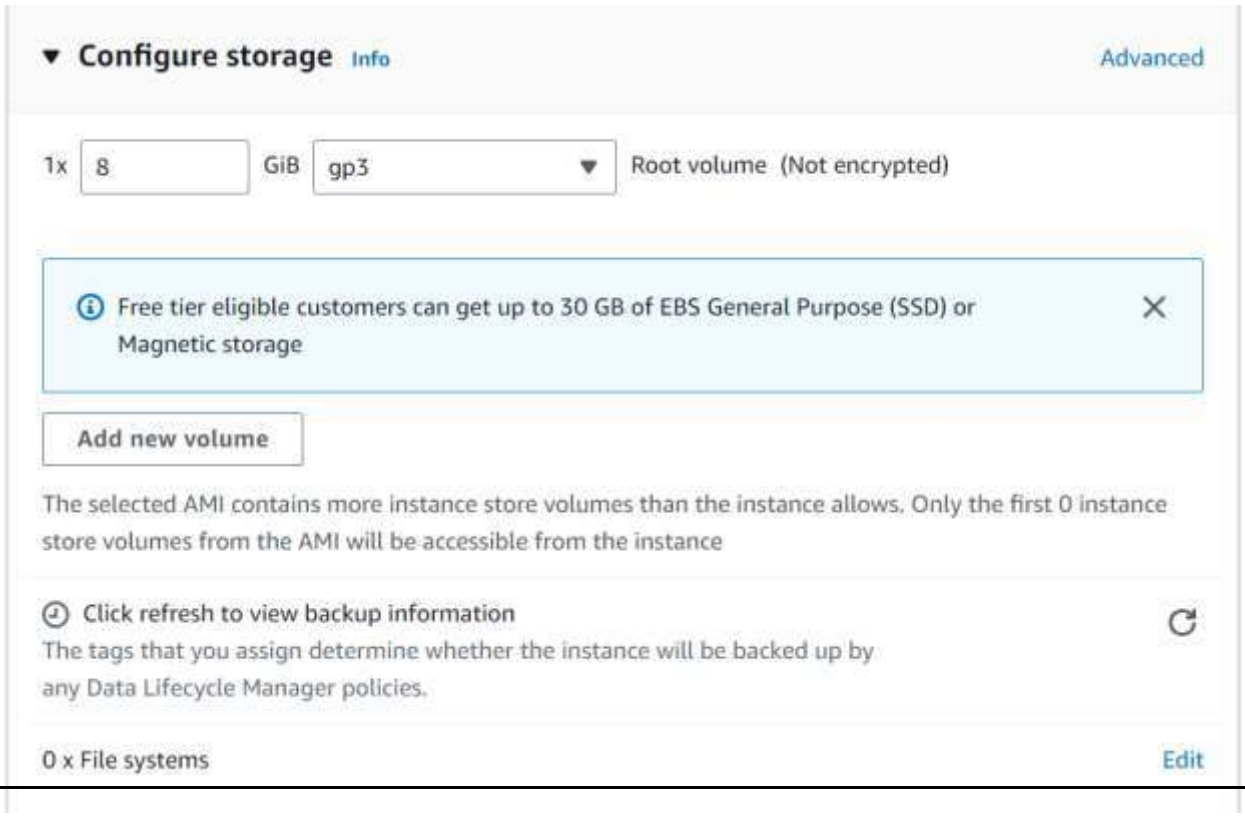

EC2 > Instances > Launch an instance
Launch an instance Info
Amazon EC2 allows you to create virtual machines, or instances, that run on the AWS Cloud. Quickly get started by following the simple steps below.
Name and tags Info
Name
SHRAVANI
Add additional tags

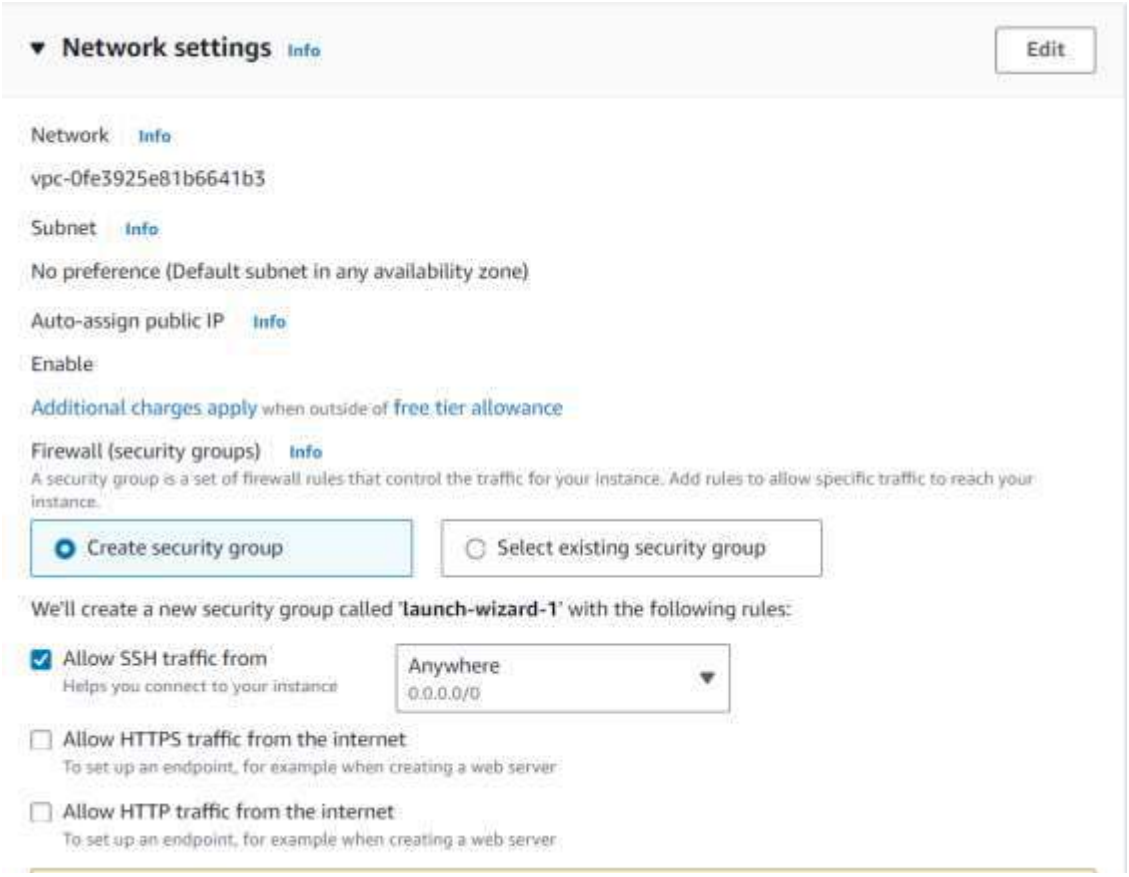

Quick Start
Amazon Linux
aws
macOS
Mac
Ubuntu
ubuntu
Windows
Microsoft
Red Hat
Red Hat
Browse more AMIs
Including AMIs from AWS, Marketplace and the Community

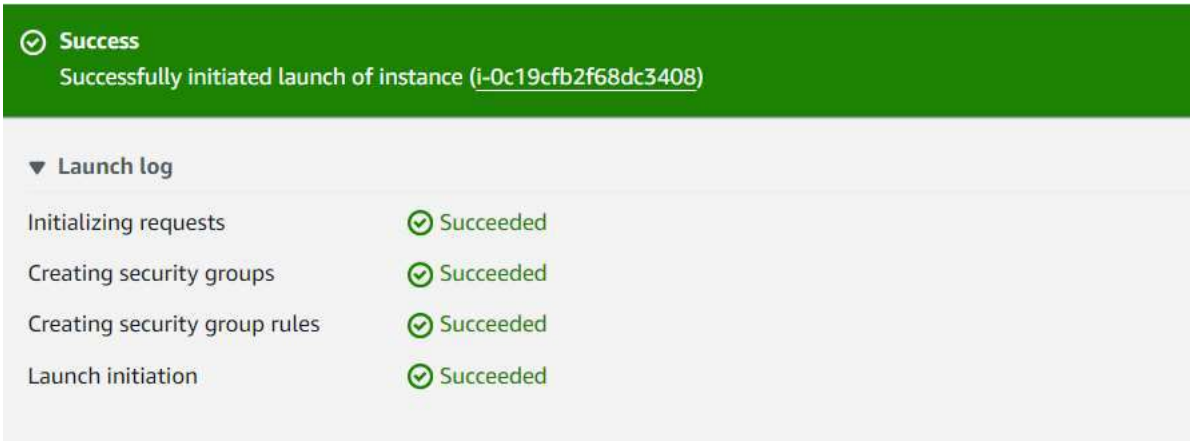

Amazon Machine Image (AMI)
Ubuntu Server 24.04 LTS (HVM), SSD Volume Type
Free tier eligible
ami-04a81a99f5ec58529 (64-bit (x86)) / ami-0c14ff330901e49ff (64-bit (Arm))
Virtualization: hvm ENA enabled: true Root device type: ebs
Description
Ubuntu Server 24.04 LTS (HVM),EBS General Purpose (SSD) Volume Type. Support available from Canonical (http://www.ubuntu.com/cloud/services).
Architecture
64-bit (x86)
AMI ID
ami-04a81a99f5ec58529
Verified provider

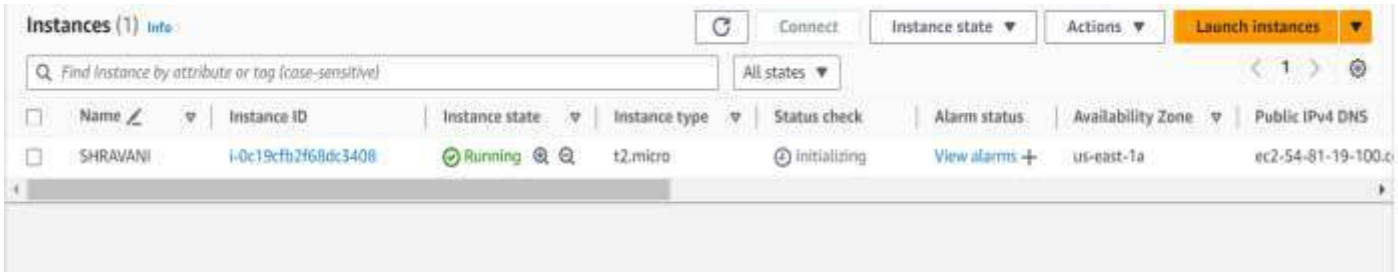

Instance type Info | Get advice
Instance type
t2.micro
Free tier eligible
Family: t2 1 vCPU 1 GiB Memory Current generation: true
On-Demand Windows base pricing: 0.0162 USD per Hour
On-Demand SUSE base pricing: 0.0116 USD per Hour
On-Demand RHEL base pricing: 0.026 USD per Hour
On-Demand Linux base pricing: 0.0116 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
vockey
Create new key pair


Configure storage Info
Advanced
1x 8 GiB gp3 Root volume (Not encrypted)
Free tier eligible customers can get up to 30 GB of EBS General Purpose (SSD) or Magnetic storage
Add new volume
The selected AMI contains more instance store volumes than the instance allows. Only the first 0 instance store volumes from the AMI will be accessible from the instance
Click refresh to view backup information
The tags that you assign determine whether the instance will be backed up by any Data Lifecycle Manager policies.
0 x File systems
Edit


Network settings Info
Edit
Network Info
vpc-0fe3925e81b6641b3
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-1' 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

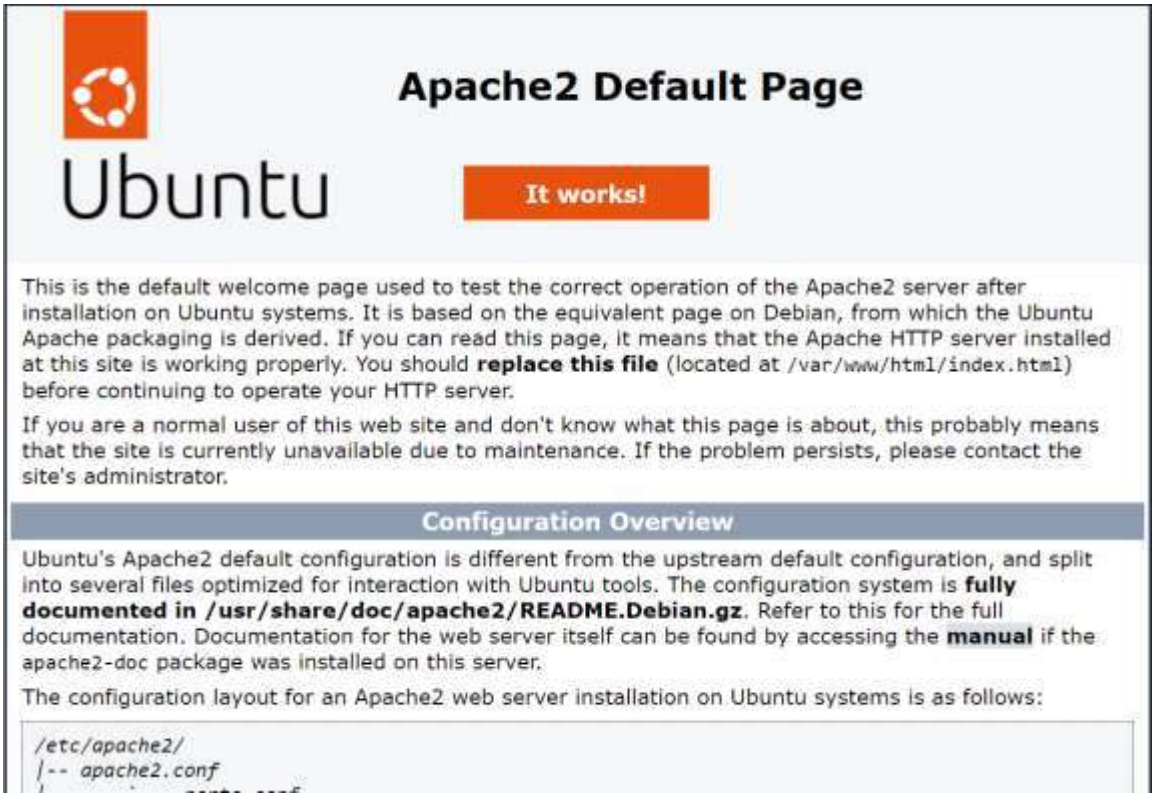

Success
Successfully initiated launch of instance (i-0c19cfb2f68dc3408)
Launch log
Initializing requests
Succeeded
Creating security groups
Succeeded
Creating security group rules
Succeeded
Launch initiation
Succeeded


Instances (1) Info
Connect
Instance state
Actions
Launch instances
Find instance by attribute or tag (case-sensitive)
All states
Name
Instance ID
Instance state
Instance type
Status check
Alarm status
Availability Zone
Public IPv4 DNS
SHRAVANI
i-0c19cfb2f68dc3408
Running
t2.micro
Initializing
View alarms
us-east-1a

Static website hosting using EC2

```
root@ip-172-31-38-150:/home/ubuntu# apt install apache2
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 libapr1t64 libaprutil1-dbd-sqlite3 l
Suggested packages:
  apache2-doc apache2-suexec-pristine | apache2-suexec-custom www-browser
The following NEW packages will be installed:
  apache2 apache2-bin apache2-data apache2-utils libapr1t64 libaprutil1-dbd-s
0 upgraded, 10 newly installed, 0 to remove and 0 not upgraded.
Need to get 1677 kB/2080 kB of archives.
After this operation, 8091 kB of additional disk space will be used.
```

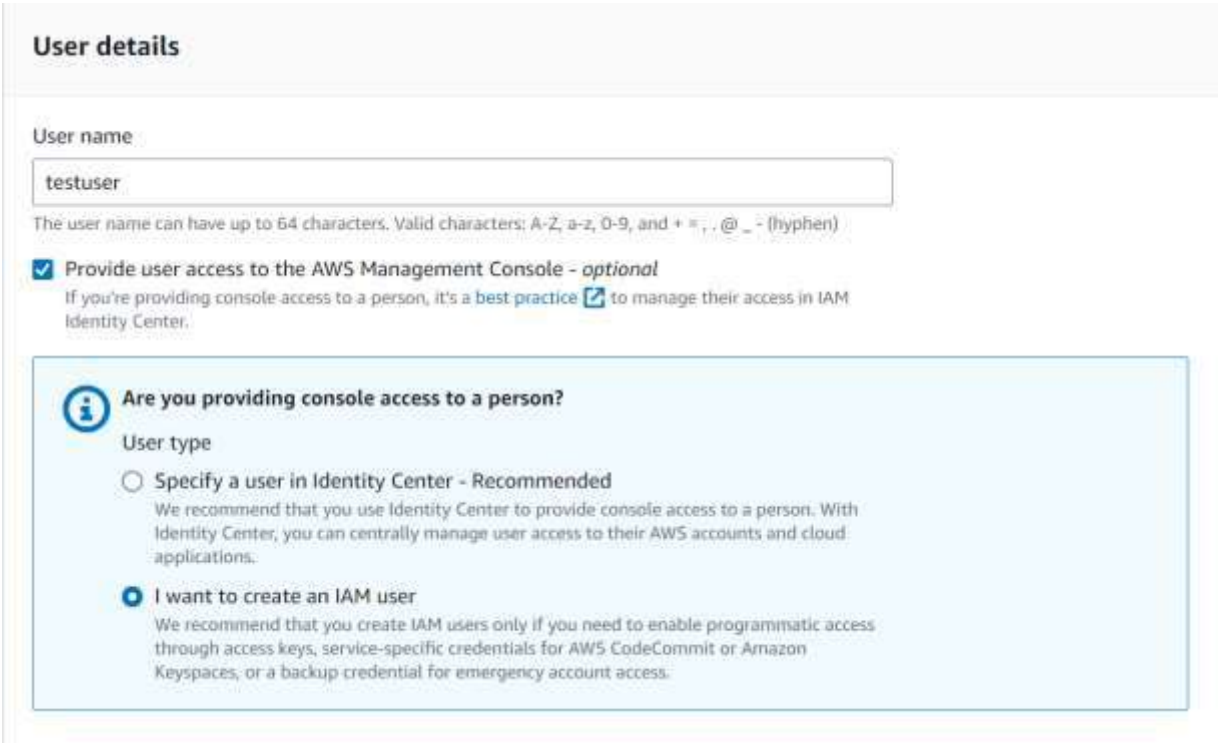
```
Jul 30 09:05:25 ip-172-31-38-150 systemd[1]: Starting apache2.service - The Apache HTTP Server...
Jul 30 09:05:25 ip-172-31-38-150 systemd[1]: Started apache2.service - The Apache HTTP Server.
root@ip-172-31-38-150:/home/ubuntu# cd /var/www/html/
root@ip-172-31-38-150:/var/www/html# nano index.html
root@ip-172-31-38-150:/var/www/html# nano index1.html
root@ip-172-31-38-150:/var/www/html# cat index1.html
<h1>Hi Shravani here...</h1>
root@ip-172-31-38-150:/var/www/html#
```

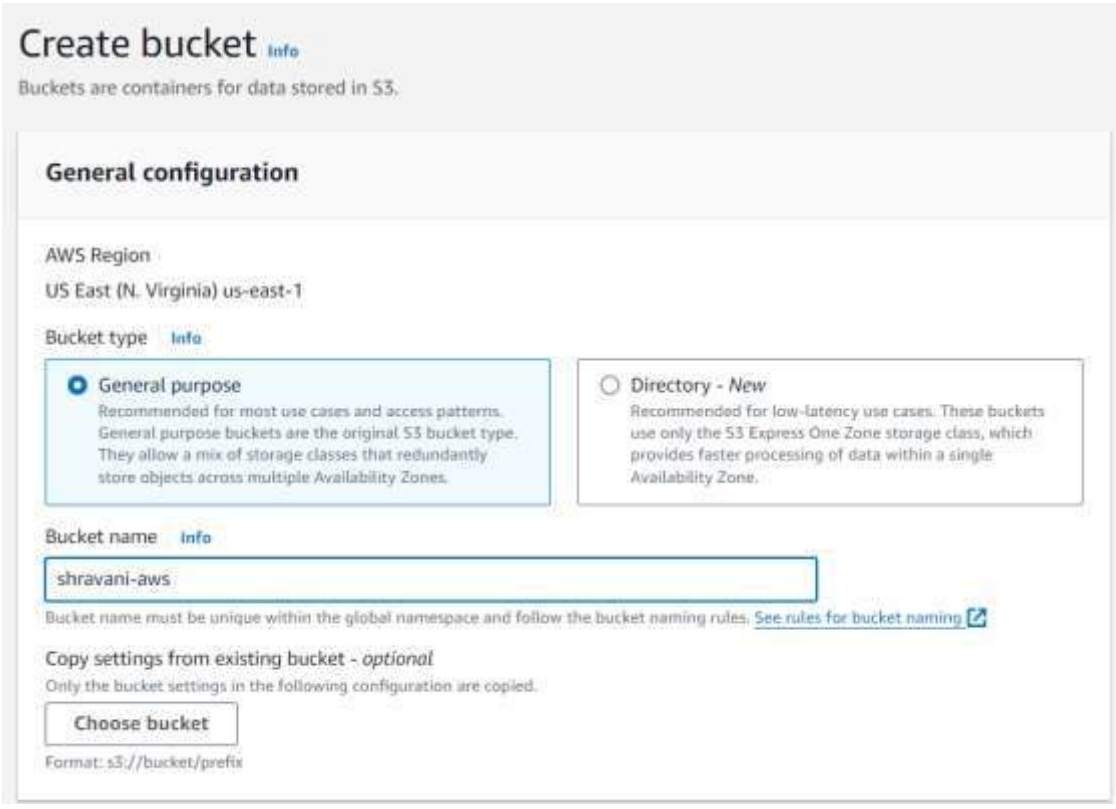


Hi Shravani here...

SHRAVANI RASAM D15A 46

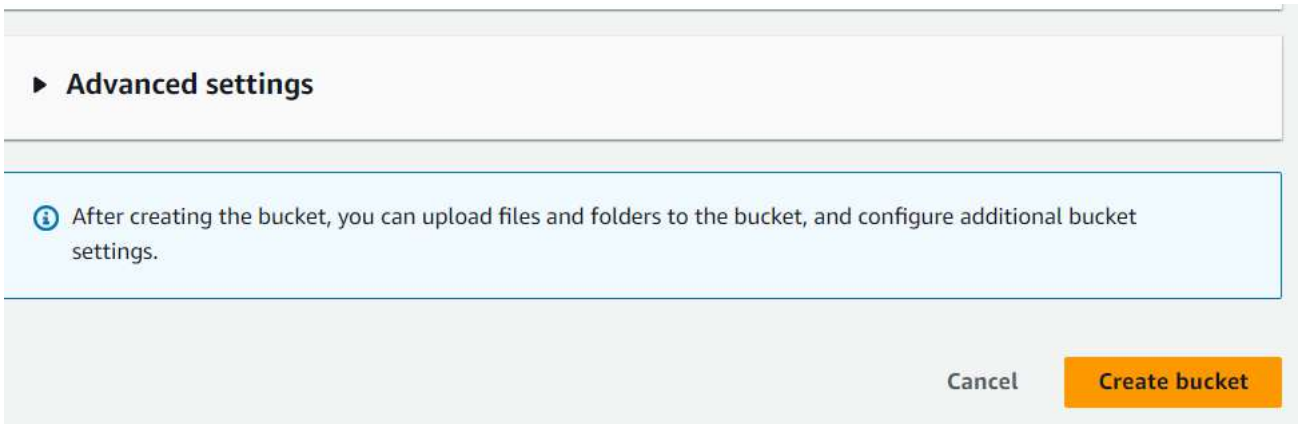
STATIC WEBSITE HOSTING USING S3 BUCKET

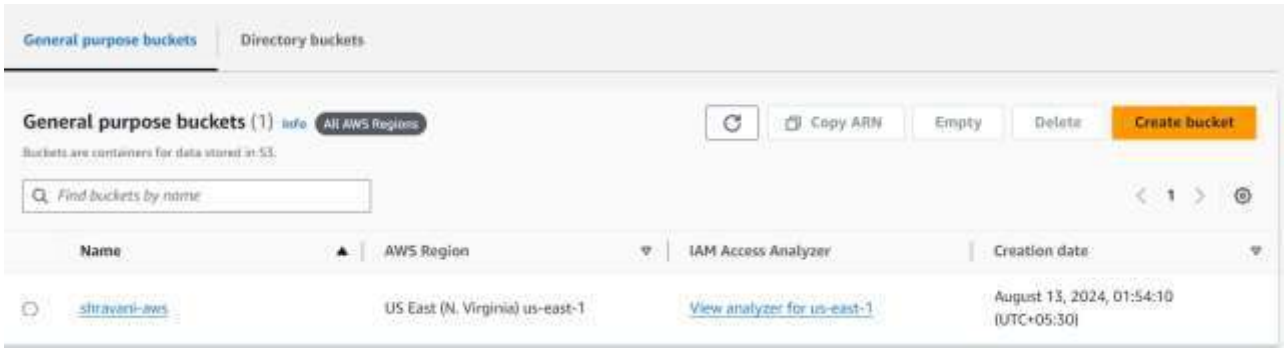


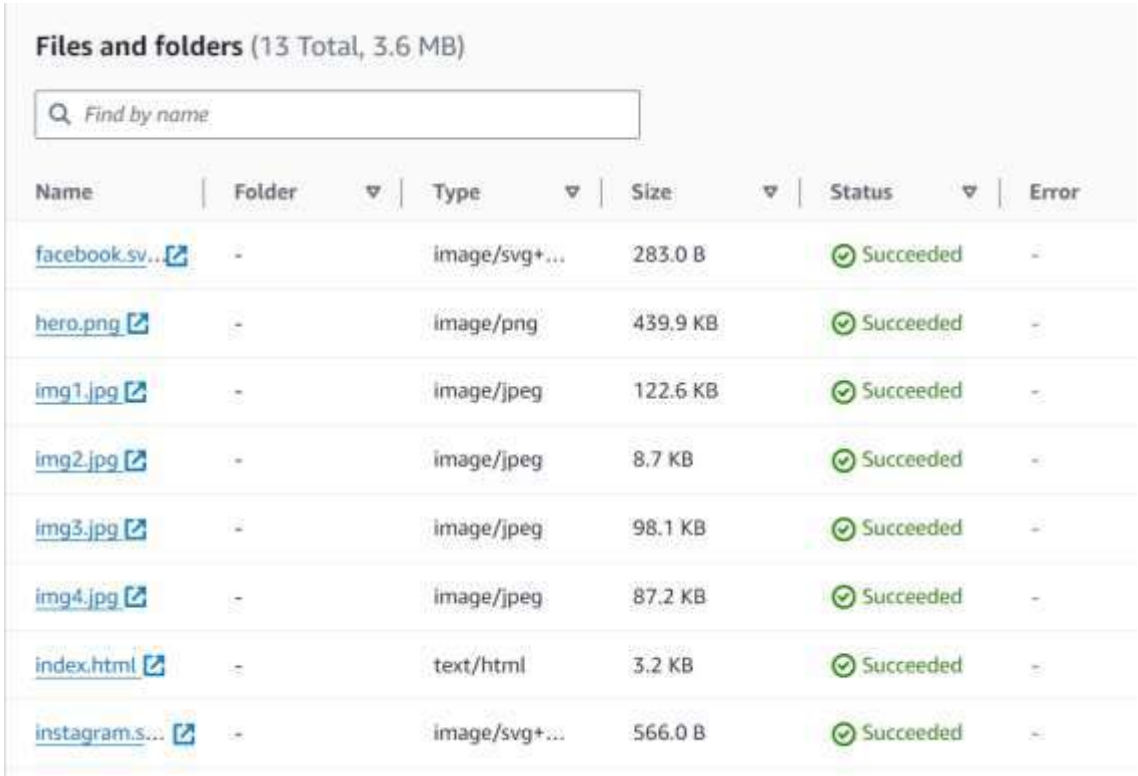


`

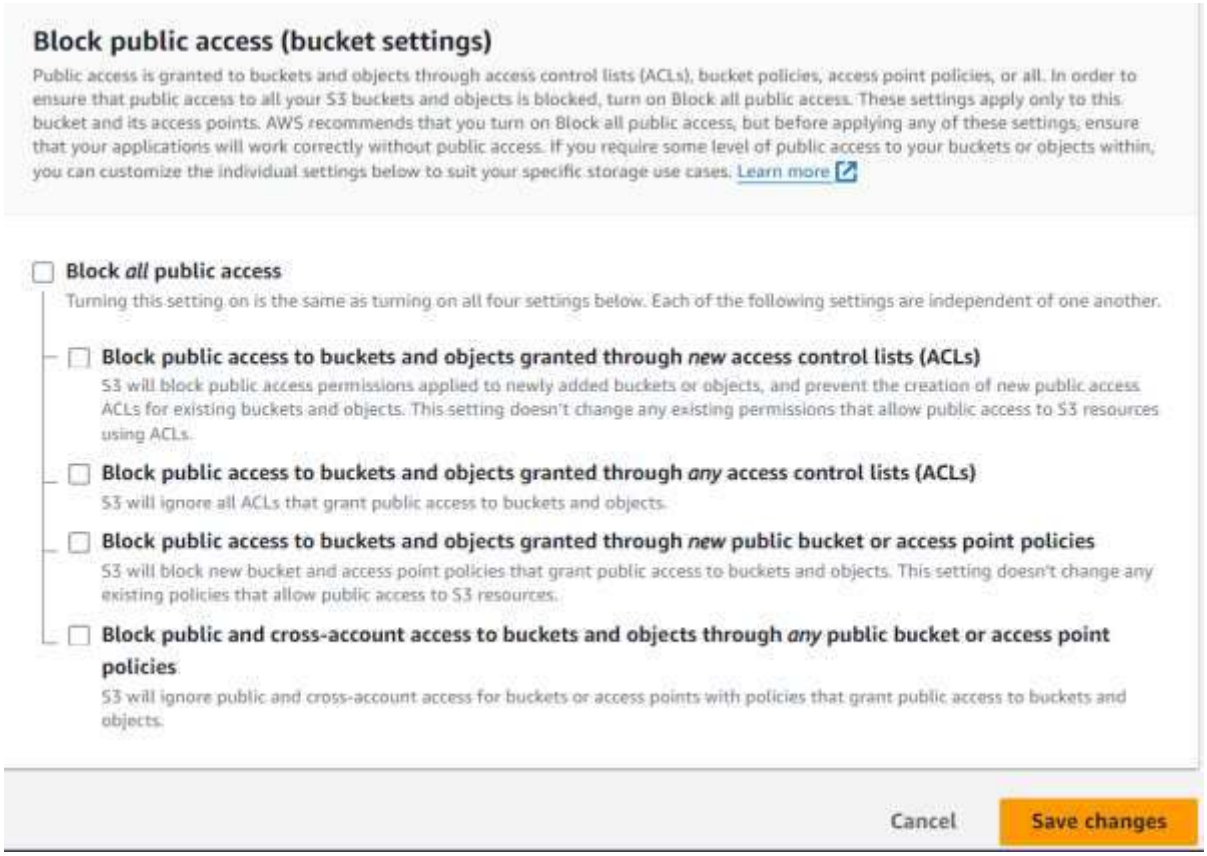
SHRAVANI RASAM D15A 46

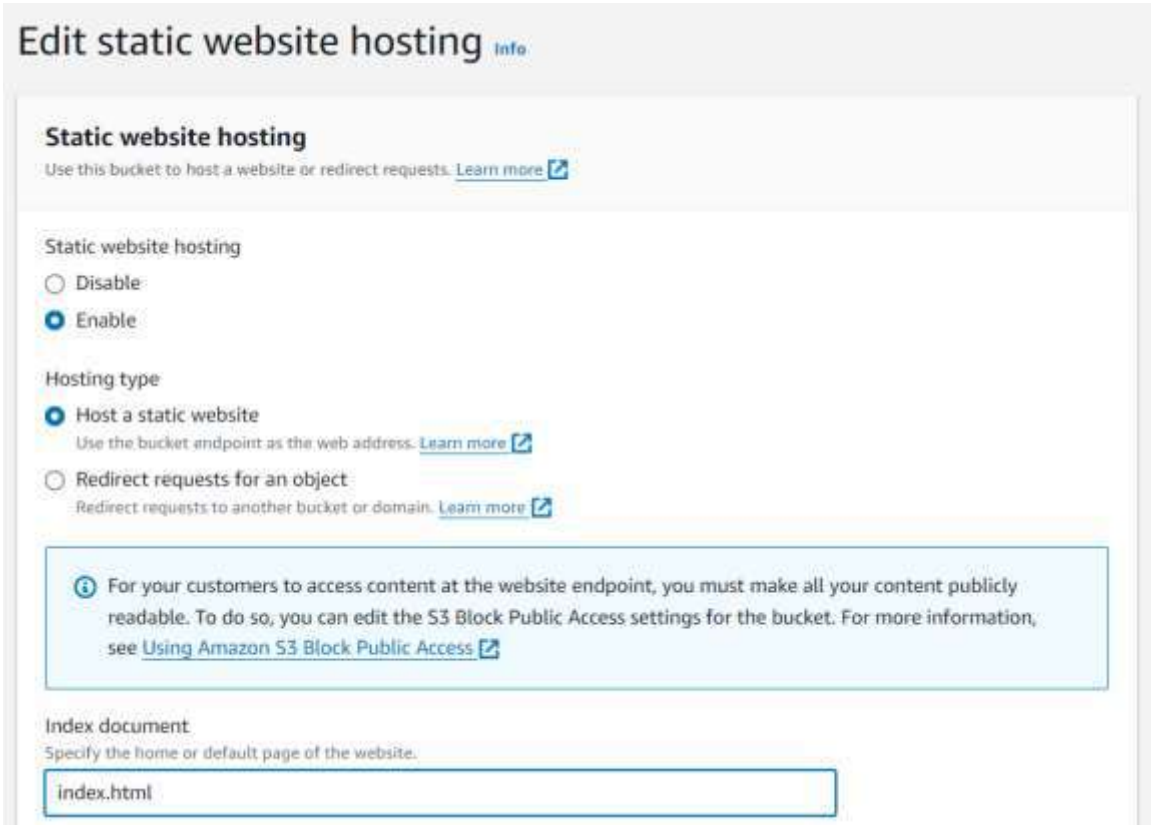






`


Block public access (bucket settings)
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
Turning this setting on is the same as turning on all four settings below. Each of the following settings are independent of one another.
Block public access to buckets and objects granted through new access control lists (ACLs)
S3 will block public access permissions applied to newly added buckets or objects, and prevent the creation of new public access ACLs for existing buckets and objects. This setting doesn't change any existing permissions that allow public access to S3 resources using ACLs.
Block public access to buckets and objects granted through any access control lists (ACLs)
S3 will ignore all ACLs that grant public access to buckets and objects.
Block public access to buckets and objects granted through new public bucket or access point policies
S3 will block new bucket and access point policies that grant public access to buckets and objects. This setting doesn't change any existing policies that allow public access to S3 resources.
Block public and cross-account access to buckets and objects through any public bucket or access point policies
S3 will ignore public and cross-account access for buckets or access points with policies that grant public access to buckets and objects.
Cancel
Save changes


Edit static website hosting Info
Static website hosting
Use this bucket to host a website or redirect requests. Learn more
Static website hosting
Disable
Enable
Hosting type
Host a static website
Use the bucket endpoint as the web address. Learn more
Redirect requests for an object
Redirect requests to another bucket or domain. Learn more
For your customers to access content at the website endpoint, you must make all your content publicly readable. To do so, you can edit the S3 Block Public Access settings for the bucket. For more information, see Using Amazon S3 Block Public Access
Index document
Specify the home or default page of the website.
index.html

`

Object Ownership

Control ownership of objects written to this bucket from other AWS accounts and the use of access control lists (ACLs). Object ownership determines who can specify access to objects.

- ○ ACLs disabled (recommended)
  All objects in this bucket are owned by this account. Access to this bucket and its objects is specified using only policies.
- ◉ ACLs enabled
  Objects in this bucket can be owned by other AWS accounts. Access to this bucket and its objects can be specified using ACLs.

⚠ We recommend disabling ACLs, unless you need to control access for each object individually or to have the object writer own the data they upload. Using a bucket policy instead of ACLs to share data with users outside of your account simplifies permissions management and auditing.

⚠ **Enabling ACLs turns off the bucket owner enforced setting for Object Ownership**
Once the bucket owner enforced setting is turned off, access control lists (ACLs) and their associated permissions are restored. Access to objects that you do not own will be based on ACLs and not the bucket policy.

☑ I acknowledge that ACLs will be restored.

`

SHRAVANI RASAM D15A 46

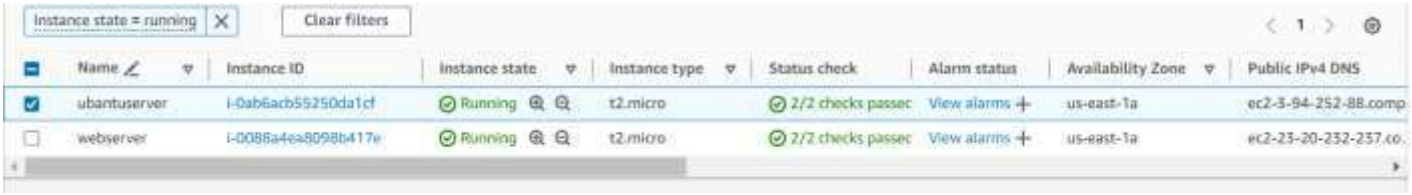
HOSTED LINK:

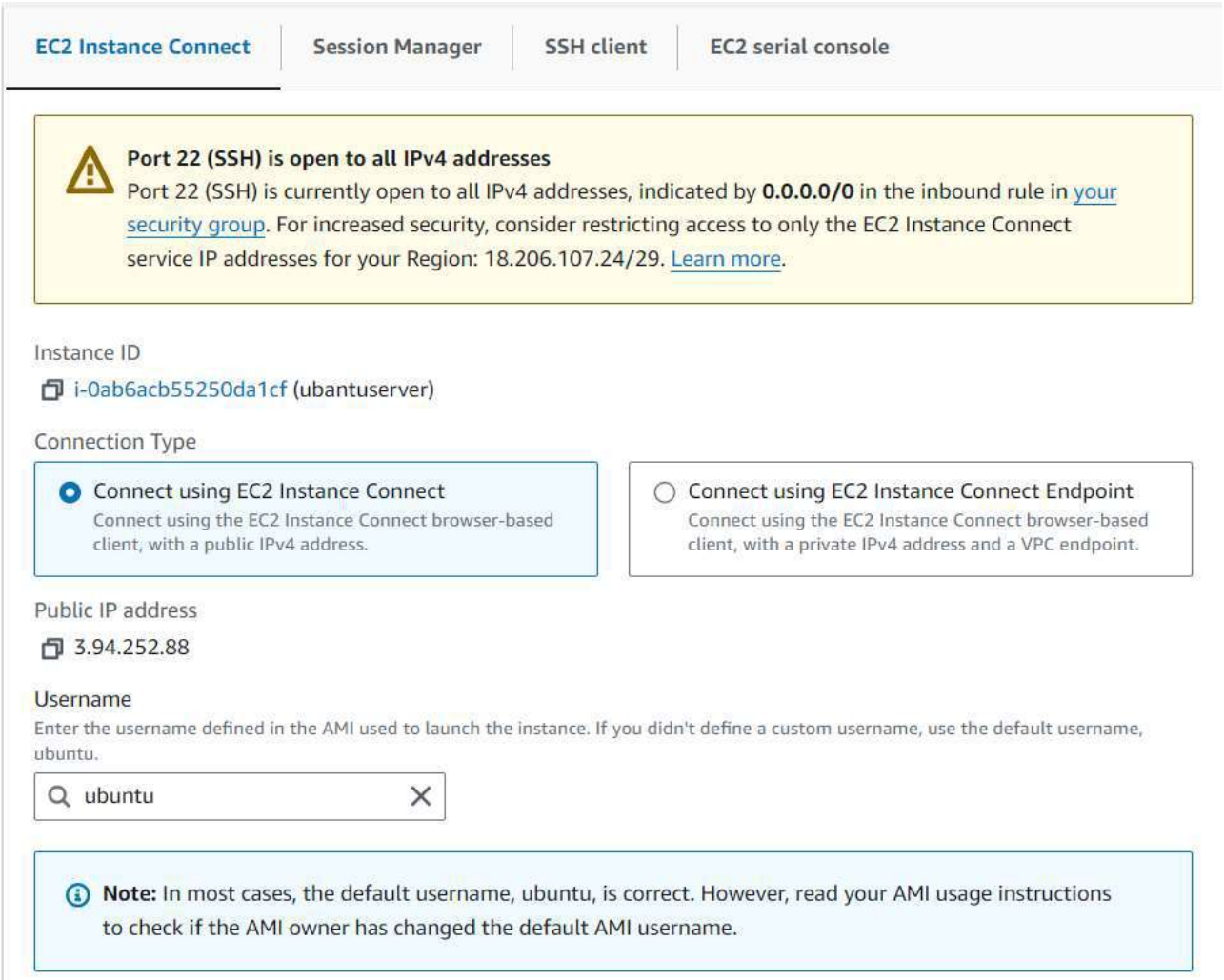
https://shravani-aws.s3.amazonaws.com/index.html

`

SHRAVANI RASAM D15A 46

EC2 DYNAMIC HOSTING





`

```
ubuntu@ip-172-31-38-150:~$ mkdir aws2
ubuntu@ip-172-31-38-150:~$ cd aws2
ubuntu@ip-172-31-38-150:~/aws2$ git clone https://github.com/ShravaniR2412/dynamic-hosting.git
Cloning into 'dynamic-hosting'...
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 0 (delta 0), pack-reused 0
Receiving objects: 100% (6/6), 11.68 KiB | 2.92 MiB/s, done.
ubuntu@ip-172-31-38-150:~/aws2$
```

```
ubuntu@ip-172-31-38-150:~/aws2$ ls
dynamic-hosting
ubuntu@ip-172-31-38-150:~/aws2$ cd dynamic-hosting/
ubuntu@ip-172-31-38-150:~/aws2/dynamic-hosting$ ls
index.js  package-lock.json  package.json
ubuntu@ip-172-31-38-150:~/aws2/dynamic-hosting$ npm i
```

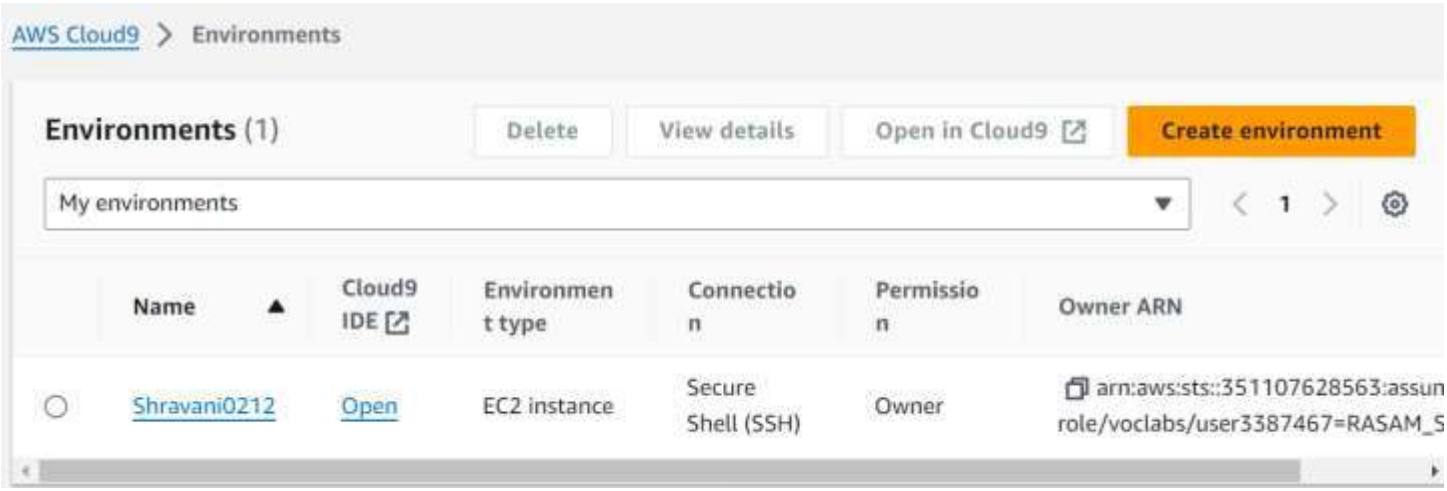
```
ubuntu@ip-172-31-38-150:~$ cd aws2
ubuntu@ip-172-31-38-150:~/aws2$ ls
dynamic-hosting
ubuntu@ip-172-31-38-150:~/aws2$ cd dynamic-hosting
ubuntu@ip-172-31-38-150:~/aws2/dynamic-hosting$ ls
index.js  package-lock.json  package.json
ubuntu@ip-172-31-38-150:~/aws2/dynamic-hosting$ npm i

added 93 packages, and audited 94 packages in 3s

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

found 0 vulnerabilities
ubuntu@ip-172-31-38-150:~/aws2/dynamic-hosting$ npm start
npm ERR! Missing script: "start"
npm ERR!
npm ERR! Did you mean one of these?
npm ERR!     npm star # Mark your favorite packages
npm ERR!     npm stars # View packages marked as favorites
npm ERR!
npm ERR! To see a list of scripts, run:
npm ERR!   npm run

npm ERR! A complete log of this run can be found in:
npm ERR!     /home/ubuntu/.npm/_logs/2024-08-12T21_17_06_213Z-debug-0.log
ubuntu@ip-172-31-38-150:~/aws2/dynamic-hosting$ node index.js
Server is running on port 3000
```

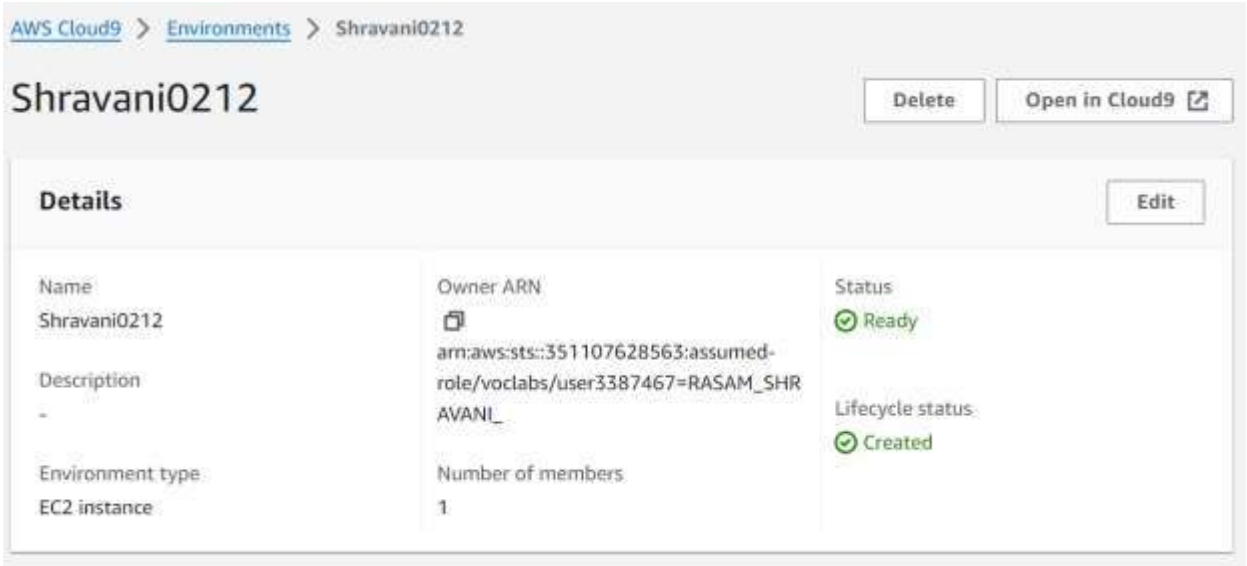
Hey Shravani this is Dynamic Website.

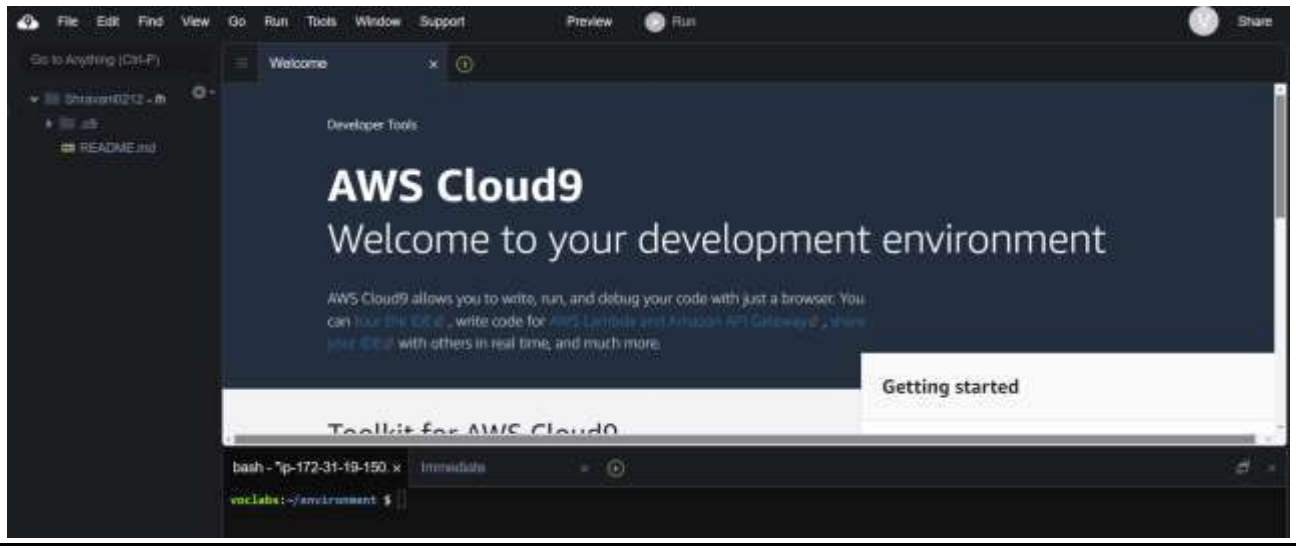
Hey this is login page.

`

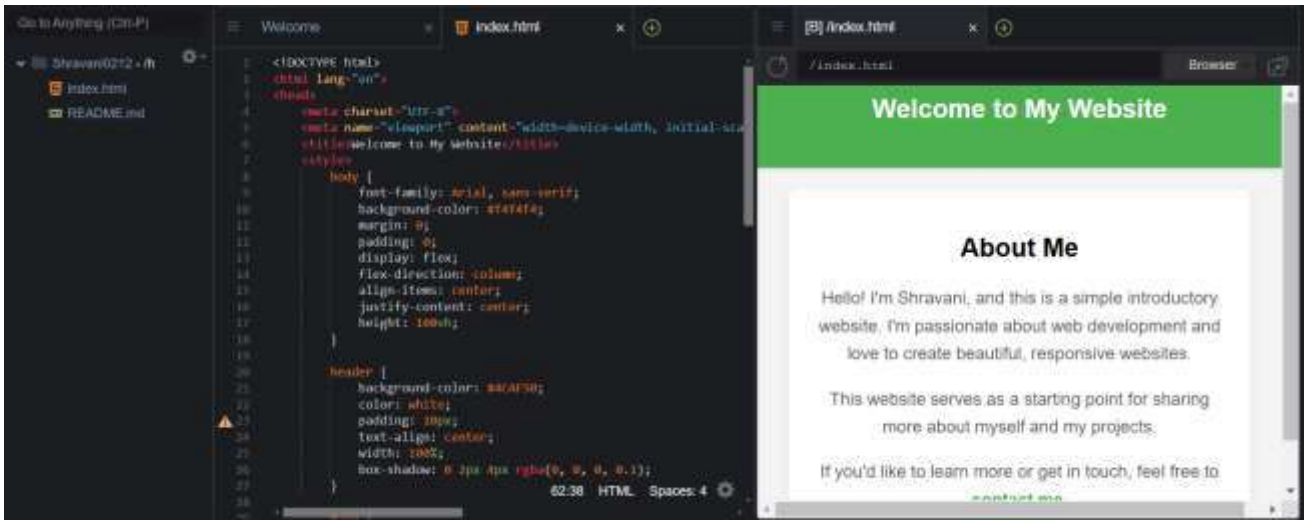
CLOUD 9 HOSTING







SHRAVANI RASAM D15A 46



User details

User name

shravani

The user name can have up to 64 characters. Valid characters: A-Z, a-z, 0-9, and + = , . @ _ - (hyphen)

☑ Provide user access to the AWS Management Console - *optional*

If you're providing console access to a person, it's a best practice to manage their access in IAM Identity Center.

Are you providing console access to a person?

User type

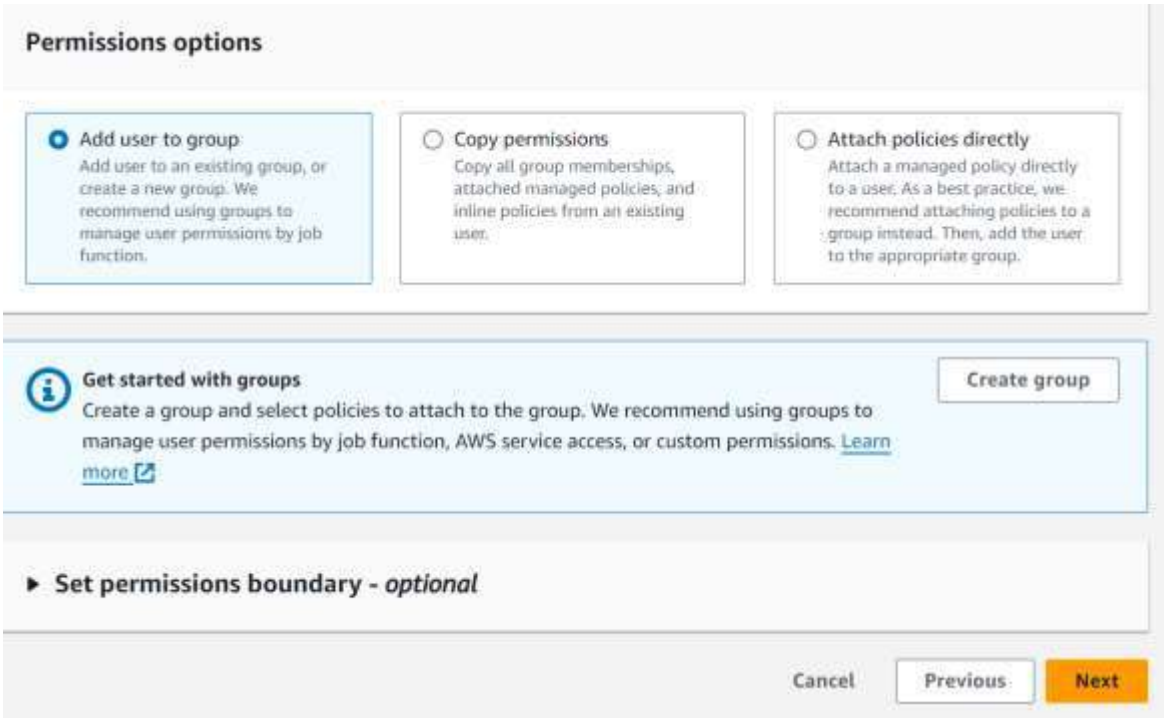
○ Specify a user in Identity Center - Recommended

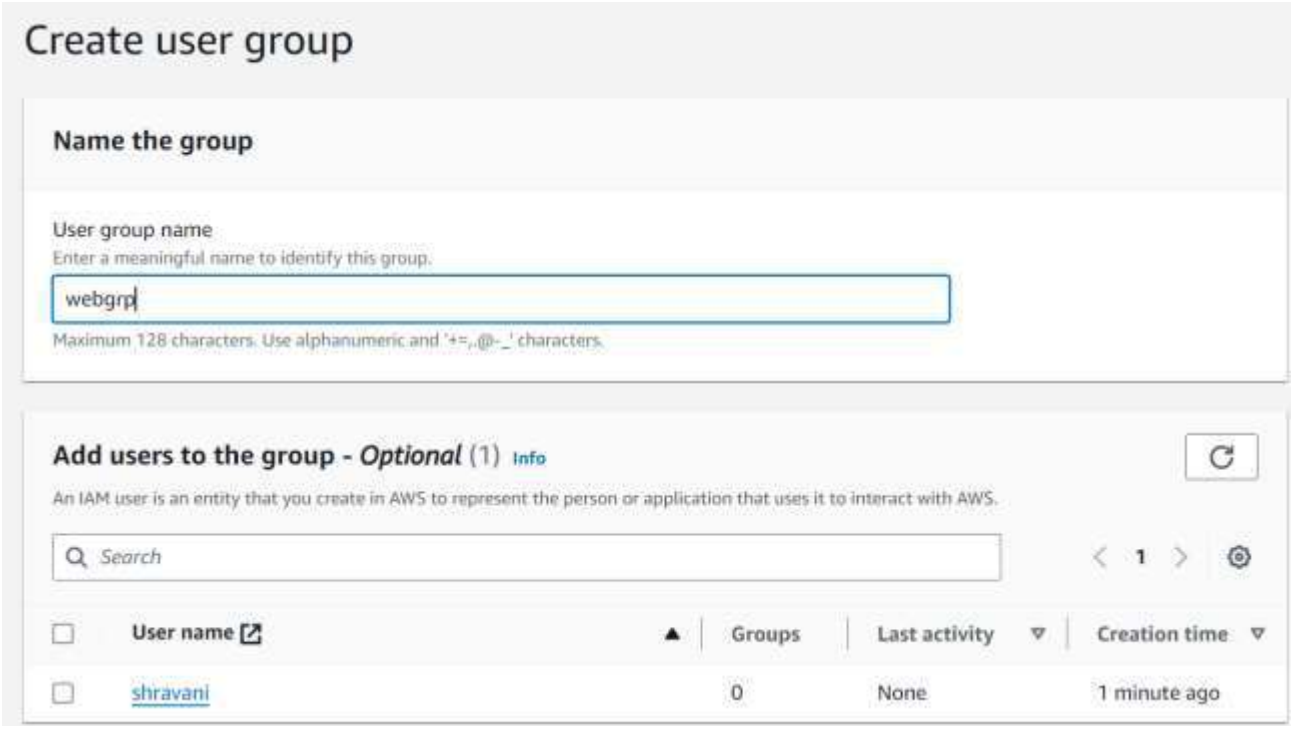
We recommend that you use Identity Center to provide console access to a person. With Identity Center, you can centrally manage user access to their AWS accounts and cloud applications.

◉ I want to create an IAM user

We recommend that you create IAM users only if you need to enable programmatic access through access keys, service-specific credentials for AWS CodeCommit or Amazon Keyspaces, or a backup credential for emergency account access.

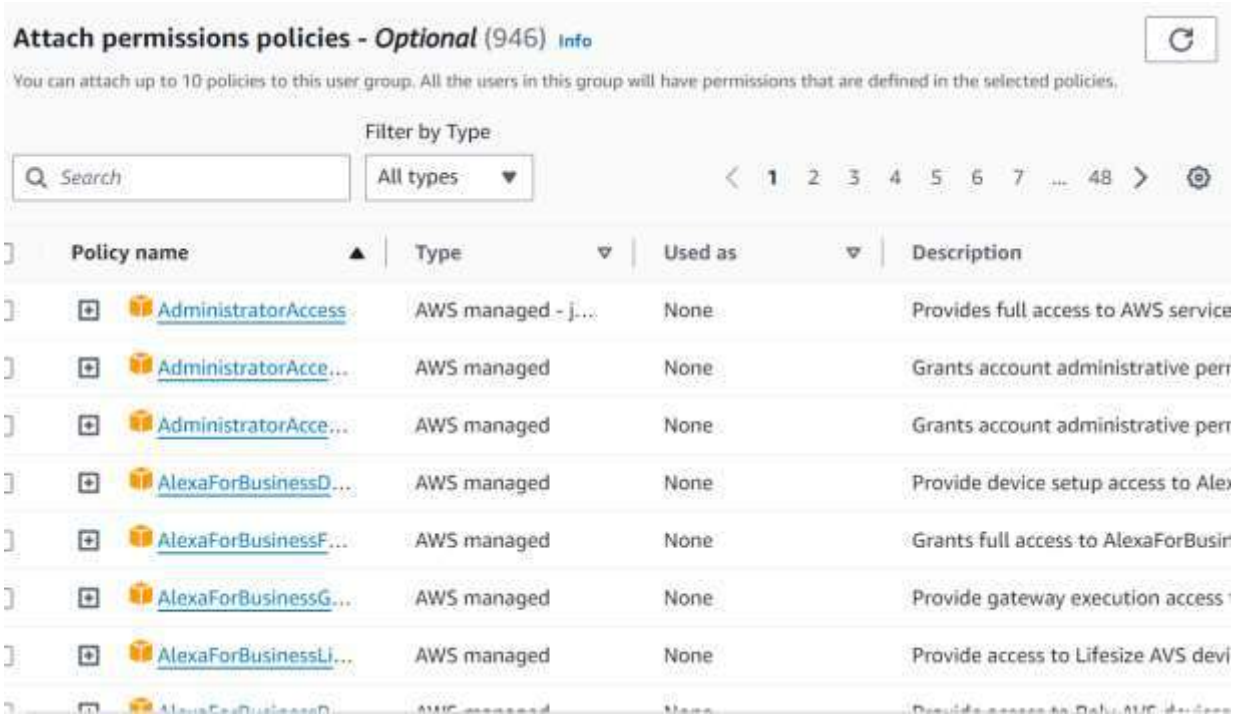
`

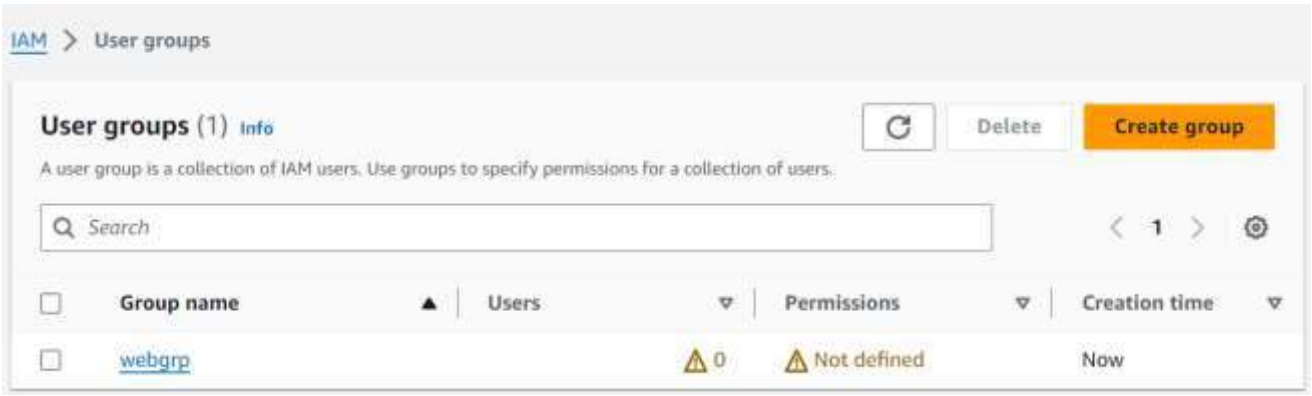

Permissions options
Add user to group
Add user to an existing group, or create a new group. We recommend using groups to manage user permissions by job function.
Copy permissions
Copy all group memberships, attached managed policies, and inline policies from an existing user.
Attach policies directly
Attach a managed policy directly to a user. As a best practice, we recommend attaching policies to a group instead. Then, add the user to the appropriate group.
Get started with groups
Create a group and select policies to attach to the group. We recommend using groups to manage user permissions by job function, AWS service access, or custom permissions. Learn more
Create group
Set permissions boundary - optional
Cancel
Previous
Next

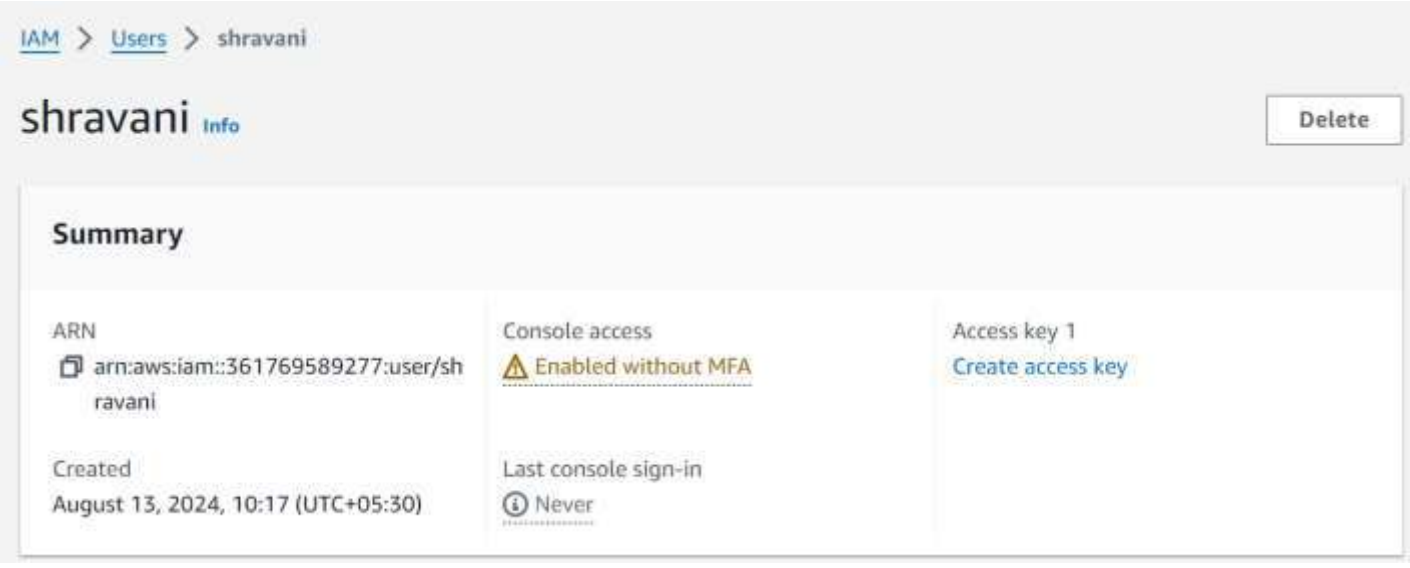

Create user group
Name the group
User group name
Enter a meaningful name to identify this group.
webgrp
Maximum 128 characters. Use alphanumeric and '+=,.@-_' characters.
Add users to the group - Optional (1) Info
An IAM user is an entity that you create in AWS to represent the person or application that uses it to interact with AWS.
Search
1
User name
Groups
Last activity
Creation time
shravani
0
None
1 minute ago

`

SHRAVANI RASAM D15A 46







`

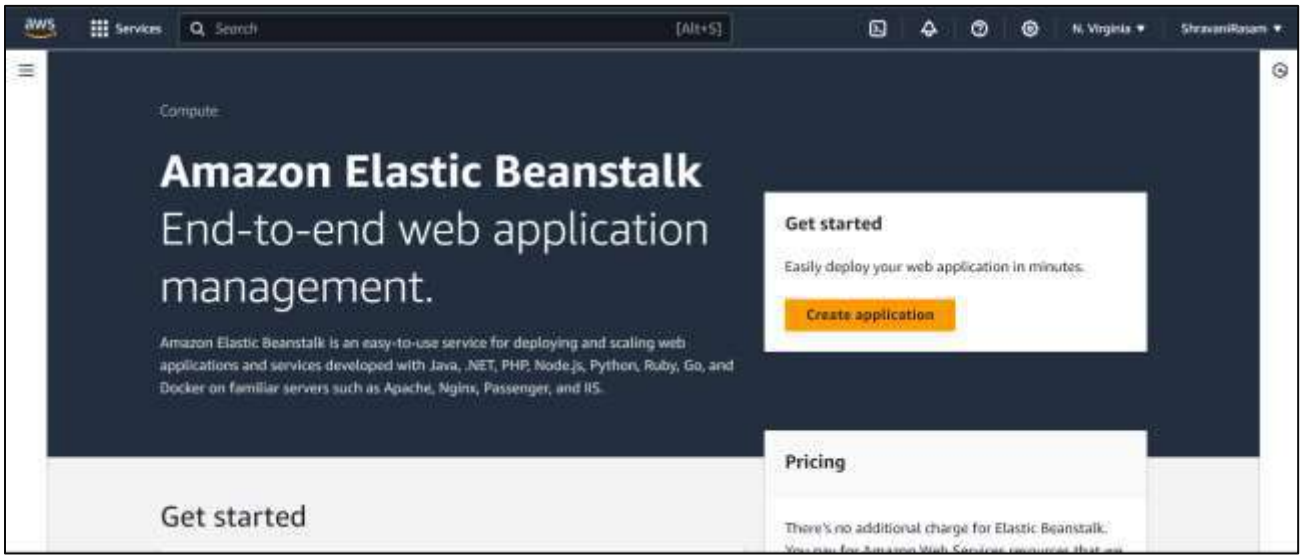
SHRAVANI RASAM D15A 46

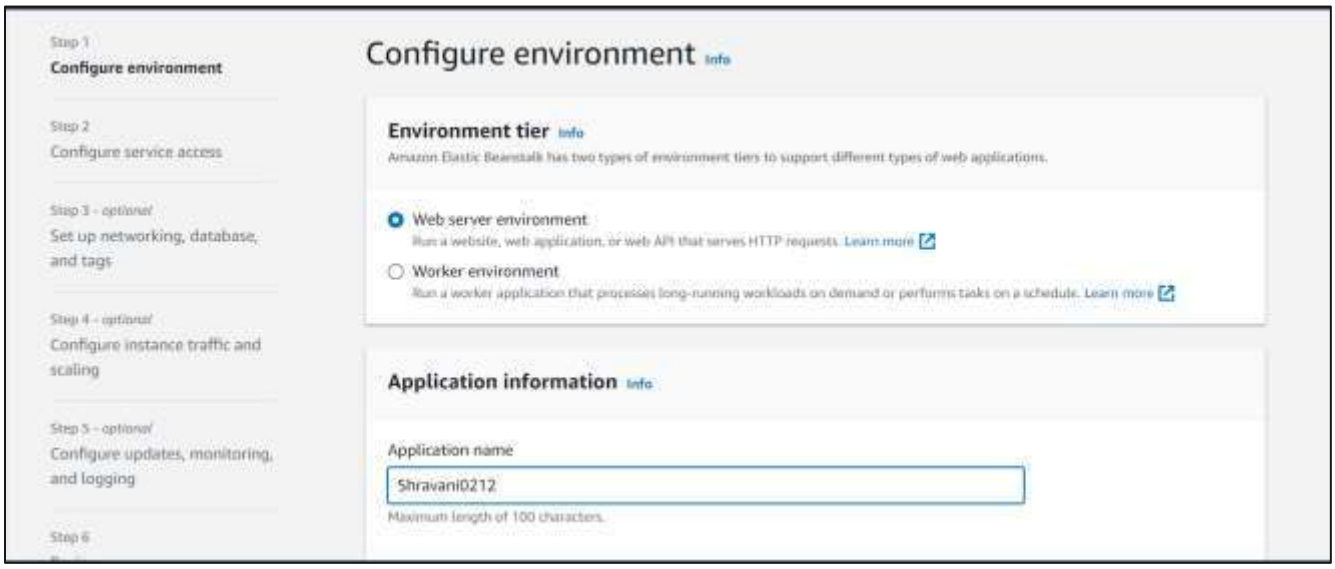
`

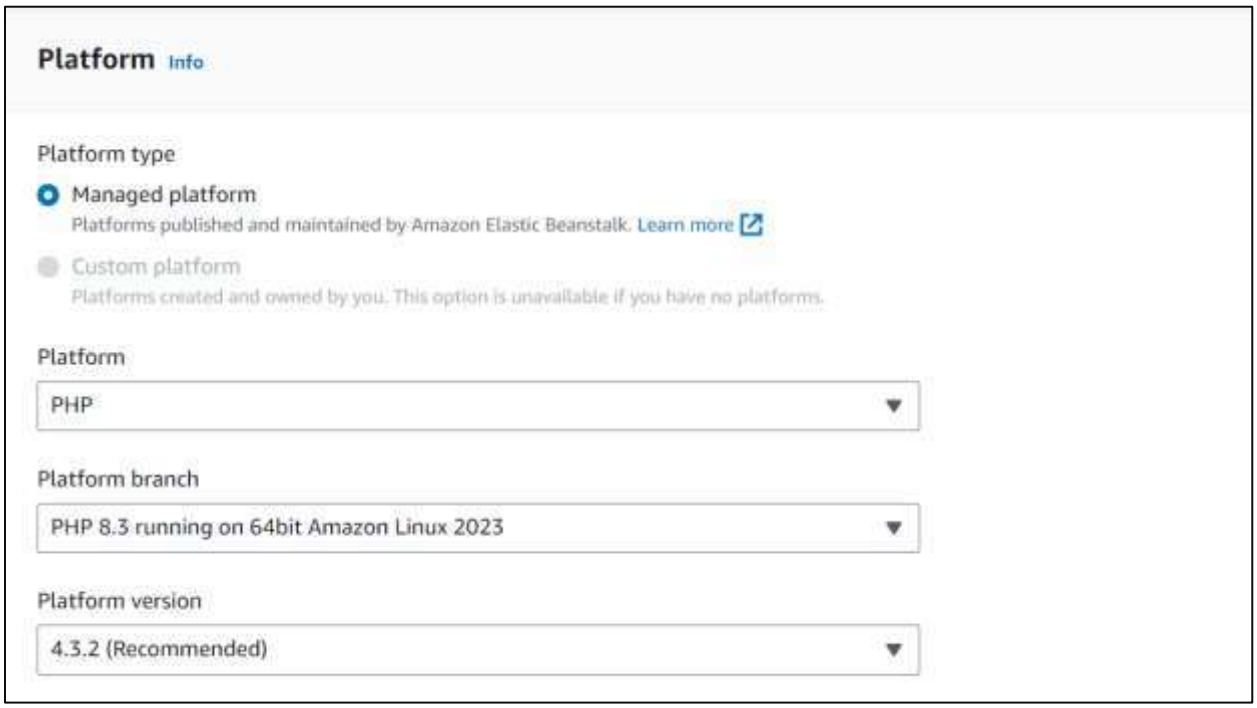
Shravani Rasam D15A 46

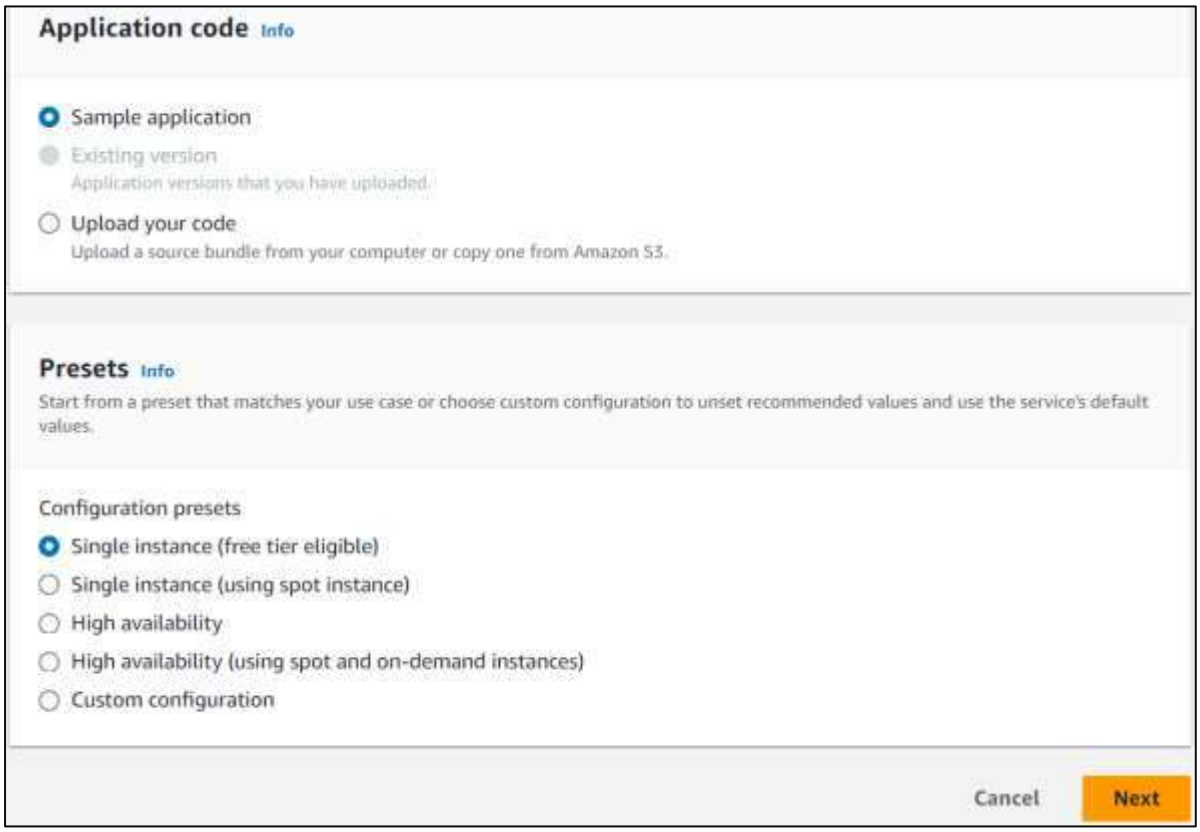
ADVANCE DEVOPS EXPERIMENT 2

Aim : To Build Your Application using AWS CodeBuild and Deploy on S3 / SEBS using AWS CodePipeline, deploy Sample Application on EC2 instance using AWS CodeDeploy






Platform Info
Platform type
Managed platform
Platforms published and maintained by Amazon Elastic Beanstalk. Learn more
Custom platform
Platforms created and owned by you. This option is unavailable if you have no platforms.
Platform
PHP
Platform branch
PHP 8.3 running on 64bit Amazon Linux 2023
Platform version
4.3.2 (Recommended)


Application code Info
Sample application
Existing version
Application versions that you have uploaded.
Upload your code
Upload a source bundle from your computer or copy one from Amazon S3.
Presets Info
Start from a preset that matches your use case or choose custom configuration to unset recommended values and use the service's default values.
Configuration presets
Single instance (free tier eligible)
Single instance (using spot instance)
High availability
High availability (using spot and on-demand instances)
Custom configuration
Cancel
Next

Service access

IAM roles, assumed by Elastic Beanstalk as a service role, and EC2 instance profiles allow Elastic Beanstalk to create and manage your environment. Both the IAM role and instance profile must be attached to IAM managed policies that contain the required permissions. Learn more

Service role

- (●) Create and use new service role
- ( ) Use an existing service role

Service role name

Enter the name for an IAM role that Elastic Beanstalk will create to assume as a service role. Beanstalk will attach the required managed policies to it.

aws-elasticbeanstalk-service-role

View permission details

EC2 key pair

Select an EC2 key pair to securely log in to your EC2 instances. Learn more

shravani02

EC2 instance profile

Choose an IAM instance profile with managed policies that allow your EC2 instances to perform required operations.

S1

View permission details

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-024c98e3c11533db9 | (172.31.0.0/16)

Create custom VPC

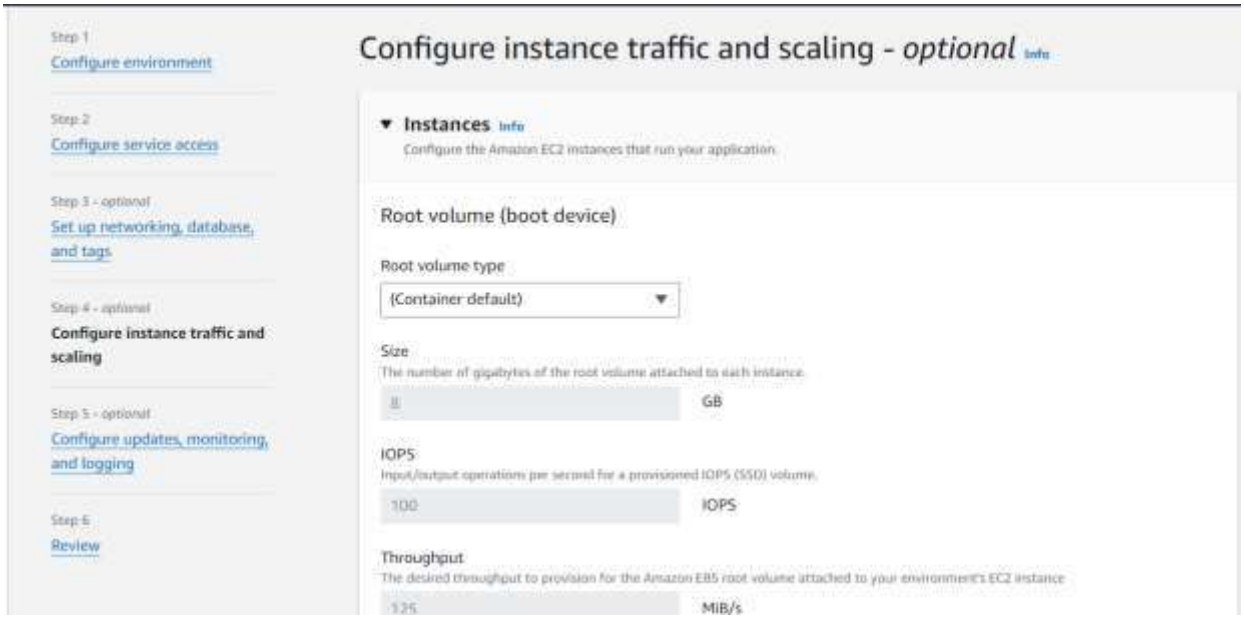
Instance settings

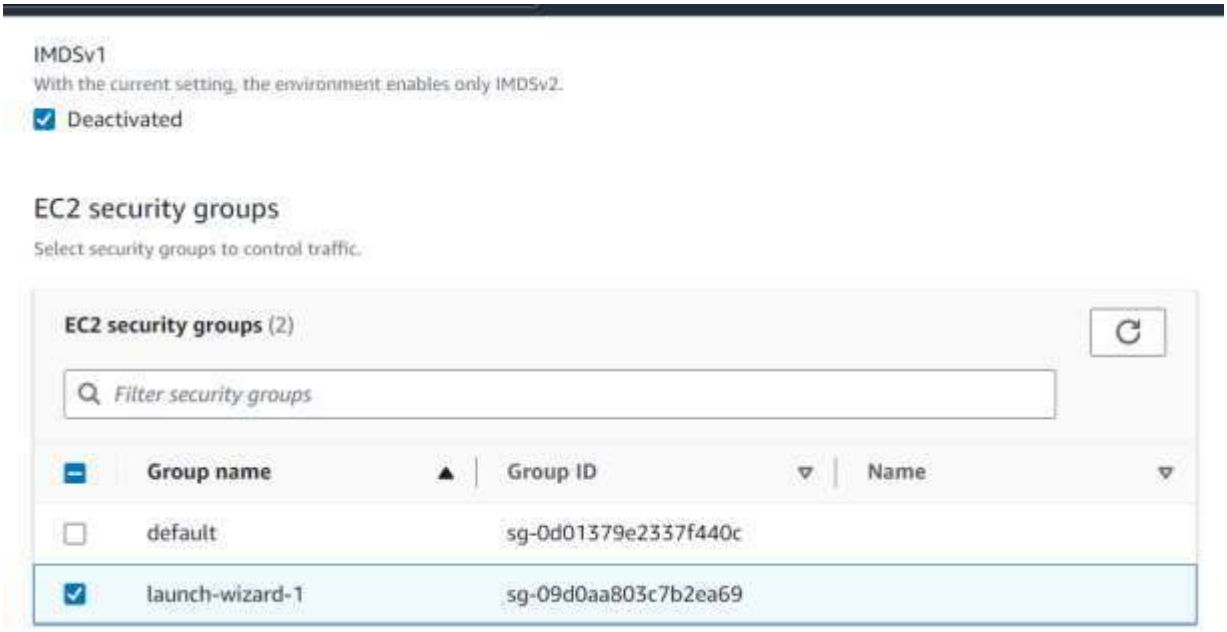
Choose a subnet in each AZ for the instances that run your application. To avoid exposing your instances to the Internet, run your instances in private subnets and load balancer in public subnets. To run your load balancer and instances in the same public subnets, assign public IP addresses to the instances. Learn more

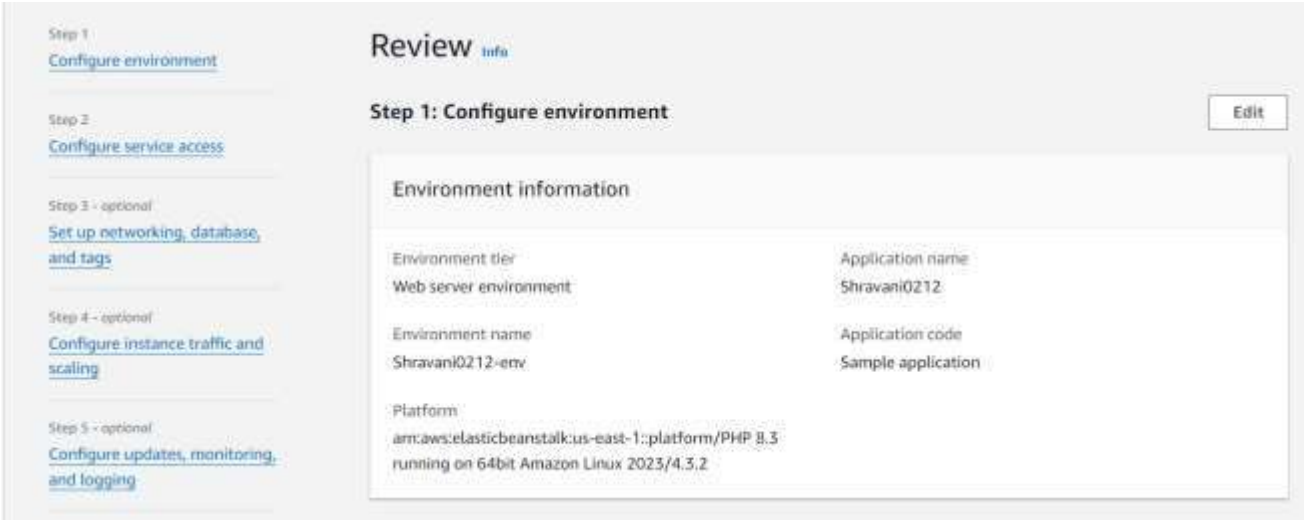
Public IP address

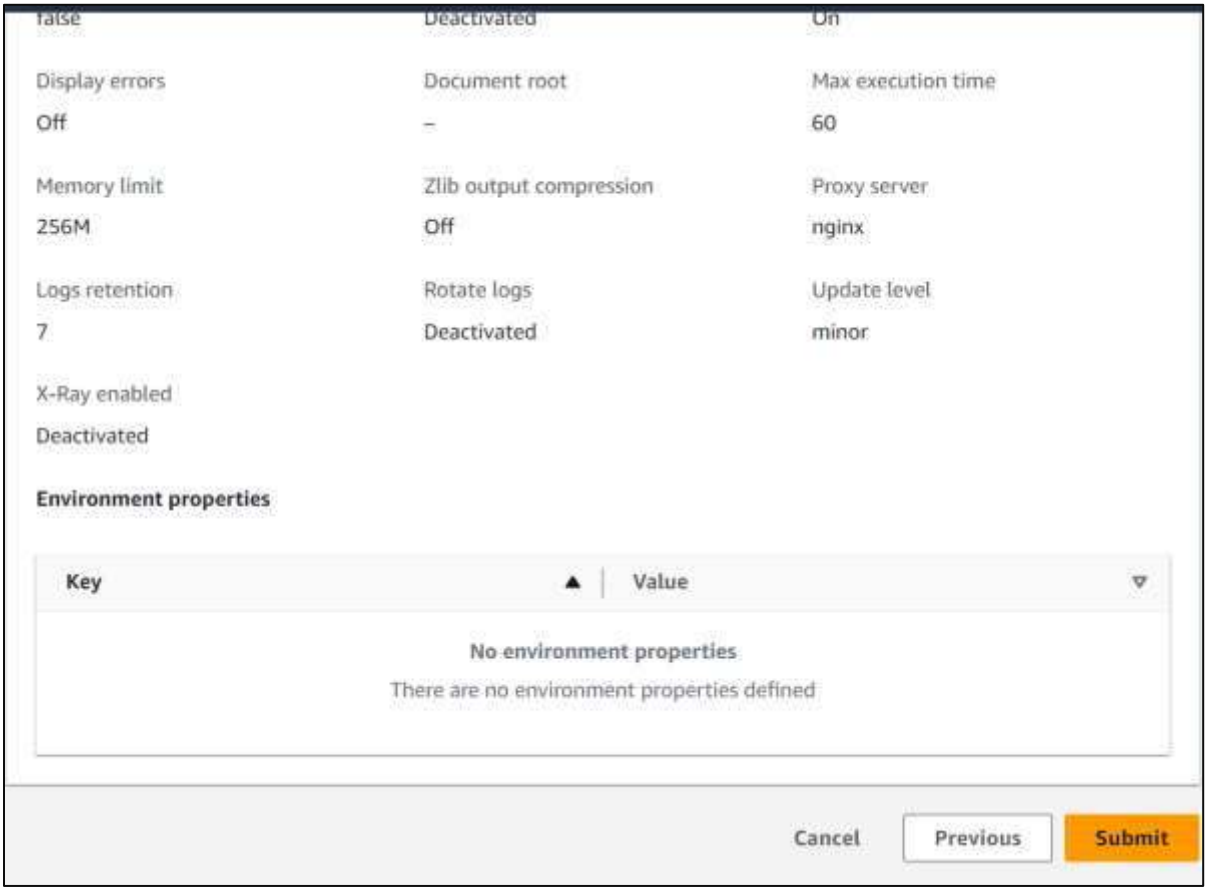
Assign a public IP address to the Amazon EC2 instances in your environment.

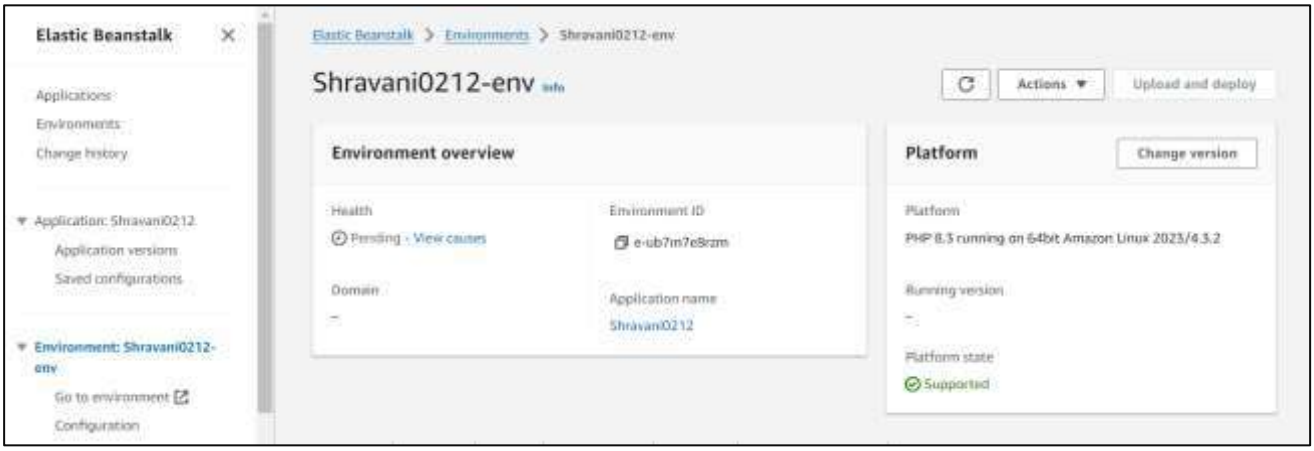
☐ Activated

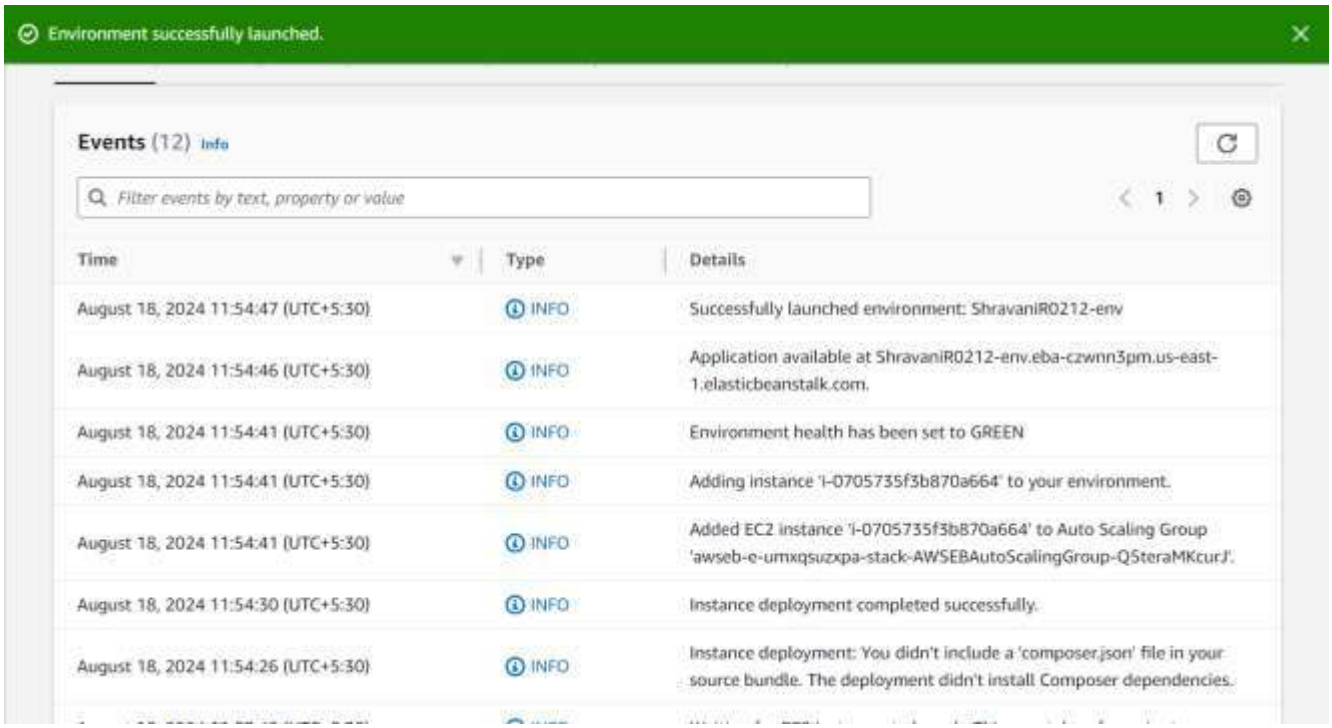

Step 1
Configure environment
Step 2
Configure service access
Step 3 - optional
Set up networking, database, and tags
Step 4 - optional
Configure instance traffic and scaling
Step 5 - optional
Configure updates, monitoring, and logging
Step 6
Review
Configure instance traffic and scaling - optional Info
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

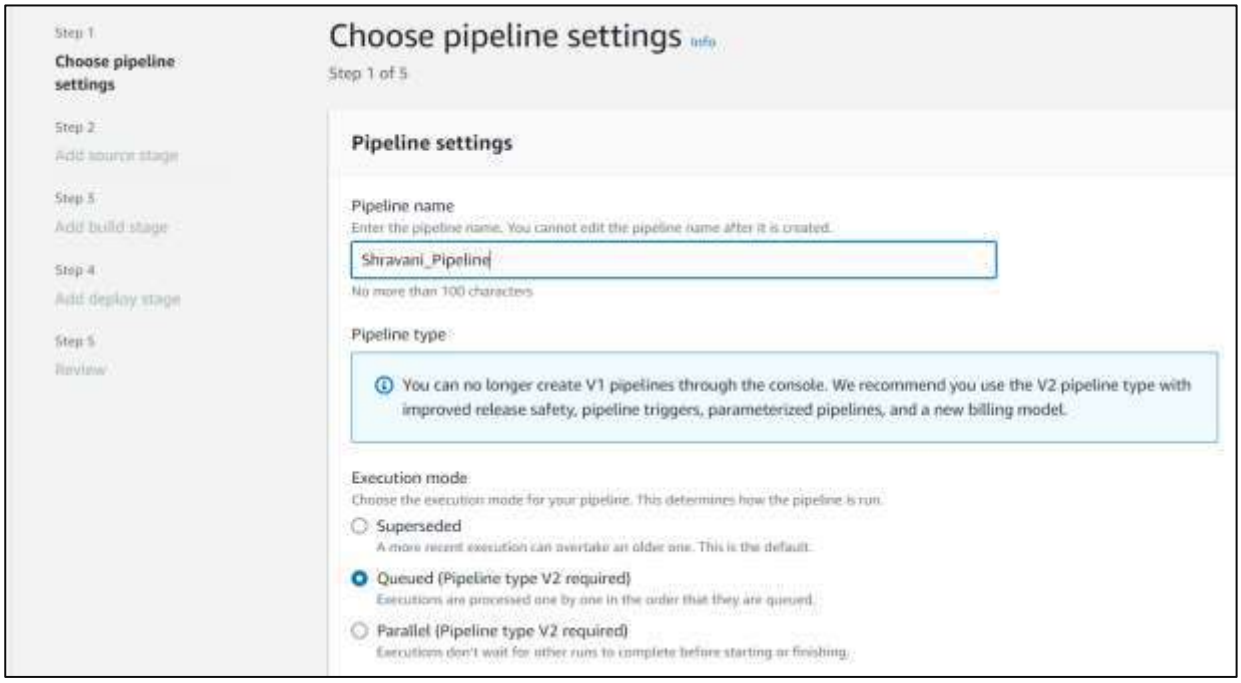

IMDSv1
With the current setting, the environment enables only IMDSv2.
Deactivated
EC2 security groups
Select security groups to control traffic.
EC2 security groups (2)
Filter security groups
Group name
Group ID
Name
default
sg-0d01379e2337f440c
launch-wizard-1
sg-09d0aa803c7b2ea69

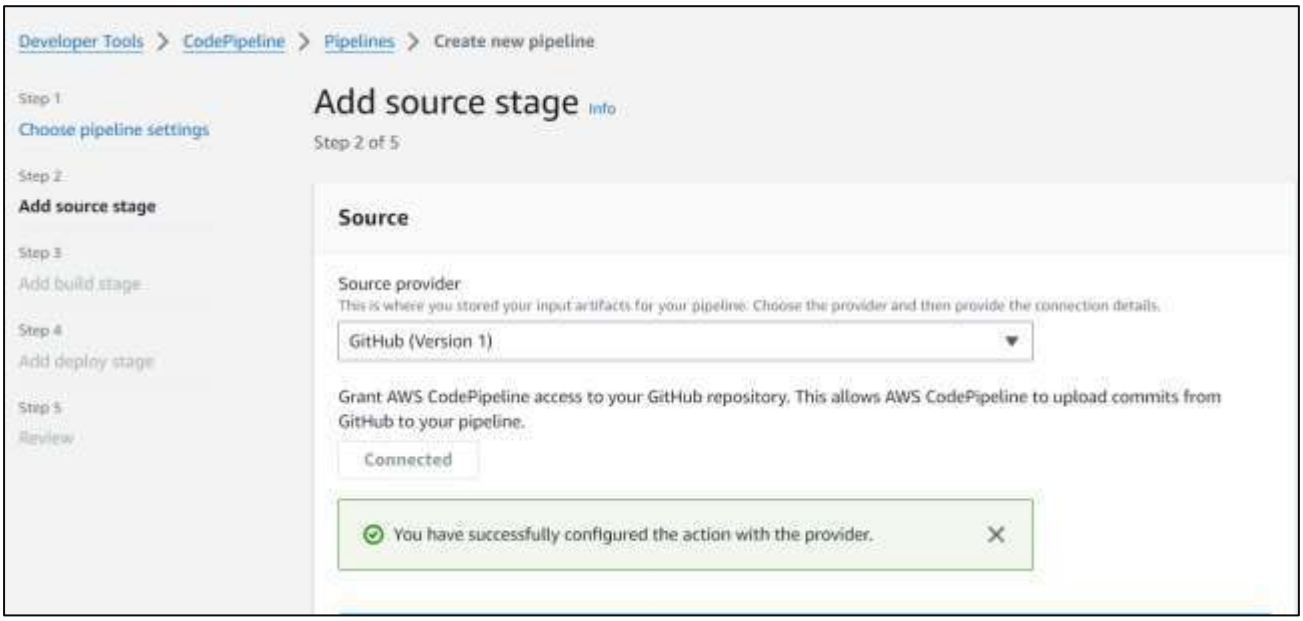

Step 1
Configure environment
Step 2
Configure service access
Step 3 - optional
Set up networking, database, and tags
Step 4 - optional
Configure instance traffic and scaling
Step 5 - optional
Configure updates, monitoring, and logging
Review Info
Step 1: Configure environment
Edit
Environment information
Environment tier
Web server environment
Application name
Shravani0212
Environment name
Shravani0212-env
Application code
Sample application
Platform
arn:aws:elasticbeanstalk:us-east-1::platform/PHP 8.3 running on 64bit Amazon Linux 2023/4.3.2

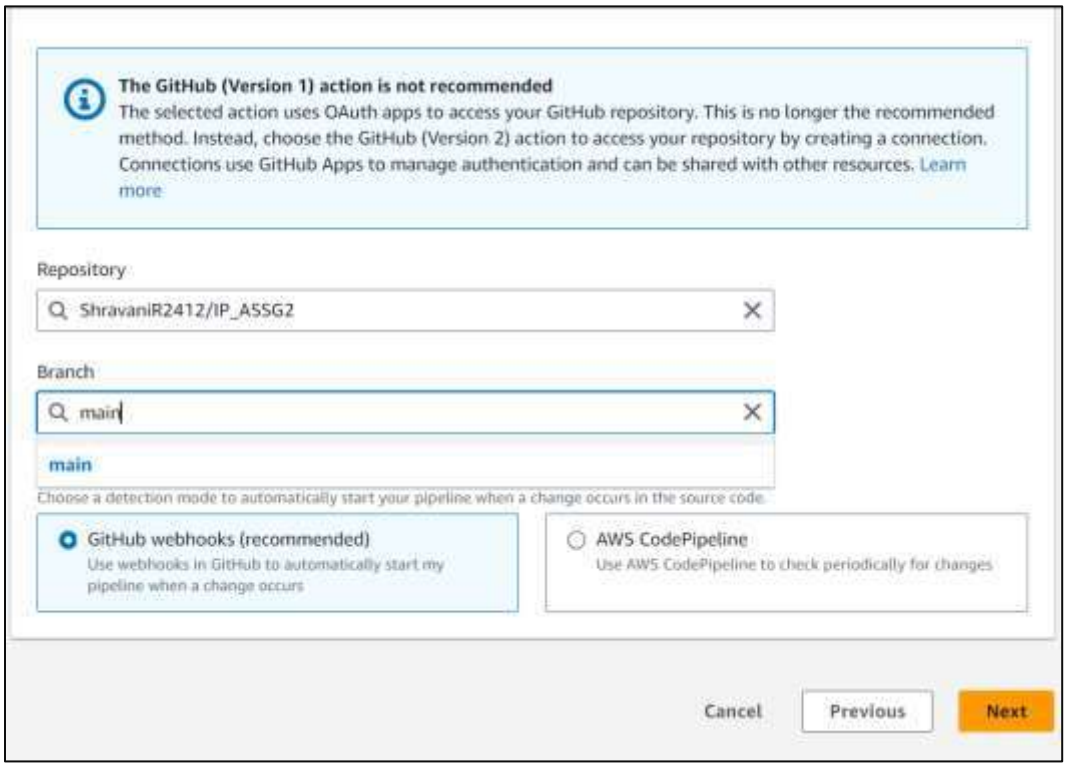

false
Deactivated
On
Display errors
Off
Document root
–
Max execution time
60
Memory limit
256M
Zlib output compression
Off
Proxy server
nginx
Logs retention
7
Rotate logs
Deactivated
Update level
minor
X-Ray enabled
Deactivated
Environment properties
Key
Value
No environment properties
There are no environment properties defined
Cancel
Previous
Submit

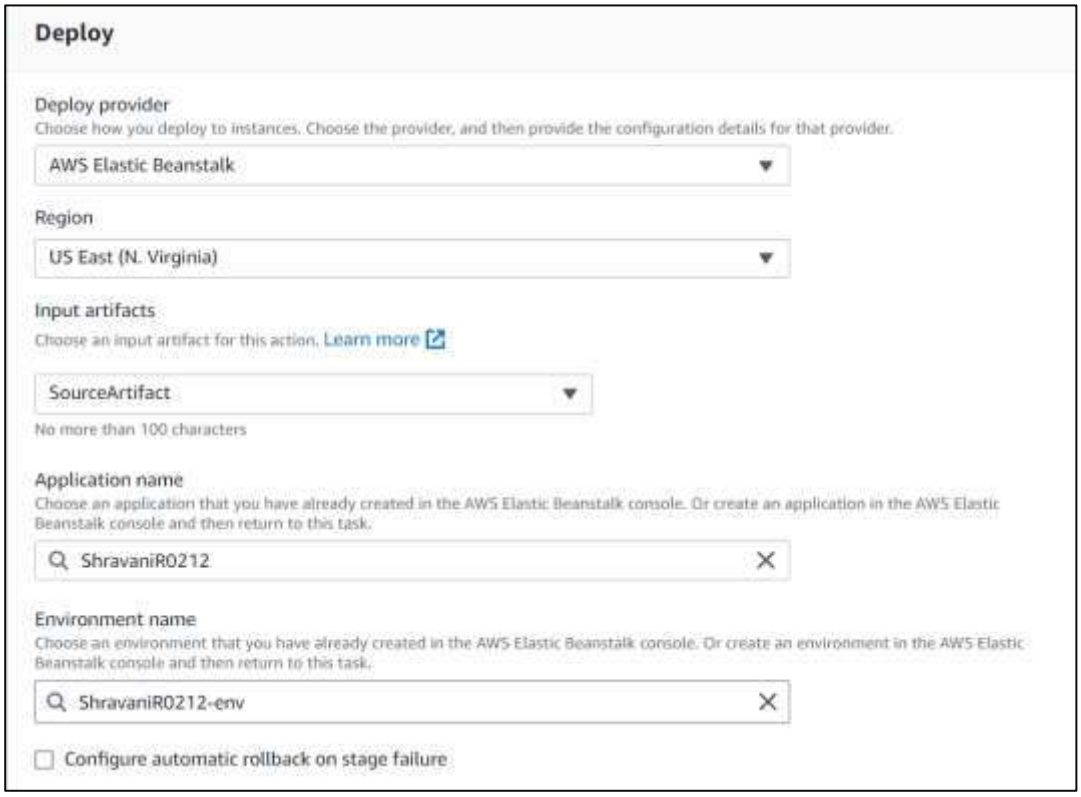

Elastic Beanstalk
Applications
Environments
Change history
Application: Shravani0212
Application versions
Saved configurations
Environment: Shravani0212-env
Go to environment
Configuration
Elastic Beanstalk > Environments > Shravani0212-env
Shravani0212-env info
Actions
Upload and deploy
Environment overview
Health
Pending · View causes
Environment ID
e-ub7m7e8rzm
Domain
-
Application name
Shravani0212
Platform
Change version
Platform
PHP 8.3 running on 64bit Amazon Linux 2023/4.3.2
Running version
-
Platform state
Supported

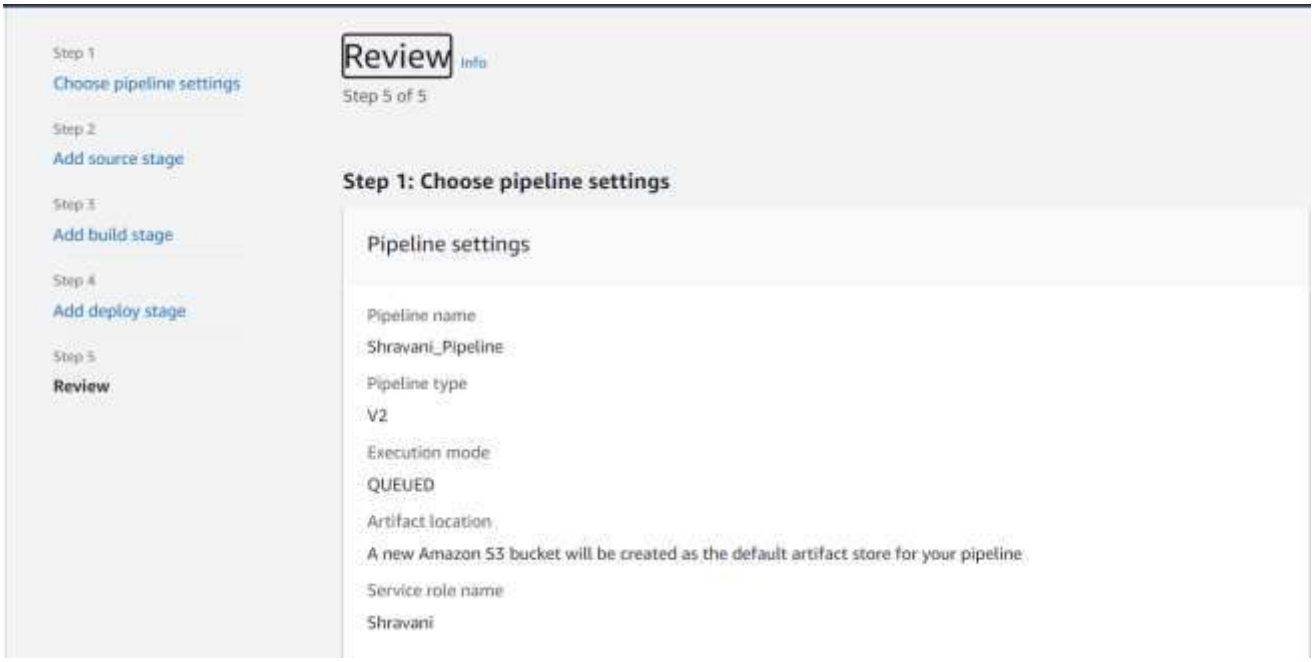

Environment successfully launched.
Events (12) Info
Filter events by text, property or value
1
Time
Type
Details
August 18, 2024 11:54:47 (UTC+5:30)
INFO
Successfully launched environment: ShravaniR0212-env
August 18, 2024 11:54:46 (UTC+5:30)
INFO
Application available at ShravaniR0212-env.eba-czwnn3pm.us-east-1.elasticbeanstalk.com.
August 18, 2024 11:54:41 (UTC+5:30)
INFO
Environment health has been set to GREEN
August 18, 2024 11:54:41 (UTC+5:30)
INFO
Adding instance 'i-0705735f3b870a664' to your environment.
August 18, 2024 11:54:41 (UTC+5:30)
INFO
Added EC2 instance 'i-0705735f3b870a664' to Auto Scaling Group 'awseb-e-umxqsuzxpa-stack-AWSEBAutoScalingGroup-Q5teraMKcurJ'.
August 18, 2024 11:54:30 (UTC+5:30)
INFO
Instance deployment completed successfully.
August 18, 2024 11:54:26 (UTC+5:30)
INFO
Instance deployment: You didn't include a 'composer.json' file in your source bundle. The deployment didn't install Composer dependencies.

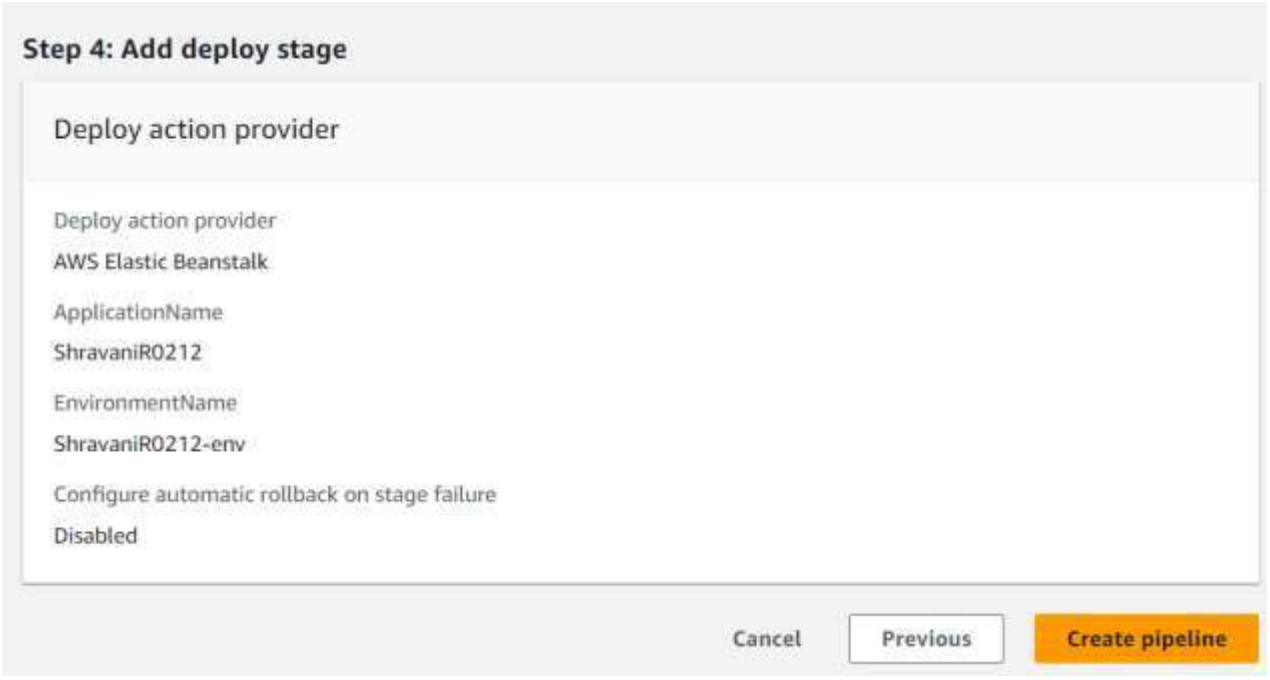

Step 1
Choose pipeline settings
Step 2
Add source stage
Step 3
Add build stage
Step 4
Add deploy stage
Step 5
Review
Choose pipeline settings Info
Step 1 of 5
Pipeline settings
Pipeline name
Enter the pipeline name. You cannot edit the pipeline name after it is created.
Shravani_Pipeline
No more than 100 characters
Pipeline type
You can no longer create V1 pipelines through the console. We recommend you use the V2 pipeline type with improved release safety, pipeline triggers, parameterized pipelines, and a new billing model.
Execution mode
Choose the execution mode for your pipeline. This determines how the pipeline is run.
Superseded
A more recent execution can overtake an older one. This is the default.
Queued (Pipeline type V2 required)
Executions are processed one by one in the order that they are queued.
Parallel (Pipeline type V2 required)
Executions don't wait for other runs to complete before starting or finishing.

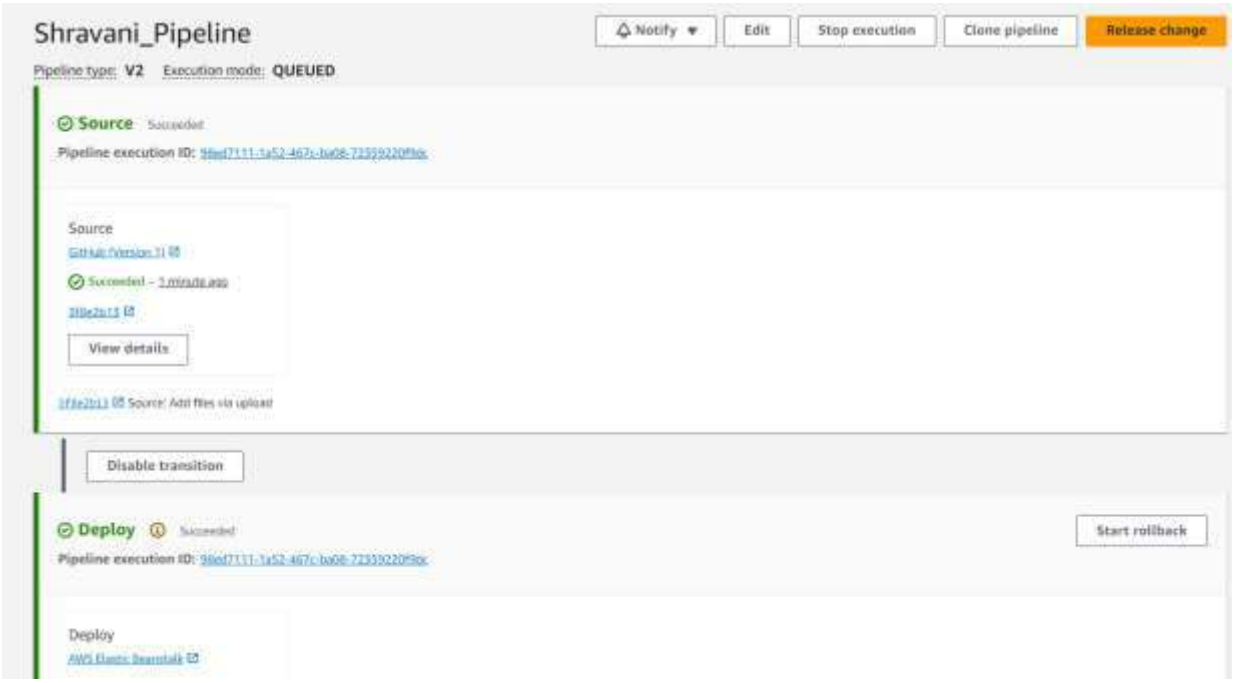

Developer Tools > CodePipeline > Pipelines > Create new pipeline
Step 1
Choose pipeline settings
Step 2
Add source stage
Step 3
Add build stage
Step 4
Add deploy stage
Step 5
Review
Add source stage Info
Step 2 of 5
Source
Source provider
This is where you stored your input artifacts for your pipeline. Choose the provider and then provide the connection details.
GitHub (Version 1)
Grant AWS CodePipeline access to your GitHub repository. This allows AWS CodePipeline to upload commits from GitHub to your pipeline.
Connected
You have successfully configured the action with the provider.

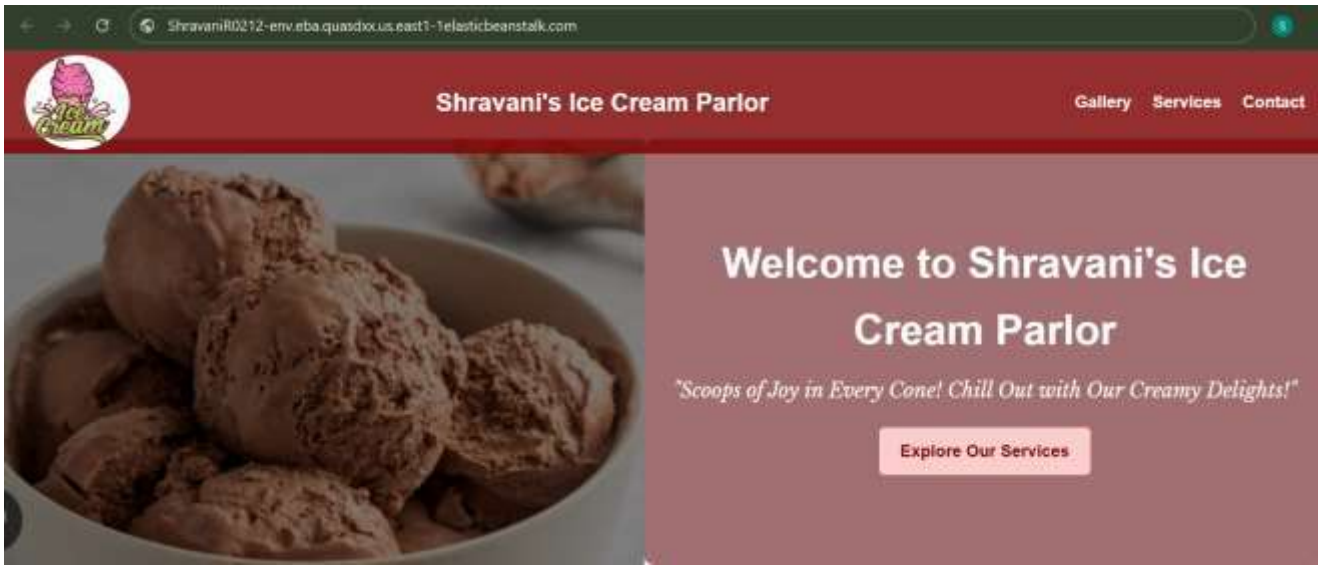

The GitHub (Version 1) action is not recommended
The selected action uses OAuth apps to access your GitHub repository. This is no longer the recommended method. Instead, choose the GitHub (Version 2) action to access your repository by creating a connection. Connections use GitHub Apps to manage authentication and can be shared with other resources. Learn more
Repository
ShravaniR2412/IP_A5SG2
Branch
main
main
Choose a detection mode to automatically start your pipeline when a change occurs in the source code.
GitHub webhooks (recommended)
Use webhooks in GitHub to automatically start my pipeline when a change occurs
AWS CodePipeline
Use AWS CodePipeline to check periodically for changes
Cancel
Previous
Next


Deploy
Deploy provider
Choose how you deploy to instances. Choose the provider, and then provide the configuration details for that provider.
AWS Elastic Beanstalk
Region
US East (N. Virginia)
Input artifacts
Choose an input artifact for this action. Learn more
SourceArtifact
No more than 100 characters
Application name
Choose an application that you have already created in the AWS Elastic Beanstalk console. Or create an application in the AWS Elastic Beanstalk console and then return to this task.
ShravaniR0212
Environment name
Choose an environment that you have already created in the AWS Elastic Beanstalk console. Or create an environment in the AWS Elastic Beanstalk console and then return to this task.
ShravaniR0212-env
Configure automatic rollback on stage failure

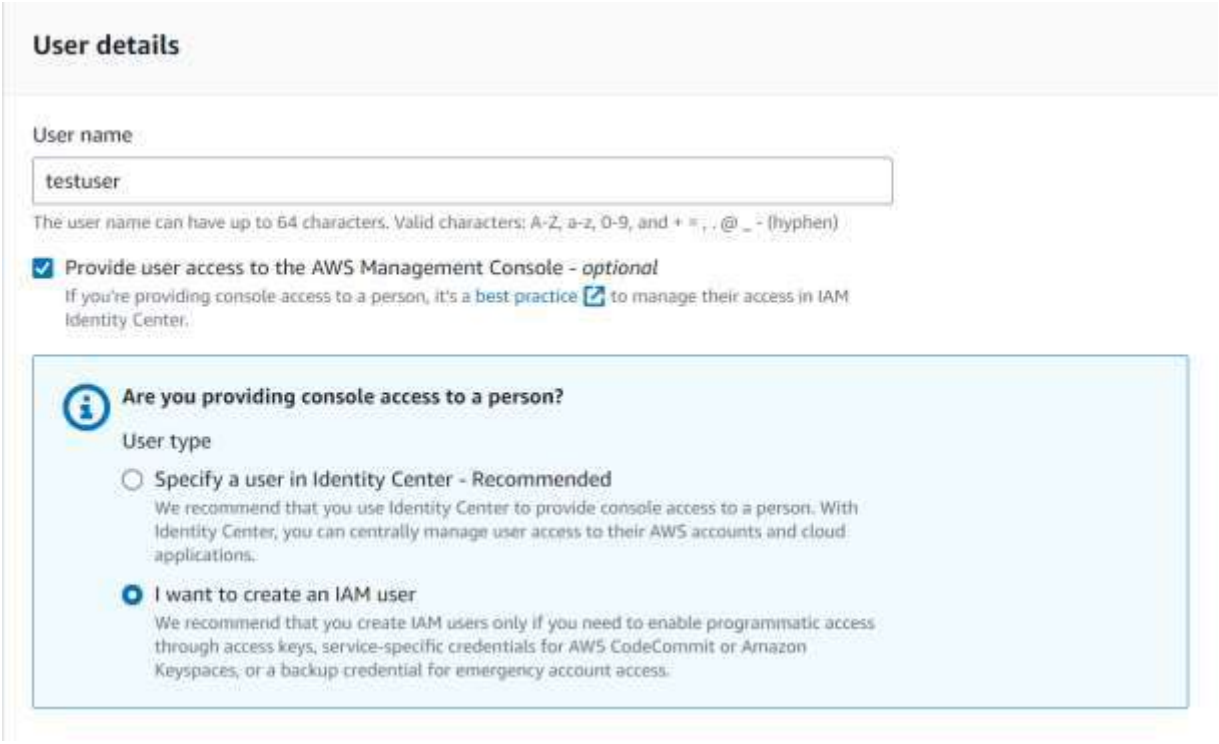

Step 1
Choose pipeline settings
Step 2
Add source stage
Step 3
Add build stage
Step 4
Add deploy stage
Step 5
Review
Review Info
Step 5 of 5
Step 1: Choose pipeline settings
Pipeline settings
Pipeline name
Shravani_Pipeline
Pipeline type
V2
Execution mode
QUEUED
Artifact location
A new Amazon S3 bucket will be created as the default artifact store for your pipeline
Service role name
Shravani

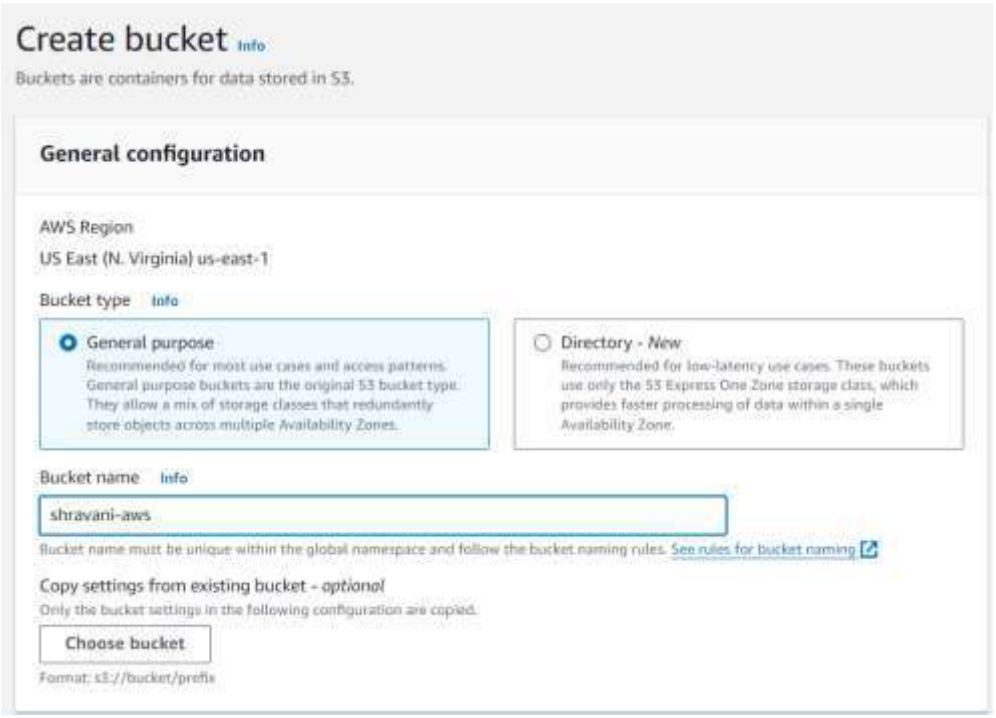

Step 4: Add deploy stage
Deploy action provider
Deploy action provider
AWS Elastic Beanstalk
ApplicationName
ShravaniR0212
EnvironmentName
ShravaniR0212-env
Configure automatic rollback on stage failure
Disabled
Cancel
Previous
Create pipeline


Shravani_Pipeline
Notify
Edit
Stop execution
Clone pipeline
Release change
Pipeline type: V2
Execution mode: QUEUED
Source
Pipeline execution ID:
Source
View details
Disable transition
Deploy
Pipeline execution ID:
Start rollback
Deploy


ShravaniR0212-env.eba.quasdxx.us.east1-1elasticbeanstalk.com
Shravani's Ice Cream Parlor
Gallery
Services
Contact
Welcome to Shravani's Ice Cream Parlor
"Scoops of Joy in Every Cone! Chill Out with Our Creamy Delights!"
Explore Our Services

USING S3 BUCKET





Files and folders (13 Total, 3.6 MB)

Find by name

Name	Folder	Type	Size	Status	Error
facebook.sv...	-	image/svg+...	283.0 B	Succeeded	-
hero.png	-	image/png	439.9 KB	Succeeded	-
img1.jpg	-	image/jpeg	122.6 KB	Succeeded	-
img2.jpg	-	image/jpeg	8.7 KB	Succeeded	-
img3.jpg	-	image/jpeg	98.1 KB	Succeeded	-
img4.jpg	-	image/jpeg	87.2 KB	Succeeded	-
index.html	-	text/html	3.2 KB	Succeeded	-
instagram.s...	-	image/svg+...	566.0 B	Succeeded	-

Edit static website hosting Info

Static website hosting

Use this bucket to host a website or redirect requests. Learn more

Static website hosting

- ○ Disable
- ● Enable

Hosting type

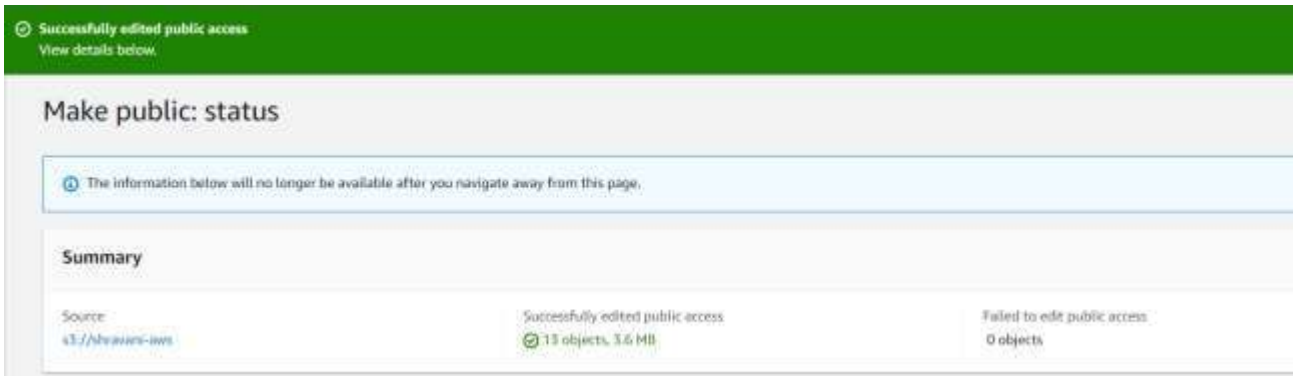
- ● Host a static website
  Use the bucket endpoint as the web address. Learn more
- ○ Redirect requests for an object
  Redirect requests to another bucket or domain. Learn more

For your customers to access content at the website endpoint, you must make all your content publicly readable. To do so, you can edit the S3 Block Public Access settings for the bucket. For more information, see Using Amazon S3 Block Public Access

Index document

Specify the home or default page of the website

index.html


Successfully edited public access
View details below
Make public: status
The information below will no longer be available after you navigate away from this page.
Summary
Source
Successfully edited public access
13 objects, 3.6 MB
Failed to edit public access
0 objects


shravani-aws.s3.amazonaws.com/index.html
Shravani's Ice Cream Parlor
Gallery
Services
Contact
Welcome to Shravani's Ice Cream Parlor
"Scoops of Joy in Every Cone! Chill Out with Our Creamy Delights!"
Explore Our Services

ADVANCE DEVOPS EXP-3

Aim: To understand the Kubernetes Cluster Architecture, install and Spin Up Kubernetes Cluster on Linux Machines/Cloud Platforms.

Theory:
Container-based microservices architectures have revolutionized how development and operations teams test and deploy modern software. Containers allow companies to scale and deploy applications more efficiently, but they also introduce new challenges, adding complexity by creating a whole new infrastructure ecosystem.

Today, both large and small software companies are deploying thousands of container instances daily. Managing this level of complexity at scale requires advanced tools. Enter Kubernetes.
Originally developed by Google, Kubernetes is an open-source container orchestration platform designed to automate the deployment, scaling, and management of containerized applications. Kubernetes has quickly become the de facto standard for container orchestration and is the flagship project of the Cloud Native Computing Foundation (CNCF), supported by major players like Google, AWS, Microsoft, IBM, Intel, Cisco, and Red Hat.

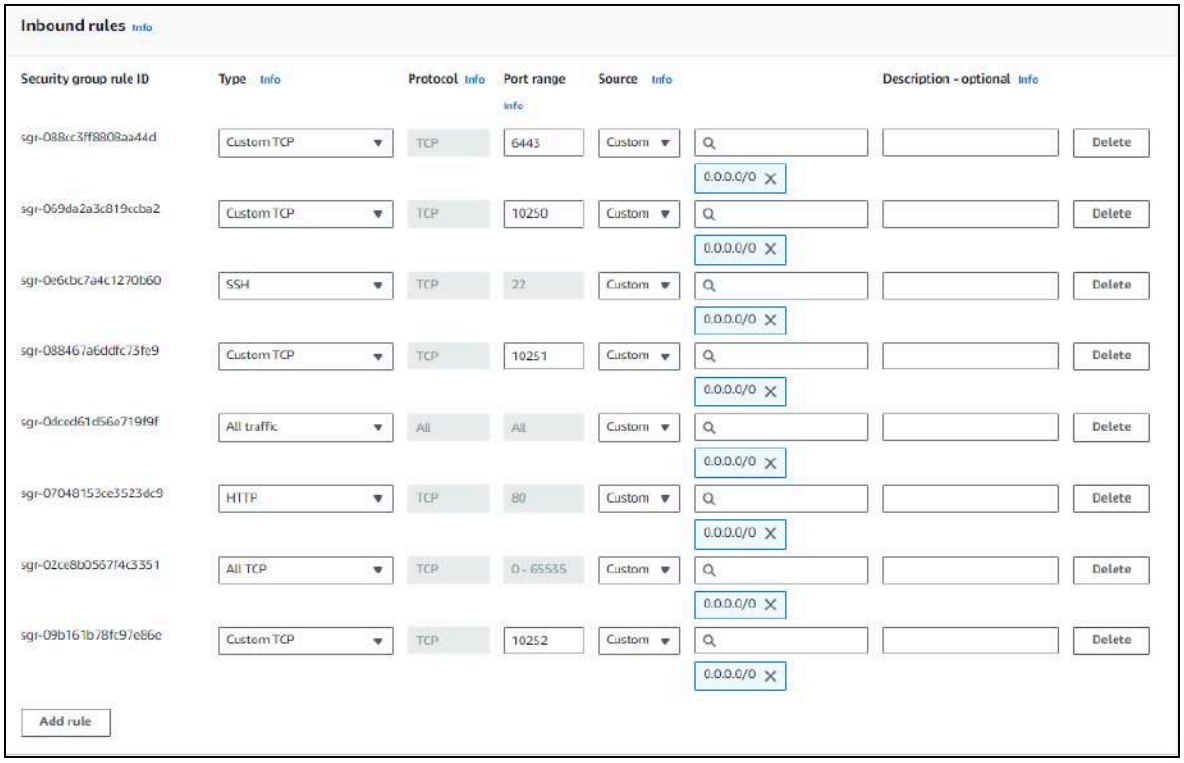
Kubernetes simplifies the deployment and operation of applications in a microservice architecture by providing an abstraction layer over a group of hosts. This allows development teams to deploy their applications while Kubernetes takes care of key tasks, including:

- ● Managing resource consumption by applications or teams
- ● Distributing application load evenly across the infrastructure
- ● Automatically load balancing requests across multiple instances of an application
- ● Monitoring resource usage to prevent applications from exceeding resource limits and
- automatically restarting them if needed
- ● Moving application instances between hosts when resources are low or if a host fails
- ● Automatically utilizing additional resources when new hosts are added to the cluster
- ● Facilitating canary deployments and rollbacks with ease
- Necessary Requirements:
- ● EC2 Instance: The experiment required launching a t2.medium EC2 instance with 2 CPUs, as
- Kubernetes demands sufficient resources for effective functioning.
- ● Minimum Requirements:

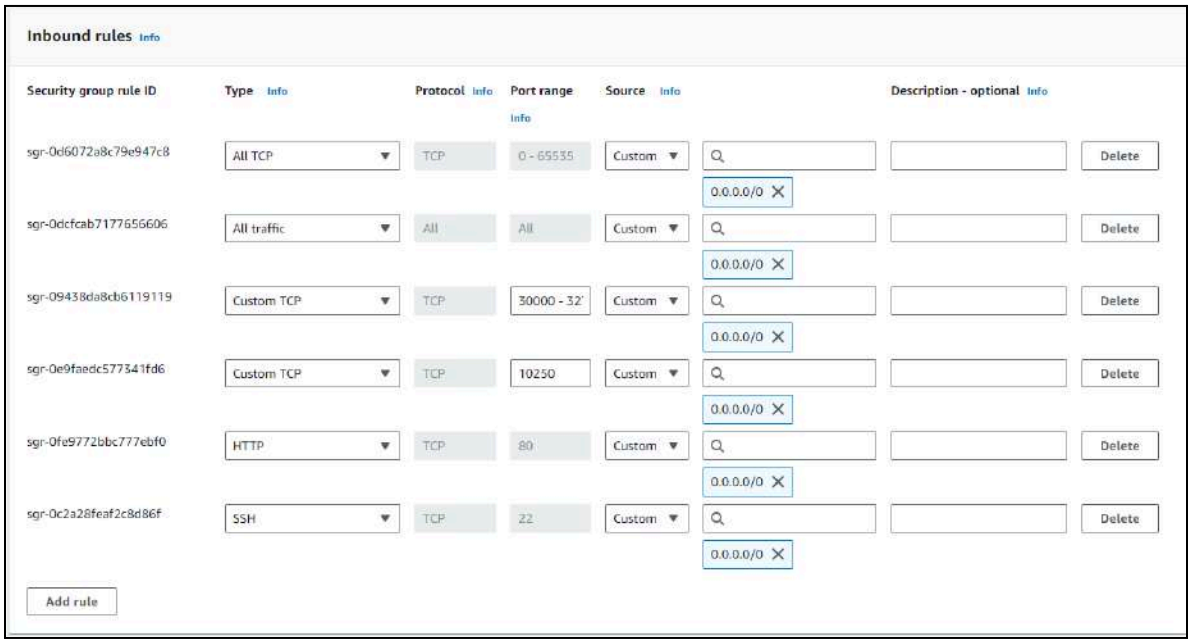
○ Instance Type: t2.medium
○ CPUs: 2
○ Memory: Adequate for container orchestration.
This ensured that the Kubernetes cluster had the necessary resources to function smoothly

Step 1: Create 2 Security Groups for Master and Nodes and add the following inbound rules in those groups

Master:

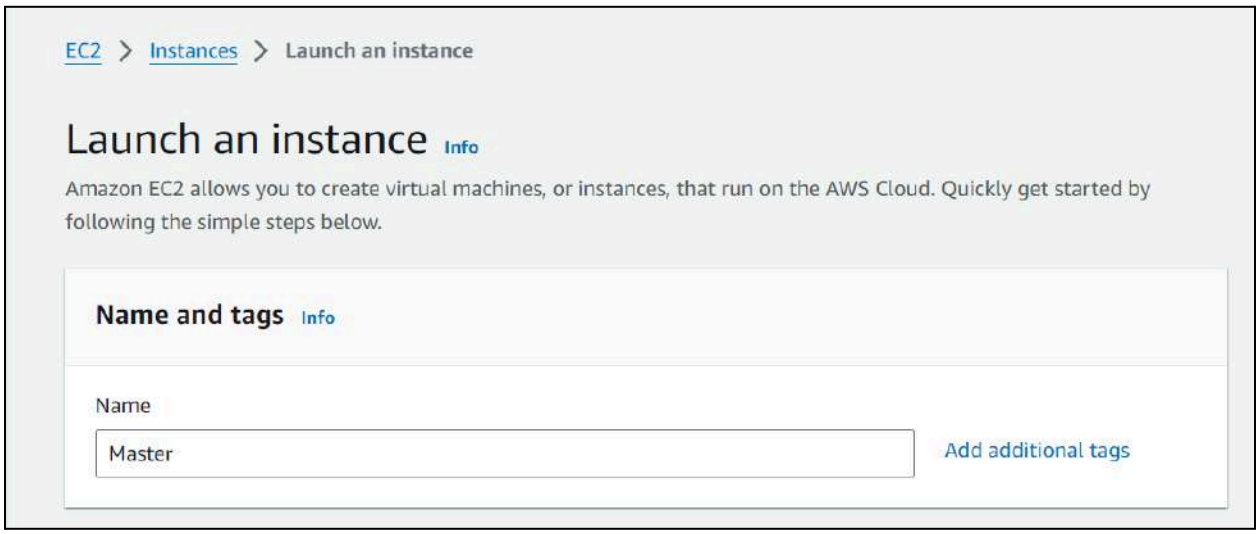


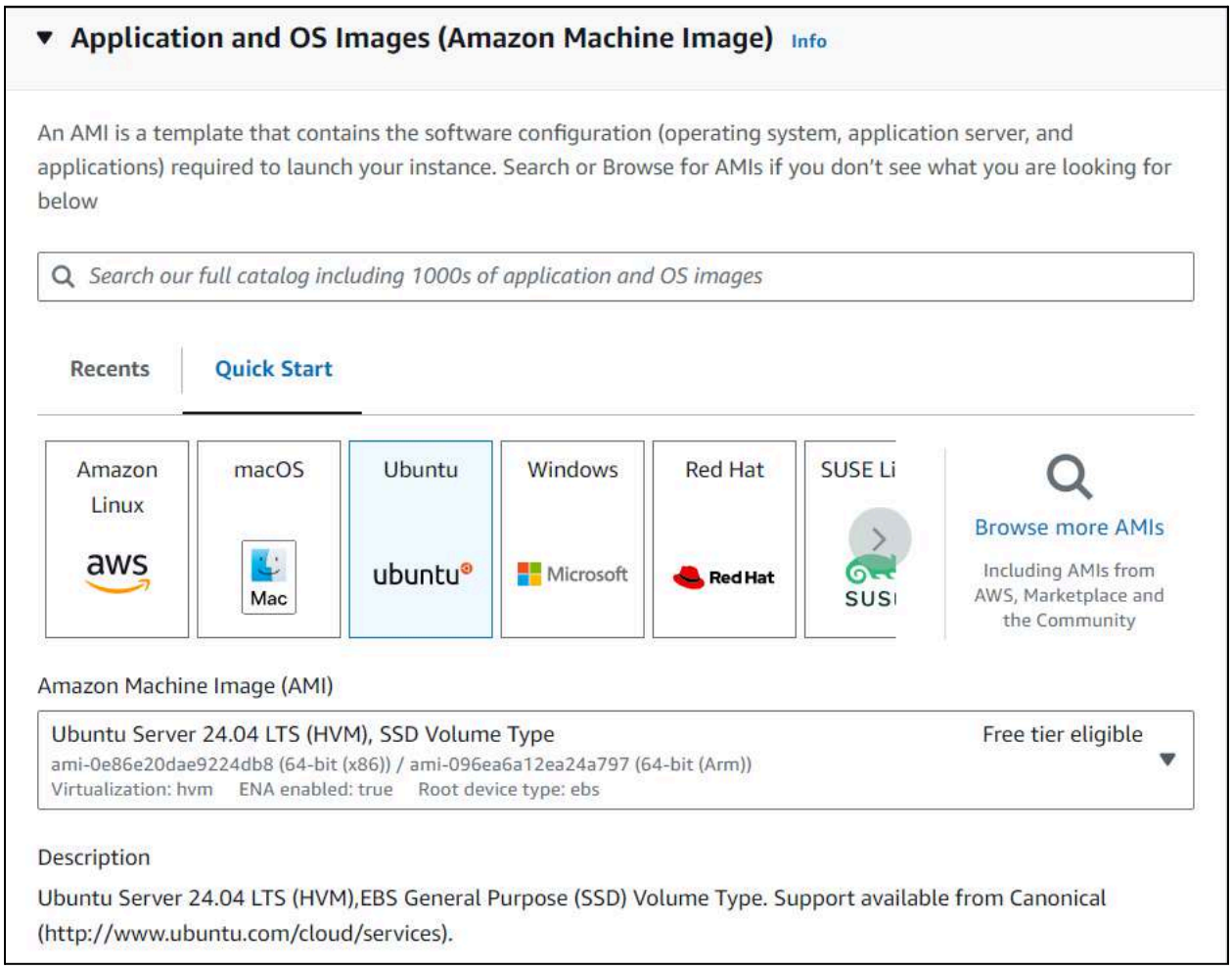
Node:

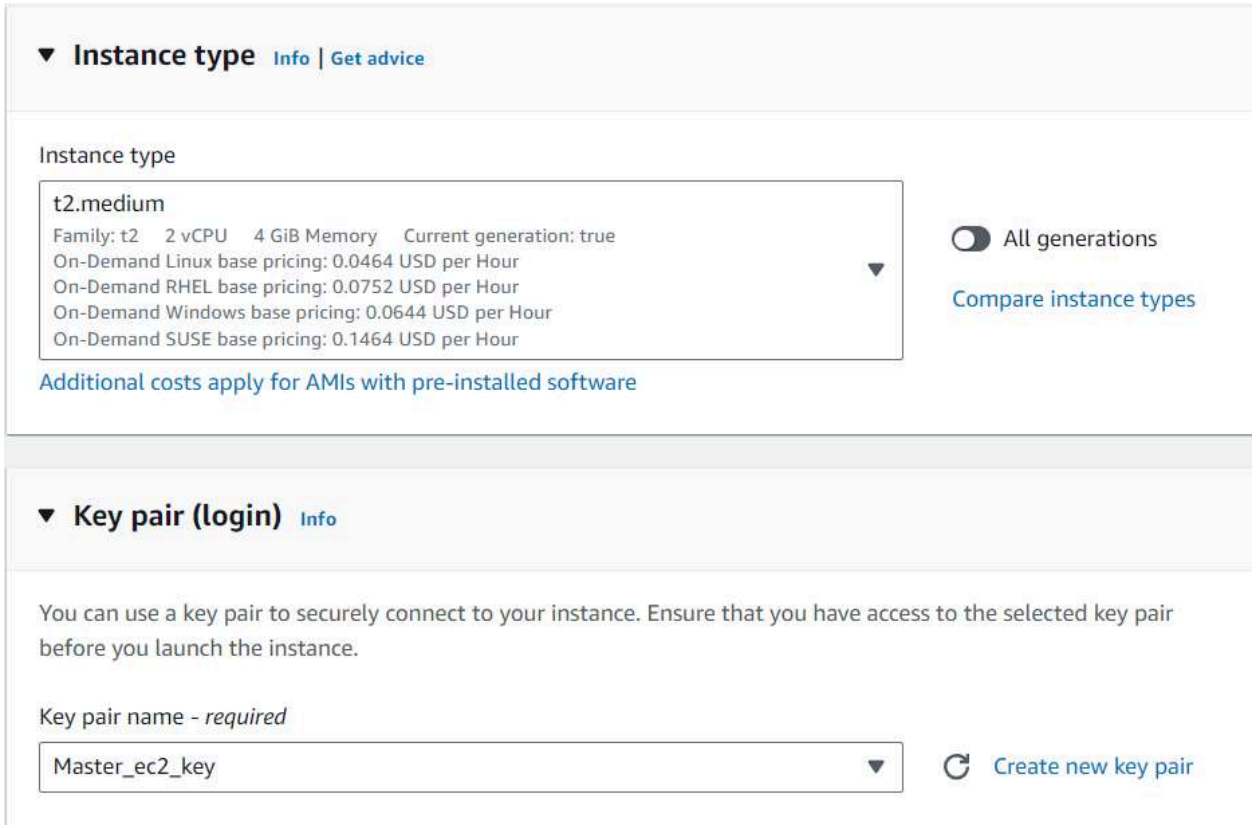


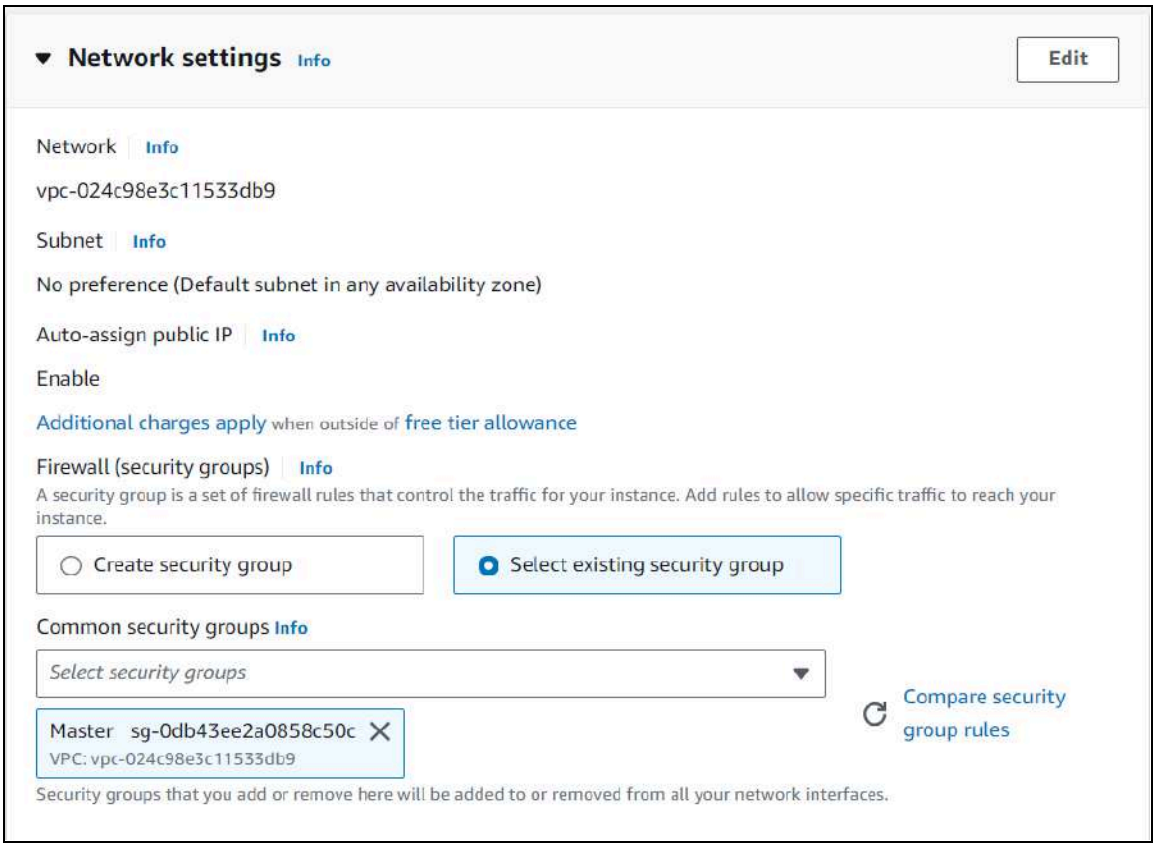
Step 2: Log in to your AWS Academy/personal account and launch 3 new Ec2 Instances(1 for Master and 2 for Node).Select Ubuntu as AMI and t2.medium as Instance Type and create a key of type RSA with .pem extension and move the downloaded key to the new folder.We can use 2 Different keys, 1 for Master and 1 for Node. Also Select Security Groups from the existing.

Master:

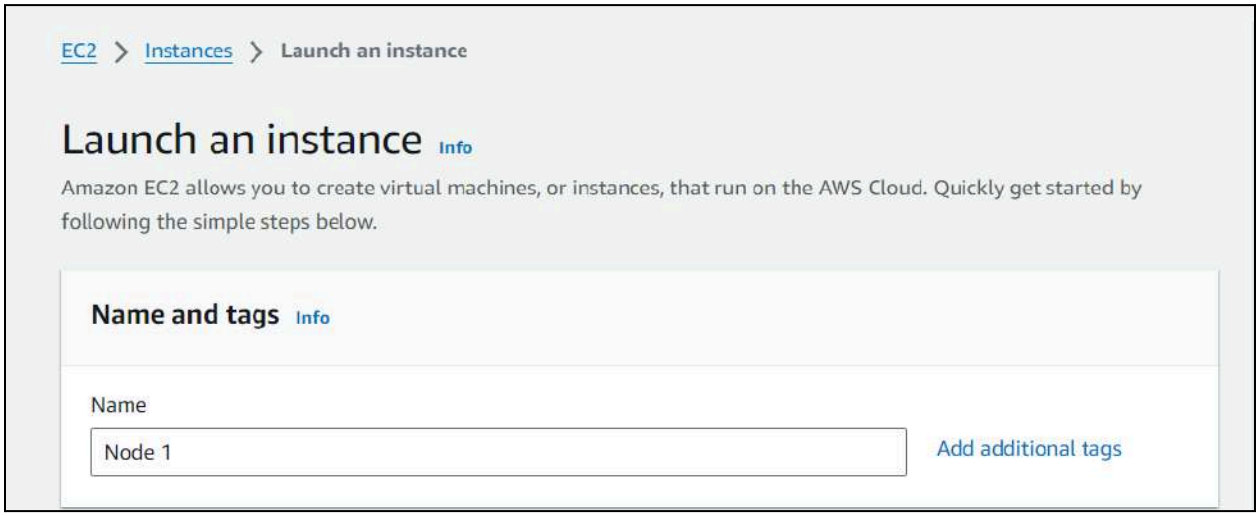


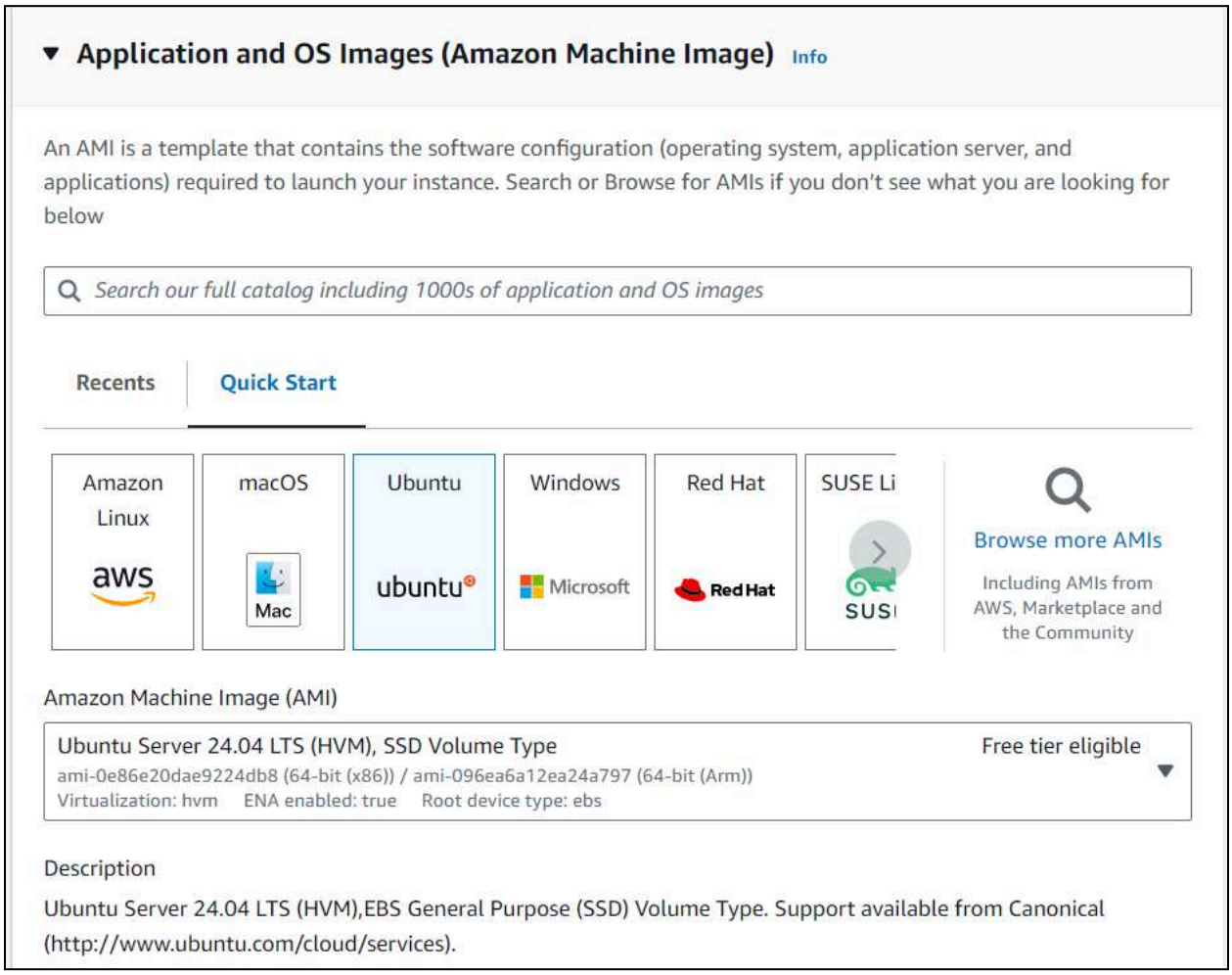



Instance type Info | Get advice
Instance type
t2.medium
Family: t2 2 vCPU 4 GiB Memory Current generation: true
On-Demand Linux base pricing: 0.0464 USD per Hour
On-Demand RHEL base pricing: 0.0752 USD per Hour
On-Demand Windows base pricing: 0.0644 USD per Hour
On-Demand SUSE base pricing: 0.1464 USD 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
Master_ec2_key
Create new key pair


Network settings Info
Edit
Network Info
vpc-024c98e3c11533db9
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
Common security groups Info
Select security groups
Master sg-0db43ee2a0858c50c
VPC: vpc-024c98e3c11533db9
Compare security group rules
Security groups that you add or remove here will be added to or removed from all your network interfaces.

Node:


EC2 > Instances > Launch an instance
Launch an instance Info
Amazon EC2 allows you to create virtual machines, or instances, that run on the AWS Cloud. Quickly get started by following the simple steps below.
Name and tags Info
Name
Node 1
Add additional tags


Application and OS Images (Amazon Machine Image) Info
An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. Search or Browse for AMIs if you don't see what you are looking for below
Search our full catalog including 1000s of application and OS images
Recents
Quick Start
Amazon Linux
macOS
Ubuntu
Windows
Red Hat
SUSE Li
Browse more AMIs
Including AMIs from AWS, Marketplace and the Community
Amazon Machine Image (AMI)
Ubuntu Server 24.04 LTS (HVM), SSD Volume Type
Free tier eligible
ami-0e86e20dae9224db8 (64-bit (x86)) / ami-096ea6a12ea24a797 (64-bit (Arm))
Virtualization: hvm ENA enabled: true Root device type: ebs
Description
Ubuntu Server 24.04 LTS (HVM),EBS General Purpose (SSD) Volume Type. Support available from Canonical (http://www.ubuntu.com/cloud/services).

Network settings Info

Edit

Network Info

vpc-024c98e3c11533db9

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

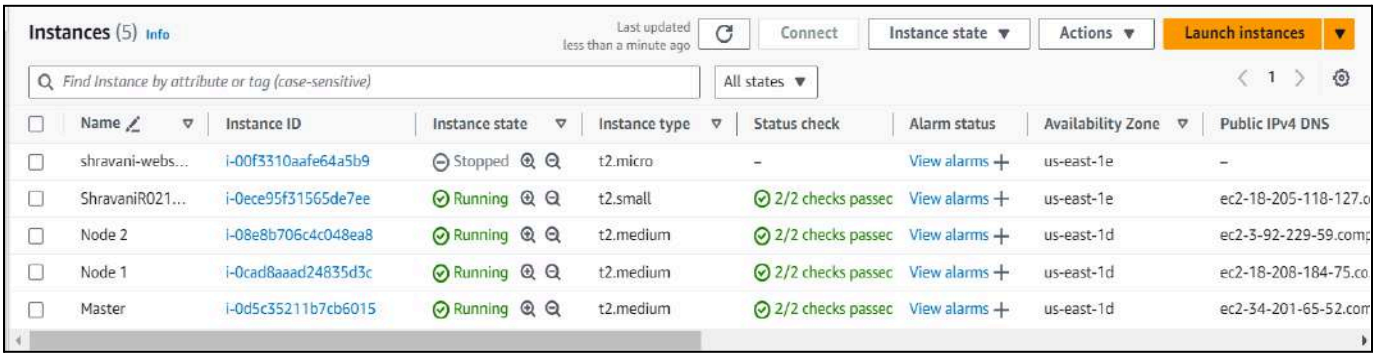
Common security groups Info

Select security groups

Nodes sg-019d7373dd3c972e8 ✕
VPC: vpc-024c98e3c11533db9

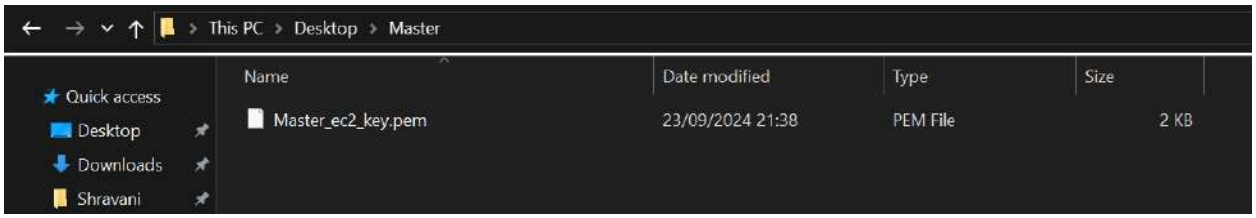
Compare security group rules

Security groups that you add or remove here will be added to or removed from all your network interfaces.

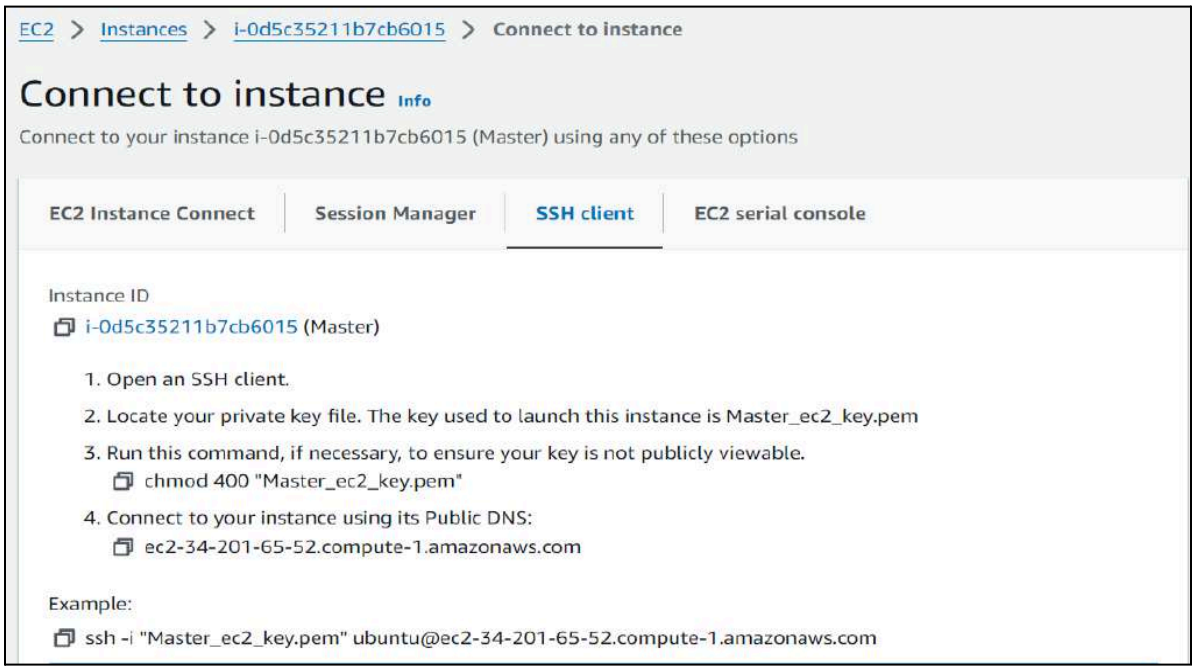


Step 3: Connect the instance and navigate to SSH client and copy the example command. Now open the folder in the terminal 3 times for Master, Node1 & Node 2 where our .pem key is stored and paste the Example command from ssh client (starting with ssh -i .....) in the terminal.

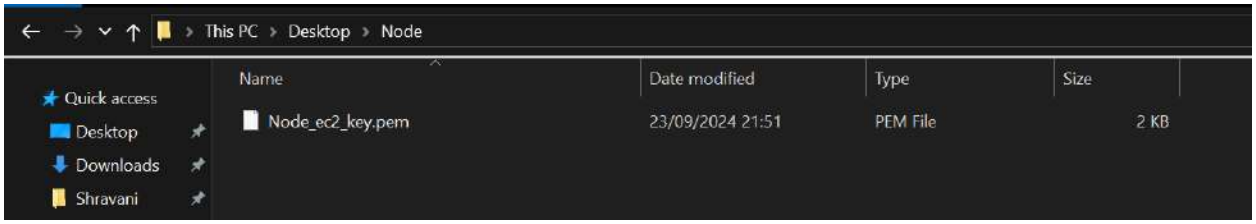
Downloaded Key:



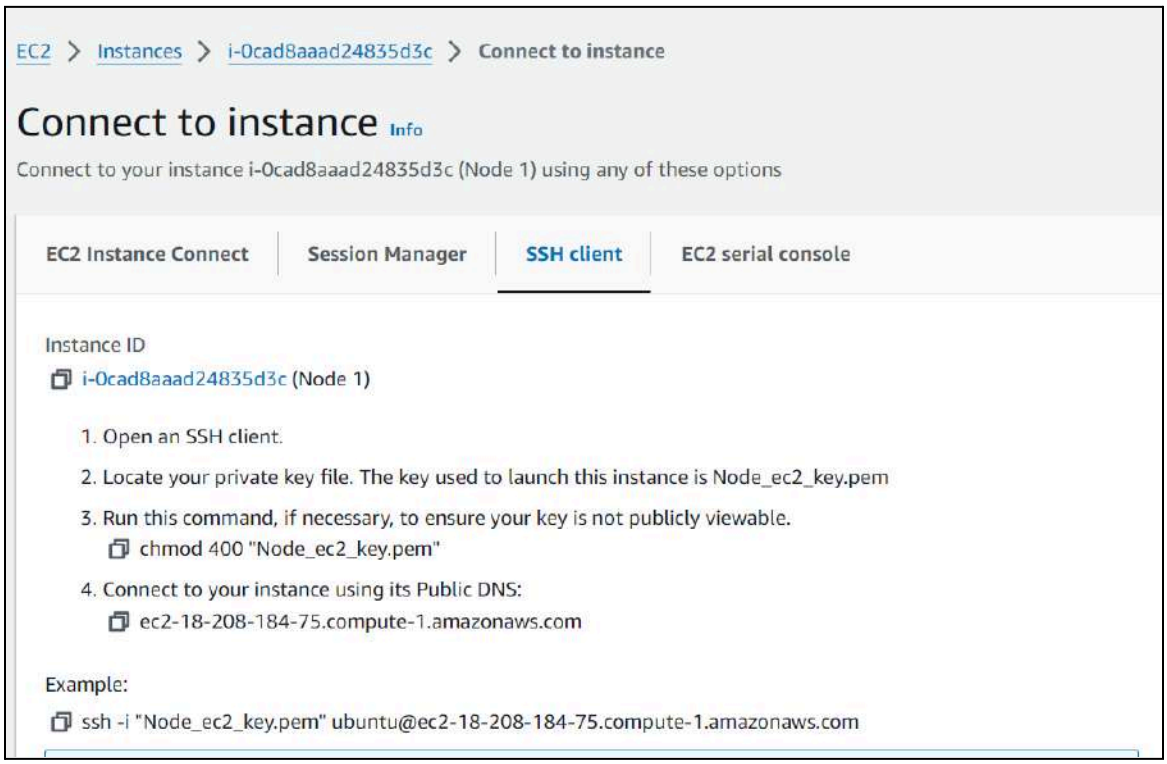
Master:




```
C:\Users\Shravani\Desktop\Master>ssh -i "Master_ec2_key.pem" ubuntu@ec2-34-201-65-52.compute-1.amazonaws.com
Welcome to Ubuntu 24.04 LTS (GNU/Linux 6.8.0-1012-aws x86_64)

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

 System information as of Mon Sep 23 16:43:43 UTC 2024

  System load:  0.0               Processes:             116
  Usage of /:   22.9% of 6.71GB   Users logged in:       0
  Memory usage: 5%                IPv4 address for enX0: 172.31.84.221
  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
```

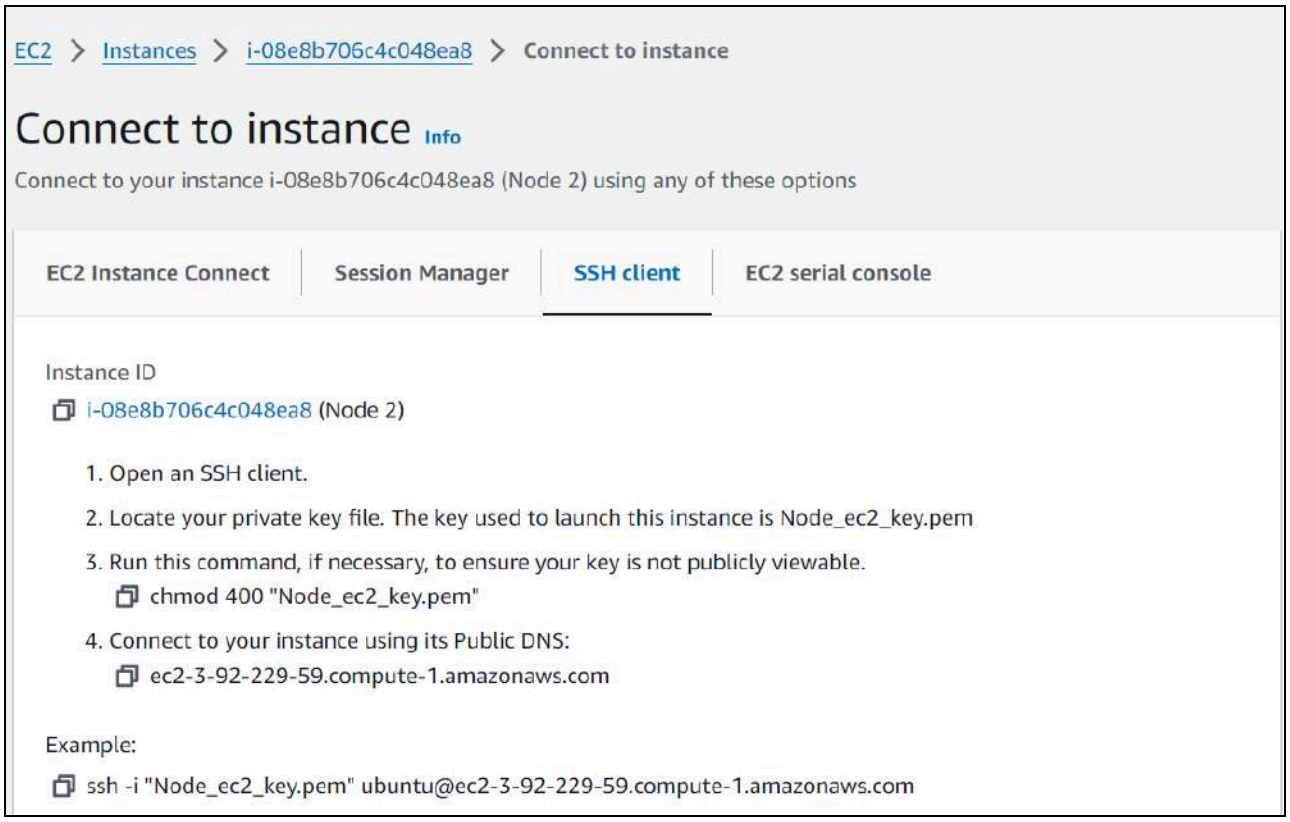


Node 1:




```
C:\Users\Shravani\Desktop\Node>ssh -i "Node_ec2_key.pem" ubuntu@ec2-18-208-184-75.compute-1.amazonaws.com
The authenticity of host 'ec2-18-208-184-75.compute-1.amazonaws.com (18.208.184.75)' can't be established.
ECDSA key fingerprint is SHA256:Mt3R8xcNRQpug+O8YjlPo+4OyaB1xn/43dC9MQA87+A.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-18-208-184-75.compute-1.amazonaws.com,18.208.184.75' (ECDSA) to the list of known hosts.
Welcome to Ubuntu 24.04 LTS (GNU/Linux 6.8.0-1012-aws x86_64)

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

 System information as of Mon Sep 23 16:48:32 UTC 2024

  System load:  0.08              Processes:             113
  Usage of /:   22.7% of 6.71GB   Users logged in:       0
  Memory usage: 5%                IPv4 address for enX0: 172.31.95.119
  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
```

Node 2:




```
C:\Users\Shravani\Desktop\Node>ssh -i "Node_ec2_key.pem" ubuntu@ec2-3-92-229-59.compute-1.amazonaws.com
The authenticity of host 'ec2-3-92-229-59.compute-1.amazonaws.com (64:ff9b::35c:e53b)' can't be established.
ECDSA key fingerprint is SHA256:jKzi3rD9OgtRdm2AJ5oS4Ndayn4cxMLRUQGQVXEMsck.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-3-92-229-59.compute-1.amazonaws.com,64:ff9b::35c:e53b' (ECDSA) to the list of known hosts.
Welcome to Ubuntu 24.04 LTS (GNU/Linux 6.8.0-1012-aws x86_64)

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

 System information as of Mon Sep 23 16:50:15 UTC 2024

  System load:  0.0               Processes:             114
  Usage of /:   22.7% of 6.71GB   Users logged in:       0
  Memory usage: 5%                IPv4 address for enX0: 172.31.80.164
  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
```

Step 4: Run on Master,Node 1,and Node 2 the below commands to install and setup Docker in Master, Node1, and Node2.

- curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
- curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo tee /etc/apt/trusted.gpg.d/docker.gpg > /dev/null
- sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable"

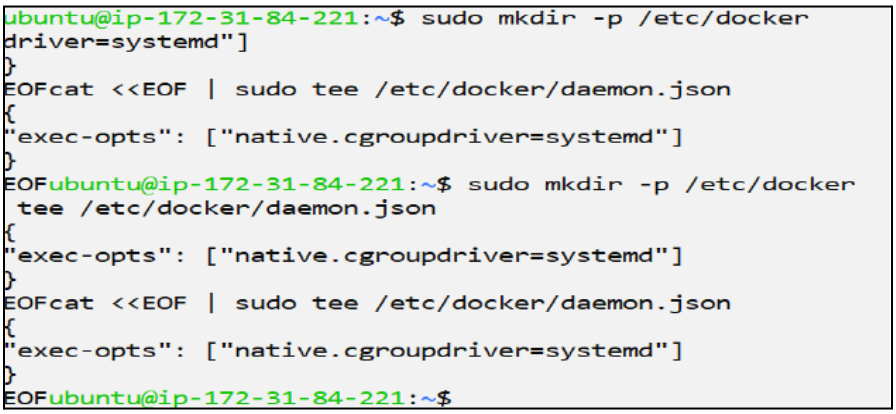
```
ubuntu@ip-172-31-84-221:~$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add
Warning: apt-key is deprecated. Manage keyring files in trusted.gpg.d instead (see apt-key(8)).
OK
ubuntu@ip-172-31-84-221:~$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo tee/etc/apt/
sudo: tee/etc/apt/trusted.gpg.d/docker.gpg: command not found
ubuntu@ip-172-31-84-221:~$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo tee
gpg.d/docker.gpg > /dev/null-----BEGIN PGP PUBLIC KEY BLOCK-----

mQINBFit2ioBEADhWpZ8/wvZ6hUTiXOwQHXMAlaFHcPH9hAtr4F1y2+OYdbtMuth
lqqwp028AqyY+PRfVMtSYMbjuQuu5byyKR01BbqYhuS3jtqQmljZ/bJvXqnmiVXh
38UuLa+z077PxyxQhu5BbqntTPQMfiyqEiU+BKbq2WmANUKQf+1AmZY/IruOXbnq
L4C1+gJ8vfmXQt99npCaxEjaNRVYfOS8QcixNzHUYnb6emjlANyEVlZzeqo7XKl7
UrwV5inawTSzWNvtjEjj4nJL8NsLwscpLPQUhTQ+7BbQXAwAmeHCUTQIvvWXqw0N
cmhh4HgeQscQHYgOJjjDVfoY5MucvglbIgCqfzAHW9jxmRL4qbMZj+b1XoePEtht
ku4bIQN1X5P07fNWzlgaRL5Z4POXDDZTlIQ/El58j9kp4bnWRCJW0lya+f8ocodo
vZZ+Doi+fy4D5ZGrL4XEcIQP/Lv5uFyf+kQtl/94VFYVJOleAv8W92KdgDkhTcTD
G7c0tIkVEKNUq48b3aQ64NOZQW7fVjfoKwEZdOqPE72Pa45jrZzvUFxSpdiNk2tZ
XYukHjlxxEgBdC/J3cMMNRE1F4NCA3ApfV1Y7/hTeOnmDuDYwr9/obA8t016Yljj
q5rdkywPf4JF8mXUW5eCN1vAFHxeg9ZWemhBtQmGxXnw9M+z6hWwc6ahmwARAQAB
tCtEb2NrZXIgUmVsZWFzZSAoQ0UgZGViKSA8ZG9ja2VyQGRvY2tlci5jb20+iQI3
```

```
Get:43 http://security.ubuntu.com/ubuntu noble-security/universe Translation-en [113 kB]
Get:44 http://security.ubuntu.com/ubuntu noble-security/universe amd64 Components [8632 B]
Get:45 http://security.ubuntu.com/ubuntu noble-security/universe amd64 c-n-f Metadata [10.1 kB]
Get:46 http://security.ubuntu.com/ubuntu noble-security/restricted amd64 Packages [353 kB]
Get:47 http://security.ubuntu.com/ubuntu noble-security/restricted Translation-en [68.1 kB]
Get:48 http://security.ubuntu.com/ubuntu noble-security/restricted amd64 c-n-f Metadata [428 B]
Get:49 http://security.ubuntu.com/ubuntu noble-security/multiverse amd64 Packages [10.9 kB]
Get:50 http://security.ubuntu.com/ubuntu noble-security/multiverse Translation-en [2808 B]
Get:51 http://security.ubuntu.com/ubuntu noble-security/multiverse amd64 Components [208 B]
Get:52 http://security.ubuntu.com/ubuntu noble-security/multiverse amd64 c-n-f Metadata [344 B]
Fetched 29.1 MB in 4s (7159 kB/s)
Reading package lists... Done
W: https://download.docker.com/linux/ubuntu/dists/noble/InRelease: Key is stored in legacy trusted.gpg keyring
key(8) for details.
ubuntu@ip-172-31-84-221:~$
```

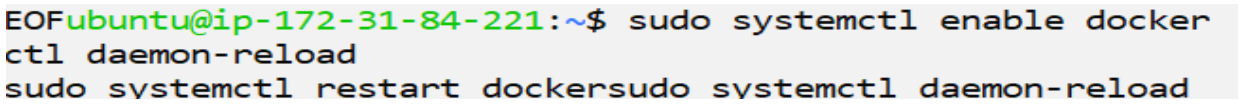
- sudo apt-get update
- sudo apt-get install -y docker-c

```
ubuntu@ip-172-31-84-221:~$ sudo apt-get update
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 https://download.docker.com/linux/ubuntu noble InRelease
Hit:5 http://security.ubuntu.com/ubuntu noble-security InRelease
Reading package lists... Done
W: https://download.docker.com/linux/ubuntu/dists/noble/InRelease: Key is stored in legacy trusted.gpg keyring
key(8) for details.
ubuntu@ip-172-31-84-221:~$
```

● sudo mkdir -p /etc/docker
cat <<EOF | sudo tee /etc/docker/daemon.json
{
"exec-opts": ["native.cgroupdriver=systemd"]
}
EOF



● sudo systemctl enable docker
● sudo systemctl daemon-reload
● sudo systemctl restart docker



Step 5: Run the below command to install Kubernetes.

● curl -fsSL https://pkgs.k8s.io/core:/stable:/v1.31/deb/Release.key | sudo gpg --dearmor -o /etc/apt/keyrings/kubernetes-apt-keyring.gpg

● echo 'deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg] https://pkgs.k8s.io/core:/stable:/v1.31/deb/ /' | sudo tee /etc/apt/sources.list.d/kubernetes.list

```
ubuntu@ip-172-31-84-221:~$ curl -fsSL https://pkgs.k8s.io/core:/stable:/v1.31/deb/Release.key | sudo gpg --dearmor -o
ngs/kubernetes-apt-keyring.gpg
echo 'deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg]
https://pkgs.k8s.io/core:/stable:/v1.31/deb/ /' | gpg: missing argument for option "-o"
sudo tee /etc/apt/sources.list.d/kubernetes.list


ubuntu@ip-172-31-84-221:~$ /etc/apt/keyrings/kubernetes-apt-keyring.gpg
-bash: /etc/apt/keyrings/kubernetes-apt-keyring.gpg: No such file or directory
ubuntu@ip-172-31-84-221:~$ echo 'deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg]
> https://pkgs.k8s.io/core:/stable:/v1.31/deb/ /' | sudo tee /etc/apt/sources.list.d/kubernetes.list
deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg]
https://pkgs.k8s.io/core:/stable:/v1.31/deb/ /
```

```
ubuntu@ip-172-31-84-221:~$ sudo apt-get install -y kubelet kubeadm kubectl
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  conntrack cri-tools kubernetes-cni
The following NEW packages will be installed:
  conntrack cri-tools kubeadm kubectl kubelet kubernetes-cni
0 upgraded, 6 newly installed, 0 to remove and 136 not upgraded.
Need to get 87.4 MB of archives.
After this operation, 335 MB of additional disk space will be used.
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 conntrack amd64 1:1.4.8-1ubuntu1 [37.9 kB]
Get:2 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.28/deb  cri-tools 1.28.0-1.1 [19.6 MB]
Get:3 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.28/deb  kubernetes-cni 1.2.0-2.1 [27.6 MB]
Get:4 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.28/deb  kubelet 1.28.14-2.1 [19.6 MB]
Get:5 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.28/deb  kubectl 1.28.14-2.1 [10.4 MB]
Get:6 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.28/deb  kubeadm 1.28.14-2.1 [10.1 MB]
Fetched 87.4 MB in 1s (77.5 MB/s)
```

```
ubuntu@ip-172-31-84-221:~$ sudo apt-mark hold kubelet kubeadm kubectl
kubelet set on hold.
kubeadm set on hold.
kubectl set on hold.
ubuntu@ip-172-31-84-221:~$
```

- sudo apt-get update
- sudo apt-get install -y kubelet kubeadm kubectl
- sudo apt-mark hold kubelet kubeadm kubectl

```
ubuntu@ip-172-31-84-221:~$ sudo apt-get update
Warning: The unit file, source configuration file or drop-ins of apt-news.service changed on disk. Run 'systemctl daemon-reload' to reload u
nits.
Warning: The unit file, source configuration file or drop-ins of esm-cache.service changed on disk. Run 'systemctl daemon-reload' to reload
units.
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://download.docker.com/linux/ubuntu noble InRelease
Get:6 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.31/deb  InRelease [1186 B]
Err:6 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.31/deb  InRelease
  The following signatures couldn't be verified because the public key is not available: NO_PUBKEY 234654DA9A296436
Reading package lists... Done
W: https://download.docker.com/linux/ubuntu/dists/noble/InRelease: Key is stored in legacy trusted.gpg keyring (/etc/apt/trusted.gpg), see t
he DEPRECATION section in apt-key(8) for details.
W: GPG error: https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.31/deb  InRelease: The following signatures co
uldn't be verified because the public key is not available: NO_PUBKEY 234654DA9A296436
E: The repository 'https://pkgs.k8s.io/core:/stable:/v1.31/deb  InRelease' is not signed.
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.
```

Err:7 https://packages.cloud.google.com/apt kubernetes-xenial Release 404 Not Found [IP: 64.233.180.139 443]

- sudo rm /etc/apt/sources.list.d/kubernetes.list
- sudo nano /etc/apt/sources.list.d/kubernetes.list
- deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg] https://pkgs.k8s.io/core:/stable:/v1.28/deb/ /

```
ubuntu@ip-172-31-84-221:~$ sudo systemctl enable --now kubelet
ubuntu@ip-172-31-84-221:~$ sudo apt-get install -y containerd
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following packages were automatically installed and are no longer required:
  docker-buildx-plugin docker-ce-cli docker-ce-rootless-extras docker-compose-plugin libltdl7 libslirp0 pigz slirp4netns
Use 'sudo apt autoremove' to remove them.
The following additional packages will be installed:
  runc
The following packages will be REMOVED:
  containerd.io docker-ce
The following NEW packages will be installed:
  containerd runc
0 upgraded, 2 newly installed, 2 to remove and 136 not upgraded.
Need to get 47.2 MB of archives.
After this operation, 53.1 MB disk space will be freed.
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 runc amd64 1.1.12-0ubuntu3.1 [8599 kB]
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 containerd amd64 1.7.12-0ubuntu4.1 [38.6 MB]
Fetched 47.2 MB in 1s (90.1 MB/s)
```

- sudo mkdir -p /etc/containerd
- sudo containerd config default | sudo tee /etc/containerd/config.toml

```
ubuntu@ip-172-31-84-221:~$ sudo containerd config default | sudo tee /etc/containerd/config.toml
disabled_plugins = []
imports = []
oom_score = 0
plugin_dir = ""
required_plugins = []
root = "/var/lib/containerd"
state = "/run/containerd"
temp = ""
version = 2

[cgroup]
  path = ""

[debug]
  address = ""
  format = ""
  gid = 0
  level = ""
  uid = 0

[grpc]
  address = "/run/containerd/containerd.sock"
  gid = 0
  max_recv_message_size = 16777216
  max_send_message_size = 16777216
  tcp_address = ""
  tcp_tls_ca = ""
  tcp_tls_cert = ""
  tcp_tls_key = ""
  uid = 0
```

- sudo systemctl restart containerd
- sudo systemctl enable containerd
- sudo systemctl status containerd

```
ubuntu@ip-172-31-84-221:~$ sudo systemctl restart containerd
ubuntu@ip-172-31-84-221:~$ sudo systemctl enable containerd
ubuntu@ip-172-31-84-221:~$ sudo systemctl status containerd
● containerd.service - containerd container runtime
     Loaded: loaded (/usr/lib/systemd/system/containerd.service; enabled; preset: enabled)
     Active: active (running) since Mon 2024-09-23 20:47:25 UTC; 14s ago
       Docs: https://containerd.io
   Main PID: 19202 (containerd)
      Tasks: 7
     Memory: 13.0M (peak: 13.8M)
        CPU: 113ms
     CGroup: /system.slice/containerd.service
             └─19202 /usr/bin/containerd

Sep 23 20:47:25 ip-172-31-84-221 containerd[19202]: time="2024-09-23T20:47:25.572213616Z" level=info
Sep 23 20:47:25 ip-172-31-84-221 containerd[19202]: time="2024-09-23T20:47:25.572255061Z" level=info
Sep 23 20:47:25 ip-172-31-84-221 containerd[19202]: time="2024-09-23T20:47:25.572281095Z" level=info
Sep 23 20:47:25 ip-172-31-84-221 containerd[19202]: time="2024-09-23T20:47:25.572298184Z" level=info
Sep 23 20:47:25 ip-172-31-84-221 containerd[19202]: time="2024-09-23T20:47:25.572313100Z" level=info
Sep 23 20:47:25 ip-172-31-84-221 containerd[19202]: time="2024-09-23T20:47:25.572322058Z" level=info
Sep 23 20:47:25 ip-172-31-84-221 containerd[19202]: time="2024-09-23T20:47:25.572328397Z" level=info
Sep 23 20:47:25 ip-172-31-84-221 containerd[19202]: time="2024-09-23T20:47:25.572313683Z" level=info
Sep 23 20:47:25 ip-172-31-84-221 containerd[19202]: time="2024-09-23T20:47:25.572786584Z" level=info
Sep 23 20:47:25 ip-172-31-84-221 systemd[1]: Started containerd.service - containerd container runtim
lines 1-21/21 (END)...skipping...
```

● sudo apt-get install -y socat

```
ubuntu@ip-172-31-84-221:~$ sudo apt-get install -y socat
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following packages were automatically installed and are no longer required:
  docker-buildx-plugin docker-ce-cli docker-ce-rootless-extras docker-compose-plugin libltdl7 lib
Use 'sudo apt autoremove' to remove them.
The following NEW packages will be installed:
  socat
0 upgraded, 1 newly installed, 0 to remove and 136 not upgraded.
Need to get 374 kB of archives.
After this operation, 1649 kB of additional disk space will be used.
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 socat amd64 1.8.0.0-4build3
Fetched 374 kB in 0s (13.8 MB/s)
Selecting previously unselected package socat.
(Reading database ... 68108 files and directories currently installed.)
Preparing to unpack .../socat_1.8.0.0-4build3_amd64.deb ...
Unpacking socat (1.8.0.0-4build3) ...
Setting up socat (1.8.0.0-4build3) ...
Processing triggers for man-db (2.12.0-4build2) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
ubuntu@ip-172-31-84-221:~$
```

Step 6: Initialize the Kubecluster .Now Perform this Command only for Master.

● sudo kubeadm init --pod-network-cidr=10.244.0.0/16

```
ubuntu@ip-172-31-84-221:~$ sudo kubeadm init --pod-network-cidr=10.244.0.0/16
I0923 20:56:13.230794   19947 version.go:256] remote version is much newer: v1.31.0; falling back to: stable-1.28
[init] Using Kubernetes version: v1.28.14
[preflight] Running pre-flight checks
[preflight] Pulling images required for setting up a Kubernetes cluster
[preflight] This might take a minute or two, depending on the speed of your internet connection
[preflight] You can also perform this action in beforehand using 'kubeadm config images pull'
W0923 20:56:20.561492   19947 checks.go:835] detected that the sandbox image "registry.k8s.io/pause:3.8" of the container r
used by kubeadm. It is recommended that using "registry.k8s.io/pause:3.9" as the CRI sandbox image.
[certs] Using certificateDir folder "/etc/kubernetes/pki"
[certs] Generating "ca" certificate and key
[certs] Generating "apiserver" certificate and key
[certs] apiserver serving cert is signed for DNS names [ip-172-31-84-221 kubernetes kubernetes.default kubernetes.default.s
.local] and IPs [10.96.0.1 172.31.84.221]
[certs] Generating "apiserver-kubelet-client" certificate and key
[certs] Generating "front-proxy-ca" certificate and key
[certs] Generating "front-proxy-client" certificate and key
[certs] Generating "etcd/ca" certificate and key
[certs] Generating "etcd/server" certificate and key
[certs] etcd/server serving cert is signed for DNS names [ip-172-31-84-221 localhost] and IPs [172.31.84.221 127.0.0.1 ::1]
[certs] Generating "etcd/peer" certificate and key
[certs] etcd/peer serving cert is signed for DNS names [ip-172-31-84-221 localhost] and IPs [172.31.84.221 127.0.0.1 ::1]
[certs] Generating "etcd/healthcheck-client" certificate and key
[certs] Generating "apiserver-etcd-client" certificate and key
[certs] Generating "sa" key and public key
```

```
[bootstrap-token] Configured RBAC rules to allow Node Bootstrap tokens to post CSRs in order for nodes to get l
[bootstrap-token] Configured RBAC rules to allow the csrapprover controller automatically approve CSRs from a N
[bootstrap-token] Configured RBAC rules to allow certificate rotation for all node client certificates in the c
[bootstrap-token] Creating the "cluster-info" ConfigMap in the "kube-public" namespace
[kubelet-finalize] Updating "/etc/kubernetes/kubelet.conf" to point to a rotatable kubelet client certificate a
[addons] Applied essential addon: CoreDNS
[addons] Applied essential addon: kube-proxy

Your Kubernetes control-plane has initialized successfully!

To start using your cluster, you need to run the following as a regular user:

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

Alternatively, if you are the root user, you can run:

  export KUBECONFIG=/etc/kubernetes/admin.conf

You should now deploy a pod network to the cluster.
Run "kubectl apply -f [podnetwork].yaml" with one of the options listed at:
  https://kubernetes.io/docs/concepts/cluster-administration/addons/

Then you can join any number of worker nodes by running the following on each as root:

kubeadm join 172.31.84.221:6443 --token yjt10w.maqlf98vcw88kw96 \
        --discovery-token-ca-cert-hash sha256:ffdb051e04077afecd5ea7a5702131537f9aa5c3dd13785ed4442327fb39f9cf
ubuntu@ip-172-31-84-221:~$
```

Copy the kudeadm join any number of worker nodes command to use it later for joining Node 1 and Node 2 with master

sudo kubeadm join 172.31.84.221:6443 --token yjt10w.maqlf98vcw88kw96 \--discovery-token-ca-cert -hash sha256:ffdb051e04077afecd5ea7a5702131537f9aa5c3dd13785ed4442327fb39f9cf

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-172-31-84-221:~$ mkdir -p $HOME/.kube
ubuntu@ip-172-31-84-221:~$ sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
cp: overwrite '/home/ubuntu/.kube/config'? y
ubuntu@ip-172-31-84-221:~$ sudo chown $(id -u):$(id -g) $HOME/.kube/config
ubuntu@ip-172-31-84-221:~$
```

Step 7: Now Run the command kubectl get nodes to see the nodes before executing Join command on nodes.

```
ubuntu@ip-172-31-84-221:~$ kubectl get nodes
NAME                STATUS     ROLES           AGE     VERSION
ip-172-31-84-221    NotReady   control-plane   8m27s   v1.28.14
ubuntu@ip-172-31-84-221:~$
```

Step 8: Now Run the following command on Node 1 and Node 2 to Join to master.
● sudo kubeadm join 172.31.95.244:6443 --token kzfth2.ug3970lp3qeeieb4\
--discovery-token-ca-cert-hash
sha256:dec27d33f1bfd1dca7a50caa2c05d4cad1d0a18aa88ad75c7ea83f15c529f4ca

Node 1:

```
ubuntu@ip-172-31-95-119:~$ sudo kubeadm join 172.31.84.221:6443 --token yjt10w.maqlf98vcw88kw96 --discovery-token-ca-cert-hash sha256:f
a5c3dd13785ed4442327fb39f9cf --ignore-preflight-errors=FileContent--proc-sys-net-bridge-bridge-nf-call-iptables
[preflight] Running pre-flight checks
        [WARNING FileContent--proc-sys-net-bridge-bridge-nf-call-iptables]: /proc/sys/net/bridge/bridge-nf-call-iptables does not exist
[preflight] Reading configuration from the cluster...
[preflight] FYI: You can look at this config file with 'kubectl -n kube-system get cm kubeadm-config -o yaml'
[kubelet-start] Writing kubelet configuration to file "/var/lib/kubelet/config.yaml"
[kubelet-start] Writing kubelet environment file with flags to file "/var/lib/kubelet/kubeadm-flags.env"
[kubelet-start] Starting the kubelet
[kubelet-start] Waiting for the kubelet to perform the TLS Bootstrap...

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

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

ubuntu@ip-172-31-95-119:~$
```

Node 2:

```
ubuntu@ip-172-31-80-164:~$ sudo kubeadm join 172.31.84.221:6443 --token yjt10w.maqlf98vcw88kw96 --discove
a5c3dd13785ed4442327fb39f9cf --ignore-preflight-errors=FileContent--proc-sys-net-bridge-bridge-nf-call-ip
[preflight] Running pre-flight checks
        [WARNING FileContent--proc-sys-net-bridge-bridge-nf-call-iptables]: /proc/sys/net/bridge/bridge-n
[preflight] Reading configuration from the cluster...
[preflight] FYI: You can look at this config file with 'kubectl -n kube-system get cm kubeadm-config -o y
[kubelet-start] Writing kubelet configuration to file "/var/lib/kubelet/config.yaml"
[kubelet-start] Writing kubelet environment file with flags to file "/var/lib/kubelet/kubeadm-flags.env"
[kubelet-start] Starting the kubelet
[kubelet-start] Waiting for the kubelet to perform the TLS Bootstrap...

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

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

Step 9: Now Run the command kubectl get nodes to see the nodes after executing Join command on nodes.

```
ubuntu@ip-172-31-84-221:~$ kubectl get nodes
NAME                STATUS     ROLES           AGE     VERSION
ip-172-31-80-164    NotReady   <none>          16s     v1.28.14
ip-172-31-84-221    NotReady   control-plane   30m     v1.28.14
ip-172-31-95-119    NotReady   <none>          6m43s   v1.28.14
ubuntu@ip-172-31-84-221:~$
```

Step 10: Since Status is NotReady we have to add a network plugin. And also we have to give the name to the nodes.

● kubectl apply -f https://docs.projectcalico.org/manifests/calico.yaml

```
ubuntu@ip-172-31-84-221:~$ kubectl apply -f https://docs.projectcalico.org/manifests/calico.yaml
poddisruptionbudget.policy/calico-kube-controllers created
serviceaccount/calico-kube-controllers created
serviceaccount/calico-node created
configmap/calico-config created
customresourcedefinition.apiextensions.k8s.io/bgpconfigurations.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
clusterrolebinding.rbac.authorization.k8s.io/calico-kube-controllers created
clusterrolebinding.rbac.authorization.k8s.io/calico-node created
daemonset.apps/calico-node created
deployment.apps/calico-kube-controllers created
ubuntu@ip-172-31-84-221:~$
```

● sudo systemctl status kubelet

```
ubuntu@ip-172-31-84-221:~$ sudo systemctl status kubelet
● kubelet.service - kubelet: The Kubernetes Node Agent
     Loaded: loaded (/usr/lib/systemd/system/kubelet.service; enabled; preset: enabled)
    Drop-In: /usr/lib/systemd/system/kubelet.service.d
             └─10-kubeadm.conf
     Active: active (running) since Mon 2024-09-23 20:56:33 UTC; 32min ago
       Docs: https://kubernetes.io/docs/
   Main PID: 20621 (kubelet)
      Tasks: 10 (limit: 4676)
     Memory: 38.0M (peak: 38.5M)
        CPU: 25.017s
     CGroup: /system.slice/kubelet.service
             └─20621 /usr/bin/kubelet --bootstrap-kubeconfig=/etc/kubernetes/bootstrap-kube

Sep 23 21:29:20 ip-172-31-84-221 kubelet[20621]:  > pod="kube-system/calico-kube-controller
Sep 23 21:29:20 ip-172-31-84-221 kubelet[20621]: E0923 21:29:20.385530   20621 remote_runti
Sep 23 21:29:20 ip-172-31-84-221 kubelet[20621]:         rpc error: code = Unknown desc = f
Sep 23 21:29:20 ip-172-31-84-221 kubelet[20621]:         : unknown
Sep 23 21:29:20 ip-172-31-84-221 kubelet[20621]:  > podSandboxID="0ac51787037fdb883ecf57aad
Sep 23 21:29:20 ip-172-31-84-221 kubelet[20621]: E0923 21:29:20.385606   20621 kuberuntime_
Sep 23 21:29:20 ip-172-31-84-221 kubelet[20621]: E0923 21:29:20.505019   20621 kubelet.go:1
Sep 23 21:29:21 ip-172-31-84-221 kubelet[20621]: I0923 21:29:21.388923   20621 pod_startup_
Sep 23 21:29:26 ip-172-31-84-221 kubelet[20621]: I0923 21:29:26.828223   20621 scope.go:117
Sep 23 21:29:26 ip-172-31-84-221 kubelet[20621]: E0923 21:29:26.828431   20621 pod_workers.
lines 1-23/23 (END)
```

● Now Run command kubectl get nodes -o wide we can see Status is ready.

```
ubuntu@ip-172-31-84-221:~$ kubectl get nodes -o wide
NAME              STATUS   ROLES           AGE     VERSION   INTERNAL-IP     EXTERNAL-IP   OS-IMAGE          KERNEL-VERSION   CONTAINER-RUNTIME
ip-172-31-80-164  Ready    <none>          3m29s   v1.28.14  172.31.80.164   <none>        Ubuntu 24.04 LTS  6.8.0-1012-aws   containerd://1.7.12
ip-172-31-84-221  Ready    control-plane   33m     v1.28.14  172.31.84.221   <none>        Ubuntu 24.04 LTS  6.8.0-1012-aws   containerd://1.7.12
ip-172-31-95-119  Ready    <none>          9m56s   v1.28.14  172.31.95.119   <none>        Ubuntu 24.04 LTS  6.8.0-1012-aws   containerd://1.7.12
ubuntu@ip-172-31-84-221:~$
```

The Roles are not yet assigned to the Nodes

```
ubuntu@ip-172-31-84-221:~$ kubectl get nodes
NAME                STATUS   ROLES           AGE     VERSION
ip-172-31-80-164    Ready    <none>          4m14s   v1.28.14
ip-172-31-84-221    Ready    control-plane   34m     v1.28.14
ip-172-31-95-119    Ready    <none>          10m     v1.28.14
ubuntu@ip-172-31-84-221:~$
```

- Rename to Node 1: kubectl label node ip-172-31-28-117 kubernetes.io/role=Node1
- Rename to Node 2: kubectl label node ip-172-31-18-135 kubernetes.io/role=Node2

```
ubuntu@ip-172-31-84-221:~$ kubectl label node ip-172-31-80-164 kubernetes.io/role=Node1
node/ip-172-31-80-164 labeled
ubuntu@ip-172-31-84-221:~$ kubectl label node ip-172-31-95-119 kubernetes.io/role=Node2
node/ip-172-31-95-119 labeled
ubuntu@ip-172-31-84-221:~$
```

- Run kubectl get nodes to check if roles are assigned now to the nodes

```
ubuntu@ip-172-31-84-221:~$ kubectl get nodes
NAME                STATUS   ROLES           AGE    VERSION
ip-172-31-80-164    Ready    Node1           8m2s   v1.28.14
ip-172-31-84-221    Ready    control-plane   38m    v1.28.14
ip-172-31-95-119    Ready    Node2           14m    v1.28.14
ubuntu@ip-172-31-84-221:~$
```

Conclusion: In this experiment, we successfully set up a Kubernetes cluster with one master and two worker nodes on AWS EC2 instances. After installing Docker, Kubernetes tools (kubelet, kubeadm, kubectl), and containerd on all nodes, the master node was initialized and the worker nodes were joined to the cluster. Initially, the nodes were in the NotReady state, which was resolved by installing the Calico network plugin. We also labeled the nodes with appropriate roles (control-plane and worker). The cluster became fully functional with all nodes in the Ready state, demonstrating the successful configuration and orchestration of Kubernetes.

SHRAVANI RASAM D15A 46

ADVANCE DEVOPS EXP-4

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

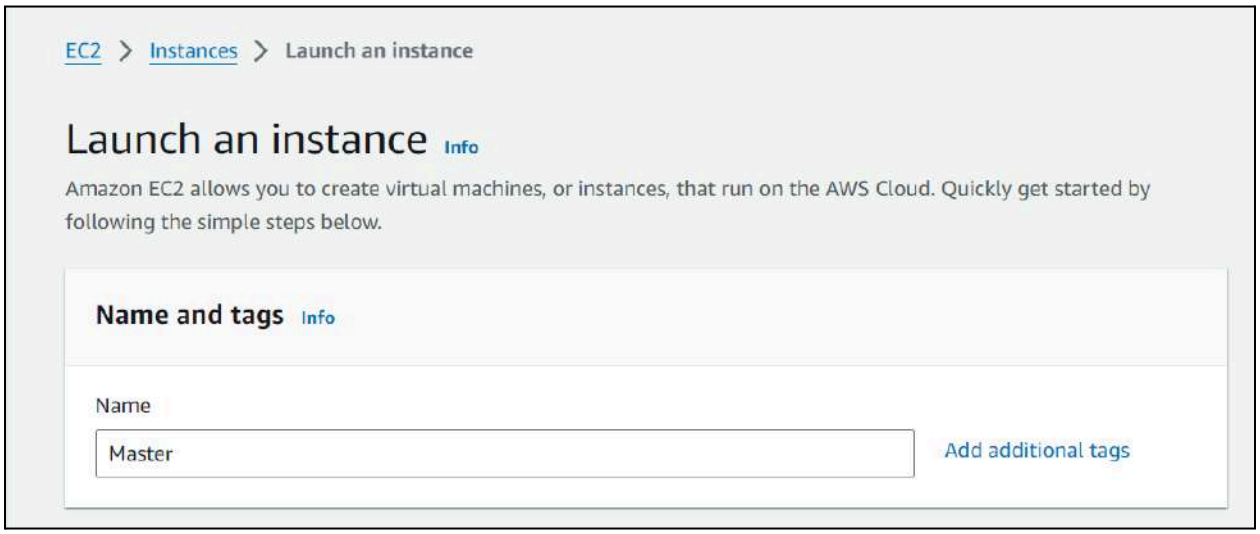
Theory:
Kubernetes, originally developed by Google, is an open-source container orchestration platform. It
automates the deployment, scaling, and management of containerized applications, ensuring high
availability and fault tolerance. Kubernetes is now the industry standard for container orchestration and
is governed by the Cloud Native Computing Foundation (CNCF), with contributions from major cloud
and software providers like Google, AWS, Microsoft, IBM, Intel, Cisco, and Red Hat.

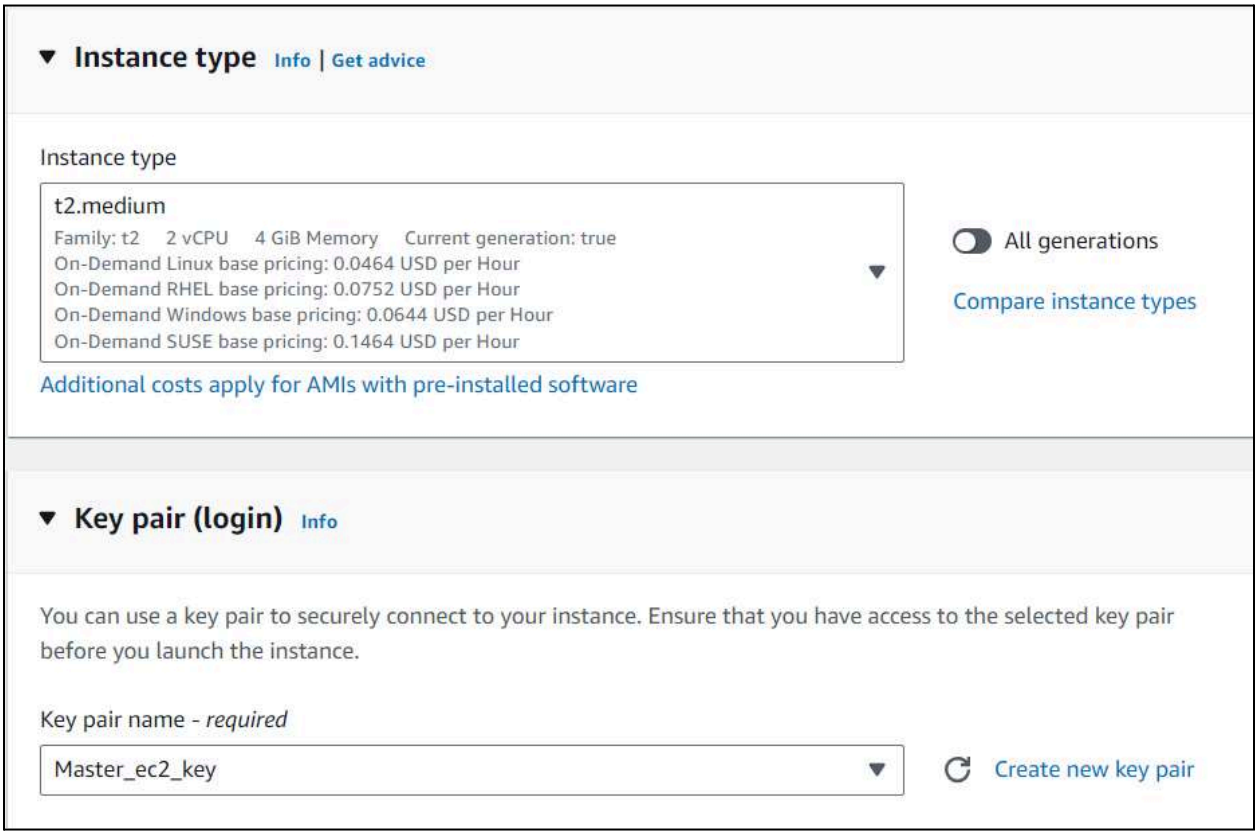
Kubernetes Deployment: Is a resource in Kubernetes that provides declarative updates for Pods and
ReplicaSets. With a Deployment, you can define how many replicas of a pod should run, roll out new
versions of an application, and roll back to previous versions if necessary. It ensures that the desired
number of pod replicas are running at all times.

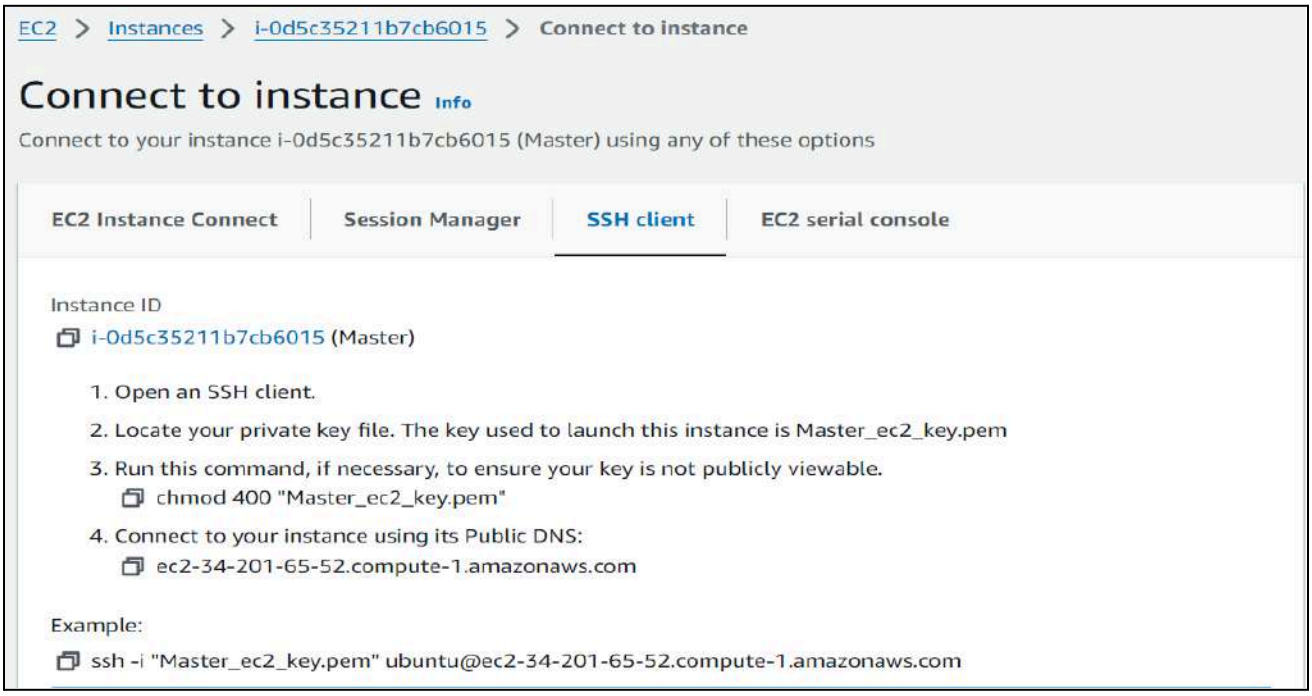
Necessary Requirements:
● **EC2 Instance:** The experiment required launching a t2.medium EC2 instance with 2 CPUs, as
Kubernetes demands sufficient resources for effective functioning.
● **Minimum Requirements:**
○ Instance Type: t2.medium
○ CPUs: 2
○ Memory: Adequate for container orchestration.

This ensured that the Kubernetes cluster had the necessary resources to function smoothly.

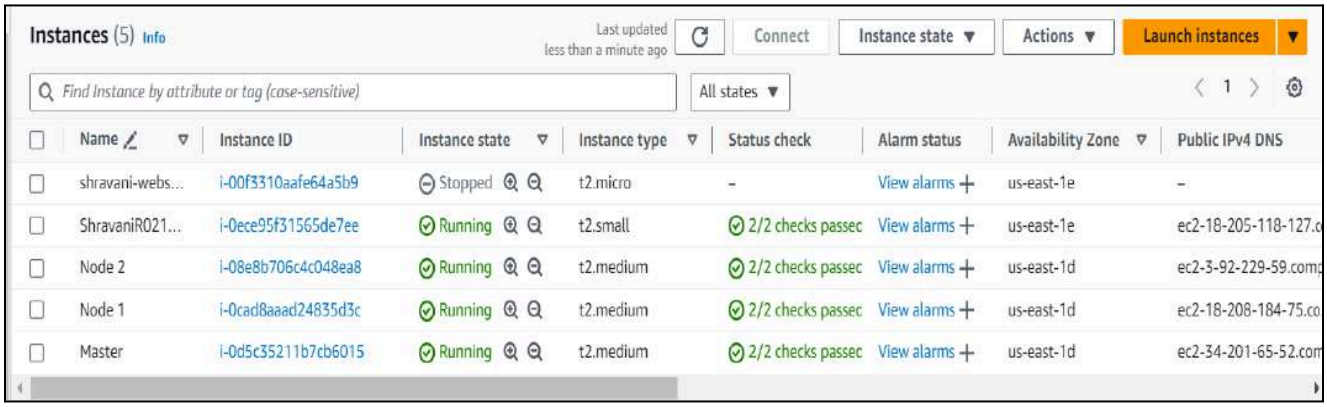
Step 1: Log in to your AWS Academy/personal account and launch a new Ec2 Instance. Select Ubuntu as AMI and t2.medium as Instance Type, create a key of type RSA with .pem extension,
and move the downloaded key to the new folder.
Note: A minimum of 2 CPUs are required so Please select t2.medium and do not forget to stop the
instance after the experiment because it is not available in the free tier.







Step 2: After creating the instance click on Connect the instance and navigate to SSH Client.



Step 3: Now open the folder in the terminal where our .pem key is stored and paste the Example command (starting with ssh -i .....) in the terminal.(ssh -i "Master_Ec2_Key.pem" ubuntu@ec2-54-196-129-215.compute-1.amazonaws.com)

```
C:\Users\Shravani\Desktop\Master>ssh -i "Master_ec2_key.pem" ubuntu@ec2-34-201-65-52.compute-1.amazonaws.com
Welcome to Ubuntu 24.04 LTS (GNU/Linux 6.8.0-1012-aws x86_64)

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

 System information as of Mon Sep 23 16:43:43 UTC 2024

  System load:  0.0               Processes:             116
  Usage of /:   22.9% of 6.71GB   Users logged in:       0
  Memory usage: 5%                IPv4 address for enX0: 172.31.84.221
  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
```

Step 4: Run the below commands to install and setup Docker.

- curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
- curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo tee /etc/apt/trusted.gpg.d/docker.gpg > /dev/null
- sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable"

```
ubuntu@ip-172-31-84-221:~$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add
Warning: apt-key is deprecated. Manage keyring files in trusted.gpg.d instead (see apt-key(8)).
OK
ubuntu@ip-172-31-84-221:~$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo tee/etc/apt/
sudo: tee/etc/apt/trusted.gpg.d/docker.gpg: command not found
ubuntu@ip-172-31-84-221:~$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo tee
gpg.d/docker.gpg > /dev/null-----BEGIN PGP PUBLIC KEY BLOCK-----

mQINBFit2ioBEADhWpZ8/wvZ6hUTiXOwQHXMAlaFHcPH9hAtr4F1y2+OYdbtMuth
lqqwp028AqyY+PRfVMtSYMbjuQuu5byyKR01BbqYhuS3jtqQmljZ/bJvXqnmiVXh
38UuLa+z077PxyxQhu5BbqntTPQMfiyqEiU+BKbq2WmANUKQf+1AmZY/IruOXbnq
L4C1+gJ8vfmXQt99npCaxEjaNRVYfOS8QcixNzHUYnb6emjlANyEVlZzeqo7XKl7
UrwV5inawTSzWNvtjEjj4nJL8NsLwscpLPQUhTQ+7BbQXAwAmeHCUTQIvvWXqw0N
cmhh4HgeQscQHYgOJjjDVfoY5MucvglbIgCqfzAHW9jxmRL4qbMZj+b1XoePEtht
ku4bIQN1X5P07fNWzlgaRL5Z4POXDDZTlIQ/El58j9kp4bnWRCJW0lya+f8ocodo
vZZ+Doi+fy4D5ZGrL4XEcIQP/Lv5uFyf+kQtl/94VFYVJOleAv8W92KdgDkhTcTD
G7c0tIkVEKNUq48b3aQ64NOZQW7fVjfoKwEZdOqPE72Pa45jrZzvUFxSpdiNk2tZ
XYukHjlxxEgBdC/J3cMMNRE1F4NCA3ApfV1Y7/hTeOnmDuDYwr9/obA8t016Yljj
q5rdkywPf4JF8mXUW5eCN1vAFHxeg9ZWemhBtQmGxXnw9M+z6hWwc6ahmwARAQAB
tCtEb2NrZXIgUmVsZWFzZSAoQ0UgZGViKSA8ZG9ja2VyQGRvY2tlci5jb20+iQI3
```

```
Get:43 http://security.ubuntu.com/ubuntu noble-security/universe Translation-en [113 kB]
Get:44 http://security.ubuntu.com/ubuntu noble-security/universe amd64 Components [8632 B]
Get:45 http://security.ubuntu.com/ubuntu noble-security/universe amd64 c-n-f Metadata [10.1 kB]
Get:46 http://security.ubuntu.com/ubuntu noble-security/restricted amd64 Packages [353 kB]
Get:47 http://security.ubuntu.com/ubuntu noble-security/restricted Translation-en [68.1 kB]
Get:48 http://security.ubuntu.com/ubuntu noble-security/restricted amd64 c-n-f Metadata [428 B]
Get:49 http://security.ubuntu.com/ubuntu noble-security/multiverse amd64 Packages [10.9 kB]
Get:50 http://security.ubuntu.com/ubuntu noble-security/multiverse Translation-en [2808 B]
Get:51 http://security.ubuntu.com/ubuntu noble-security/multiverse amd64 Components [208 B]
Get:52 http://security.ubuntu.com/ubuntu noble-security/multiverse amd64 c-n-f Metadata [344 B]
Fetched 29.1 MB in 4s (7159 kB/s)
Reading package lists... Done
W: https://download.docker.com/linux/ubuntu/dists/noble/InRelease: Key is stored in legacy trusted.gpg keyring
key(8) for details.
ubuntu@ip-172-31-84-221:~$
```

- sudo apt-get update
- sudo apt-get install -y docker-c

```
ubuntu@ip-172-31-84-221:~$ sudo apt-get update
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 https://download.docker.com/linux/ubuntu noble InRelease
Hit:5 http://security.ubuntu.com/ubuntu noble-security InRelease
Reading package lists... Done
W: https://download.docker.com/linux/ubuntu/dists/noble/InRelease: Key is stored in legacy trusted.gpg keyring
key(8) for details.
ubuntu@ip-172-31-84-221:~$
```

```
ubuntu@ip-172-31-95-119:~$ sudo apt-get install -y docker-ce
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done

The following additional packages will be installed:
  containerd.io
Suggested packages:
  aufs-tools cgroupfs-mount | cgroup-lite
The following packages will be REMOVED:
  containerd runc
The following NEW packages will be installed:
  containerd.io docker-ce
0 upgraded, 2 newly installed, 2 to remove and 105 not upgraded.
Need to get 0 B/55.0 MB of archives.
After this operation, 53.1 MB of additional disk space will be used.
(Reading database ... 98747 files and directories currently installed.)
Removing containerd (1.7.12-0ubuntu4.1) ...
Removing runc (1.1.12-0ubuntu3.1) ...
Selecting previously unselected package containerd.io.
(Reading database ... 98685 files and directories currently installed.)
Preparing to unpack .../containerd.io_1.7.22-1_amd64.deb ...
Unpacking containerd.io (1.7.22-1) ...
Selecting previously unselected package docker-ce.
Preparing to unpack .../docker-ce_5%3a27.3.1-1~ubuntu.24.04~noble_amd64.deb ...
Unpacking docker-ce (5:27.3.1-1~ubuntu.24.04~noble) ...
Setting up containerd.io (1.7.22-1) ...
Setting up docker-ce (5:27.3.1-1~ubuntu.24.04~noble) ...
Processing triggers for man-db (2.12.0-4build2) ...
Scanning processes...
```

● sudo mkdir -p /etc/docker
cat <<EOF | sudo tee /etc/docker/daemon.json
{
"exec-opts": ["native.cgroupdriver=systemd"]
}
EOF

```
ubuntu@ip-172-31-84-221:~$ sudo mkdir -p /etc/docker
driver=systemd"]
}
EOFcat <<EOF | sudo tee /etc/docker/daemon.json
{
"exec-opts": ["native.cgroupdriver=systemd"]
}
EOFubuntu@ip-172-31-84-221:~$ sudo mkdir -p /etc/docker
 tee /etc/docker/daemon.json
{
"exec-opts": ["native.cgroupdriver=systemd"]
}
EOFcat <<EOF | sudo tee /etc/docker/daemon.json
{
"exec-opts": ["native.cgroupdriver=systemd"]
}
EOFubuntu@ip-172-31-84-221:~$
```

● sudo systemctl enable docker
● sudo systemctl daemon-reload
● sudo systemctl restart docker

```
EOFubuntu@ip-172-31-95-119:~$ sudo systemctl enable docker
Synchronizing state of docker.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable docker
ubuntu@ip-172-31-95-119:~$ sudo systemctl daemon-reload
ubuntu@ip-172-31-95-119:~$ sudo systemctl restart docker
```

Step 5: Run the below command to install Kubernetes.

● curl -fsSL https://pkgs.k8s.io/core:/stable:/v1.31/deb/Release.key | sudo gpg --dearmor -o /etc/apt/keyrings/kubernetes-apt-keyring.gpg
● echo 'deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg] https://pkgs.k8s.io/core:/stable:/v1.31/deb/ /' | sudo tee /etc/apt/sources.list.d/kubernetes.list

```
ubuntu@ip-172-31-84-221:~$ curl -fsSL https://pkgs.k8s.io/core:/stable:/v1.31/deb/Release.key | sudo gpg --dearmor -o
ngs/kubernetes-apt-keyring.gpg
echo 'deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg]
https://pkgs.k8s.io/core:/stable:/v1.31/deb/ /' | gpg: missing argument for option "-o"
sudo tee /etc/apt/sources.list.d/kubernetes.list


ubuntu@ip-172-31-84-221:~$ /etc/apt/keyrings/kubernetes-apt-keyring.gpg
-bash: /etc/apt/keyrings/kubernetes-apt-keyring.gpg: No such file or directory
ubuntu@ip-172-31-84-221:~$ echo 'deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg]
> https://pkgs.k8s.io/core:/stable:/v1.31/deb/ /' | sudo tee /etc/apt/sources.list.d/kubernetes.list
deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg]
https://pkgs.k8s.io/core:/stable:/v1.31/deb/ /
```

```
ubuntu@ip-172-31-84-221:~$ sudo apt-get install -y kubelet kubeadm kubectl
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  conntrack cri-tools kubernetes-cni
The following NEW packages will be installed:
  conntrack cri-tools kubeadm kubectl kubelet kubernetes-cni
0 upgraded, 6 newly installed, 0 to remove and 136 not upgraded.
Need to get 87.4 MB of archives.
After this operation, 335 MB of additional disk space will be used.
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 conntrack amd64 1:1.4.8-1ubuntu1 [37.9 kB]
Get:2 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.28/deb  cri-tools 1.28.0-1.1 [19.6 MB]
Get:3 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.28/deb  kubernetes-cni 1.2.0-2.1 [27.6 MB]
Get:4 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.28/deb  kubelet 1.28.14-2.1 [19.6 MB]
Get:5 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.28/deb  kubectl 1.28.14-2.1 [10.4 MB]
Get:6 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.28/deb  kubeadm 1.28.14-2.1 [10.1 MB]
Fetched 87.4 MB in 1s (77.5 MB/s)
```

```
ubuntu@ip-172-31-84-221:~$ sudo apt-mark hold kubelet kubeadm kubectl
kubelet set on hold.
kubeadm set on hold.
kubectl set on hold.
ubuntu@ip-172-31-84-221:~$
```

- sudo apt-get update
- sudo apt-get install -y kubelet kubeadm kubectl
- sudo apt-mark hold kubelet kubeadm kubectl

```
ubuntu@ip-172-31-84-221:~$ sudo apt-get update
Warning: The unit file, source configuration file or drop-ins of apt-news.service changed on disk. Run 'systemctl daemon-reload' to reload u
nits.
Warning: The unit file, source configuration file or drop-ins of esm-cache.service changed on disk. Run 'systemctl daemon-reload' to reload
units.
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://download.docker.com/linux/ubuntu noble InRelease
Get:6 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.31/deb  InRelease [1186 B]
Err:6 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.31/deb  InRelease
  The following signatures couldn't be verified because the public key is not available: NO_PUBKEY 234654DA9A296436
Reading package lists... Done
W: https://download.docker.com/linux/ubuntu/dists/noble/InRelease: Key is stored in legacy trusted.gpg keyring (/etc/apt/trusted.gpg), see t
he DEPRECATION section in apt-key(8) for details.
W: GPG error: https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.31/deb  InRelease: The following signatures co
uldn't be verified because the public key is not available: NO_PUBKEY 234654DA9A296436
E: The repository 'https://pkgs.k8s.io/core:/stable:/v1.31/deb  InRelease' is not signed.
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.
```

Err:7 https://packages.cloud.google.com/apt kubernetes-xenial Release 404 Not Found [IP: 64.233.180.139 443]

- sudo rm /etc/apt/sources.list.d/kubernetes.list
- sudo nano /etc/apt/sources.list.d/kubernetes.list
- deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg] https://pkgs.k8s.io/core:/stable:/v1.28/deb/ /

```
ubuntu@ip-172-31-84-221:~$ sudo systemctl enable --now kubelet
ubuntu@ip-172-31-84-221:~$ sudo apt-get install -y containerd
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following packages were automatically installed and are no longer required:
  docker-buildx-plugin docker-ce-cli docker-ce-rootless-extras docker-compose-plugin libltdl7 libslirp0 pigz slirp4netns
Use 'sudo apt autoremove' to remove them.
The following additional packages will be installed:
  runc
The following packages will be REMOVED:
  containerd.io docker-ce
The following NEW packages will be installed:
  containerd runc
0 upgraded, 2 newly installed, 2 to remove and 136 not upgraded.
Need to get 47.2 MB of archives.
After this operation, 53.1 MB disk space will be freed.
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 runc amd64 1.1.12-0ubuntu3.1 [8599 kB]
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 containerd amd64 1.7.12-0ubuntu4.1 [38.6 MB]
Fetched 47.2 MB in 1s (90.1 MB/s)
```

- sudo mkdir -p /etc/containerd
- sudo containerd config default | sudo tee /etc/containerd/config.toml

```
ubuntu@ip-172-31-84-221:~$ sudo containerd config default | sudo tee /etc/containerd/config.toml
disabled_plugins = []
imports = []
oom_score = 0
plugin_dir = ""
required_plugins = []
root = "/var/lib/containerd"
state = "/run/containerd"
temp = ""
version = 2

[cgroup]
  path = ""

[debug]
  address = ""
  format = ""
  gid = 0
  level = ""
  uid = 0

[grpc]
  address = "/run/containerd/containerd.sock"
  gid = 0
  max_recv_message_size = 16777216
  max_send_message_size = 16777216
  tcp_address = ""
  tcp_tls_ca = ""
  tcp_tls_cert = ""
  tcp_tls_key = ""
  uid = 0
```

- sudo systemctl restart containerd
- sudo systemctl enable containerd
- sudo systemctl status containerd

```
ubuntu@ip-172-31-84-221:~$ sudo systemctl restart containerd
ubuntu@ip-172-31-84-221:~$ sudo systemctl enable containerd
ubuntu@ip-172-31-84-221:~$ sudo systemctl status containerd
● containerd.service - containerd container runtime
     Loaded: loaded (/usr/lib/systemd/system/containerd.service; enabled; preset: enabled)
     Active: active (running) since Mon 2024-09-23 20:47:25 UTC; 14s ago
       Docs: https://containerd.io
   Main PID: 19202 (containerd)
      Tasks: 7
     Memory: 13.0M (peak: 13.8M)
        CPU: 113ms
     CGroup: /system.slice/containerd.service
             └─19202 /usr/bin/containerd

Sep 23 20:47:25 ip-172-31-84-221 containerd[19202]: time="2024-09-23T20:47:25.572213616Z" level=info
Sep 23 20:47:25 ip-172-31-84-221 containerd[19202]: time="2024-09-23T20:47:25.572255061Z" level=info
Sep 23 20:47:25 ip-172-31-84-221 containerd[19202]: time="2024-09-23T20:47:25.572281095Z" level=info
Sep 23 20:47:25 ip-172-31-84-221 containerd[19202]: time="2024-09-23T20:47:25.572298184Z" level=info
Sep 23 20:47:25 ip-172-31-84-221 containerd[19202]: time="2024-09-23T20:47:25.572313100Z" level=info
Sep 23 20:47:25 ip-172-31-84-221 containerd[19202]: time="2024-09-23T20:47:25.572322058Z" level=info
Sep 23 20:47:25 ip-172-31-84-221 containerd[19202]: time="2024-09-23T20:47:25.572328397Z" level=info
Sep 23 20:47:25 ip-172-31-84-221 containerd[19202]: time="2024-09-23T20:47:25.572313683Z" level=info
Sep 23 20:47:25 ip-172-31-84-221 containerd[19202]: time="2024-09-23T20:47:25.572786584Z" level=info
Sep 23 20:47:25 ip-172-31-84-221 systemd[1]: Started containerd.service - containerd container runtim
lines 1-21/21 (END)...skipping...
```

- sudo apt-get install -y socat

```
ubuntu@ip-172-31-84-221:~$ sudo apt-get install -y socat
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following packages were automatically installed and are no longer required:
  docker-buildx-plugin docker-ce-cli docker-ce-rootless-extras docker-compose-plugin libltdl7 lib
Use 'sudo apt autoremove' to remove them.
The following NEW packages will be installed:
  socat
0 upgraded, 1 newly installed, 0 to remove and 136 not upgraded.
Need to get 374 kB of archives.
After this operation, 1649 kB of additional disk space will be used.
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 socat amd64 1.8.0.0-4build3
Fetched 374 kB in 0s (13.8 MB/s)
Selecting previously unselected package socat.
(Reading database ... 68108 files and directories currently installed.)
Preparing to unpack .../socat_1.8.0.0-4build3_amd64.deb ...
Unpacking socat (1.8.0.0-4build3) ...
Setting up socat (1.8.0.0-4build3) ...
Processing triggers for man-db (2.12.0-4build2) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
ubuntu@ip-172-31-84-221:~$
```

Step 6: Initialize the Kubecluster .Now Perform this Command only for Master.

- sudo kubeadm init --pod-network-cidr=10.244.0.0/16

```
ubuntu@ip-172-31-84-221:~$ sudo kubeadm init --pod-network-cidr=10.244.0.0/16
I0923 20:56:13.230794   19947 version.go:256] remote version is much newer: v1.31.0; falling back to: stable-1.28
[init] Using Kubernetes version: v1.28.14
[preflight] Running pre-flight checks
[preflight] Pulling images required for setting up a Kubernetes cluster
[preflight] This might take a minute or two, depending on the speed of your internet connection
[preflight] You can also perform this action in beforehand using 'kubeadm config images pull'
W0923 20:56:20.561492   19947 checks.go:835] detected that the sandbox image "registry.k8s.io/pause:3.8" of the container r
used by kubeadm. It is recommended that using "registry.k8s.io/pause:3.9" as the CRI sandbox image.
[certs] Using certificateDir folder "/etc/kubernetes/pki"
[certs] Generating "ca" certificate and key
[certs] Generating "apiserver" certificate and key
[certs] apiserver serving cert is signed for DNS names [ip-172-31-84-221 kubernetes kubernetes.default kubernetes.default.s
.local] and IPs [10.96.0.1 172.31.84.221]
[certs] Generating "apiserver-kubelet-client" certificate and key
[certs] Generating "front-proxy-ca" certificate and key
[certs] Generating "front-proxy-client" certificate and key
[certs] Generating "etcd/ca" certificate and key
[certs] Generating "etcd/server" certificate and key
[certs] etcd/server serving cert is signed for DNS names [ip-172-31-84-221 localhost] and IPs [172.31.84.221 127.0.0.1 ::1]
[certs] Generating "etcd/peer" certificate and key
[certs] etcd/peer serving cert is signed for DNS names [ip-172-31-84-221 localhost] and IPs [172.31.84.221 127.0.0.1 ::1]
[certs] Generating "etcd/healthcheck-client" certificate and key
[certs] Generating "apiserver-etcd-client" certificate and key
[certs] Generating "sa" key and public key
```

```
[bootstrap-token] Configured RBAC rules to allow Node Bootstrap tokens to post CSRs in order for nodes to get l
[bootstrap-token] Configured RBAC rules to allow the csrapprover controller automatically approve CSRs from a N
[bootstrap-token] Configured RBAC rules to allow certificate rotation for all node client certificates in the c
[bootstrap-token] Creating the "cluster-info" ConfigMap in the "kube-public" namespace
[kubelet-finalize] Updating "/etc/kubernetes/kubelet.conf" to point to a rotatable kubelet client certificate a
[addons] Applied essential addon: CoreDNS
[addons] Applied essential addon: kube-proxy

Your Kubernetes control-plane has initialized successfully!

To start using your cluster, you need to run the following as a regular user:

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

Alternatively, if you are the root user, you can run:

  export KUBECONFIG=/etc/kubernetes/admin.conf

You should now deploy a pod network to the cluster.
Run "kubectl apply -f [podnetwork].yaml" with one of the options listed at:
  https://kubernetes.io/docs/concepts/cluster-administration/addons/

Then you can join any number of worker nodes by running the following on each as root:

kubeadm join 172.31.84.221:6443 --token yjt10w.maqlf98vcw88kw96 \
        --discovery-token-ca-cert-hash sha256:ffdb051e04077afecd5ea7a5702131537f9aa5c3dd13785ed4442327fb39f9cf
ubuntu@ip-172-31-84-221:~$
```

STEP 7:

- 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-172-31-84-221:~$ mkdir -p $HOME/.kube
ubuntu@ip-172-31-84-221:~$ sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
cp: overwrite '/home/ubuntu/.kube/config'? y
ubuntu@ip-172-31-84-221:~$ sudo chown $(id -u):$(id -g) $HOME/.kube/config
ubuntu@ip-172-31-84-221:~$
```

Add a common networking plugin called flannel as mentioned in the code.
kubectl apply -f
https://raw.githubusercontent.com/coreos/flannel/master/Documentation/kube-flannel.yml

```
ubuntu@ip-172-31-84-221:~$ kubectl apply -f https://raw.githubusercontent.com/coreos/flannel/master/Documentation/kube-flannel.yml
namespace/kube-flannel created
clusterrole.rbac.authorization.k8s.io/flannel created
clusterrolebinding.rbac.authorization.k8s.io/flannel created
serviceaccount/flannel created
configmap/kube-flannel-cfg created
daemonset.apps/kube-flannel-ds created
```

Step 7: Now that the cluster is up and running, we can deploy our nginx server on this cluster.Apply this deployment file using this command to create a deployment
kubectl apply -f https://k8s.io/examples/application/deployment.yaml

```
ubuntu@ip-172-31-95-119:~$ kubectl apply -f https://k8s.io/examples/application/deployment.yaml
deployment.apps/nginx-deployment unchanged
```

kubectl get pods

```
ubuntu@ip-172-31-95-119:~$ kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
nginx-deployment-86dcfdf4c6-9xvm2   0/1     Pending   0          33m
nginx-deployment-86dcfdf4c6-zmddb   0/1     Pending   0          33m
```

```
ubuntu@ip-172-31-95-119:~$ kubectl get nodes
NAME                STATUS   ROLES           AGE   VERSION
ip-172-31-95-119    Ready    control-plane   40m   v1.28.14
```

POD_NAME=$(kubectl get pods -l app=nginx -o jsonpath="{.items[0].metadata.name}")
kubectl port-forward $POD_NAME 8087:80

```
^Cubuntu@ip-172-31-95-119:~$ kubectl get pods -l app=nginx
NAME                                READY   STATUS    RESTARTS   AGE
nginx-deployment-66df5888d5-c85tq   1/1     Running   0          82m
nginx-deployment-66df5888d5-q2jxg   1/1     Running   0          88m
```

Step 8: Verify your deployment
Open up a new terminal and ssh to your EC2 instance.
Then, use this curl command to check if the Nginx server is running.
curl --head http://127.0.0.1:8080

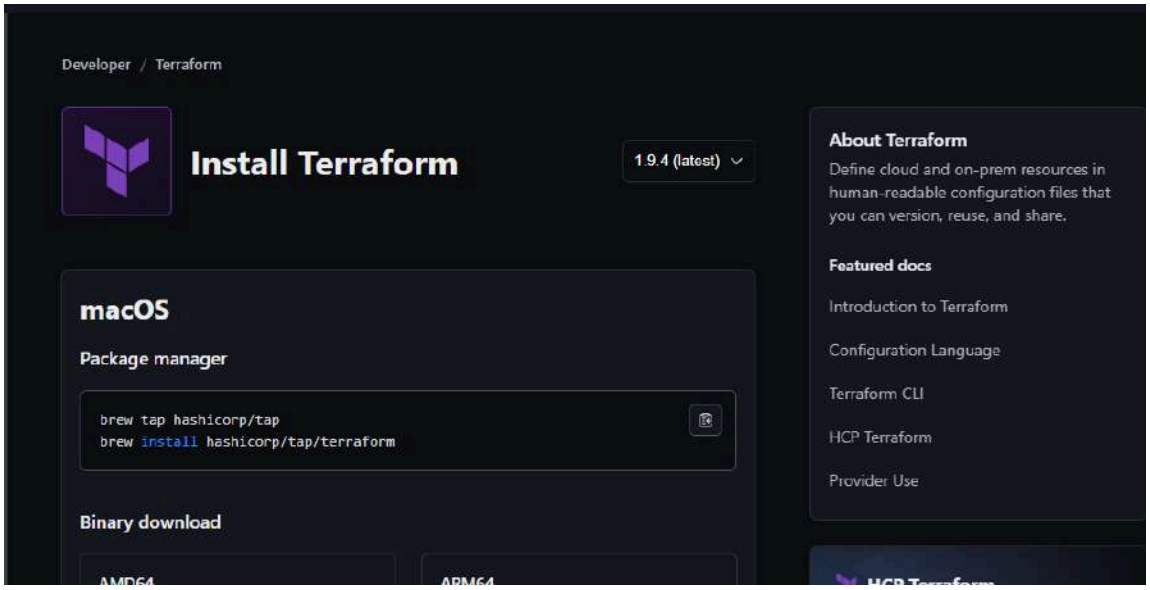
```
ubuntu@ip-172-31-20-171:~$ curl --head http://127.0.0.1:8080
HTTP/1.1 200 OK
Server: nginx/1.14.2
Content-Type: text/html
Content-Length: 612
Last-Modified: Tue, 04 Dec 2018 14:44:49 GMT
Connection: keep-alive
ETag: "5c0692e1-264"
Accept-Ranges: bytes
```

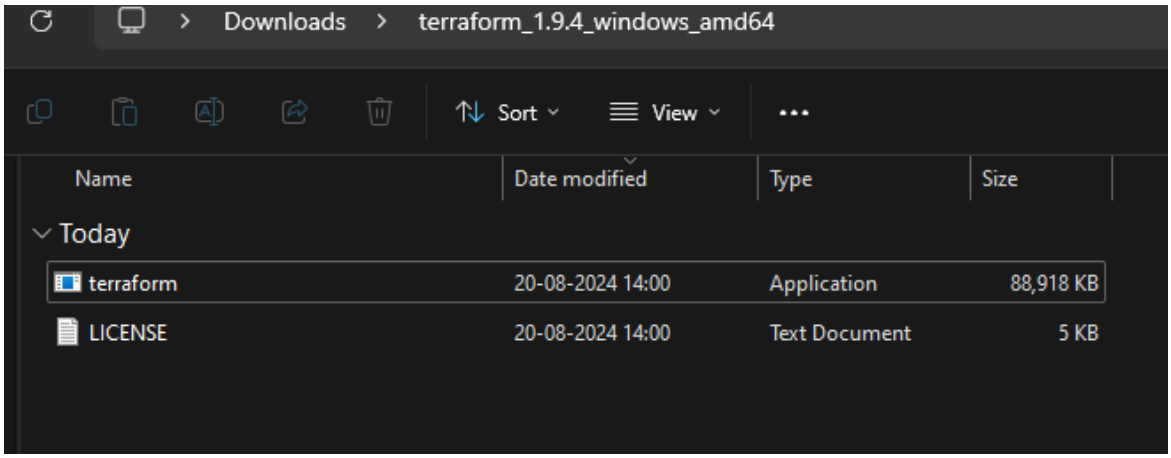
If the response is 200 OK and you can see the Nginx server name, your deployment was successful.
We have successfully deployed our Nginx server on our EC2 instance.

Conclusion:
In this experiment, we successfully installed Kubernetes on an EC2 instance and deployed an Nginx server using Kubectl commands. During the process, we encountered two main errors: the Kubernetes pod was initially in a pending state, which was resolved by removing the control-plane taint using kubectl taint nodes --all, and we also faced an issue with the missing containerd runtime, which was fixed by installing and starting containerd. We used a t2.medium EC2 instance with 2 CPUs to meet the necessary resource requirements for the Kubernetes setup and deployment.

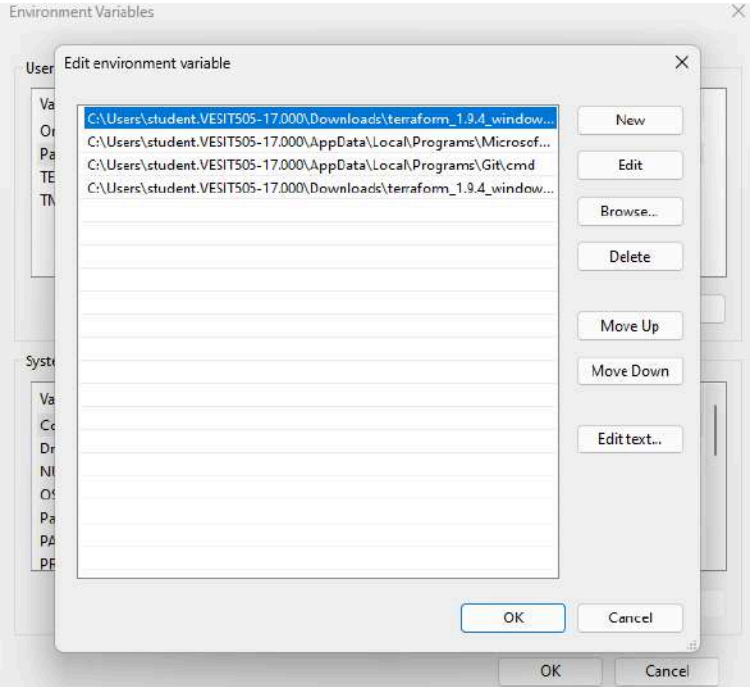
ADVANCED DEVOPS EXP 5

AIM: Installation and configuration of terraform on Windows





SHRAVANI RASAM D15A 46



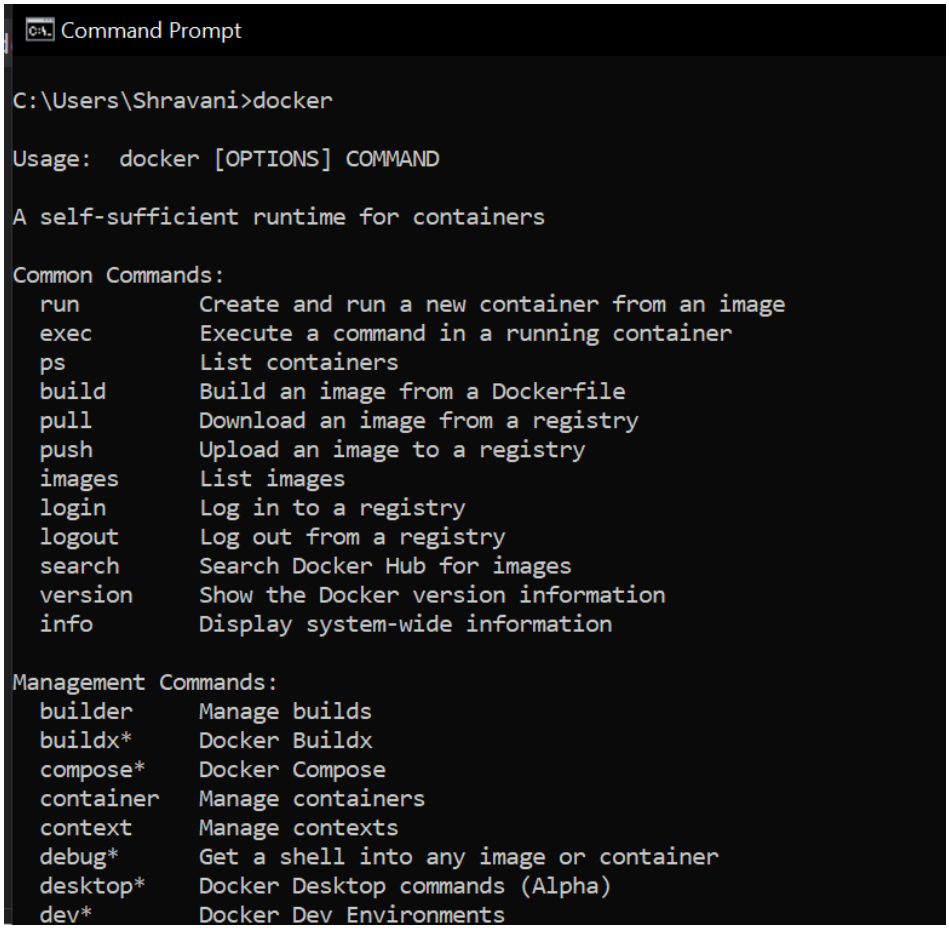

```
Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\student.VESIT505-17.000\Downloads\terraform_1.9.4_windows_amd64> 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
  test          Execute integration tests for Terraform modules
  untaint       Remove the 'tainted' state from a resource instance
```

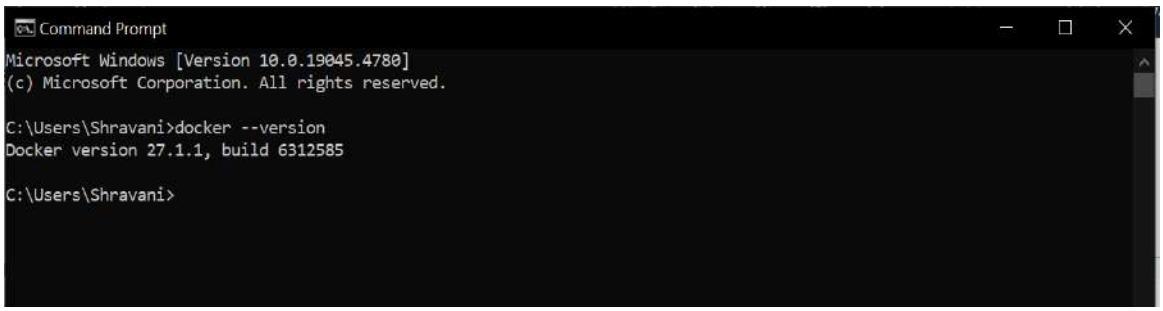
```
PS C:\Users\student.VESIT505-17.000\Downloads\terraform_1.9.4_windows_amd64> terraform --version
Terraform v1.9.4
on windows_amd64
PS C:\Users\student.VESIT505-17.000\Downloads\terraform_1.9.4_windows_amd64>
```

ADVANCED DEVOPS EXP 6

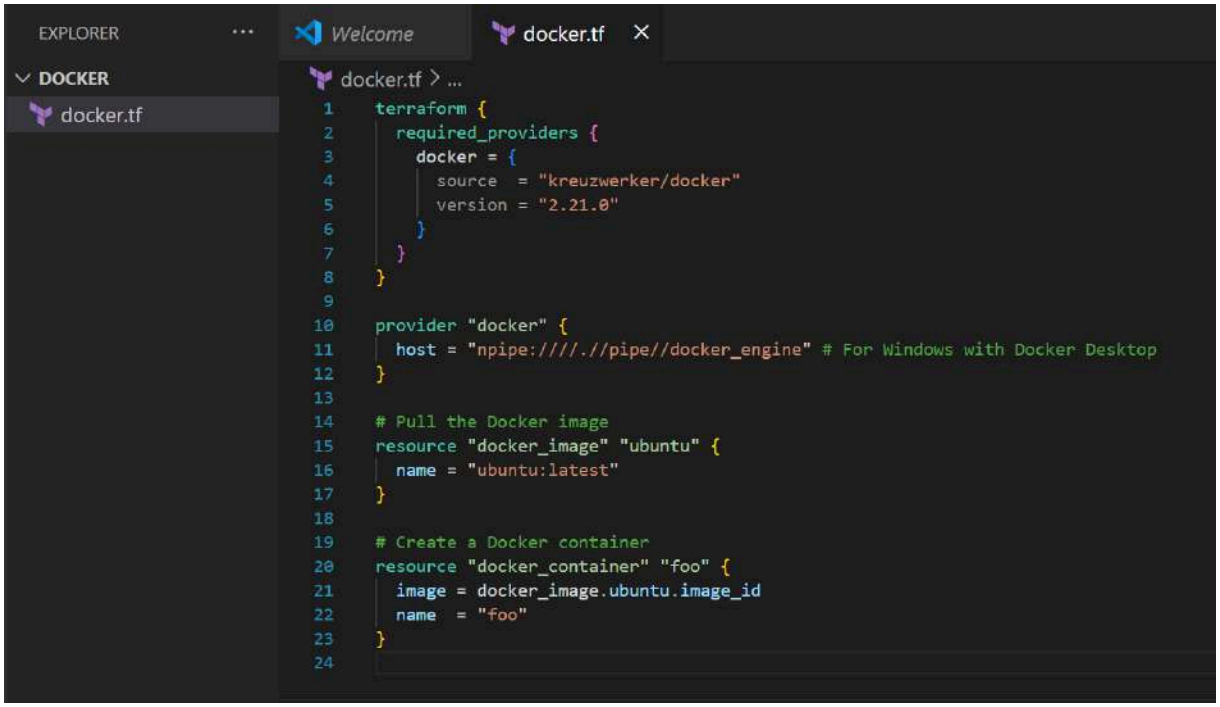
AIM: Creating docker image using terraform

Step 1: Check the docker functionality

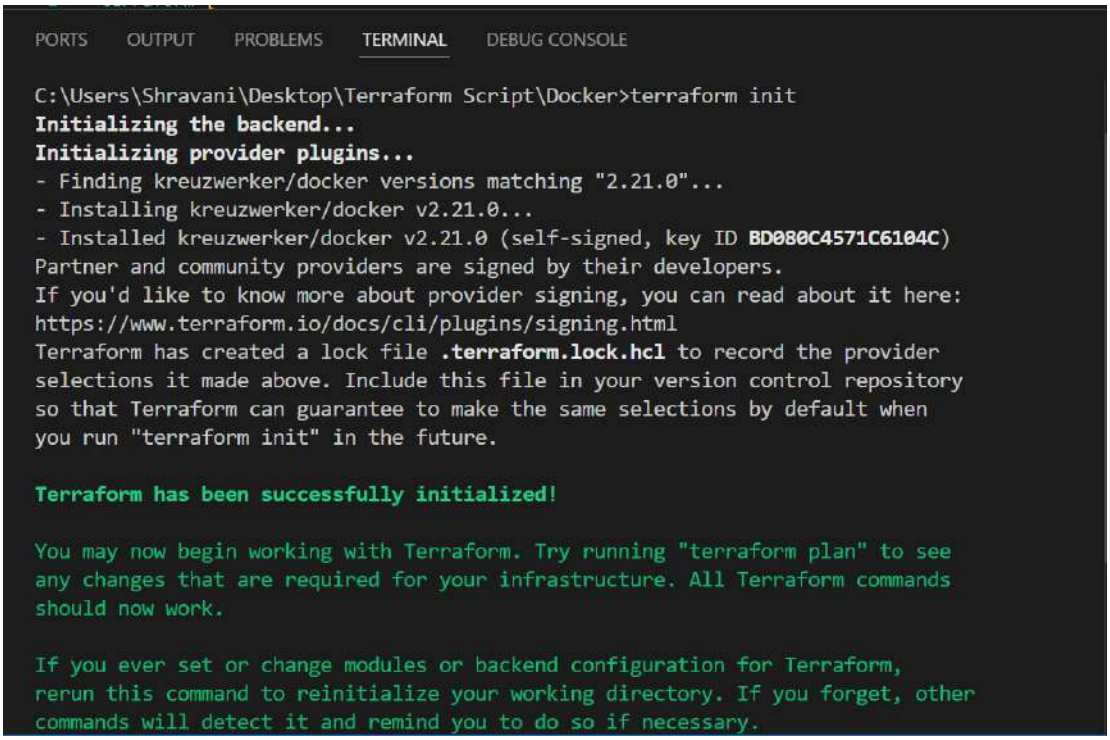




Step 2: Firstly create a new folder named ‘Docker’ in the ‘TerraformScripts’ folder. Then create a new docker.tf file using Atom editor and write the followingcontents into it to create a Ubuntu Linux container.



Step 3: Execute Terraform Init command to initialize the resources



Step 4: Execute Terraform plan to see the available resources

```
PORTS   OUTPUT   PROBLEMS   TERMINAL   DEBUG CONSOLE

C:\Users\Shravani\Desktop\Terraform Script\Docker>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.foo will be created
  + resource "docker_container" "foo" {
      + attach           = false
      + bridge           = (known after apply)
      + command          = (known after apply)
      + container_logs   = (known after apply)
      + entrypoint       = (known after apply)
      + env              = (known after apply)
      + exit_code        = (known after apply)
      + gateway          = (known after apply)
      + hostname         = (known after apply)
      + id               = (known after apply)
      + image            = (known after apply)
      + init             = (known after apply)
      + ip_address       = (known after apply)
```

```
PORTS   OUTPUT   PROBLEMS   TERMINAL   DEBUG CONSOLE

      + healthcheck (known after apply)

      + labels (known after apply)
    }

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

Plan: 2 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.

C:\Users\Shravani\Desktop\Terraform Script\Docker>
```

Step 5: Execute Terraform apply to apply the configuration, which will automatically create and run the Ubuntu Linux container based on our configuration. Using command : “terraform apply”

Docker images, Before Executing Apply step:

```
C:\Users\Shravani\Desktop\Terraform Script\Docker>docker images
REPOSITORY   TAG       IMAGE ID   CREATED   SIZE
```

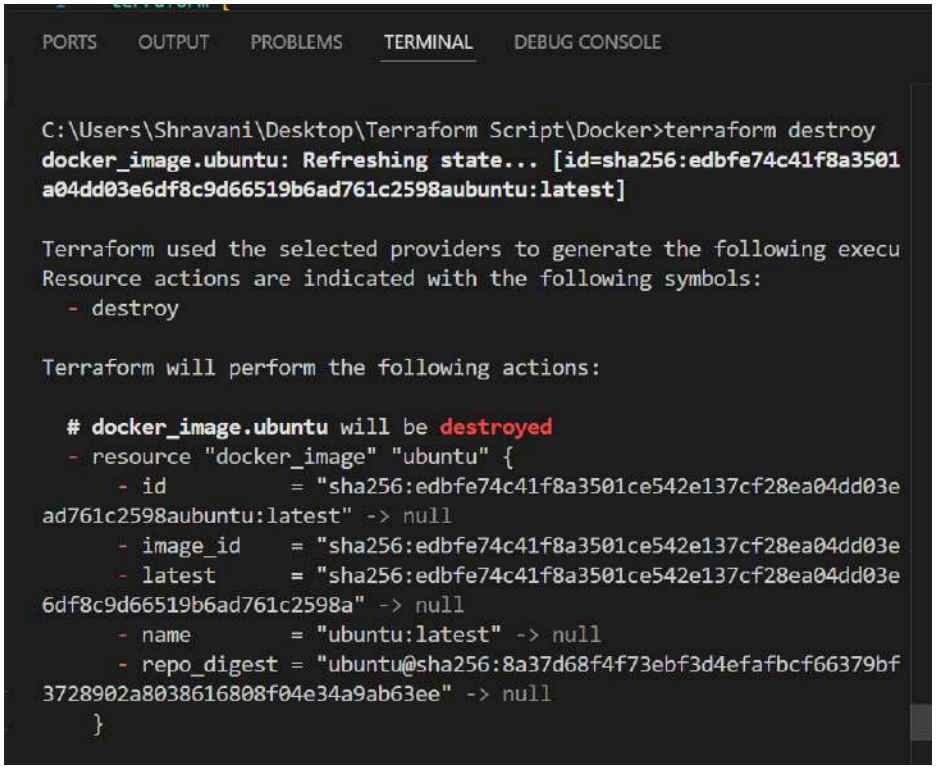
terraform apply

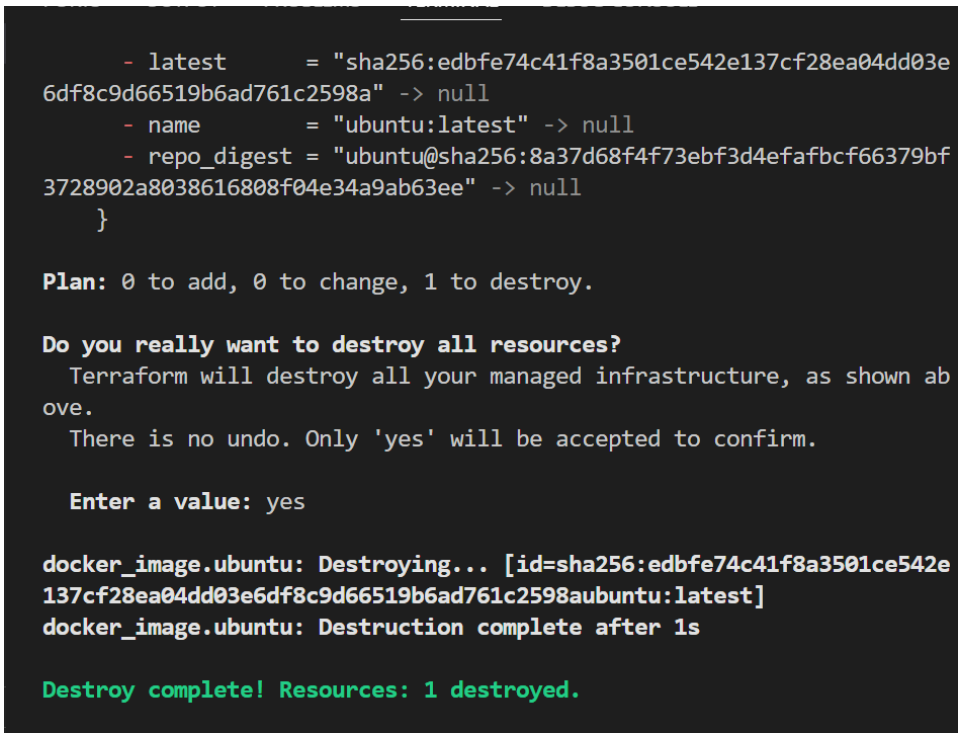
```
PORTS   OUTPUT   PROBLEMS   TERMINAL   DEBUG CONSOLE

C:\Users\Shravani\Desktop\Terraform Script\Docker>terraform apply
docker_image.ubuntu: Refreshing state... [id=sha256:edbfe74c41f8a3501ce542e137cf28e
3e6df8c9d66519b6ad761c2598aubuntu:latest]

Note: Objects have changed outside of Terraform

Terraform detected the following changes made outside of Terraform since the last
"terraform apply" which may have affected this plan:

  # docker_image.ubuntu has been deleted
  - resource "docker_image" "ubuntu" {
        id          = "sha256:edbfe74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6
2598aubuntu:latest"
      - image_id    = "sha256:edbfe74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6
2598a" -> null
        name        = "ubuntu:latest"
        # (2 unchanged attributes hidden)
    }


Unless you have made equivalent changes to your configuration, or ignored the relev
attributes using ignore_changes, the following plan may include actions to undo or
respond to these changes.
```

Docker images, After Executing Apply step:

```
C:\Users\Shravani\Desktop\Terraform Script\Docker>docker images
REPOSITORY   TAG       IMAGE ID       CREATED       SIZE
ubuntu       latest    edbfe74c41f8   3 weeks ago   78.1MB
```

Step 6: Execute Terraform destroy to delete the configuration, which will automatically delete the Ubuntu Container.





Docker images After Executing Destroy step

```
Destroy complete! Resources: 1 destroyed.

C:\Users\Shravani\Desktop\Terraform Script\Docker>docker images
REPOSITORY   TAG       IMAGE ID   CREATED   SIZE
```

SHRAVANI RASAM D15A 46

ADVANCED DEVOPS EXP 7

Static Application Security Testing (SAST)

SAST is a method of security testing that analyzes source code to identify vulnerabilities **without executing the program**. It is also known as **white-box testing**.

SAST Process Breakdown

1. **Code Parsing**
    - The source code is parsed to create an **Abstract Syntax Tree (AST)**, which represents the code structure.
2. **Pattern Matching**
    - The AST is analyzed using predefined rules to detect patterns that may indicate security vulnerabilities.
3. **Data Flow Analysis**
    - This step examines how data moves through the code to identify potential security issues like **SQL Injection** or **Cross-Site Scripting (XSS)**.
4. **Control Flow Analysis**
    - Involves analyzing the paths that the code execution might take to find logical errors or vulnerabilities.
5. **Reporting**
    - The tool generates a report highlighting the vulnerabilities found, their severity, and recommendations for fixing them.

Benefits of SAST

- **Early Detection**
    - Identifies vulnerabilities early in the development lifecycle, reducing the cost and effort required to fix them.
- **Comprehensive Coverage**
    - Can analyze **100% of the codebase**, including all possible execution paths.
- **Automated and Scalable**
    - Suitable for large codebases and can be integrated into **CI/CD pipelines** for continuous monitoring.

SHRAVANI RASAM D15A 46

SonarQube and SAST

SonarQube is a popular tool that provides static code analysis to detect bugs, code smells, and security vulnerabilities. Here's how SonarQube fits into the SAST process:

1. **Integration**
   - SonarQube can be integrated into your **CI/CD pipeline** to automatically analyze code every time it is committed.
2. **Rule Sets**
   - It uses a comprehensive set of rules to detect **security vulnerabilities**, **coding standards violations**, and **code quality issues**.
3. **Detailed Reporting**
   - SonarQube generates detailed reports that help developers understand and fix the identified issues efficiently.
4. **Continuous Feedback**
   - Provides continuous feedback to developers, enabling them to maintain high code quality and security standards throughout the development process.
5. **Customization**
   - Allows customization of rule sets to match the specific needs and standards of your project or organization.

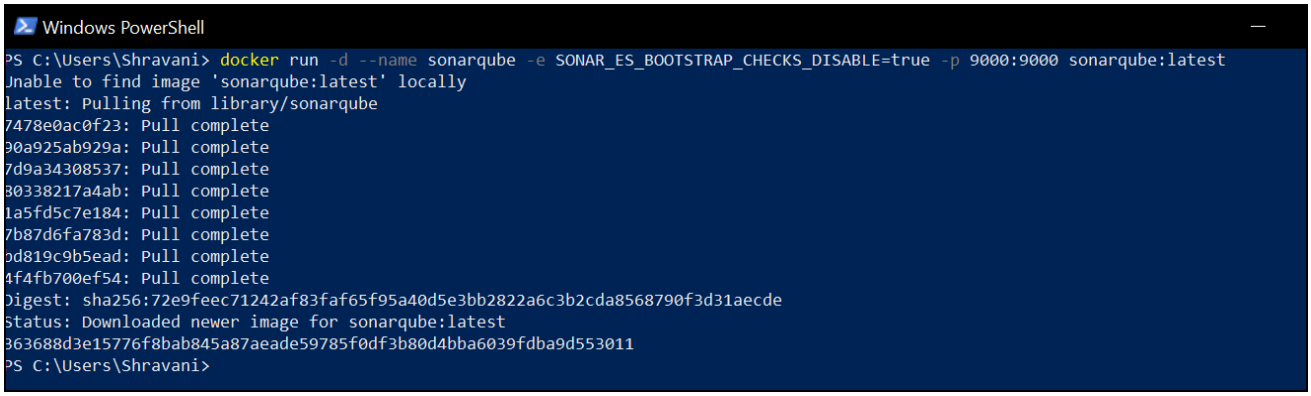
Implementation:

1. Open Jenkins Dashboard

● Access your Jenkins Dashboard by navigating to http://localhost:8080 (or the port you have configured Jenkins to run on).

2. Run SonarQube in a Docker Container

● Open a terminal and run the following command to start SonarQube in a Docker container Command - docker run -d --name sonarqube -e SONAR_ES_BOOTSTRAP_CHECKS_DISABLE=true -p 9000:9000 sonarqube:latest

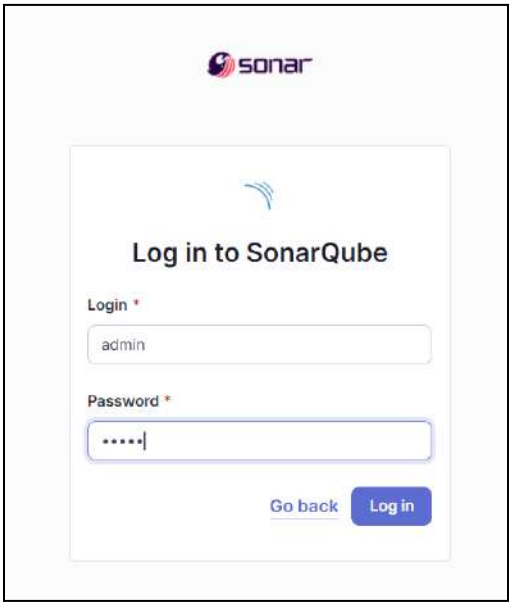


3. Check SonarQube Status

● Once the container is up and running, check the status of SonarQube by navigating to http://localhost:9000

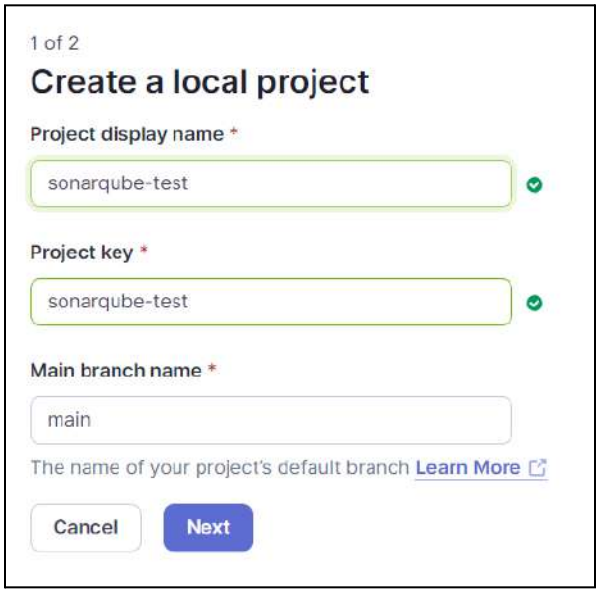
4. Login to SonarQube

● Use the default credentials to log in:
  ○ Username: admin
  ○ Password: admin



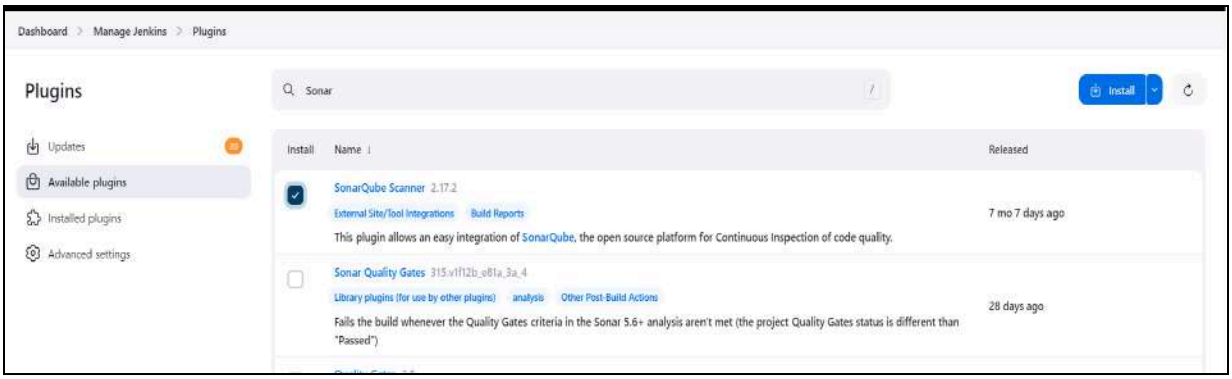
5. Create a Project in SonarQube

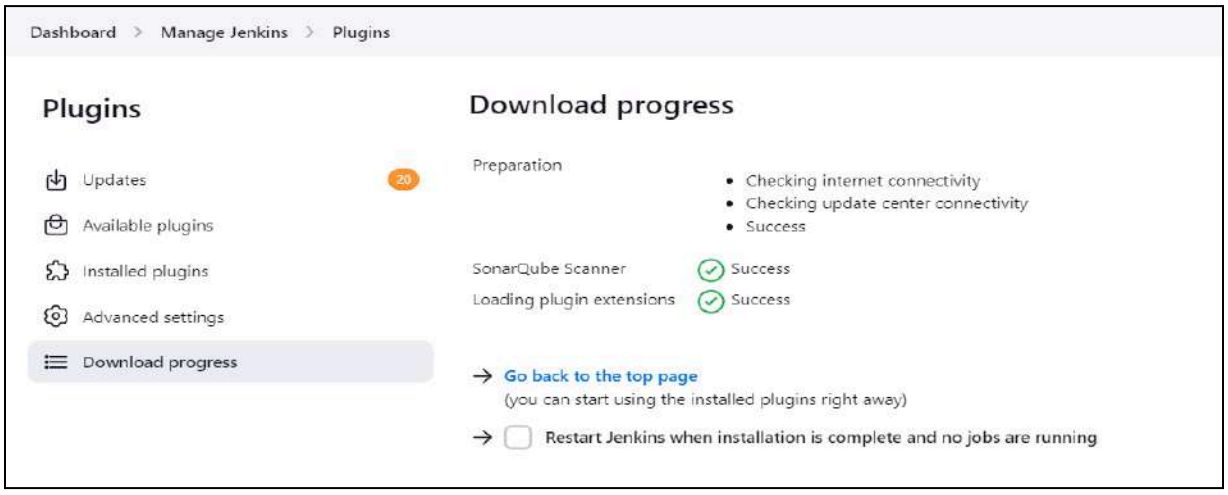
● Create a new project manually in SonarQube and name it sonarqube



6. Install SonarQube Scanner for Jenkins

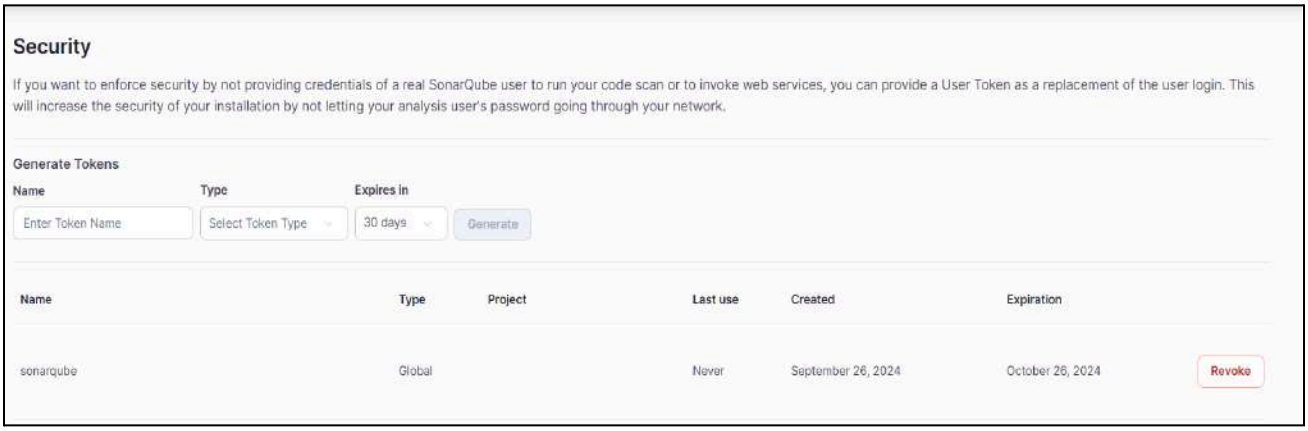
- Go back to the Jenkins Dashboard.
- Navigate to Manage Jenkins > Manage Plugins.
- Search for SonarQube Scanner for Jenkins and install it.





7. Configure SonarQube in Jenkins

- Go to Manage Jenkins > Configure System
- Scroll down to the SonarQube Servers section and enter the required details:
  - **Name**: Any name you prefer.
  - **Server URL**: http://localhost:9000
  - **Server Authentication Token**: (Generate this token in SonarQube under My Account > Security > Generate Tokens).
  - **Add Jenkins**: Select Kind - Secret Text > Secret (Paste Generated Token)

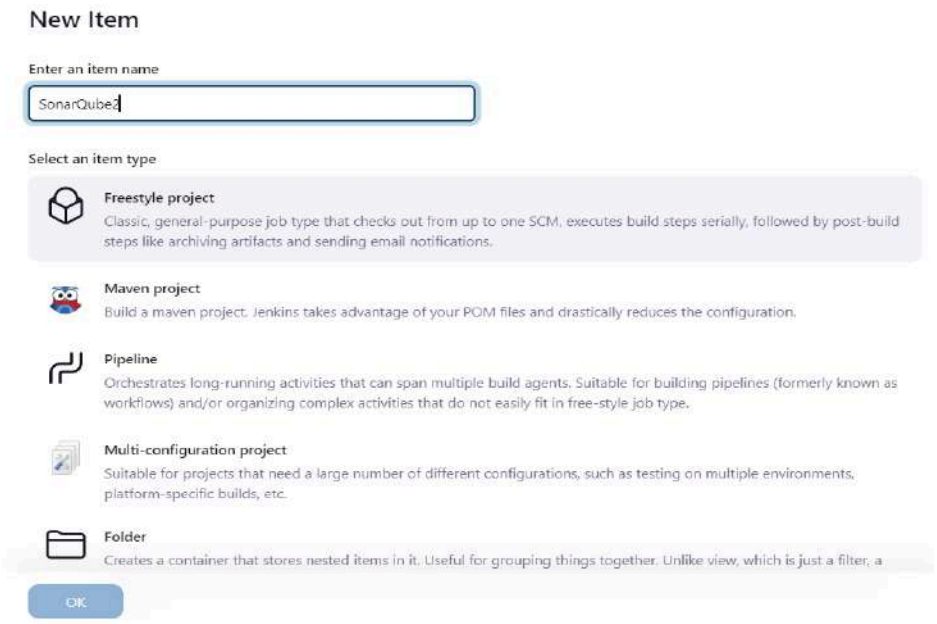


8. Configure SonarQube Scanner in Jenkins

- Go to Manage Jenkins > Global Tool Configuration.
- Scroll down to SonarQube Scanner.
- Choose the latest version and select Install automatically

9. Create a New Jenkins Job

- In Jenkins, create a new item and select Freestyle project.
- Under Source Code Management, choose Git and enter the repository URL: https://github.com/shazforiot/MSBuild_firstproject.git



10. Configure Build Steps

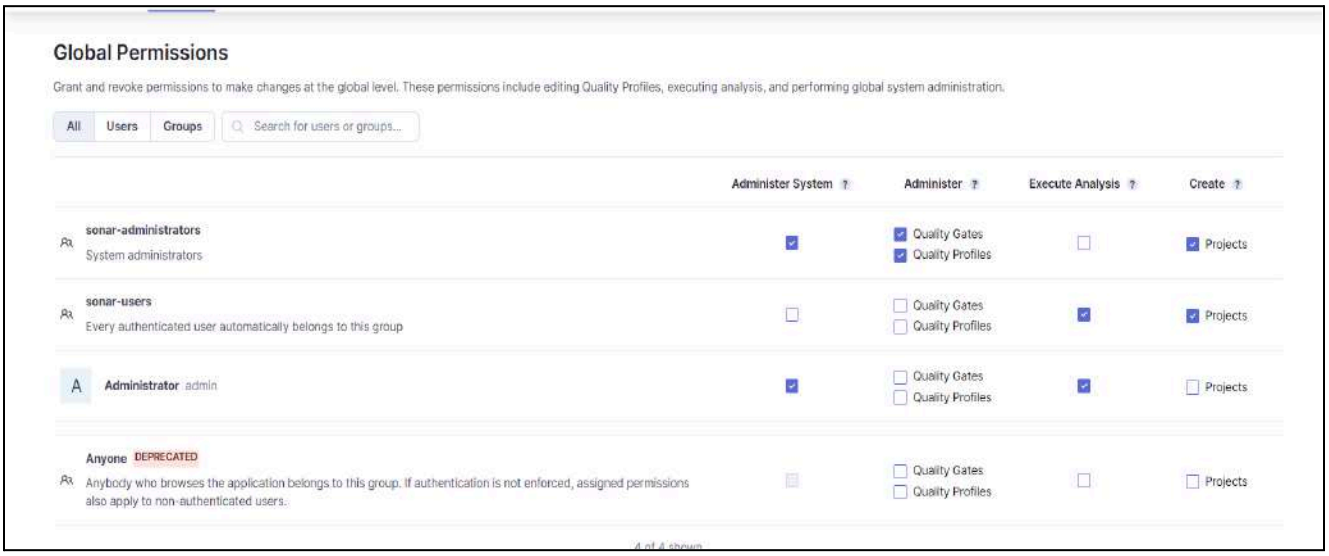
- Under the Build section, add a build step to Execute SonarQube Scanner

. ● Enter the following analysis properties:

  - sonar.projectKey=my_project_name
  - sonar.login=your_generated_token
  - sonar.sources=HelloWorldCore
  - sonar.host.url=http://localhost:9000

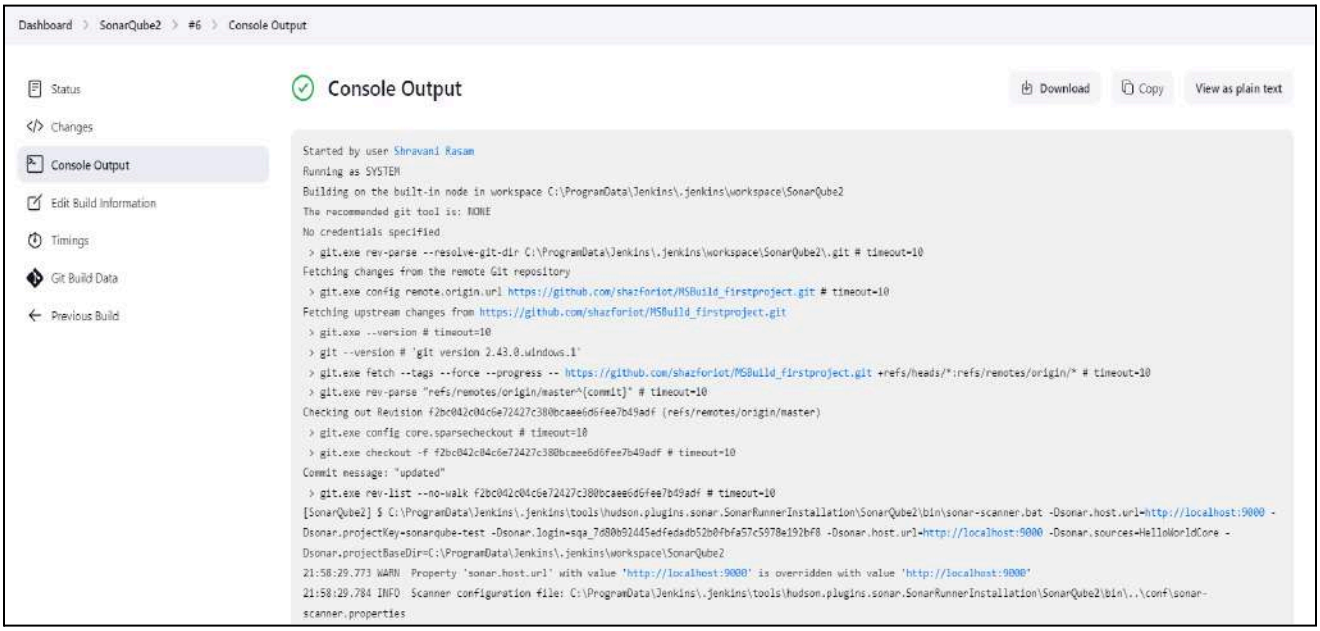
11. Set Permissions in SonarQube

- Navigate to http://localhost:9000/<user-name>/permissions.
- Allow Execute Permissions to the Admin user.



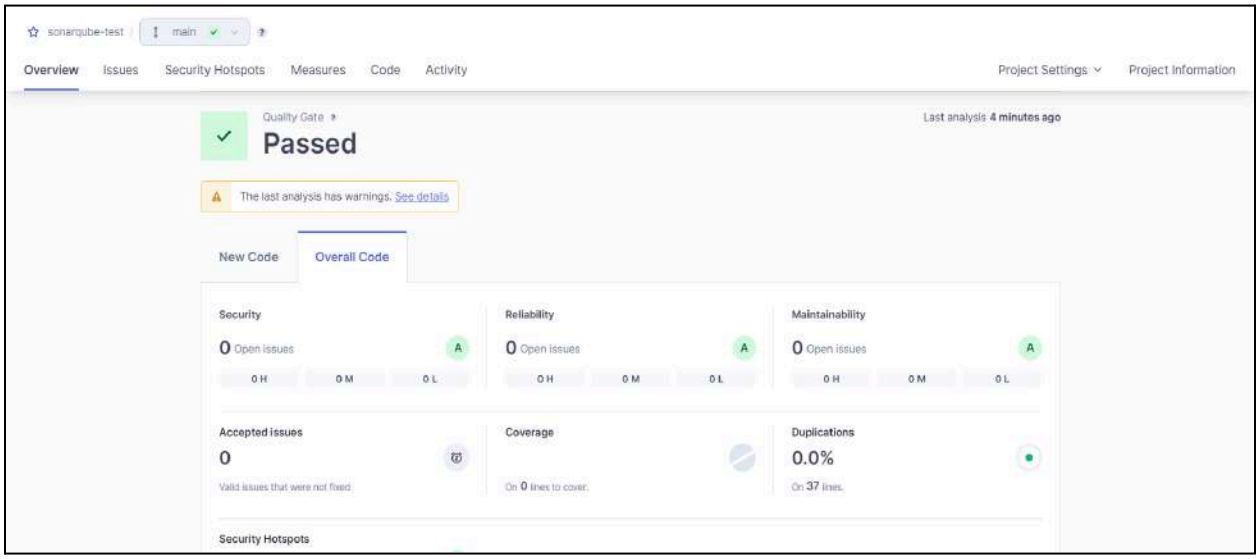
12. Run the Build

- Go back to Jenkins and run the build.
- Check the console output for any errors or issues.



13. Verify in SonarQube

- Once the build is complete, check the project in SonarQube to see the analysis results.



ADVANCED DEVOPS EXP 8

AIM: Create a Jenkins CICD Pipeline with SonarQube / GitLab Integration to perform a static analysis of the code to detect bugs, code smells, and security vulnerabilities on a sample Web /Java / Python application.

THEORY:

Static Application Security Testing (SAST)
SAST is a methodology for testing an application's source code to identify security vulnerabilities before the code is compiled. This type of testing, also referred to as white-box testing, helps improve application security by finding weaknesses early in development.

Problems SAST Solves

- **Early Detection**: SAST finds vulnerabilities early in the Software Development Life Cycle (SDLC), allowing developers to fix issues without affecting builds or passing vulnerabilities to the final release.
- **Real-Time Feedback**: Developers receive immediate feedback during coding, helping them address security issues before moving to the next stage of development.
- **Graphical Representations**: SAST tools often provide visual aids to help developers navigate the code and identify the exact location of vulnerabilities, offering suggestions for fixes.
- **Regular Scanning**: SAST tools can be configured to scan code regularly, such as during daily builds, code check-ins, or before releases.

Importance of SAST

- **Resource Efficiency**: With a larger number of developers than security experts, SAST allows full codebase analysis quickly and efficiently, without relying on manual code reviews.
- **Speed**: SAST tools can analyze millions of lines of code within minutes, detecting critical vulnerabilities such as buffer overflows, SQL injection, and cross-site scripting (XSS) with high accuracy.

CI/CD Pipeline
A Continuous Integration/Continuous Delivery (CI/CD) pipeline is a sequence of automated tasks designed to build, test, and deploy new software versions rapidly and consistently. It plays a crucial role in DevOps practices, ensuring fast and reliable software releases.

SonarQube
SonarQube is an open-source platform from SonarSource that performs continuous code quality inspections through static code analysis. It identifies bugs, code smells, security vulnerabilities, and code

duplications in a wide range of programming languages. SonarQube is extendable with plugins and integrates seamlessly into CI/CD pipelines.

Benefits of SonarQube

- **Sustainability**: By reducing complexity and vulnerabilities, SonarQube extends the lifespan of applications and helps maintain cleaner code.
- **Increased Productivity**: SonarQube minimizes maintenance costs and risks, resulting in fewer code changes and a more stable codebase.
- **Quality Code**: Ensures code quality checks are integrated into the development process.
- **Error Detection**: Automatically identifies coding errors and alerts developers to resolve them before moving to production.
- **Consistency**: Helps maintain consistent code quality by detecting and reporting violations of coding standards.
- **Business Scaling**: SonarQube supports scaling as the business grows without any restrictions.

Implementation:

Prerequisites

1. Jenkins installed on your machine.
2. Docker installed to run SonarQube.
3. SonarQube installed via Docker

1. Set Up Jenkins

- Open Jenkins Dashboard on localhost:8080 or your configured port
- Install the necessary plugins:
  - SonarQube Scanner Plugin

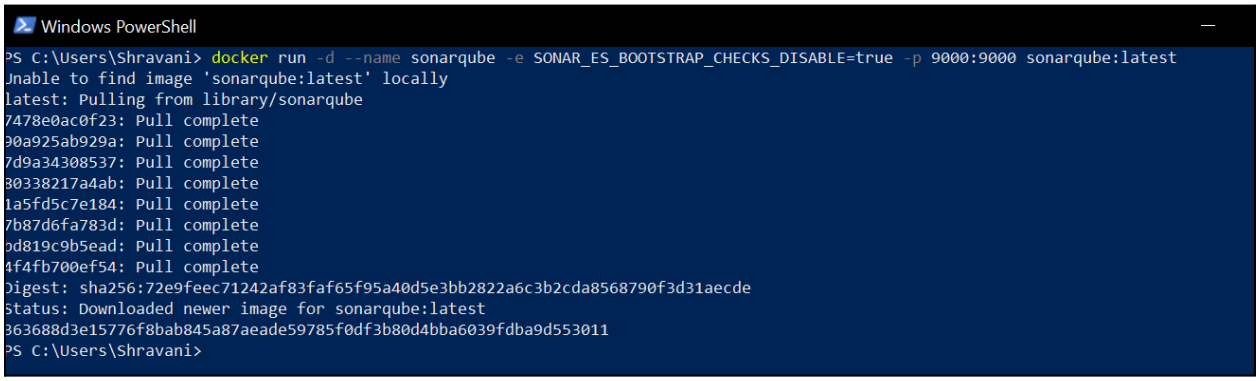
.2. Run SonarQube in Docker

Run the following command to start SonarQube in a Docker container:

command :

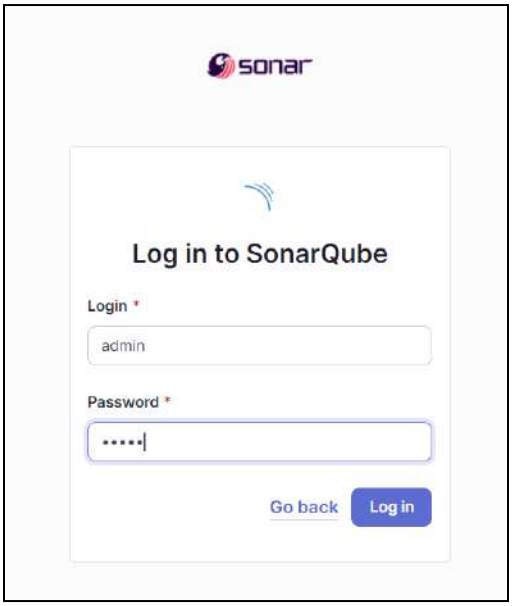
docker run -d --name sonarqube -e SONAR_ES_BOOTSTRAP_CHECKS_DISABLE=true - p 9000:9000 sonarqube:latest

- Check SonarQube status at http://localhost:9000.
- Login with your credentials:



3. Create a Project in SonarQube

- Go to Projects > Create Project.
- Name the project (e.g., sonarqube-test)

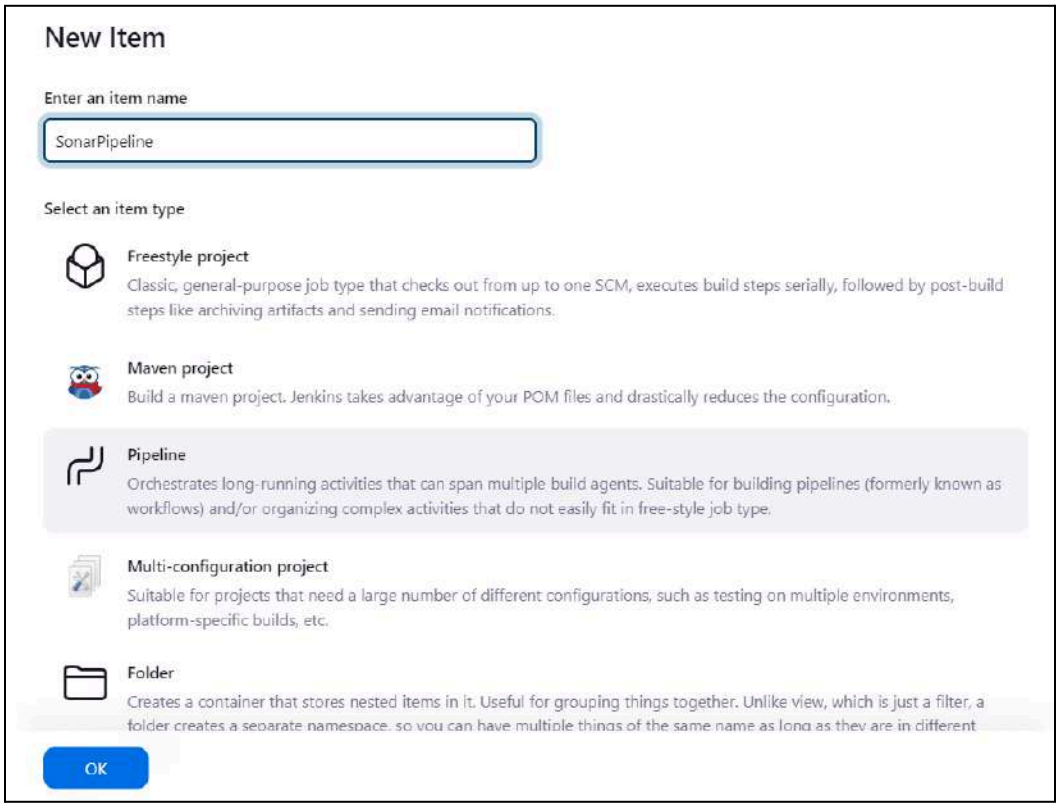


4. Generate SonarQube Token

- Go to My Account > Security > Generate Tokens.
- Copy the generated token for later use

5. Create a Jenkins Pipeline

- Go to Jenkins Dashboard, click New Item, and select Pipeline.



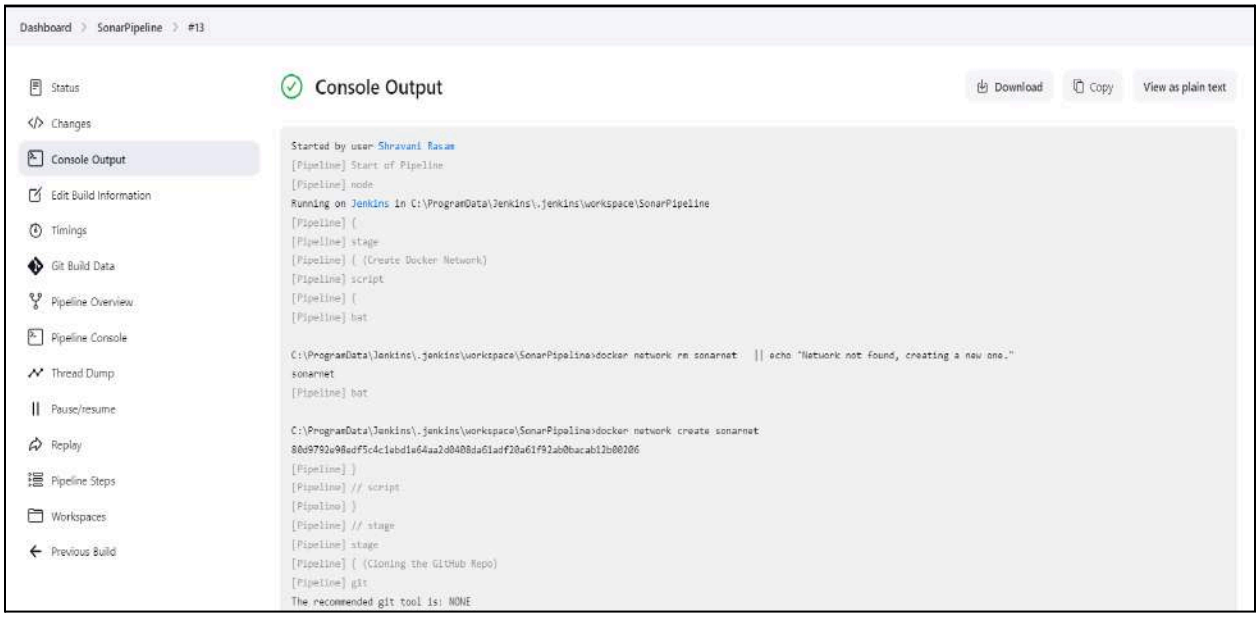
6. Under Pipeline Script, enter the following script:

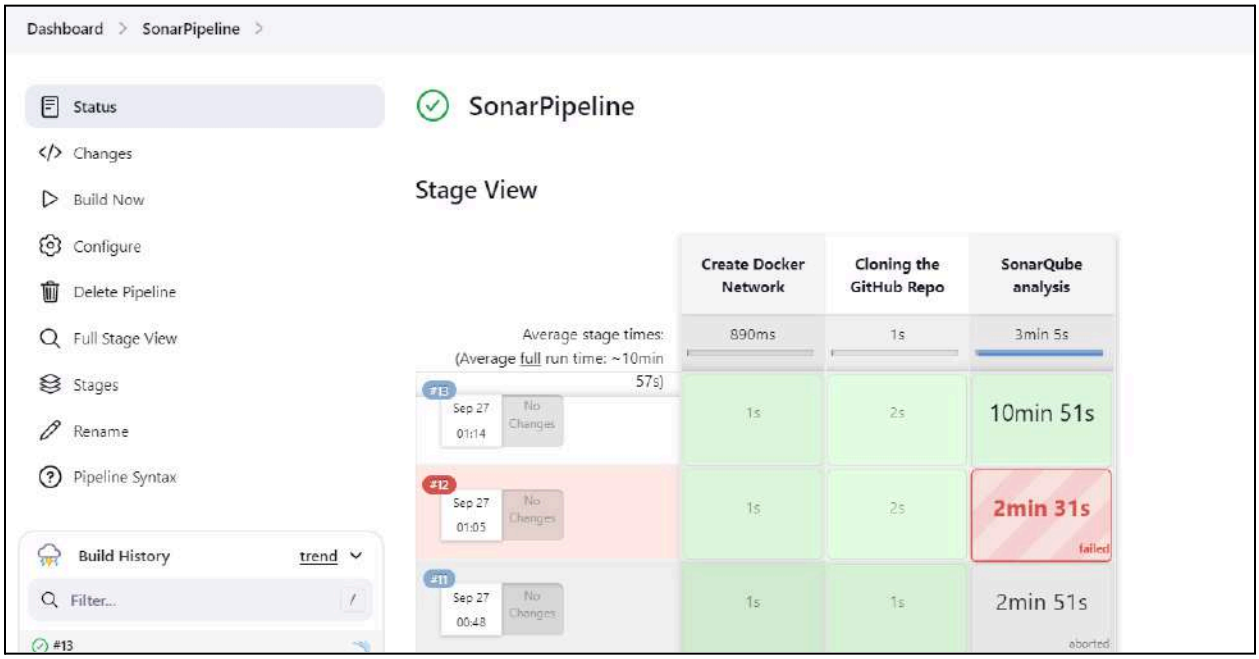
```
pipeline {
    agent any

    stages {
        stage('Create Docker Network') {
            steps {
                script {
                    bat 'docker network rm sonarnet || echo "Network not found, creating a new one."'
                    bat 'docker network create sonarnet'
                }
            }
        }

        stage('Cloning the GitHub Repo') {
            steps {
                git 'https://github.com/shazforiot/GOL.git'
```

```
        }
    }

    stage('SonarQube analysis') {
        steps {
            withSonarQubeEnv('sonarqube') {
                bat """
                docker run --rm --network sonarnet ^
                    -e SONAR_HOST_URL=http://192.168.133.16:9000 ^
                    -e SONAR_LOGIN=admin ^
                    -e SONAR_PASSWORD=Shravani@0212 ^
                    -e SONAR_PROJECT_KEY=sonarqube-test ^
                    -v ${WORKSPACE}:/usr/src ^
                    sonarsource/sonar-scanner-cli ^
                    -Dsonar.projectKey=sonarqube-test ^
                    -Dsonar.exclusions=vendor/**,resources/**,**/*.java ^
                    -Dsonar.login=admin ^
                    -Dsonar.password=Shravani@0212
                """
            }
        }
    }
  }
}
```

7. Run the Pipeline

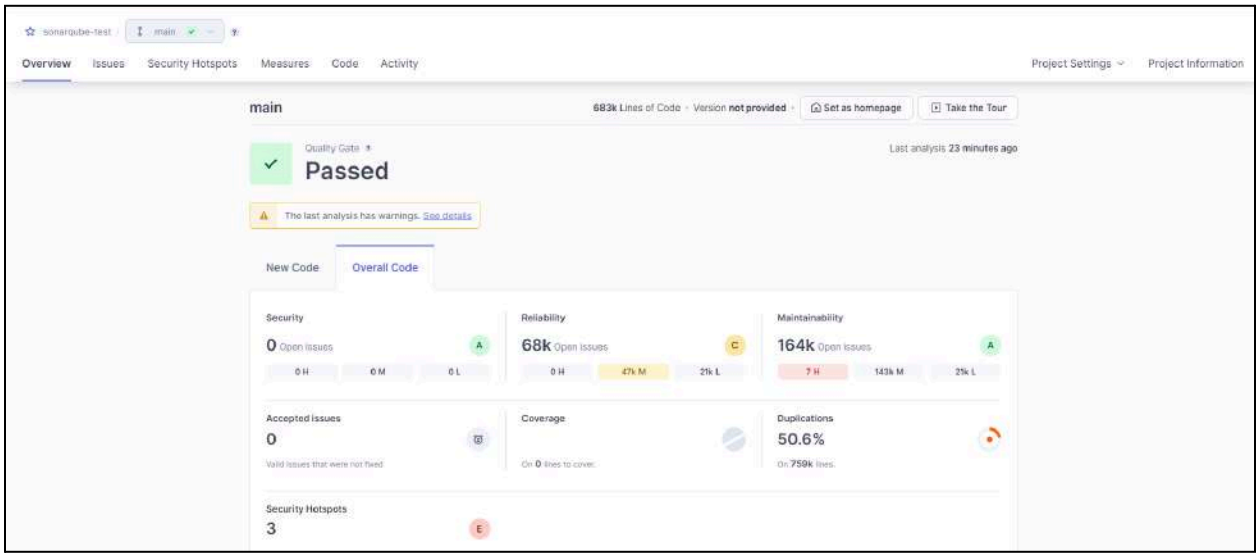
- Save the pipeline and click Build Now
- Monitor the console output for any errors





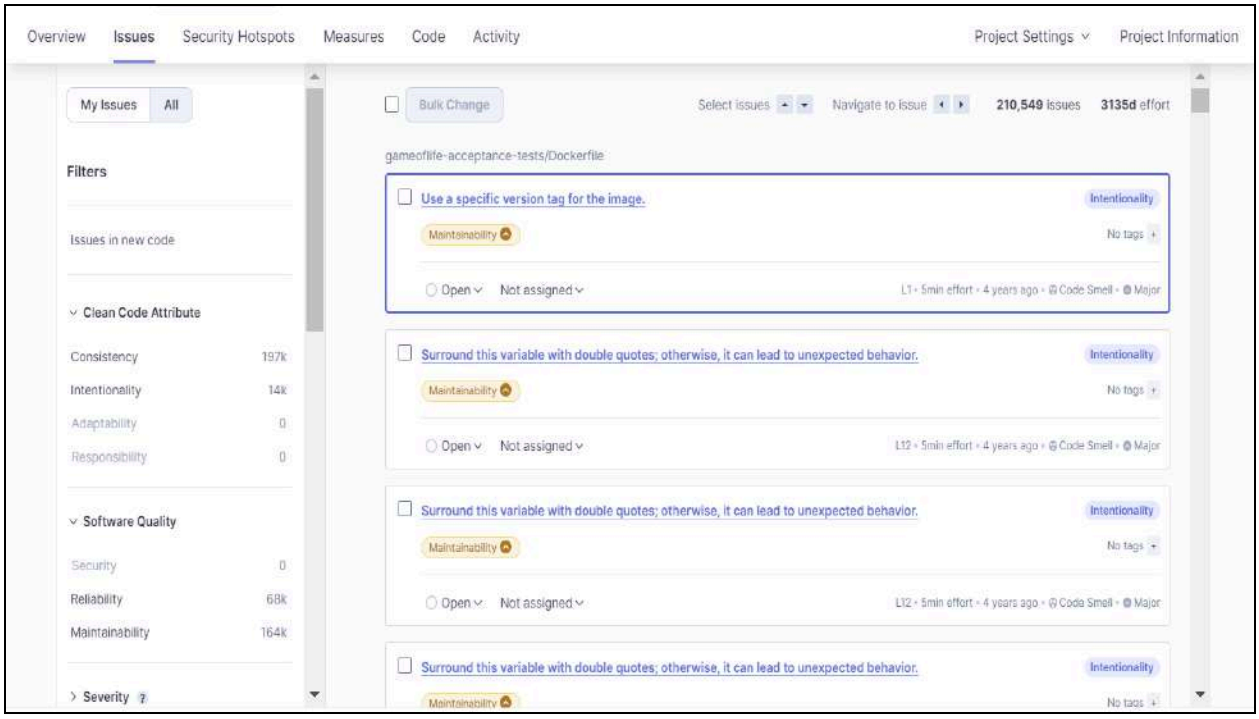
9. Check SonarQube for Analysis Results

● Go to your SonarQube dashboard and check the project for issues such as bugs, code smells, and security vulnerabilities.

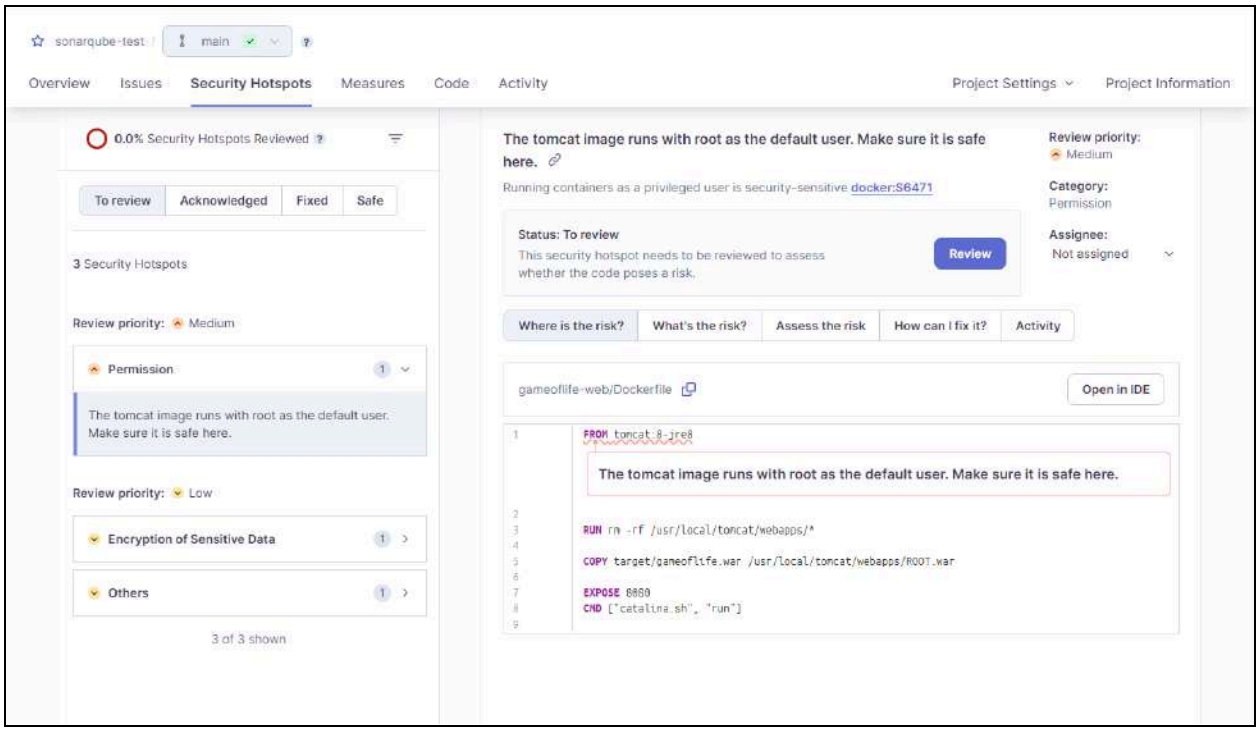


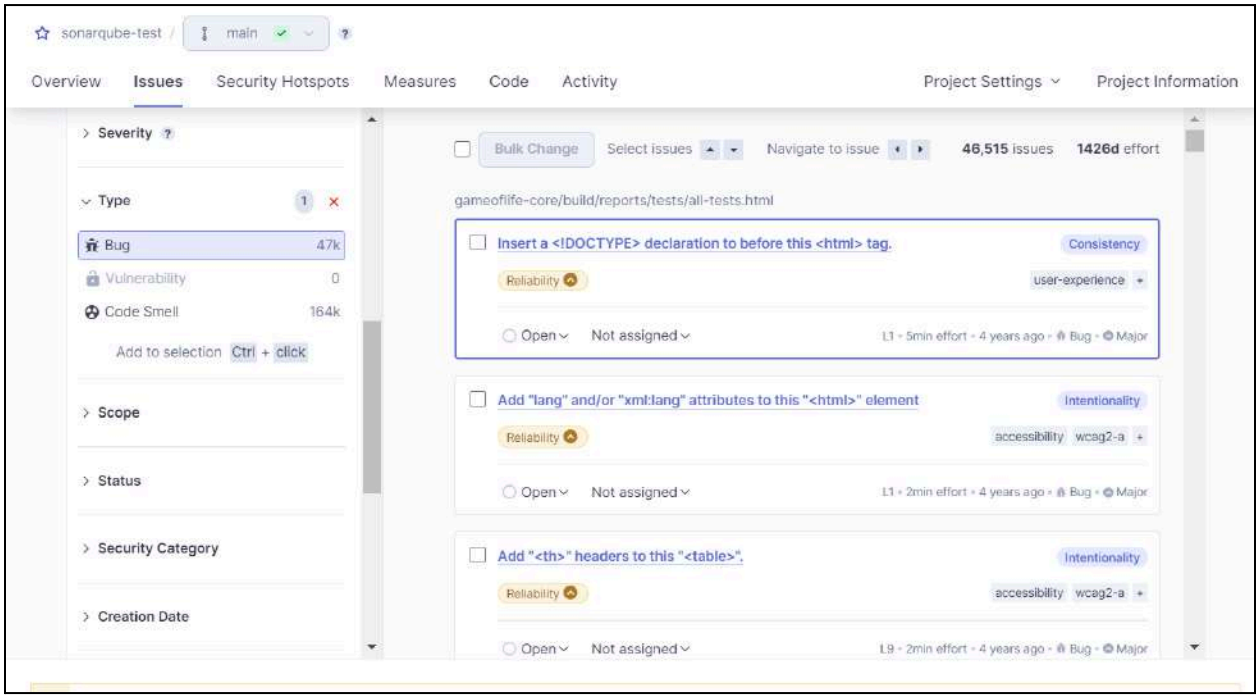
10. Checking SonarQube for Analysis Results of a Code File with Bugs , Code Smells, Security Vulnerabilities, Cyclomatic Complexities and Duplicates .

● Issues -

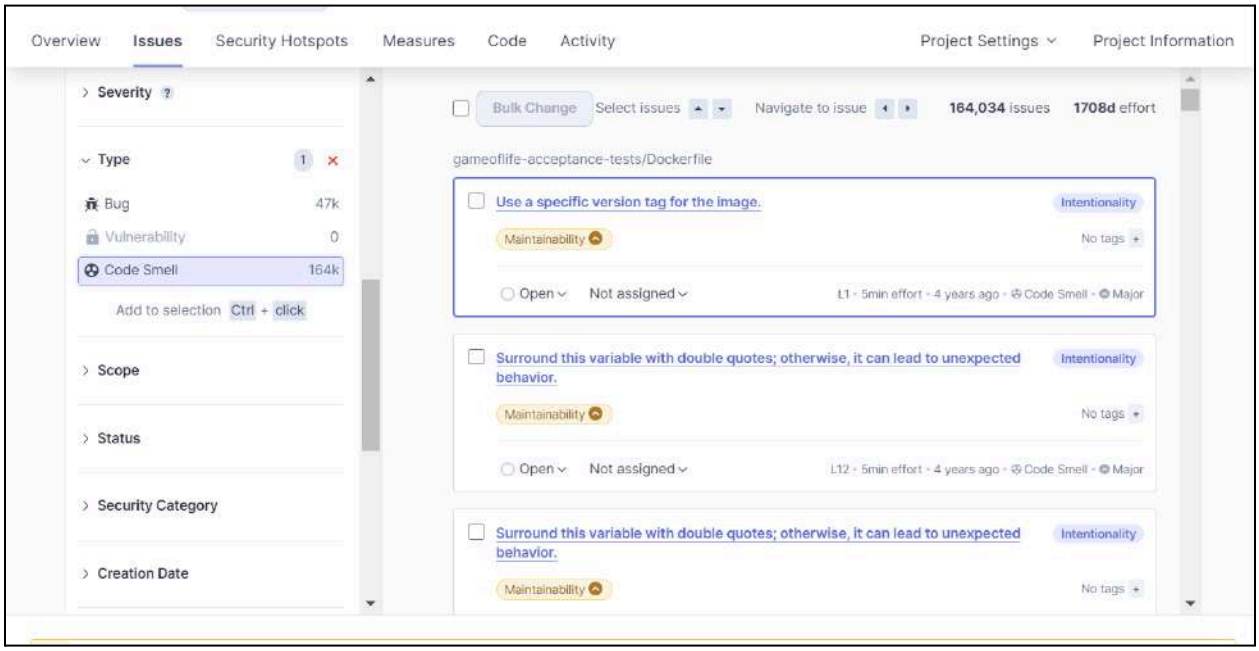


SHRAVANI RASAM D15A 46

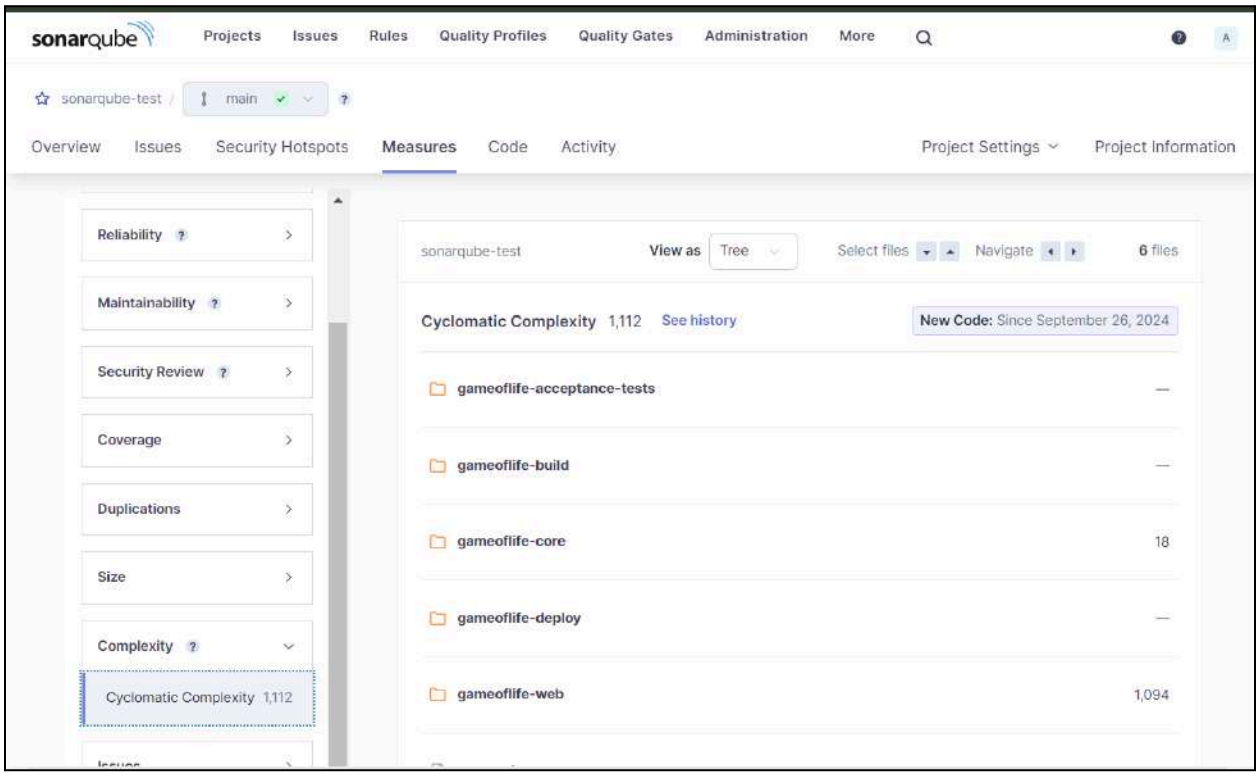


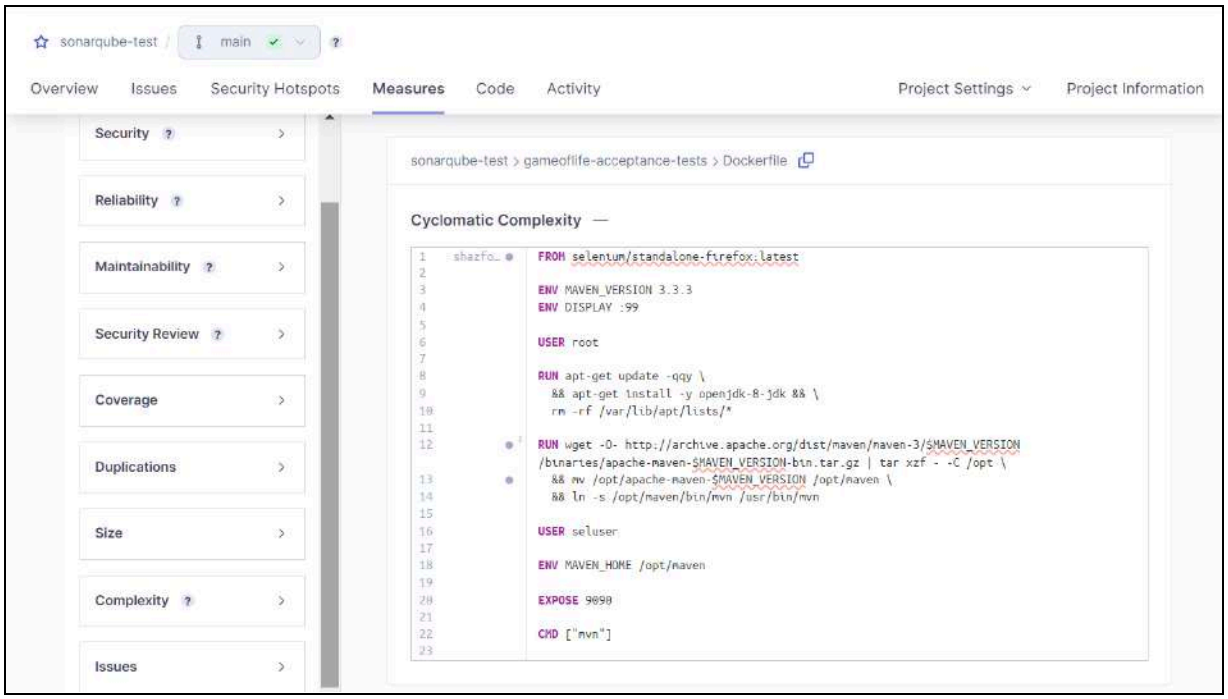


● Security Hotspot (Security Vulnerabilities) -

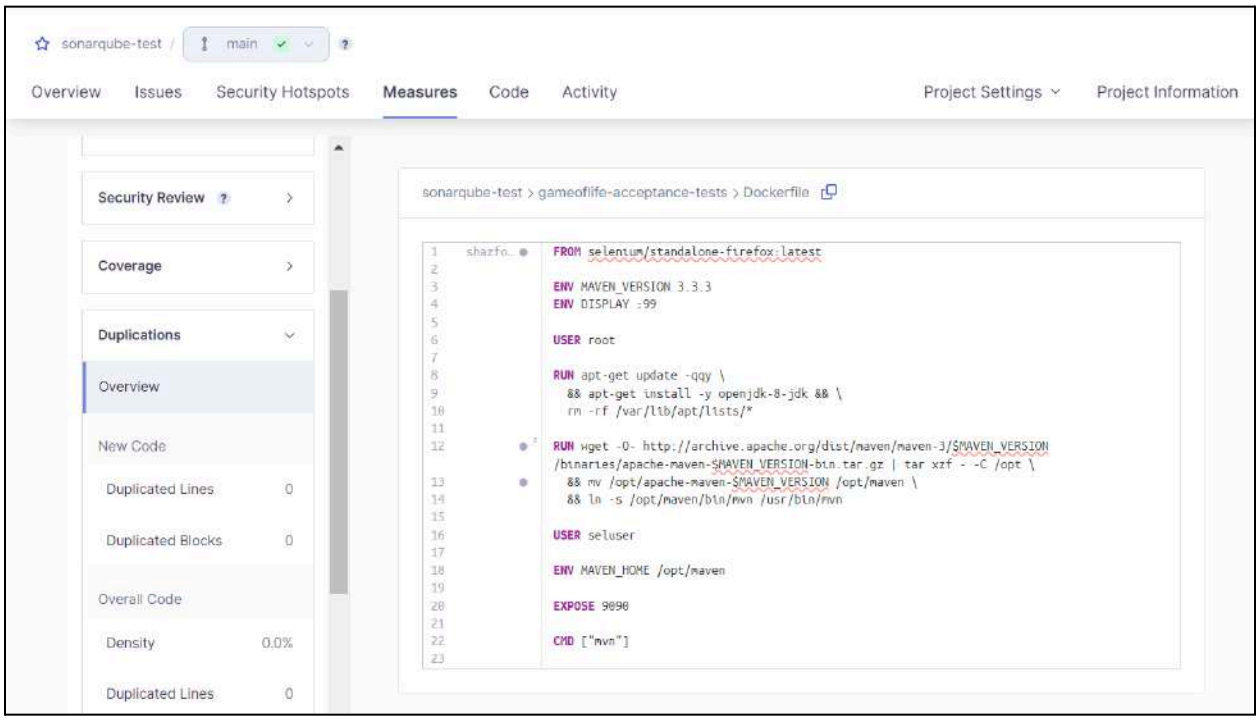


Cyclomatic Complexity -





● Duplications -



Conclusion:

In this experiment, we performed a static analysis of the code to detect bugs, code smells, and security vulnerabilities on our sample codes.

ADVANCED DEVOPS EXP 9

Aim: To Understand Continuous monitoring and Installation and configuration of Nagios Core, Nagios Plugins and NRPE (Nagios Remote Plugin Executor) on Linux Machine.

Theory:

What is Nagios?

Nagios is an open-source monitoring tool designed to monitor systems, networks, and infrastructure. It helps organizations identify and resolve IT infrastructure issues before they affect critical business processes. Nagios provides monitoring and alerting services for servers, switches, applications, and services.

Key Features of Nagios

- **Monitoring**: Nagios can monitor a wide range of network services (HTTP, SMTP, POP3, etc.), host resources (processor load, disk usage, system logs, etc.), and environmental factors (temperature, humidity, etc.).
- **Alerting:** When an issue is detected, Nagios can send alerts via email, SMS, or custom scripts to notify administrators.
- **Reporting:** Nagios provides detailed reports and logs of outages, events, notifications, and alert responses, helping in historical analysis and SLA compliance.
- **Scalability:** Nagios is designed to scale and can monitor large, complex environments. 5. Flexibility: With a wide range of plugins and add-ons, Nagios can be customized to meet specific monitoring needs.

How Nagios Works

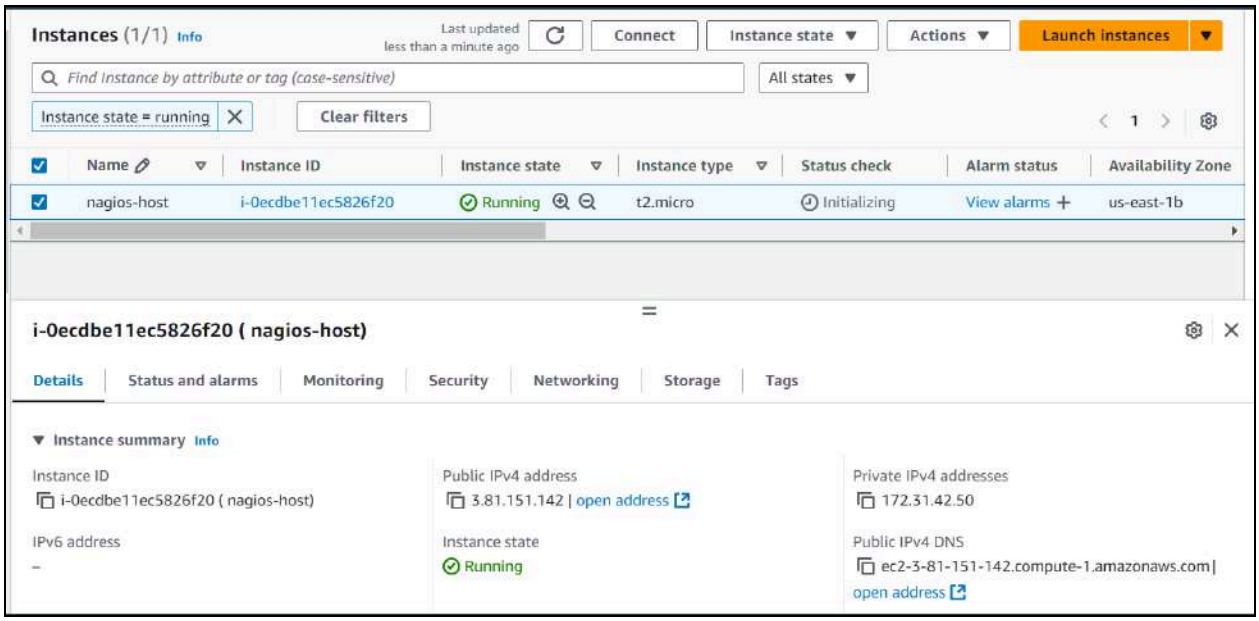
- **Configuration:** Administrators configure Nagios to monitor specific services and hosts. This involves defining what to monitor, how to monitor it, and what actions to take when issues are detected.
- **Plugins:** Nagios uses plugins to gather information about the status of various services and hosts. These plugins can be custom scripts or pre-built ones available in the Nagios community.
- **Scheduling:** Nagios schedules regular checks of the defined services and hosts using the configured plugins.
- **Alerting**: If a check indicates a problem, Nagios triggers an alert. Alerts can be configured to escalate if not acknowledged within a certain timeframe. 5. Web Interface: Nagios provides a web interface for viewing the status of monitored services and hosts, acknowledging alerts, and generating reports.

Setting Up Nagios

1. **Installation**: Install Nagios on a server, typically a Linux-based system.
2. **Configuration Files**: Edit configuration files to define what to monitor and how to monitor it. This includes defining hosts, services, contacts, and notification methods.
3. **Plugins**: Install and configure necessary plugins to monitor specific services and hosts.
4. **Web Interface**: Set up the web interface to allow easy access to monitoring data and alert management.
5. **Testing**: Test the configuration to ensure that Nagios is correctly monitoring the defined services and hosts and that alerts are being sent as expected

1. Create an Amazon Linux EC2 Instance

- Name it nagios-host.

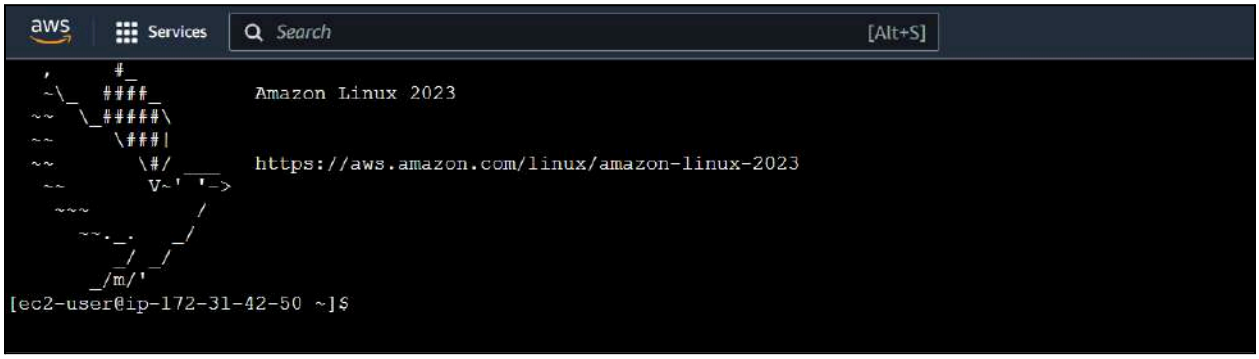


2. Configure Security Group

- Ensure HTTP, HTTPS, SSH, and ICMP are open from everywhere.
- Edit the inbound rules of the specified Security Group

3. Connect to Your EC2 Instance

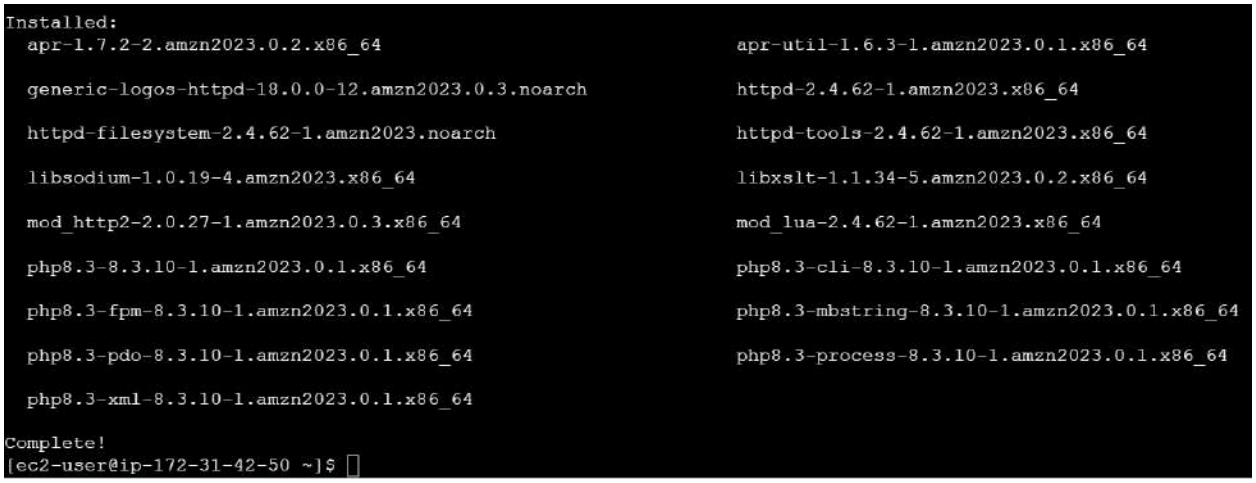
● SSH into your EC2 instance or use EC2 Instance Connect from the browse



4. Update Package Indices and Install Required Packages

Commands -

- sudo yum update sudo yum install httpd php
- sudo yum install gcc glibc glibc-common
- sudo yum install gd gd-devel



```
[ec2-user@ip-172-31-42-50 ~]$ sudo yum install -y gcc glibc glibc-common
Last metadata expiration check: 0:13:52 ago on Mon Oct  7 15:30:12 2024.
Package glibc-2.34-52.amzn2023.0.11.x86_64 is already installed.
Package glibc-common-2.34-52.amzn2023.0.11.x86_64 is already installed.
Dependencies resolved.
================================================================================
==========
 Package                                    Architecture
     Size
================================================================================
==========
Installing:
 gcc                                        x86_64
     32 M
Installing dependencies:
 annobin-docs                               noarch
```

5. Create a New Nagios User

Commands -

- sudo adduser -m nagios
- sudo passwd nagios

admin123

```
[ec2-user@ip-172-31-42-50 ~]$ sudo useradd nagios
useradd: user 'nagios' already exists
[ec2-user@ip-172-31-42-50 ~]$ sudo useradd nagios
useradd: user 'nagios' already exists
[ec2-user@ip-172-31-42-50 ~]$ sudo passwd nagios
Changing password for user nagios.
New password:
BAD PASSWORD: The password is shorter than 8 characters
Retype new password:
Sorry, passwords do not match.
New password:
[ec2-user@ip-172-31-42-50 ~]$ sudo passwd nagios
Changing password for user nagios.
New password:
BAD PASSWORD: The password fails the dictionary check - it is based on a dictionary word
Retype new password:
passwd: all authentication tokens updated successfully.
[ec2-user@ip-172-31-42-50 ~]$
```

6. Create a New User Group

Commands -

- sudo groupadd nagcmd

```
[ec2-user@ip-172-31-42-50 ~]$ sudo groupadd nagcmd
[ec2-user@ip-172-31-42-50 ~]$ sudo groupadd nagcmd
groupadd: group 'nagcmd' already exists
[ec2-user@ip-172-31-42-50 ~]$ sudo usermod -aG nagcmd nagios
sudo usermod -aG nagcmd apache
[ec2-user@ip-172-31-42-50 ~]$
```

7. Create a Directory for Nagios Downloads

Commands -

- mkdir ~/downloads
- cd ~/downloads

```
[ec2-user@ip-172-31-42-50 ~]$ mkdir ~/downloads
[ec2-user@ip-172-31-42-50 ~]$ cd ~/downloads
[ec2-user@ip-172-31-42-50 downloads]$
```

8. Download Nagios and Plugins Source Files

Commands -

- Wget https://assets.nagios.com/downloads/nagioscore/releases/nagios-4.4.6.tar.gz
- wget https://nagios-plugins.org/download/nagios-plugins-2.3.3.tar.gz

```
[ec2-user@ip-172-31-42-50 downloads]$ wget https://assets.nagios.com/downloads/nagioscore/releases/
--2024-10-07 16:07:16--  https://assets.nagios.com/downloads/nagioscore/releases/nagios-4.4.6.tar.g
Resolving assets.nagios.com (assets.nagios.com)... 45.79.49.120, 2600:3c00::f03c:92ff:fef7:45ce
Connecting to assets.nagios.com (assets.nagios.com)|45.79.49.120|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 11333414 (11M) [application/x-gzip]
Saving to: 'nagios-4.4.6.tar.gz'

nagios-4.4.6.tar.gz                           0%[
nagios-4.4.6.tar.gz                           2%[=>
nagios-4.4.6.tar.gz                          19%[=================>
nagios-4.4.6.tar.gz                          49%[=============================================>
nagios-4.4.6.tar.gz                          76%[=============================================
nagios-4.4.6.tar.gz                         100%[=============================================
 1.0s

2024-10-07 16:07:18 (11.1 MB/s) - 'nagios-4.4.6.tar.gz' saved [11333414/11333414]
```

9. Extract the Nagios Source File

Commands -

- tar zxvf nagios-4.4.6.tar.gz cd nagios-4.4.6

```
[ec2-user@ip-172-31-42-50 downloads]$ tar zxvf nagios-4.4.6.tar.gz
nagios-4.4.6/
nagios-4.4.6/.gitignore
nagios-4.4.6/.travis.yml
nagios-4.4.6/CONTRIBUTING.md
nagios-4.4.6/Changelog
nagios-4.4.6/INSTALLING
nagios-4.4.6/LEGAL
nagios-4.4.6/LICENSE
nagios-4.4.6/Makefile.in
nagios-4.4.6/README.md
nagios-4.4.6/THANKS
nagios-4.4.6/UPGRADING
nagios-4.4.6/aclocal.m4
nagios-4.4.6/autoconf-macros/
nagios-4.4.6/autoconf-macros/.gitignore
nagios-4.4.6/autoconf-macros/CHANGELOG.md
nagios-4.4.6/autoconf-macros/LICENSE
```

10. Run the Configuration Script Commands

- ./configure --with-command-group=nagcmd

```
[ec2-user@ip-172-31-42-50 downloads]$ cd nagios-4.4.6
[ec2-user@ip-172-31-42-50 nagios-4.4.6]$ ./configure --with-command-group=nagcmd
checking for a BSD-compatible install... /usr/bin/install -c
checking build system type... x86_64-pc-linux-gnu
checking host system type... x86_64-pc-linux-gnu
checking for gcc... gcc
checking whether the C compiler works... yes
checking for C compiler default output file name... a.out
checking for suffix of executables...
checking whether we are cross compiling... no
checking for suffix of object files... o
checking whether we are using the GNU C compiler... yes
checking whether gcc accepts -g... yes
checking for gcc option to accept ISO C89... none needed
checking whether make sets $(MAKE)... yes
checking whether ln -s works... yes
checking for strip... /usr/bin/strip
checking how to run the C preprocessor... gcc -E
checking for grep that handles long lines and -e... /usr/bin/grep
checking for egrep... /usr/bin/grep -E
checking for ANSI C header files... yes
```

11. Compile the Source Code

Commands -

make all

```
*** Support Notes *****************************************

If you have questions about configuring or running Nagios,
please make sure that you:

     - Look at the sample config files
     - Read the documentation on the Nagios Library at:
          https://library.nagios.com

before you post a question to one of the mailing lists.
Also make sure to include pertinent information that could
help others help you.  This might include:

     - What version of Nagios you are using
     - What version of the plugins you are using
     - Relevant snippets from your config files
     - Relevant error messages from the Nagios log file

For more information on obtaining support for Nagios, visit:

       https://support.nagios.com

************************************************************
```

12. Install Binaries, Init Script, and Sample Config Files

Commands -

- sudo make install
- sudo make install-init
- sudo make install-config
- sudo make install-commandmode

```
[ec2-user@ip-172-31-42-50 nagios-4.4.6]$ sudo make install
cd ./base && make install
make[1]: Entering directory '/home/ec2-user/downloads/nagios-4.4.6/base'
/usr/bin/install -c -m 775 -o nagios -g nagios -d /usr/local/nagios/bin
/usr/bin/install -c -s -m 774 -o nagios -g nagios nagios /usr/local/nagios/bin
/usr/bin/install -c -s -m 774 -o nagios -g nagios nagiostats /usr/local/nagios/bin
make[1]: Leaving directory '/home/ec2-user/downloads/nagios-4.4.6/base'
cd ./cgi && make install
make[1]: Entering directory '/home/ec2-user/downloads/nagios-4.4.6/cgi'
make install-basic
make[2]: Entering directory '/home/ec2-user/downloads/nagios-4.4.6/cgi'
/usr/bin/install -c -m 775 -o nagios -g nagios -d /usr/local/nagios/sbin
```

```
[ec2-user@ip-172-31-42-50 nagios-4.4.6]$ sudo make install-init
/usr/bin/install -c -m 755 -d -o root -g root /lib/systemd/system
/usr/bin/install -c -m 755 -o root -g root startup/default-service /lib/systemd/system/
[ec2-user@ip-172-31-42-50 nagios-4.4.6]$ sudo make install-config
/usr/bin/install -c -m 775 -o nagios -g nagios -d /usr/local/nagios/etc
/usr/bin/install -c -m 775 -o nagios -g nagios -d /usr/local/nagios/etc/objects
/usr/bin/install -c -b -m 664 -o nagios -g nagios sample-config/nagios.cfg /usr/local/n
/usr/bin/install -c -b -m 664 -o nagios -g nagios sample-config/cgi.cfg /usr/local/nagi
/usr/bin/install -c -b -m 660 -o nagios -g nagios sample-config/resource.cfg /usr/local
/usr/bin/install -c -b -m 664 -o nagios -g nagios sample-config/template-object/templat
/usr/bin/install -c -b -m 664 -o nagios -g nagios sample-config/template-object/command
/usr/bin/install -c -b -m 664 -o nagios -g nagios sample-config/template-object/contact
/usr/bin/install -c -b -m 664 -o nagios -g nagios sample-config/template-object/timeper
/usr/bin/install -c -b -m 664 -o nagios -g nagios sample-config/template-object/localho
/usr/bin/install -c -b -m 664 -o nagios -g nagios sample-config/template-object/windows
/usr/bin/install -c -b -m 664 -o nagios -g nagios sample-config/template-object/printer
/usr/bin/install -c -b -m 664 -o nagios -g nagios sample-config/template-object/switch.
```

```
[ec2-user@ip-172-31-42-50 nagios-4.4.6]$ sudo make install-commandmode
/usr/bin/install -c -m 775 -o nagios -g nagcmd -d /usr/local/nagios/var/rw
chmod g+s /usr/local/nagios/var/rw

*** External command directory configured ***

[ec2-user@ip-172-31-42-50 nagios-4.4.6]$
```

14. Edit the Config File to Change the Email Address

Commands -

- sudo nano /usr/local/nagios/etc/objects/contacts.cfg
- Change the email address in the contacts.cfg file to your preferred email.

```
###############################################################################
#
# CONTACTS
#
###############################################################################
odified
# 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 above)
    alias                   Nagios Admin            ; Full name of user
    email                   shravanirasam0212@gmail.com  ; <<***** CHANGE THIS TO YOUR EMAIL ADDRESS ******
}



###############################################################################
#
# CONTACT GROUPS
#
```

15. Configure the Web Interface

Commands -

sudo make install-webconf

```
[ec2-user@ip-172-31-42-50 nagios-4.4.6]$ sudo make install-webconf
/usr/bin/install -c -m 644 sample-config/httpd.conf /etc/httpd/conf.d/nagios.conf
if [ 0 -eq 1 ]; then \
        ln -s /etc/httpd/conf.d/nagios.conf /etc/apache2/sites-enabled/nagios.conf; \
fi

*** Nagios/Apache conf file installed ***

[ec2-user@ip-172-31-42-50 nagios-4.4.6]$
```

16.Create a Nagios Admin Account

Commands -

- sudo htpasswd -c /usr/local/nagios/etc/htpasswd.users nagiosadmin
- You will be prompted to enter and confirm the password for the nagiosadmin user

```
[ec2-user@ip-172-31-42-50 nagios-4.4.6]$ sudo htpasswd -c /usr/local/nagios/etc/htpasswd.users nagiosadmin
New password:
Re-type new password:
Adding password for user nagiosadmin
[ec2-user@ip-172-31-42-50 nagios-4.4.6]$
```

admin123

17. Restart Apache

Commands -

- sudo systemctl restart httpd

18. Extract the Plugins Source File

Commands -

- cd ~/downloads
- tar zxvf nagios-plugins-2.3.3.tar.gz cd nagios-plugins-2.3.3

```
[ec2-user@ip-172-31-42-50 nagios-4.4.6]$ sudo systemctl restart httpd
[ec2-user@ip-172-31-42-50 nagios-4.4.6]$ cd ~/downloads
[ec2-user@ip-172-31-42-50 downloads]$ tar zxvf nagios-plugins-2.3.3.tar.gz
nagios-plugins-2.3.3/
nagios-plugins-2.3.3/perlmods/
nagios-plugins-2.3.3/perlmods/Config-Tiny-2.14.tar.gz
nagios-plugins-2.3.3/perlmods/parent-0.226.tar.gz
nagios-plugins-2.3.3/perlmods/Test-Simple-0.98.tar.gz
nagios-plugins-2.3.3/perlmods/Makefile.in
nagios-plugins-2.3.3/perlmods/version-0.9903.tar.gz
nagios-plugins-2.3.3/perlmods/Makefile.am
nagios-plugins-2.3.3/perlmods/Module-Runtime-0.013.tar.gz
nagios-plugins-2.3.3/perlmods/Module-Metadata-1.000014.tar.gz
nagios-plugins-2.3.3/perlmods/Params-Validate-1.08.tar.gz
nagios-plugins-2.3.3/perlmods/Class-Accessor-0.34.tar.gz
```

19. Compile and Install Plugins Commands -

- ./configure --with-nagios-user=nagios --with-nagios-group=nagios make
- sudo make install

```
[ec2-user@ip-172-31-42-50 downloads]$ cd nagios-plugins-2.3.3
[ec2-user@ip-172-31-42-50 nagios-plugins-2.3.3]$ ./configure --with-nagios-user=nagios --with-nagios-group=nagios
make
sudo make install
checking for a BSD-compatible install... /usr/bin/install -c
checking whether build environment is sane... yes
checking for a thread-safe mkdir -p... /usr/bin/mkdir -p
checking for gawk... gawk
checking whether make sets $(MAKE)... yes
checking whether to disable maintainer-specific portions of Makefiles... yes
checking build system type... x86_64-unknown-linux-gnu
checking host system type... x86_64-unknown-linux-gnu
checking for gcc... gcc
checking for C compiler default output file name... a.out
checking whether the C compiler works... yes
checking whether we are cross compiling... no
checking for suffix of executables...
checking for suffix of object files... o
checking whether we are using the GNU C compiler... yes
```

20. Start Nagios

Commands

- sudo chkconfig --add nagios
- sudo chkconfig nagios on
- sudo /usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg
- sudo systemctl start nagios

```
[ec2-user@ip-172-31-42-50 nagios-plugins-2.3.3]$ sudo chkconfig --add nagios
sudo chkconfig nagios on
sudo /usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg
sudo systemctl start nagios
error reading information on service nagios: No such file or directory
Note: Forwarding request to 'systemctl enable nagios.service'.
Created symlink /etc/systemd/system/multi-user.target.wants/nagios.service → /usr/l

Nagios Core 4.4.6
Copyright (c) 2009-present Nagios Core Development Team and Community Contributors
Copyright (c) 1999-2009 Ethan Galstad
Last Modified: 2020-04-28
License: GPL

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

21. Check the Status of Nagios

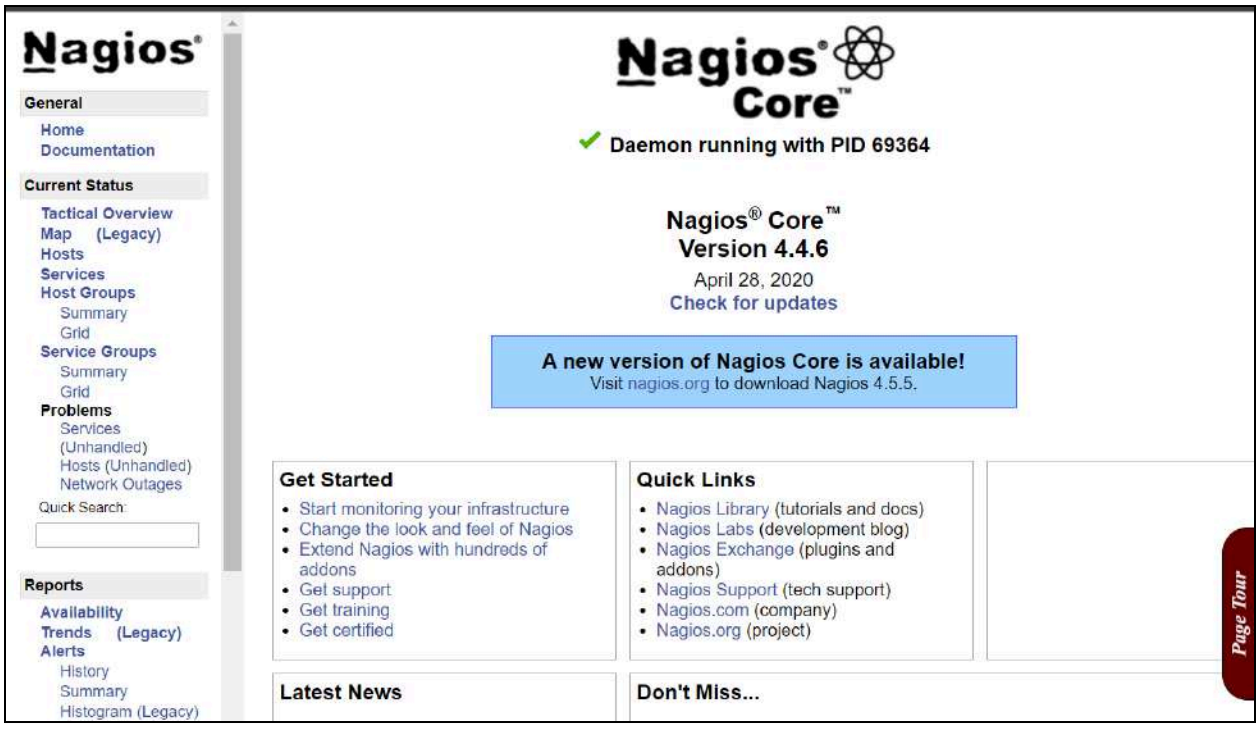
Commands -

- sudo systemctl status nagios

```
Things look okay - No serious problems were detected during the pre-flight check
[ec2-user@ip-172-31-42-50 nagios-plugins-2.3.3]$ sudo systemctl status nagios
● nagios.service - Nagios Core 4.4.6
     Loaded: loaded (/usr/lib/systemd/system/nagios.service; enabled; preset: disabled)
     Active: active (running) since Mon 2024-10-07 16:28:45 UTC; 38s ago
       Docs: https://www.nagios.org/documentation
    Process: 69362 ExecStartPre=/usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg
    Process: 69363 ExecStart=/usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg (c
   Main PID: 69364 (nagios)
      Tasks: 6 (limit: 1112)
     Memory: 2.1M
        CPU: 22ms
     CGroup: /system.slice/nagios.service
             ├─69364 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg
             ├─69365 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─69366 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─69367 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─69368 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             └─69369 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg
```

22. Access Nagios Web Interface

- Copy the Public IP address of your EC2 instance.
- Open your browser and navigate to http:///nagios.
- Enter the username nagiosadmin and the password you set in Step 16



Conclusion:

After installing and configuring Nagios Core, Plugins, and NRPE on a Linux machine, We have a robust continuous monitoring setup, ensuring proactive issue detection and optimal system performance.

ADVANCED DEVOPS EXP 10

Aim: To perform Port, Service monitoring, Windows/Linux server monitoring using Nagios.

Theory:
Nagios is a comprehensive monitoring and alerting platform designed to keep track of IT infrastructure, networks, and applications. It provides real-time monitoring, alerting, and reporting capabilities to ensure the health and performance of critical systems.

Key Components of Nagios
1. **Nagios Core**: The open-source foundation of the Nagios monitoring system. It provides the basic framework for monitoring and alerting.
2. **Nagios XI**: A commercial version of Nagios that offers advanced features, a more user-friendly interface, and additional support options.
3. **Nagios Log Server**: A tool for centralized log management, allowing you to view, analyze, and archive logs from various sources.
4. **Nagios Network Analyzer**: Provides detailed insights into network traffic and bandwidth usage.
5. **Nagios Fusion**: Centralizes monitoring data from multiple Nagios instances, providing a unified view of the entire network.

Monitoring Capabilities
1. **Port Monitoring**: Nagios can monitor specific network ports to ensure they are open and responsive. This is crucial for services that rely on these ports.
2. **Service Monitoring**: Nagios checks the status of various services (e.g., web servers, databases) to ensure they are running smoothly.
3. **Server Monitoring:** Nagios can monitor both Windows and Linux servers using agents like NSClient++ for Windows and NRPE for Linux. This includes metrics like CPU usage, memory usage, disk space, and more.

How Nagios Works
1. **Configuration**: Administrators define what to monitor and how to monitor it using configuration files.
2. **Plugins:** Nagios uses plugins to gather information about the status of various services and hosts. These plugins can be custom scripts or pre-built ones.
3. **Scheduling:** Nagios schedules regular checks of the defined services and hosts using the configured plugins.

4. **Alerting:** If a check indicates a problem, Nagios triggers an alert. Alerts can be configured to escalate if not acknowledged within a certain timeframe. 5. Log Management: Centralizing and analyzing logs from various sources to detect issues and ensure compliance.

Implementation :

Prerequisites

- AWS Free Tier
- Nagios Server running on an Amazon Linux Machine

1. Confirm Nagios is Running on the Server

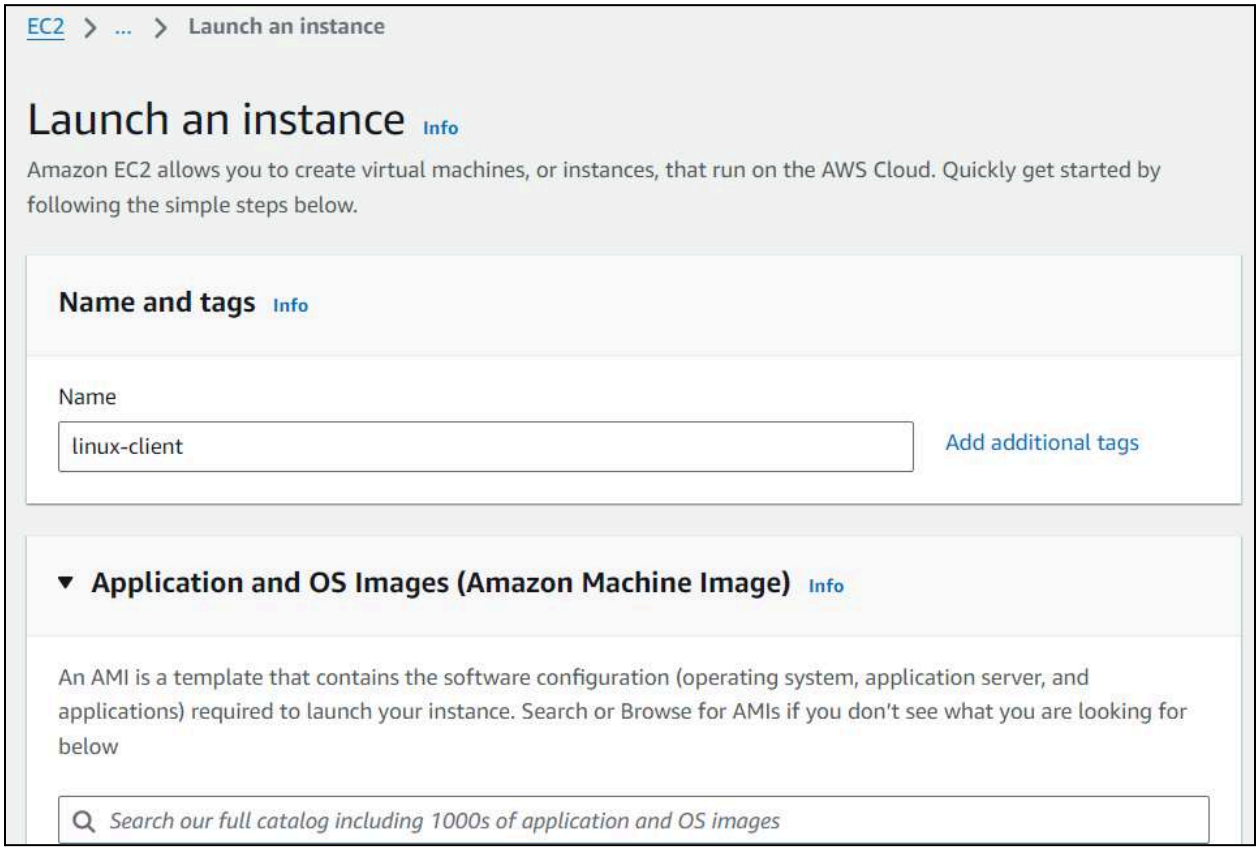
Commands -

- sudo systemctl status nagios
- Proceed if you see that Nagios is active and running.

```
Things look okay - No serious problems were detected during the pre-flight check
[ec2-user@ip-172-31-42-50 nagios-plugins-2.3.3]$ sudo systemctl status nagios
● nagios.service - Nagios Core 4.4.6
     Loaded: loaded (/usr/lib/systemd/system/nagios.service; enabled; preset: disabled)
     Active: active (running) since Mon 2024-10-07 16:28:45 UTC; 38s ago
       Docs: https://www.nagios.org/documentation
    Process: 69362 ExecStartPre=/usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg
    Process: 69363 ExecStart=/usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg (c
   Main PID: 69364 (nagios)
      Tasks: 6 (limit: 1112)
     Memory: 2.1M
        CPU: 22ms
     CGroup: /system.slice/nagios.service
             ├─69364 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg
             ├─69365 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─69366 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─69367 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─69368 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             └─69369 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg
```

2. Create an Ubuntu 20.04 Server EC2 Instance

- Name it linux-client.
- Use the same security group as the Nagios Host



3. Verify Nagios Process on the Server

Commands

- - ps -ef | grep nagios

```
[ec2-user@ip-172-31-42-50 nagios-plugins-2.3.3]$ ps -ef | grep nagios
nagios     69364       1  0 16:28 ?        00:00:00 /usr/local/nagios/bin/nagios -d
nagios     69365   69364  0 16:28 ?        00:00:00 /usr/local/nagios/bin/nagios --
nagios     69366   69364  0 16:28 ?        00:00:00 /usr/local/nagios/bin/nagios --
nagios     69367   69364  0 16:28 ?        00:00:00 /usr/local/nagios/bin/nagios --
nagios     69368   69364  0 16:28 ?        00:00:00 /usr/local/nagios/bin/nagios --
nagios     69369   69364  0 16:28 ?        00:00:00 /usr/local/nagios/bin/nagios -d
ec2-user   70969    2909  0 16:55 pts/0    00:00:00 grep --color=auto nagios
[ec2-user@ip-172-31-42-50 nagios-plugins-2.3.3]$
```

4. Become Root User and Create Directories

Commands -

- sudo su
- mkdir -p /usr/local/nagios/etc/objects/monitorhosts/linuxhosts

```
[ec2-user@ip-172-31-42-50 nagios-plugins-2.3.3]$ sudo su
mkdir -p /usr/local/nagios/etc/objects/monitorhosts/linuxhosts
[root@ip-172-31-42-50 nagios-plugins-2.3.3]#
```

5. Copy Sample Configuration File

Commands -

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

```
[root@ip-172-31-42-50 ec2-user]# cp /usr/local/nagios/etc/objects/localhost.cfg /usr/local/nagios/etc/objects/monitorhosts/
[root@ip-172-31-42-50 ec2-user]# nano /usr/local/nagios/etc/objects/monitorhosts/linuxhosts/linuxserver.cfg
[root@ip-172-31-42-50 ec2-user]#
```

6. Edit the Configuration File

Commands -

sudo nano /usr/local/nagios/etc/objects/monitorhosts/linuxhosts/linuxserver.cfg

- Change hostname to linuxserver everywhere in the file.
- Change address to the public IP address of your linux-client.

```
##############################################################################
#
# HOST DEFINITION
#
##############################################################################

# Define a host for the local machine

define host {

    use                     linux-server            ; Name of host template to use
                                                    ; This host definition will inherit all variables that are defined
                                                    ; in (or inherited by) the linux-server host template definition.
    host_name               linuxserver
    alias                   linuxserver
    address                 127.0.0.1
}

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

7. Update Nagios Configuration

Commands -

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

- Add the following line: cfg_dir=/usr/local/nagios/etc/objects/monitorhosts/
- Change hostgroup_name under hostgroup to linux-servers1

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

# Define an optional hostgroup for Linux machines

define hostgroup {

    hostgroup_name          linux-servers1          ; The name of the hostgroup
    alias                   Linux Servers           ; Long name of the group
    members                 localhost               ; Comma separated list of hosts that belong to this group
}



##############################################################################
#
```

8. Verify Configuration Files

Commands -

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

ERROR OCCURRED

```
[root@ip-172-31-42-50 ec2-user]# sudo /usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg

Nagios Core 4.4.6
Copyright (c) 2009-present Nagios Core Development Team and Community Contributors
Copyright (c) 1999-2009 Ethan Galstad
Last Modified: 2020-04-28
License: GPL

Website: https://www.nagios.org
Reading configuration data...
   Read main config file okay...
Error: Could not find any host matching 'linuxerver' (config file '/usr/local/nagios/etc/objects/monito
Error: Failed to expand host list 'linuxerver' for service 'Total Processes' (/usr/local/nagios/etc/obj
   Error processing object config files!


***> One or more problems was encountered while processing the config files...

     Check your configuration file(s) to ensure that they contain valid
     directives and data definitions.  If you are upgrading from a previous
     version of Nagios, you should be aware that some variables/definitions
     may have been removed or modified in this version.  Make sure to read
     the HTML documentation regarding the config files, as well as the
     'Whats New' section to find out what has changed.
```

Error resolved

```
[root@ip-172-31-42-50 ec2-user]# sudo /usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg

Nagios Core 4.4.6
Copyright (c) 2009-present Nagios Core Development Team and Community Contributors
Copyright (c) 1999-2009 Ethan Galstad
Last Modified: 2020-04-28
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 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-172-31-42-50 ec2-user]#
```

9. Restart Nagios Service

Commands -

- sudo systemctl restart nagios

10. SSH into the Client Machine

● Use SSH or EC2 Instance Connect to access the linux-client.

11. Update Package Index and Install Required Packages

Commands -

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

```
ubuntu@ip-172-31-33-27:~$ sudo apt update -y
sudo apt install gcc -y
sudo apt install -y nagios-nrpe-server nagios-plugins
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates InR
Get:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports I
Get:4 http://security.ubuntu.com/ubuntu noble-security InRelease [126
Get:5 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe am
Get:6 http://security.ubuntu.com/ubuntu noble-security/main amd64 Pack
Get:7 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe Tr
Get:8 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe am
Get:9 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe am
Get:10 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/multiverse
```

12. Edit NRPE Configuration File

Commands -

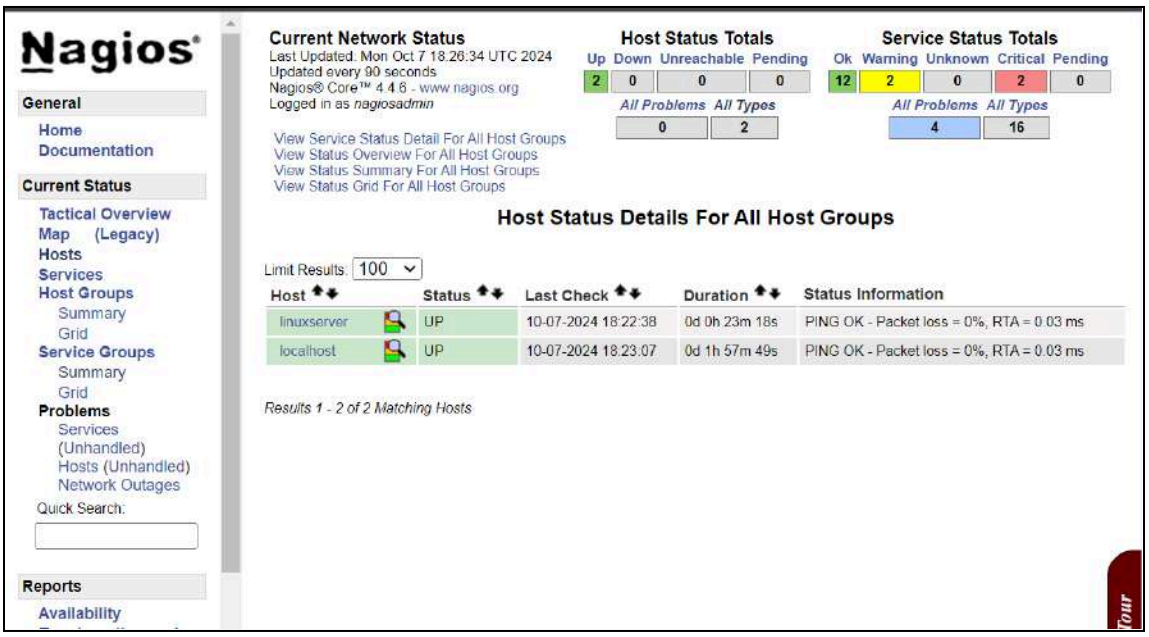
sudo nano /etc/nagios/nrpe.cfg

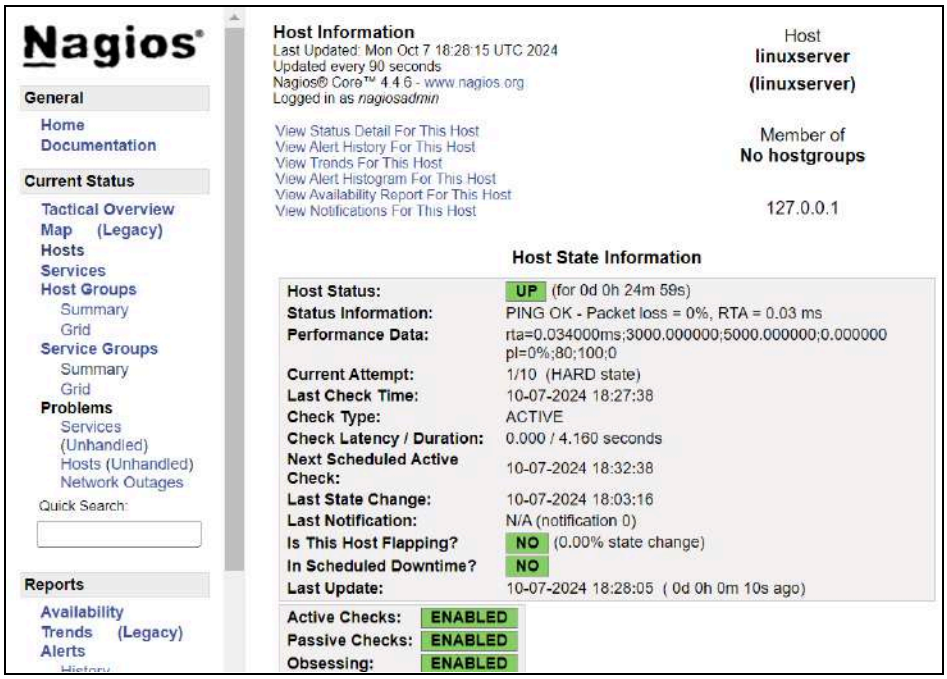
● Add your Nagios host IP address under allowed_hosts: allowed_hosts=

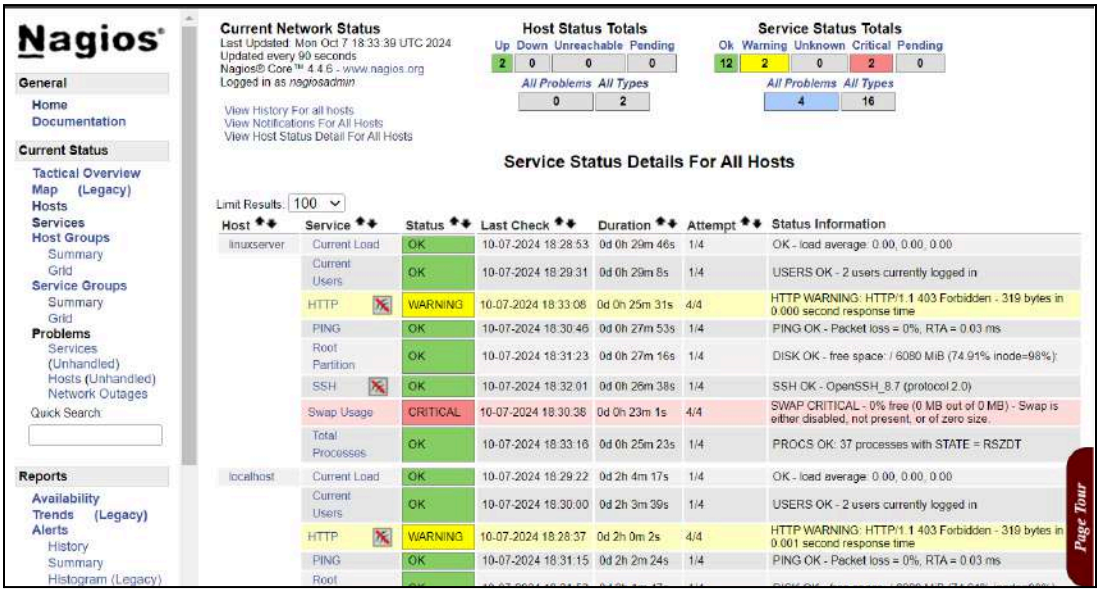
```
# supported.
#
# Note: The daemon only does rudimentary checking of the client's IP
# address.  I would highly recommend adding entries in your /etc/hosts.allow
# file to allow only the specified host to connect to the port
# you are running this daemon on.
#
# NOTE: This option is ignored if NRPE is running under either inetd or xinetd

allowed_hosts=127.0.0.1,3.81.151.142



# COMMAND ARGUMENT PROCESSING
# This option determines whether or not the NRPE daemon will allow clients
# to specify arguments to commands that are executed.  This option only works
# if the daemon was configured with the --enable-command-args configure script
# option.
#
```

14. Check Nagios Dashboard

- Open your browser and navigate to http:///nagios.
- Log in with nagiosadmin and the password you set earlier.
- You should see the new host linuxserver added.
- Click on Hosts to see the host details.
- Click on Services to see all services and ports being monitored







Conclusion:

To perform port, service, and Windows/Linux server monitoring using Nagios, configure the necessary plugins and agents, define the monitoring parameters in the configuration files, and set up alerting mechanisms to ensure timely notifications of any issues. This comprehensive approach ensures robust monitoring and quick response to potential problems, maintaining the health and performance of your IT infrastructure.

ADVANCED DEVOPS EXP 11

Aim: To understand AWS Lambda, its workflow, various functions and create your first Lambda functions using Python / Java / Nodejs.

Theory:

AWS Lambda

- AWS Lambda is a serverless computing service provided by Amazon Web Services (AWS). Users of AWS Lambda create functions, self-contained applications written in one of the supported languages and runtimes, and upload them to AWS Lambda, which executes those functions in an efficient and flexible manner.
- The Lambda functions can perform any kind of computing task, from serving web pages and processing streams of data to calling APIs and integrating with other AWS services. The concept of "serverless" computing refers to not needing to maintain your own servers to run these functions.
- AWS Lambda is a fully managed service that takes care of all the infrastructure for you. And so "serverless" doesn't mean that there are no servers involved: it just means that the servers, the operating systems, the network layer and the rest of the infrastructure have already been taken care of so that you can focus on writing application code.

Features of AWS Lambda

● AWS Lambda easily scales the infrastructure without any additional configuration. It reduces the operational work involved.

● It offers multiple options like AWS S3, CloudWatch, DynamoDB, API Gateway, Kinesis, CodeCommit, and many more to trigger an event.

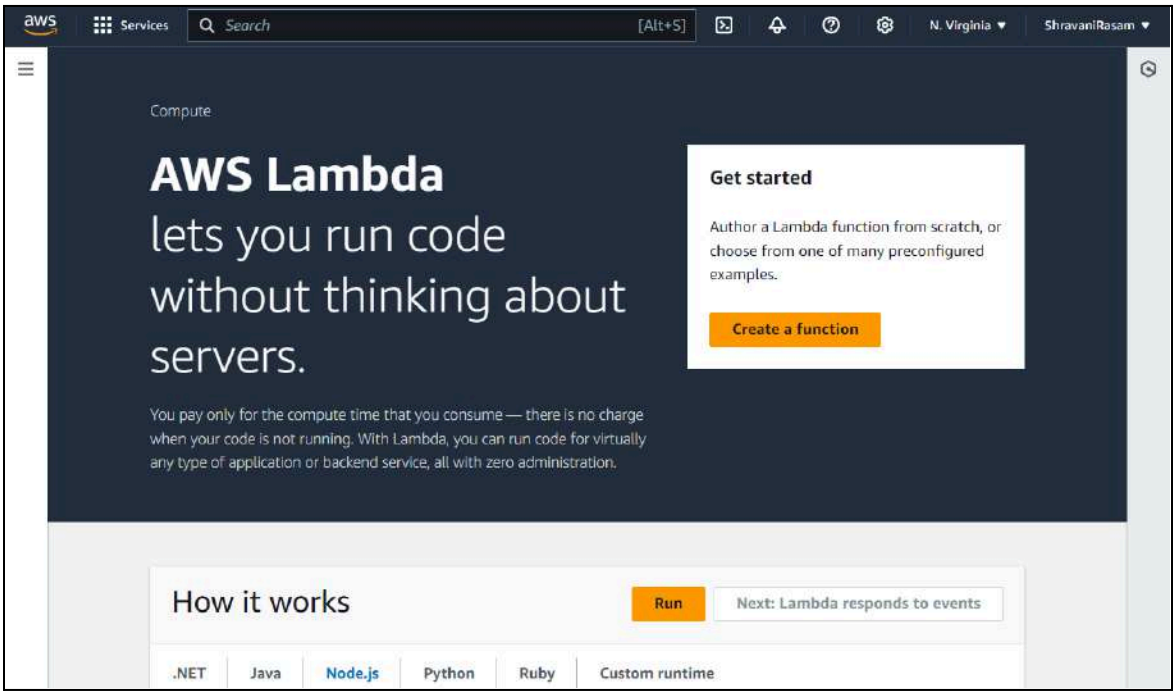
● You don't need to invest upfront. You pay only for the memory used by the lambda function and minimal cost on the number of requests hence cost-efficient.

● AWS Lambda is secure. It uses AWS IAM to define all the roles and security policies.

● It offers fault tolerance for both services running the code and the function. You do not have to worry about the application down

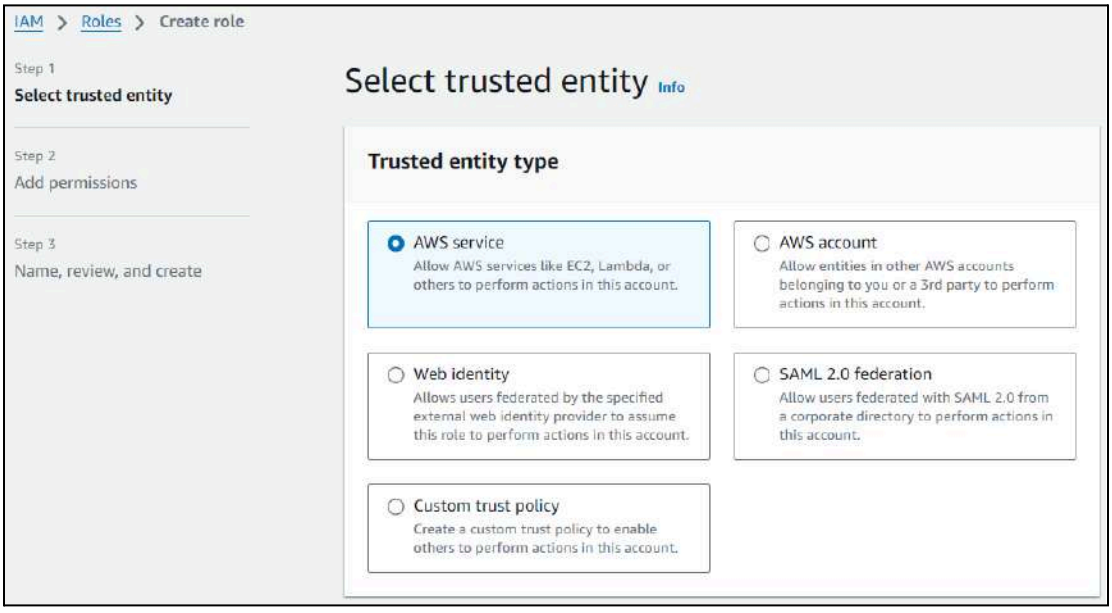
Steps to create an AWS Lambda function

1. Open up the Lambda Console and click on the Create button. Be mindful of where you create your functions since Lambda is region-dependent.



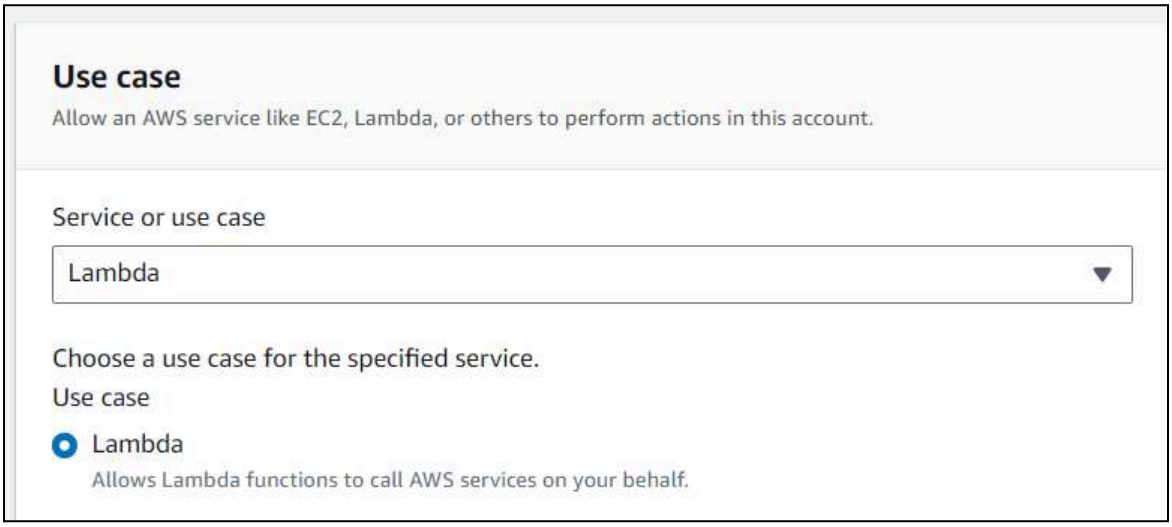
2.Choose the Lambda service:

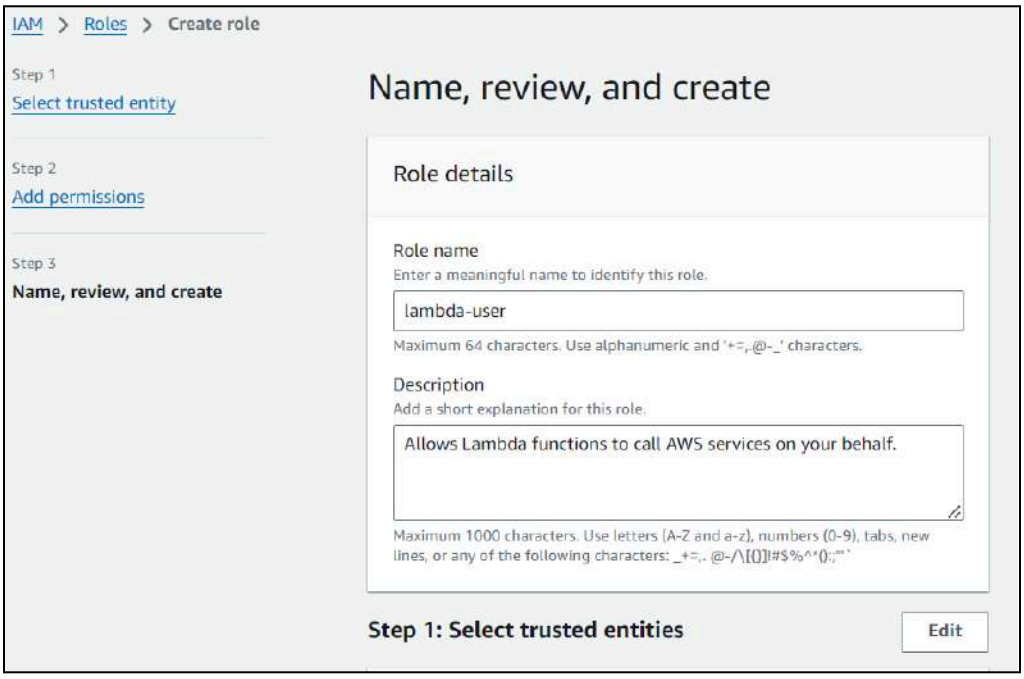
- Under "Trusted entity type," select **AWS service**.
- Choose **Lambda** from the list of services, and click **Next**.

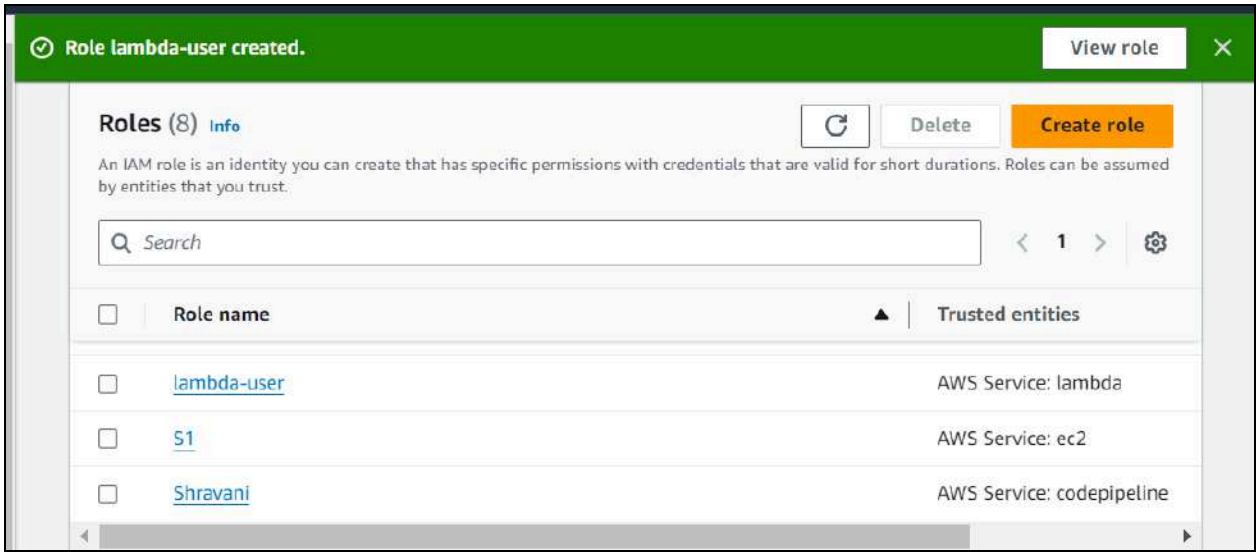


3. Attach CloudWatch Logs permissions:

- In the "Permissions" step, search for the policy called `AWSLambdaBasicExecutionRole`.
- Select this policy, which gives your Lambda function permission to write logs to CloudWatch.
- Click **Next**.

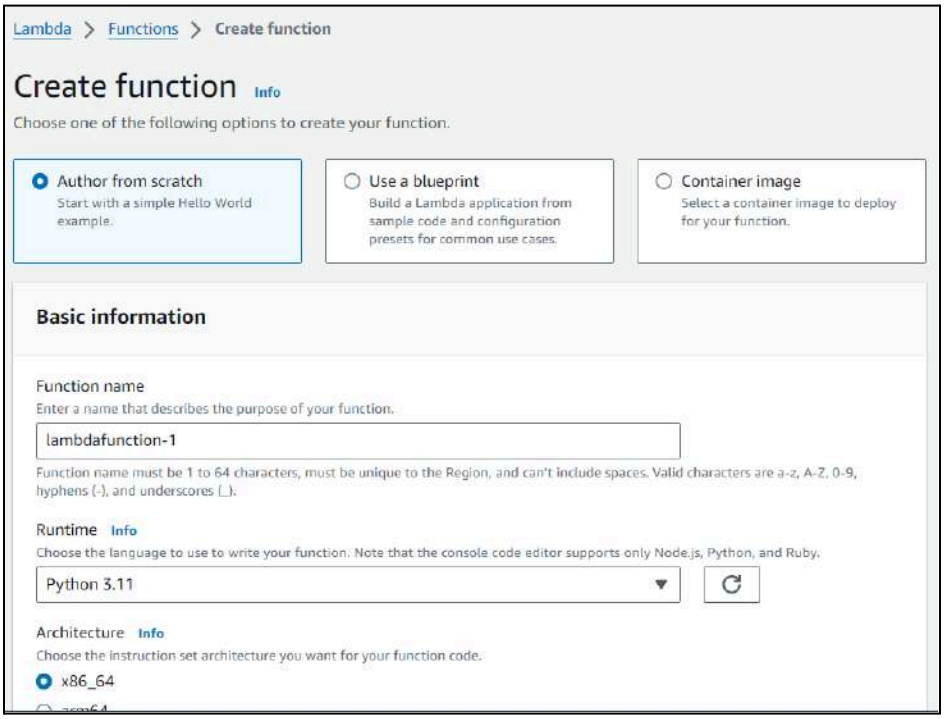


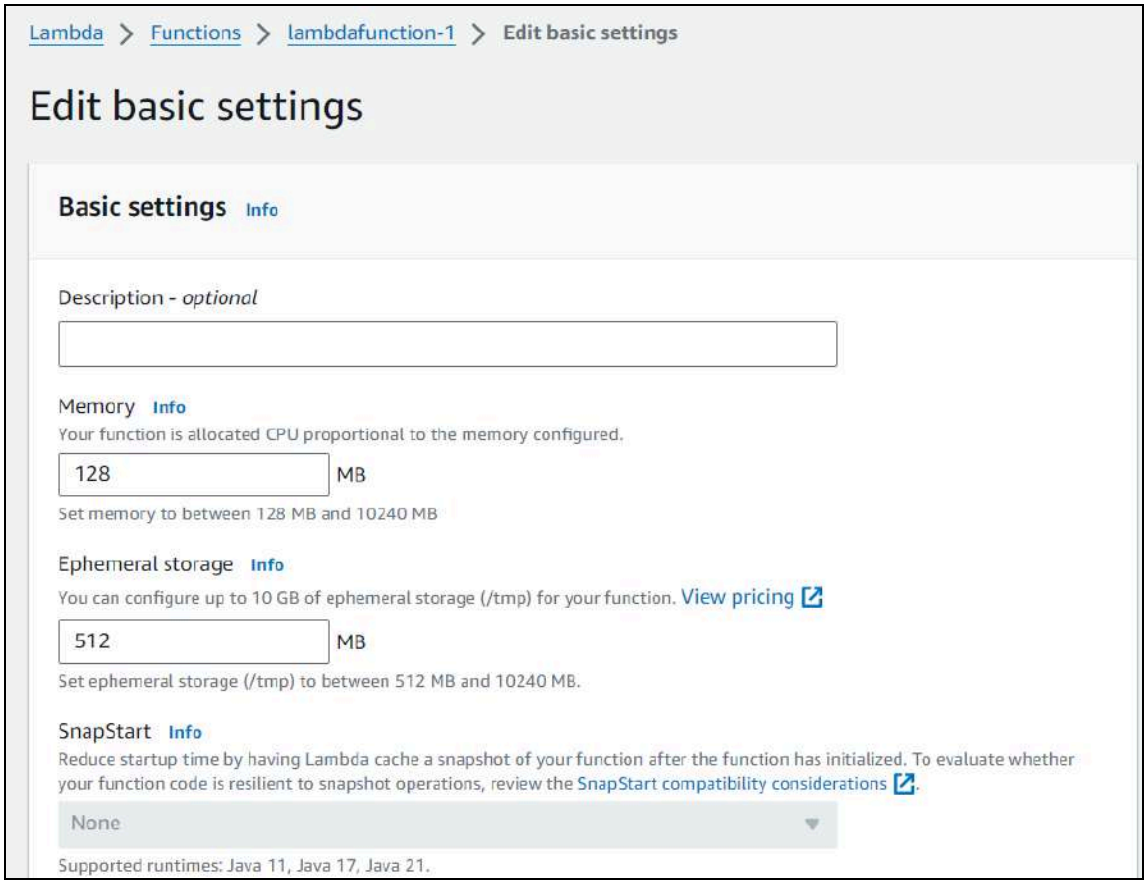


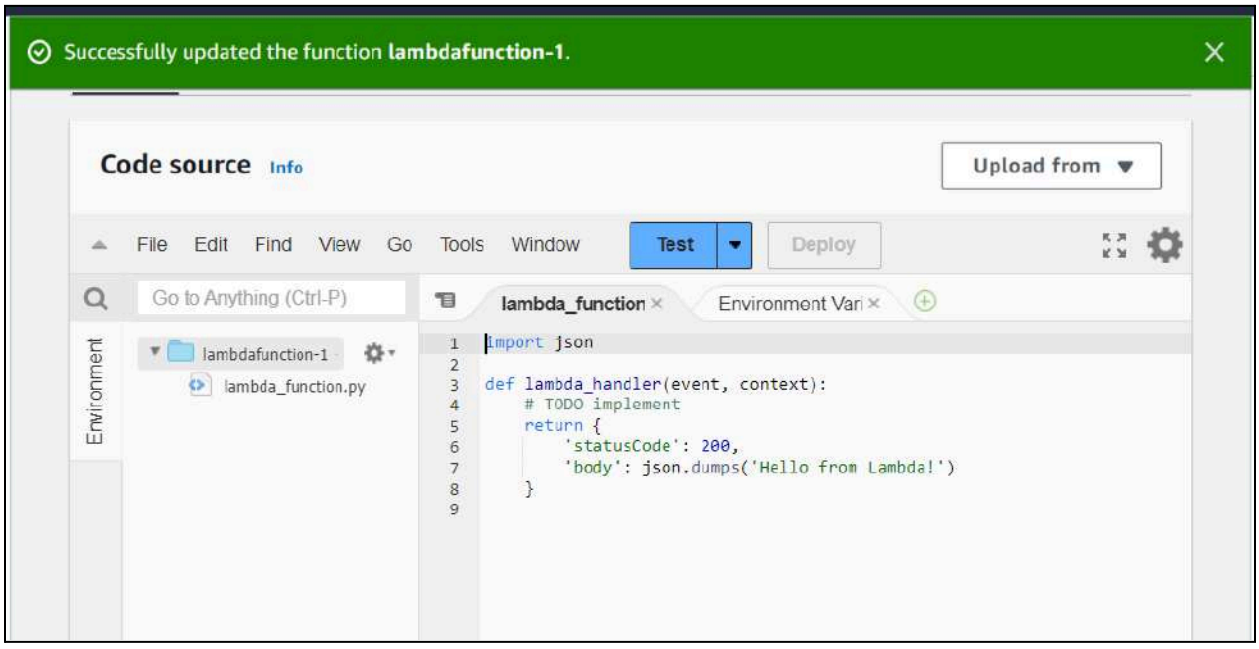


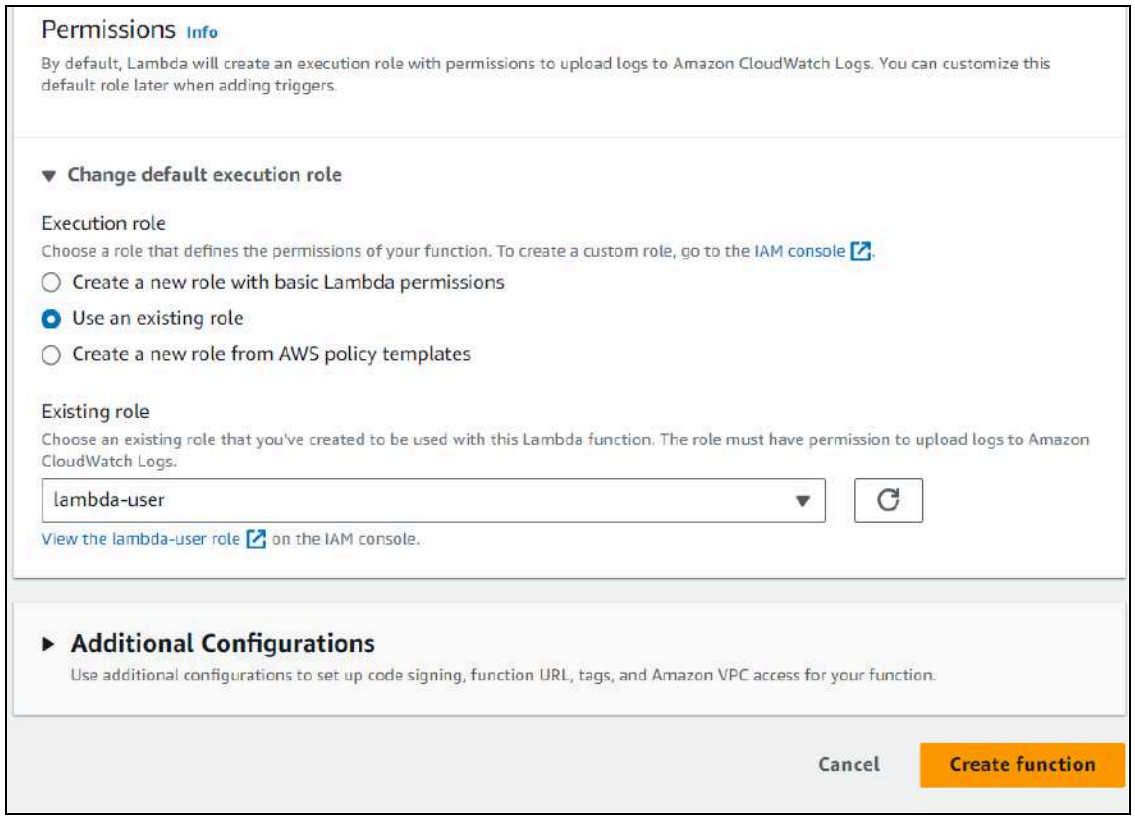
2. Choose to create a function from scratch or use a blueprint, i.e templates defined by AWS for you with all configuration presets required for the most common use cases.

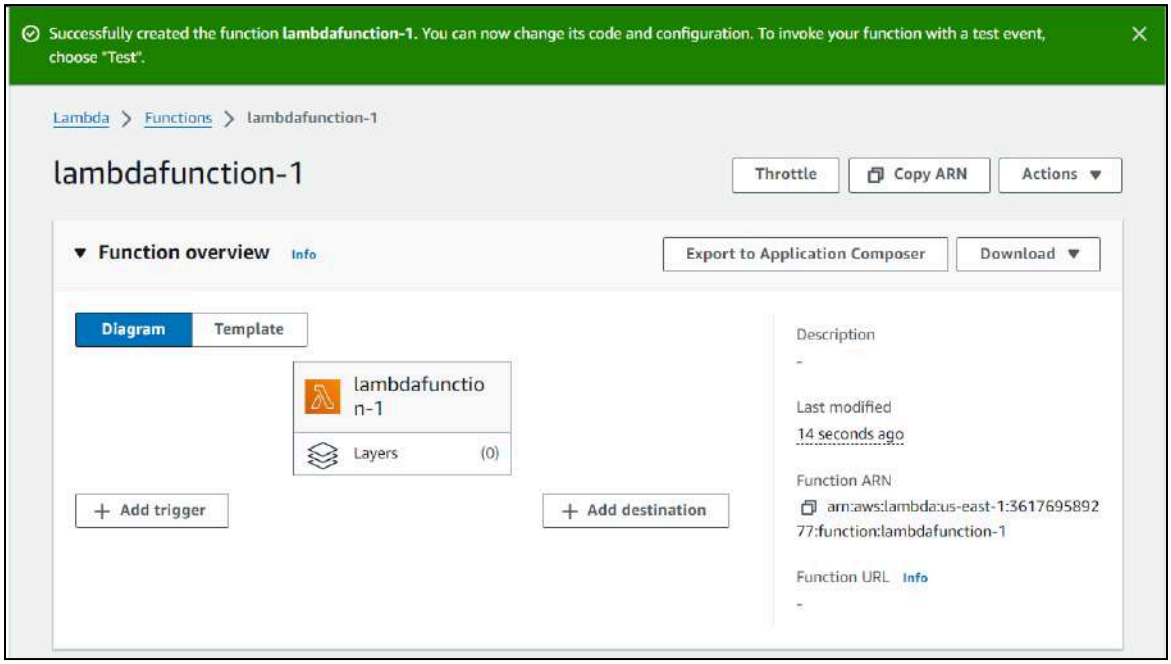
Then, choose a runtime env for your function, under the dropdown, you can see all the options AWS supports, Python, Nodejs, .NET and Java being the most popular ones. After that, choose to create a new role with basic Lambda permissions if you don't have an existing one.



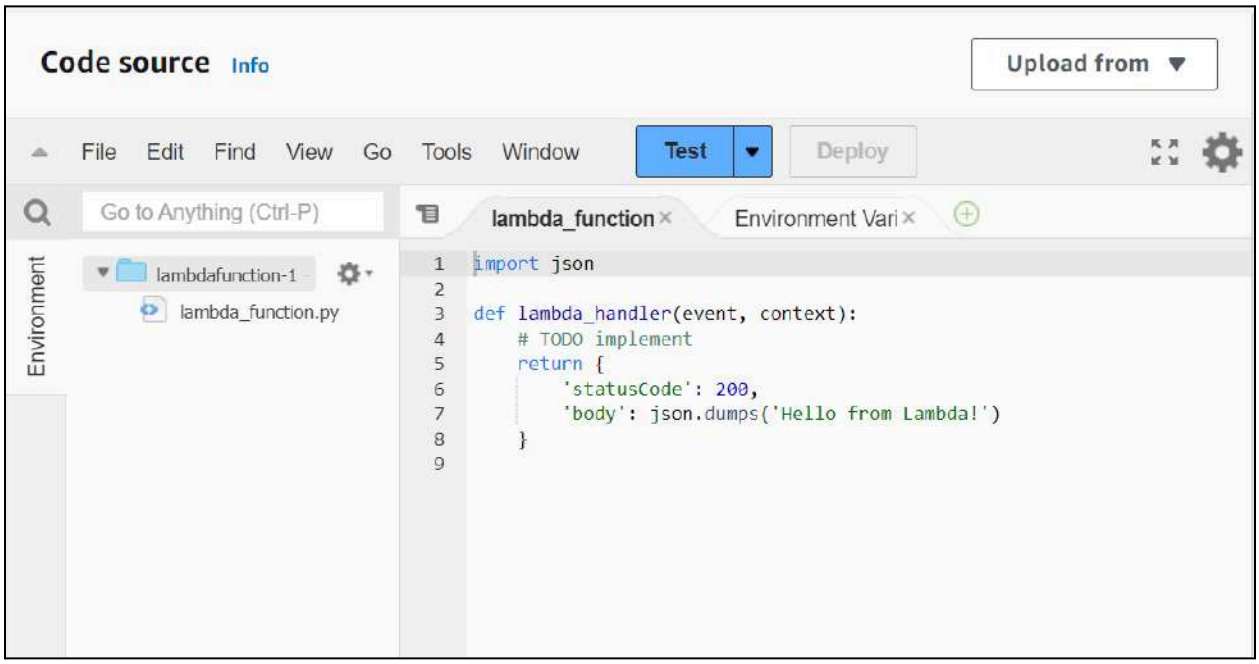

Lambda > Functions > lambdafunction-1 > Edit basic settings
Edit basic settings
Basic settings Info
Description - optional
Memory Info
Your function is allocated CPU proportional to the memory configured.
128
MB
Set memory to between 128 MB and 10240 MB
Ephemeral storage Info
You can configure up to 10 GB of ephemeral storage (/tmp) for your function. View pricing
512
MB
Set ephemeral storage (/tmp) to between 512 MB and 10240 MB.
SnapStart Info
Reduce startup time by having Lambda cache a snapshot of your function after the function has initialized. To evaluate whether your function code is resilient to snapshot operations, review the SnapStart compatibility considerations.
None
Supported runtimes: Java 11, Java 17, Java 21.


Successfully updated the function lambdafunction-1.
Code source Info
Upload from
File Edit Find View Go Tools Window
Test
Deploy
Go to Anything (Ctrl-P)
lambda_function
Environment Var
Environment
lambdafunction-1
lambda_function.py
import json
def lambda_handler(event, context):
# TODO implement
return {
'statusCode': 200,
'body': json.dumps('Hello from Lambda!')
}


Permissions Info
By default, Lambda will create an execution role with permissions to upload logs to Amazon CloudWatch Logs. You can customize this default role later when adding triggers.
Change default execution role
Execution role
Choose a role that defines the permissions of your function. To create a custom role, go to the IAM console.
Create a new role with basic Lambda permissions
Use an existing role
Create a new role from AWS policy templates
Existing role
Choose an existing role that you've created to be used with this Lambda function. The role must have permission to upload logs to Amazon CloudWatch Logs.
lambda-user
View the lambda-user role on the IAM console.
Additional Configurations
Use additional configurations to set up code signing, function URL, tags, and Amazon VPC access for your function.
Cancel
Create function

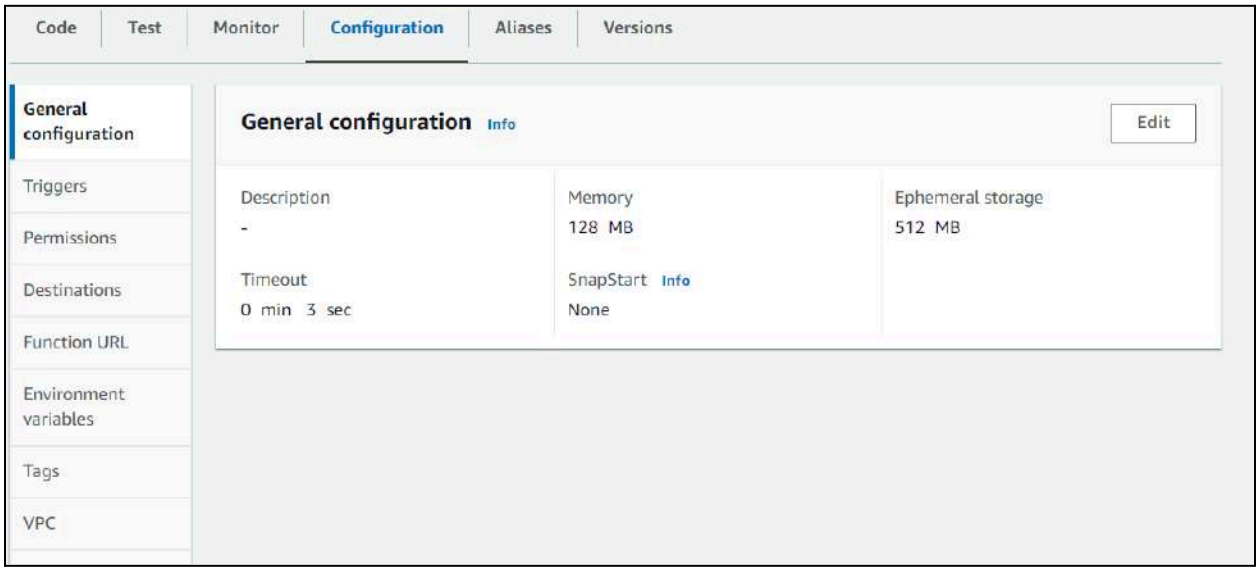

Successfully created the function lambdafunction-1. You can now change its code and configuration. To invoke your function with a test event, choose "Test".
Lambda > Functions > lambdafunction-1
lambdafunction-1
Throttle
Copy ARN
Actions
Function overview Info
Export to Application Composer
Download
Diagram
Template
lambdafunctio n-1
Layers (0)
+ Add trigger
+ Add destination
Description
-
Last modified
14 seconds ago
Function ARN
arn:aws:lambda:us-east-1:361769589277:function:lambdafunction-1
Function URL Info
-

3. This process will take a while to finish and after that, you'll get a message that your function was successfully created

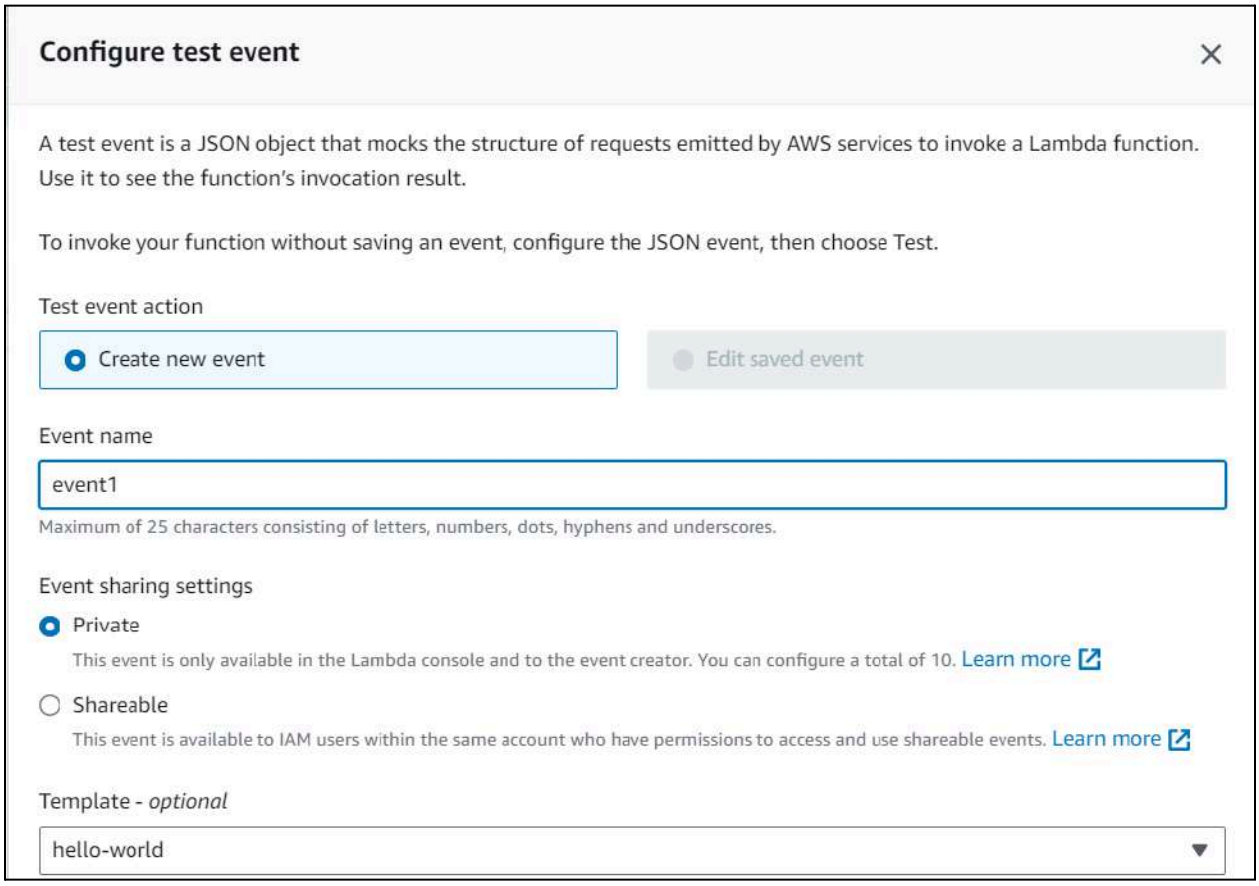


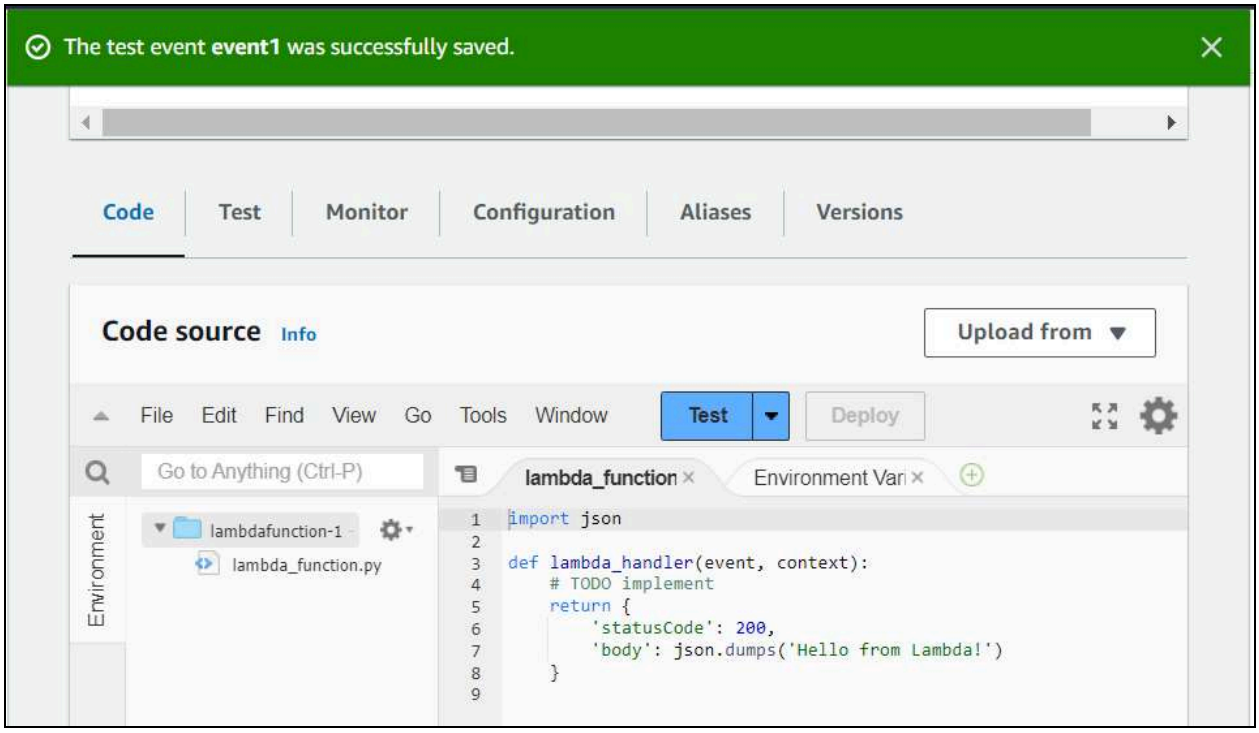
Edit Basic Settings:

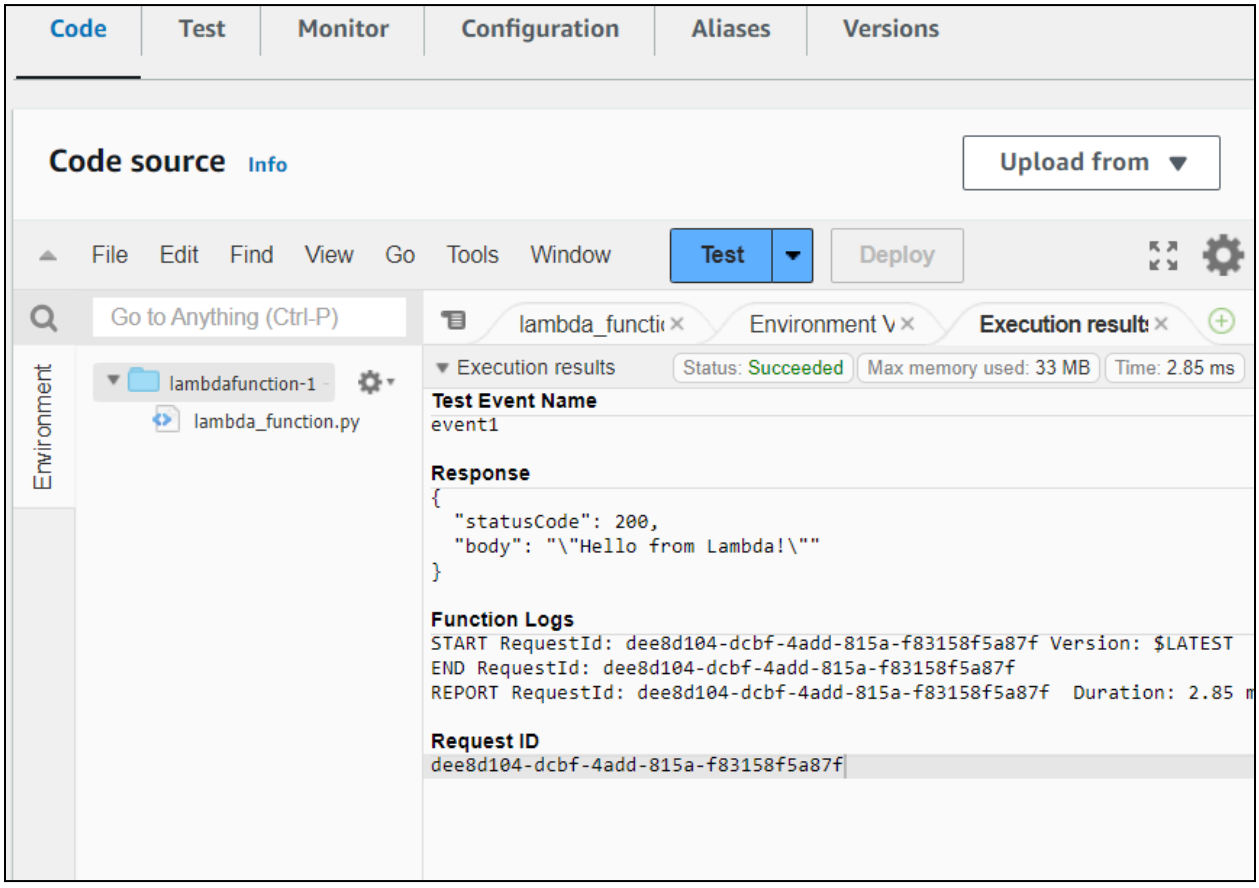
- On the function's **Configuration** tab, locate the **Basic settings** section



Configuring test event which tiggers when the function is tested

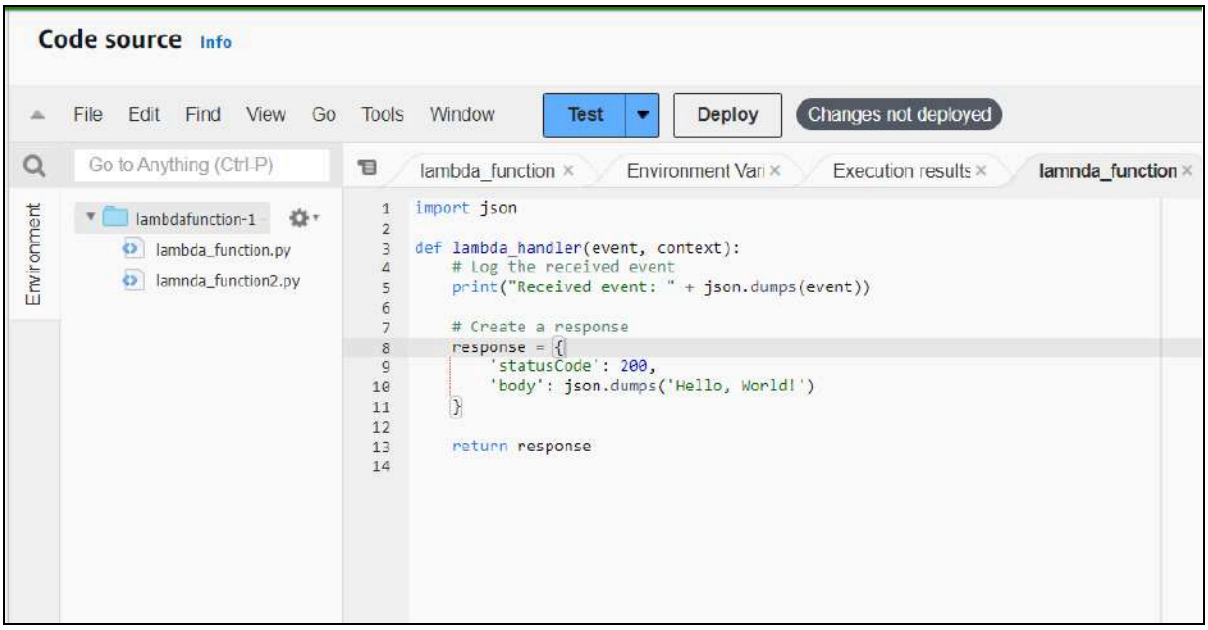


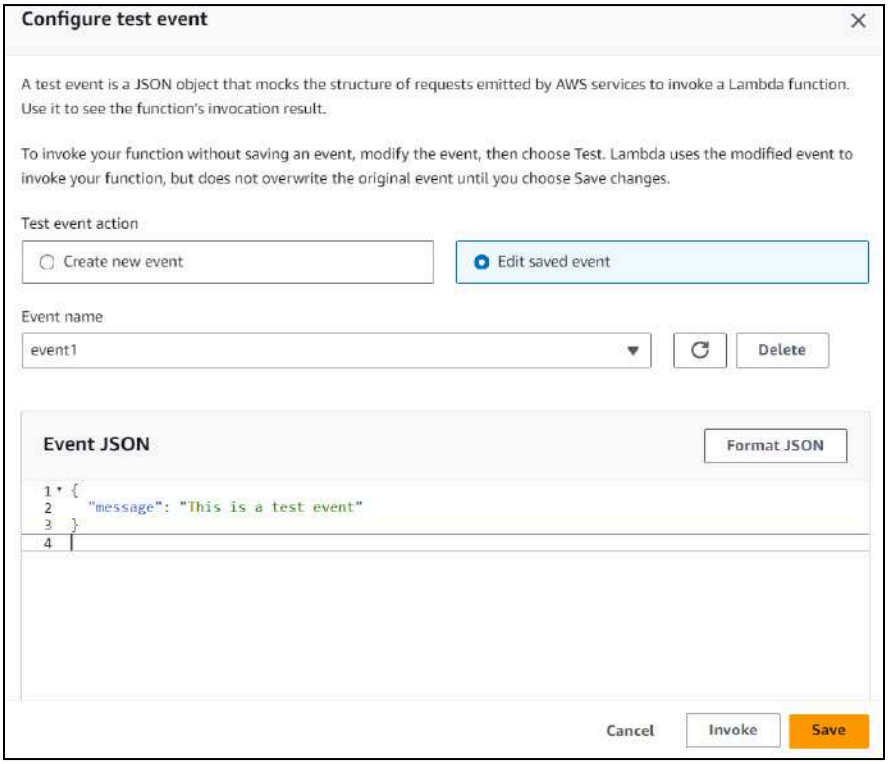

The test event event1 was successfully saved.
Code
Test
Monitor
Configuration
Aliases
Versions
Code source Info
Upload from
File Edit Find View Go Tools Window
Test
Deploy
Go to Anything (Ctrl-P)
lambda_function
Environment Vari
Environment
lambdafunction-1
lambda_function.py
1 import json
2
3 def lambda_handler(event, context):
4 # TODO implement
5 return {
6 'statusCode': 200,
7 'body': json.dumps('Hello from Lambda!')
8 }
9


Code
Test
Monitor
Configuration
Aliases
Versions
Code source Info
Upload from
File Edit Find View Go Tools Window
Test
Deploy
Go to Anything (Ctrl-P)
lambda_functi
Environment V
Execution result
Environment
lambdafunction-1
lambda_function.py
Execution results
Status: Succeeded
Max memory used: 33 MB
Time: 2.85 ms
Test Event Name
event1
Response
{
"statusCode": 200,
"body": "\"Hello from Lambda!\""
}
Function Logs
START RequestId: dee8d104-dcbf-4add-815a-f83158f5a87f Version: $LATEST
END RequestId: dee8d104-dcbf-4add-815a-f83158f5a87f
REPORT RequestId: dee8d104-dcbf-4add-815a-f83158f5a87f Duration: 2.85 m
Request ID
dee8d104-dcbf-4add-815a-f83158f5a87f

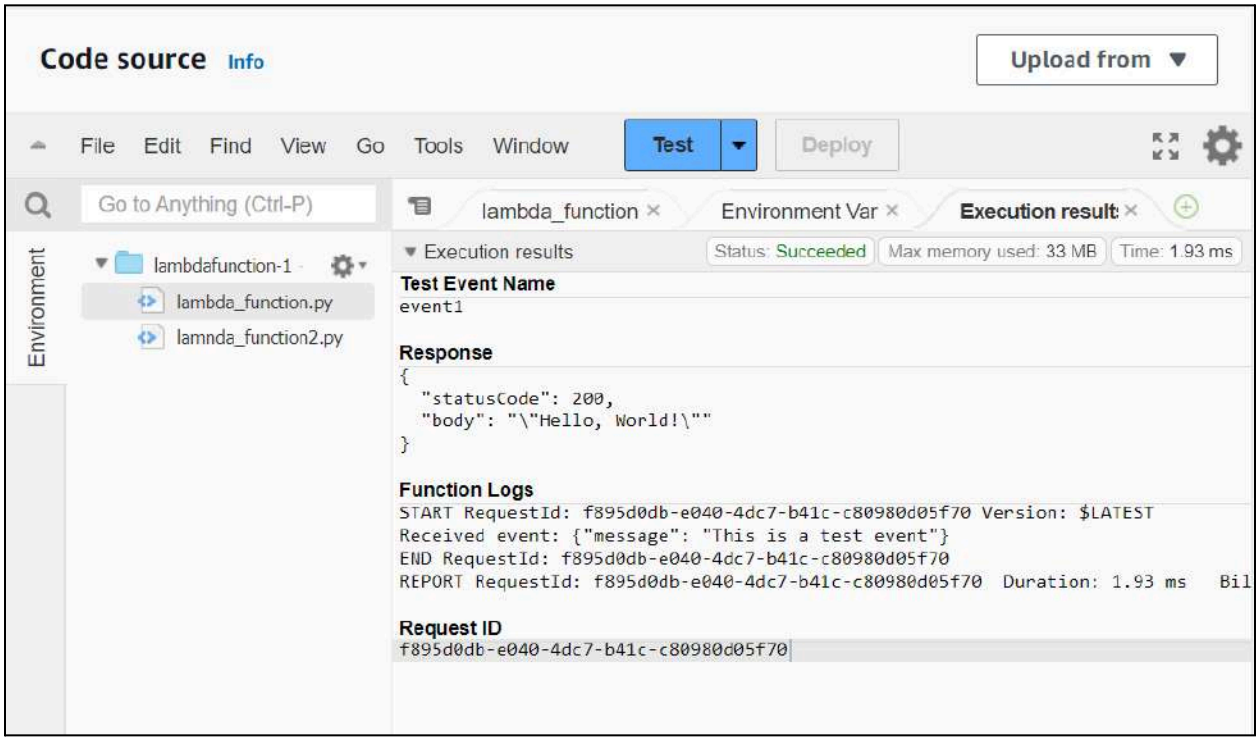
Basic "Hello, World!" Lambda Function (Python)

1. Open the AWS Lambda Console.
2. Create a new function.
3. Use the following code to test the Lambda function.





- The `lambda_handler` function is the entry point for Lambda execution.
- The function logs the received event, processes it (in this case, just returns "Hello, World!"), and sends a response with HTTP status code `200`.



Conclusion:

AWS Lambda is a serverless computing service that allows you to run code without managing servers, making it highly scalable, cost-effective, and easy to use. It automatically manages the compute resources, executes your code in response to specific events such as API calls, file uploads, or database updates, and scales based on the demand

ADVANCED DEVOPS EXP 12

Aim: To create a Lambda function which will log “An Image has been added” once you add an object to a specific bucket in S3

Theory:

AWS Lambda and S3 Integration: AWS Lambda allows you to execute code in response to various events, including those triggered by Amazon S3. When an object is added to an S3 bucket, it can trigger a Lambda function to execute, allowing for event-driven processing without managing servers.

Workflow:

1. Create an S3 Bucket: First, create an S3 bucket that will store the objects. This bucket will act as the trigger source for the Lambda function.

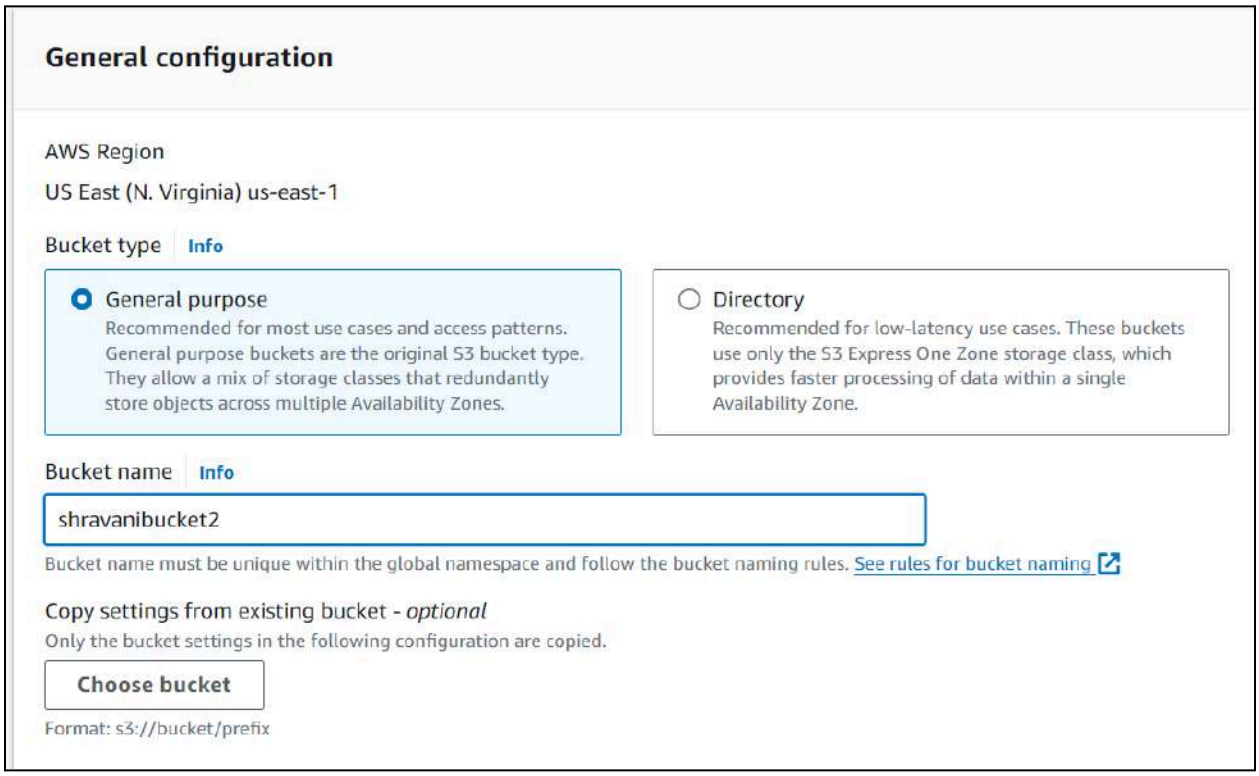
2. Create the Lambda Function: Set up a new Lambda function using AWS Lambda’s console. You can choose a runtime environment like Python, Node.js, or Java. Write code that logs a message like “An Image has been added” when triggered.

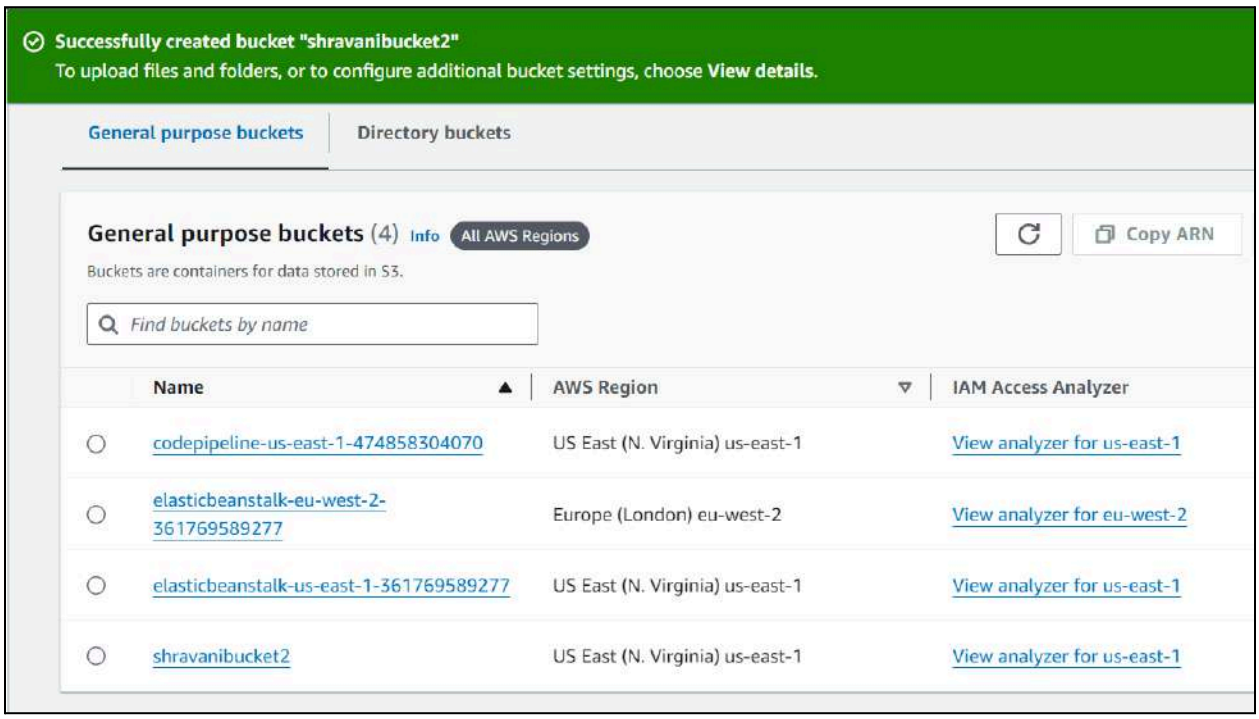
3. Set Up Permissions: Ensure that the Lambda function has the necessary permissions to access S3. You can do this by attaching an IAM role with policies that allow reading from the bucket and writing logs to CloudWatch.

4. Configure S3 Trigger: Link the S3 bucket to the Lambda function by setting up a trigger. Specify that the function should be triggered when an object is created in the bucket (e.g., when an image is uploaded).

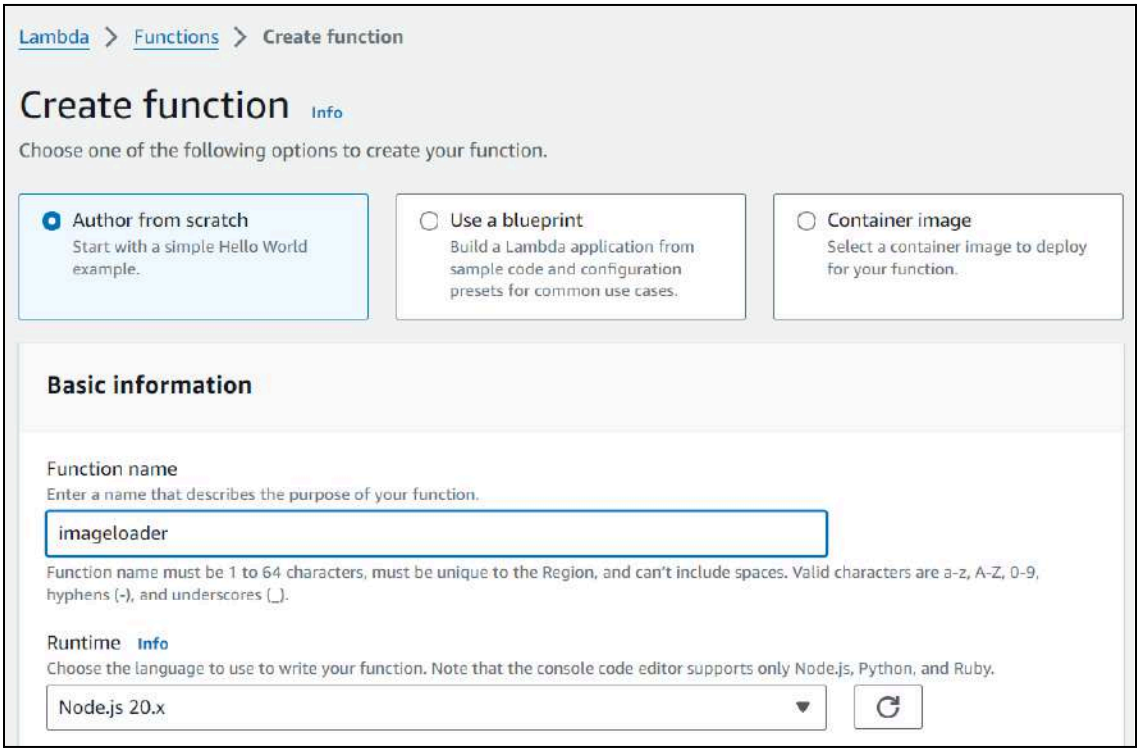
5. Test the Setup: Upload an object (e.g., an image) to the S3 bucket to test the trigger. The Lambda function should execute and log the message “An Image has been added” in AWS CloudWatch Logs

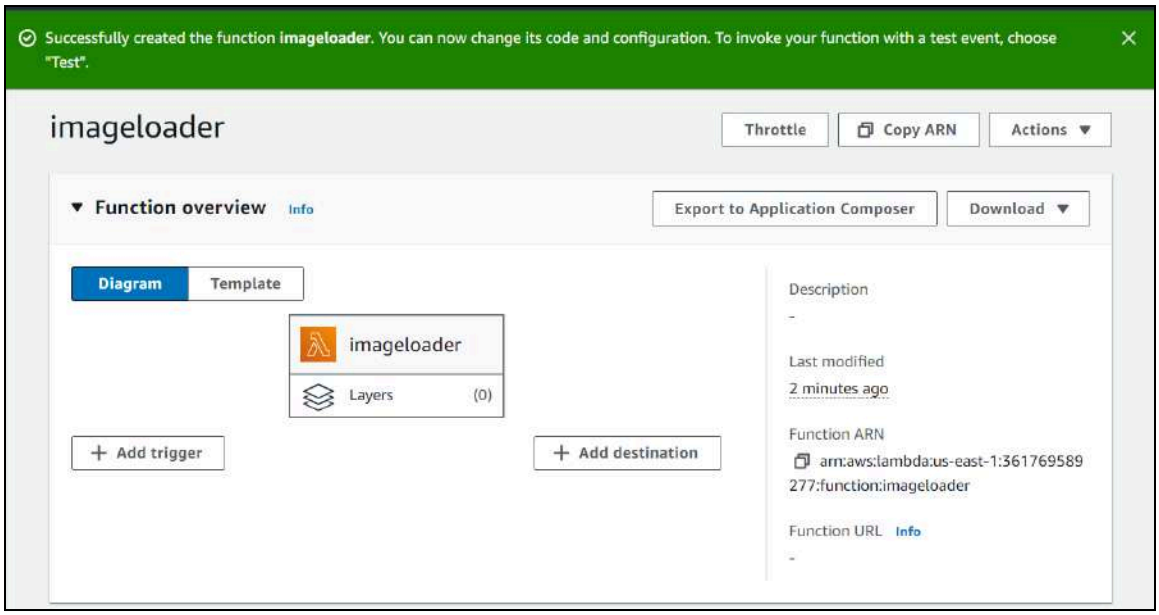
1. Create an S3 Bucket: First, create an S3 bucket that will store the objects. This bucket will act as the trigger source for the Lambda function.



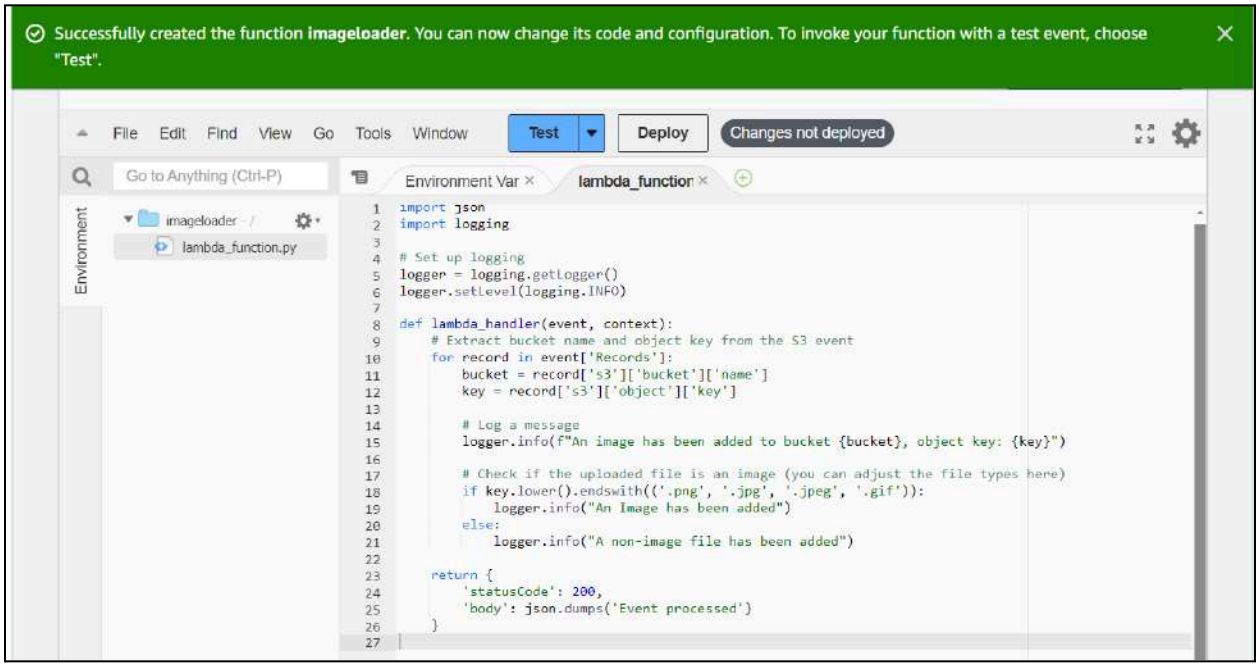


2. Create the Lambda Function: Set up a new Lambda function using AWS Lambda's console. You can choose a runtime environment like Python, Node.js, or Java. Write code that logs a message like "An Image has been added" when triggered

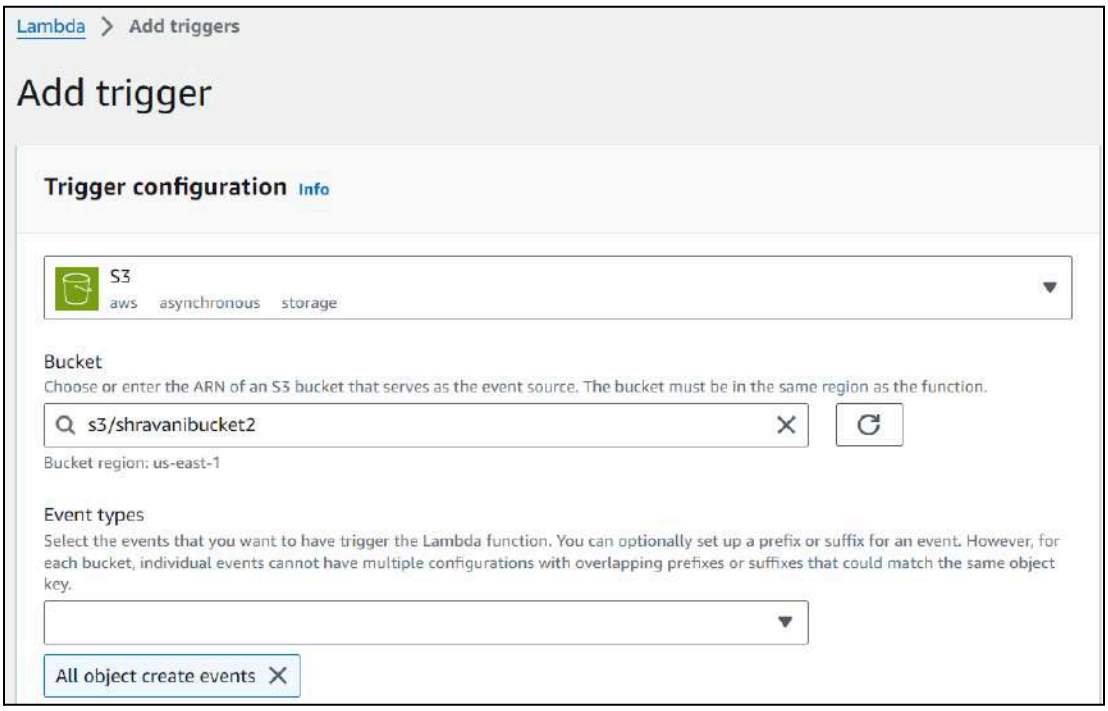


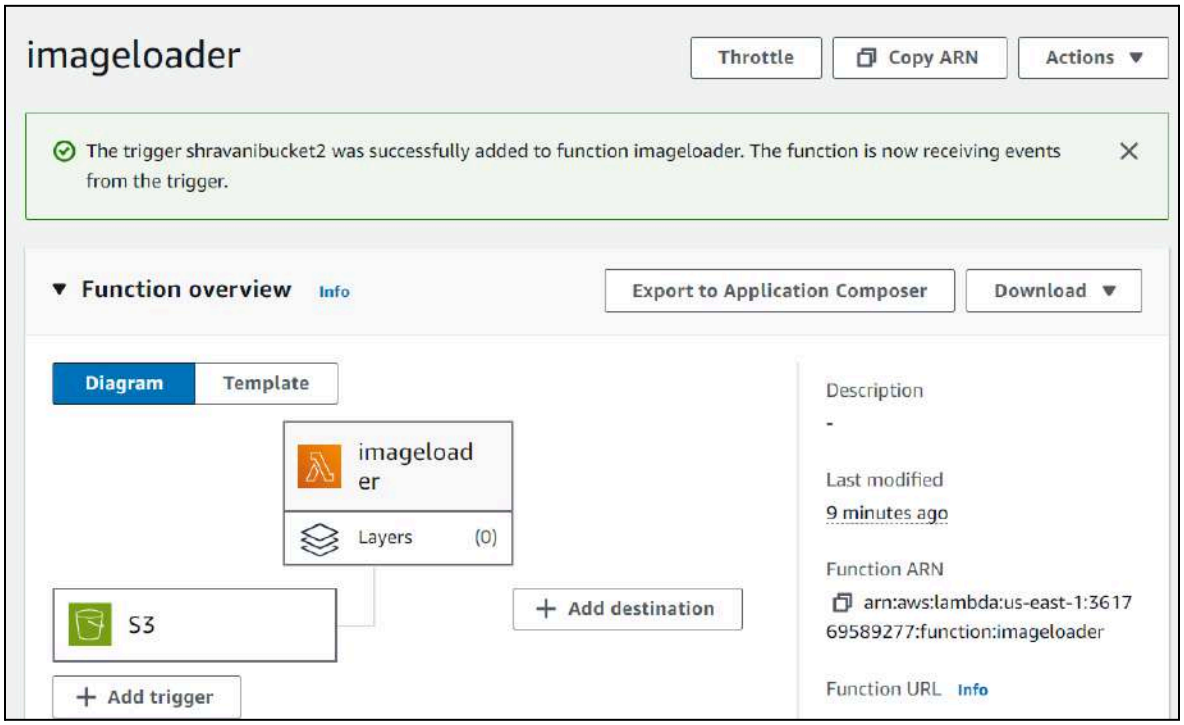


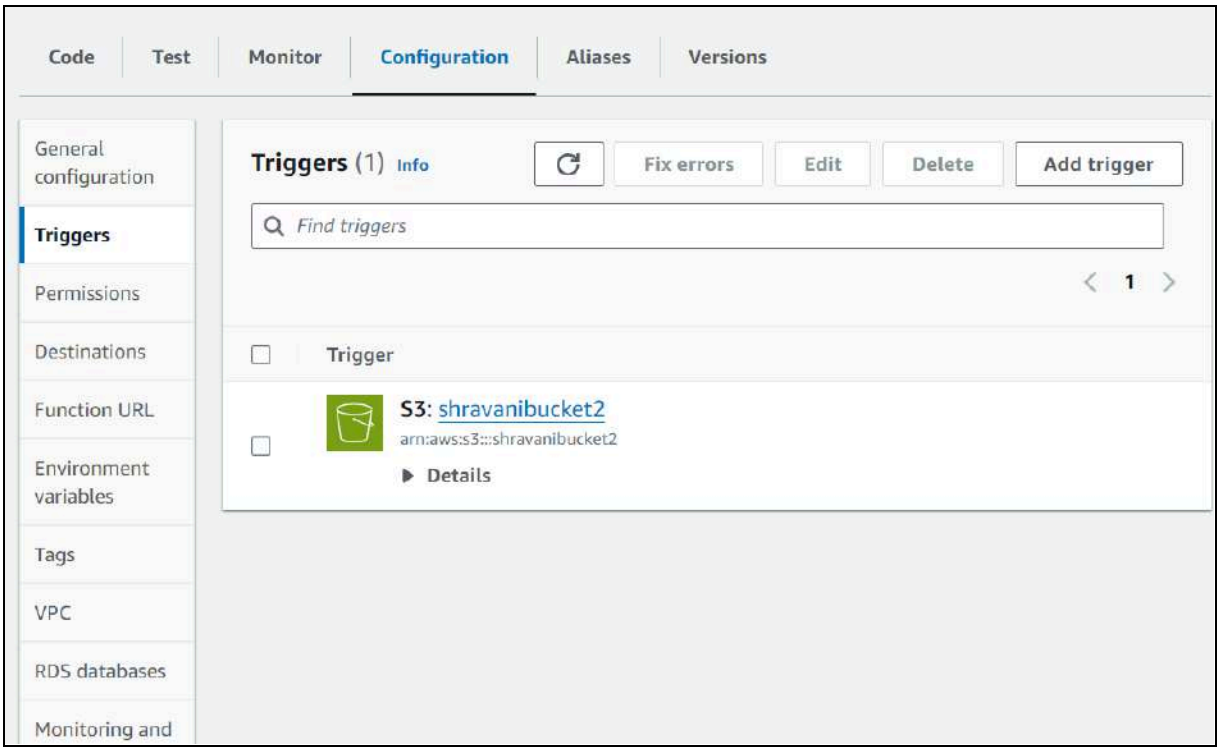
```
import json
import logging

# Set up logging
logger = logging.getLogger()
logger.setLevel(logging.INFO)

def lambda_handler(event, context):
    # Extract bucket name and object key from the S3 event
    for record in event['Records']:
        bucket = record['s3']['bucket']['name']
        key = record['s3']['object']['key']

        # Log a message
        logger.info(f"An image has been added to bucket {bucket}, object key: {key}")

        # Check if the uploaded file is an image (you can adjust the file types here)
        if key.lower().endswith(('.png', '.jpg', '.jpeg', '.gif')):
            logger.info("An Image has been added")
        else:
            logger.info("A non-image file has been added")

    return {
        'statusCode': 200,
        'body': json.dumps('Event processed')
    }
```



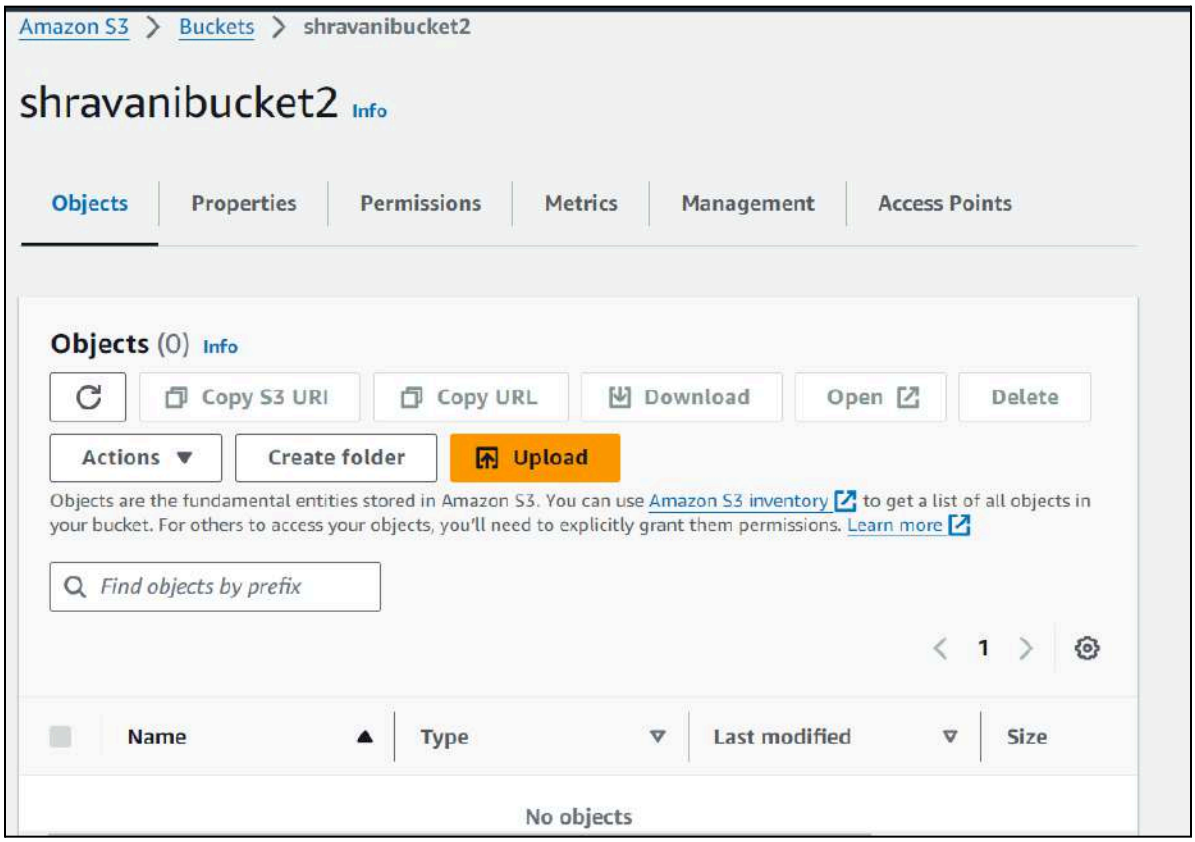
3. Configure S3 Trigger: Link the S3 bucket to the Lambda function by setting up a trigger. Specify that the function should be triggered when an object is created in the bucket (e.g., when an image is uploaded).

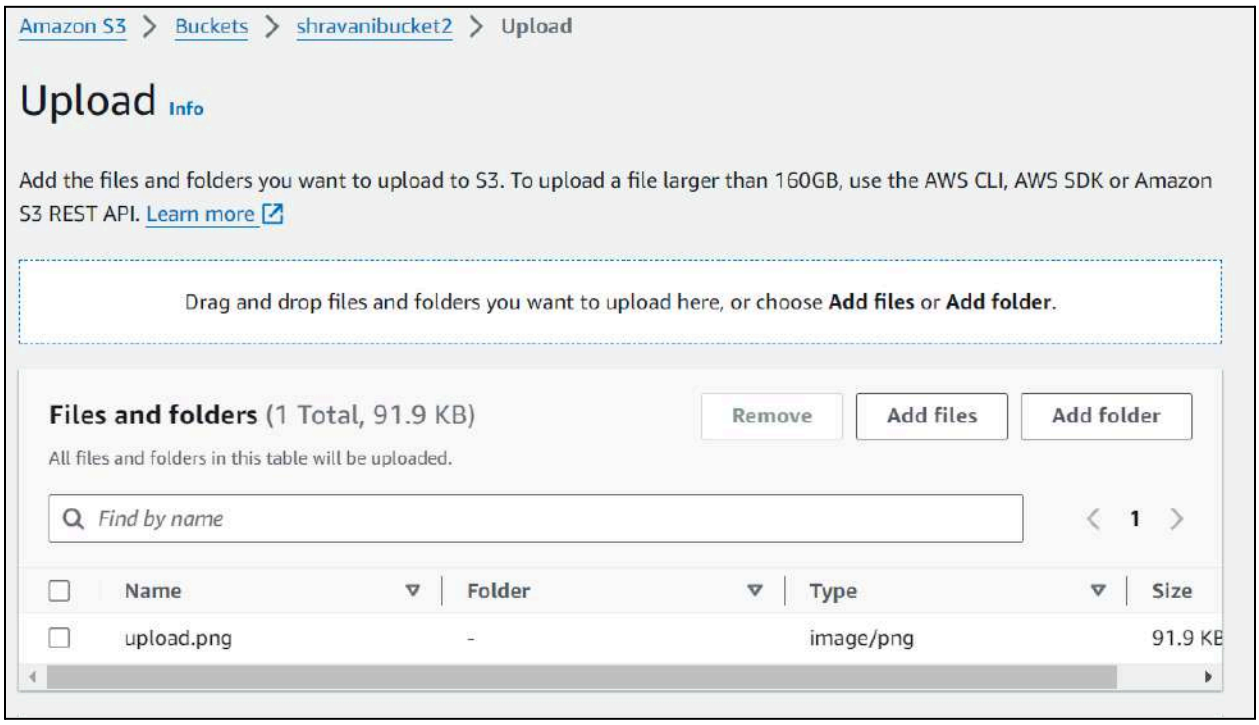


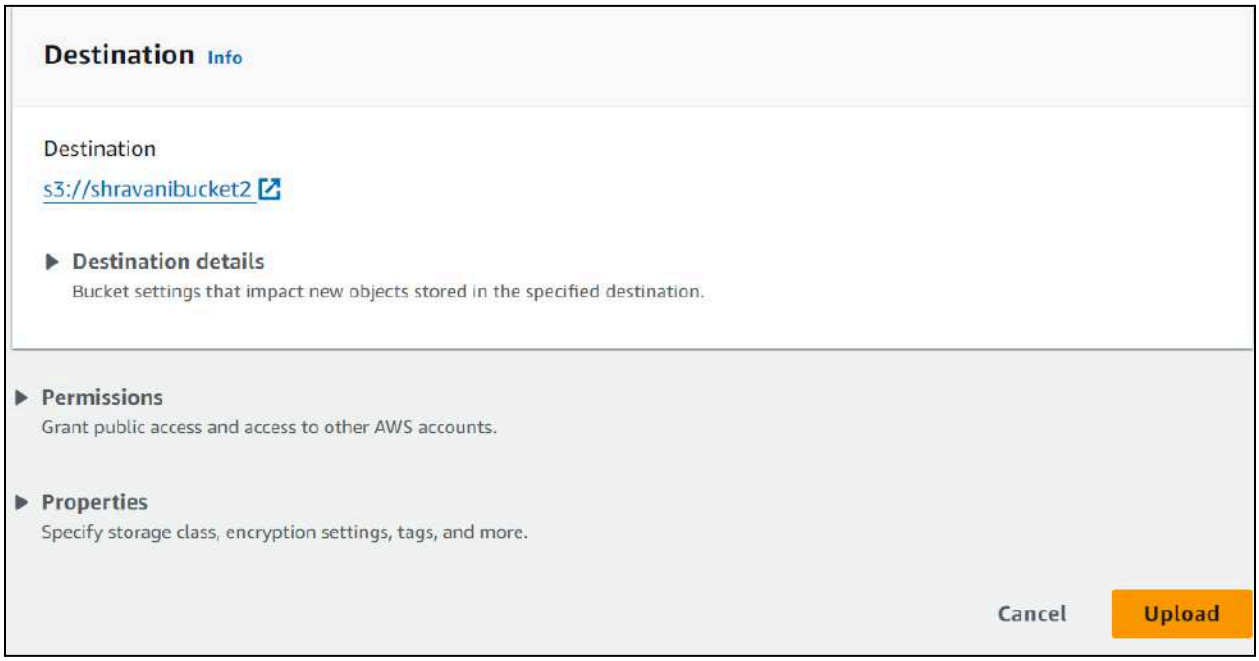


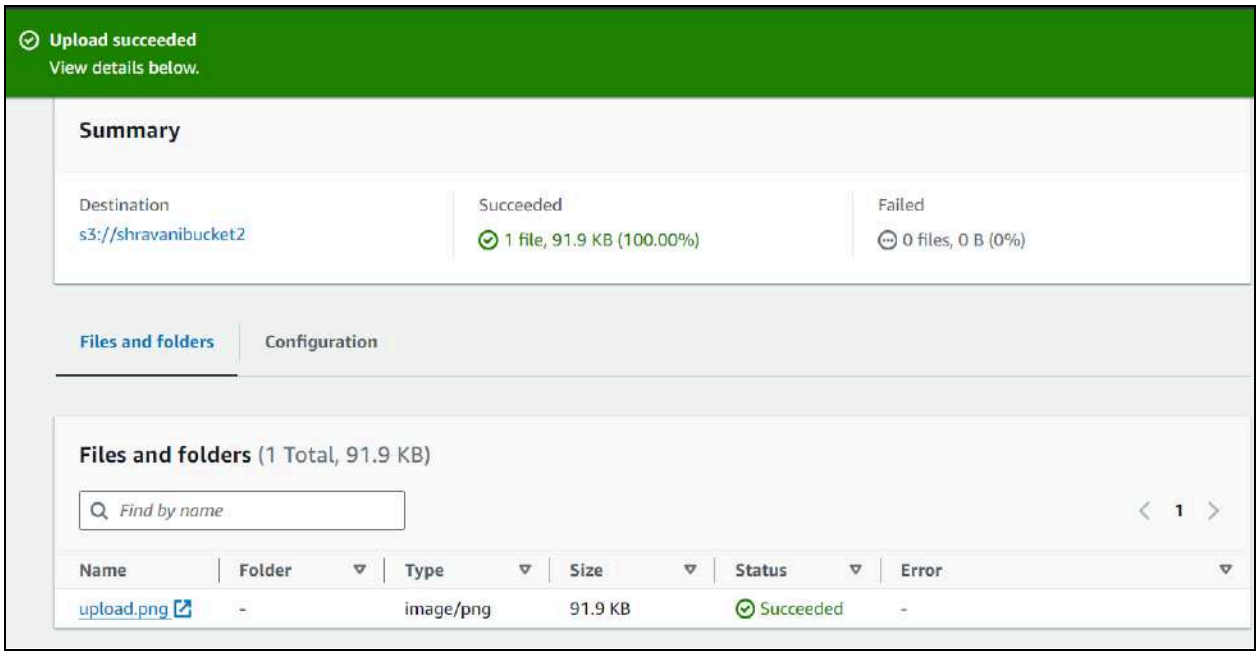


4. Upload an object (e.g., an image) to the S3 bucket to test the trigger

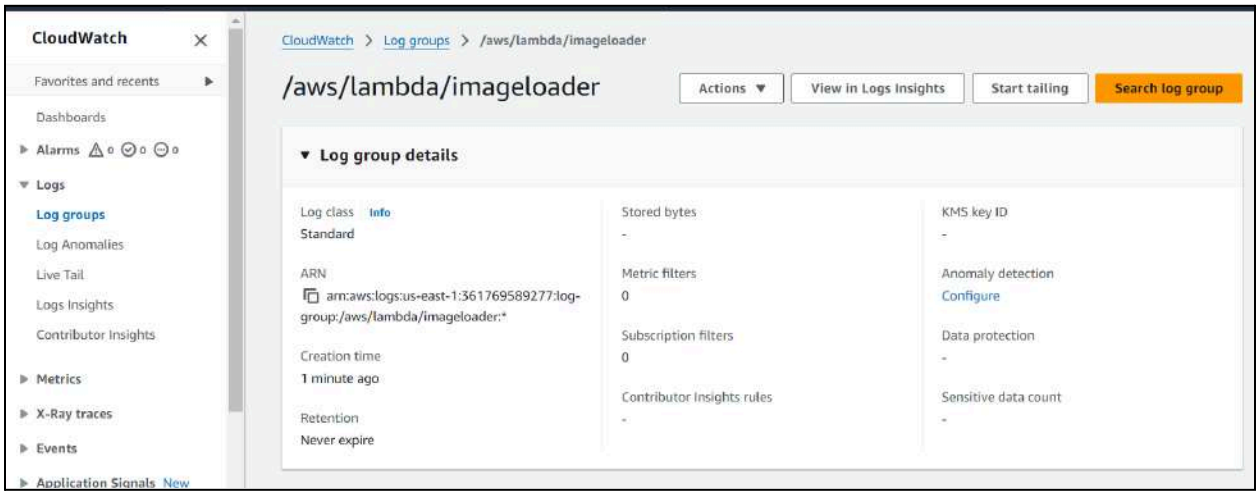


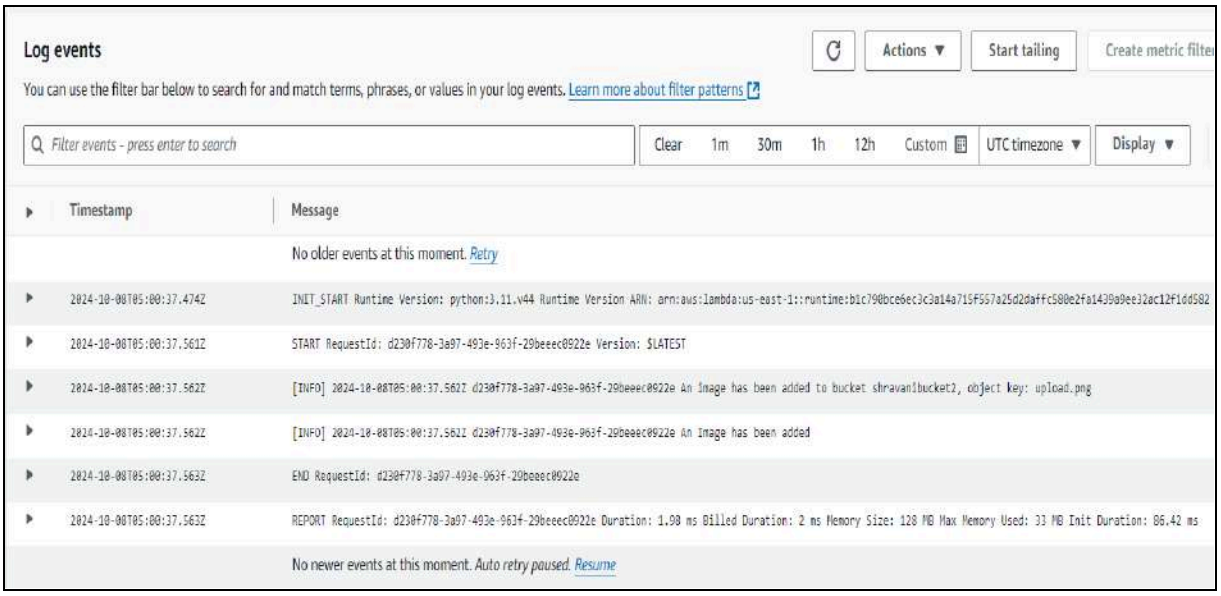

Amazon S3 > Buckets > shravanibucket2 > Upload
Upload Info
Add the files and folders you want to upload to S3. To upload a file larger than 160GB, use the AWS CLI, AWS SDK or Amazon S3 REST API. Learn more
Drag and drop files and folders you want to upload here, or choose Add files or Add folder.
Files and folders (1 Total, 91.9 KB)
Remove
Add files
Add folder
All files and folders in this table will be uploaded.
Find by name
1
Name
Folder
Type
Size
upload.png
-
image/png
91.9 KB


Destination Info
Destination
s3://shravanibucket2
Destination details
Bucket settings that impact new objects stored in the specified destination.
Permissions
Grant public access and access to other AWS accounts.
Properties
Specify storage class, encryption settings, tags, and more.
Cancel
Upload



5. Test the Setup: Upload an object (e.g., an image) to the S3 bucket to test the trigger. The Lambda function should execute and log the message "An Image has been added" in AWS CloudWatch Logs






```
An image has been added to bucket shravanibucket2, object key: upload.png

An Image has been added
```

Conclusion:

Integrating AWS Lambda with S3 allows for real-time, automated processing of events such as file uploads. In this example, a Lambda function is configured to log a message whenever an image is added to a specific S3 bucket. This setup demonstrates the power and flexibility of serverless computing by automating tasks without requiring manual intervention or server management. By leveraging AWS Lambda, developers can efficiently handle event-driven workflows, reduce operational overhead, and quickly deploy scalable solutions that respond to specific actions within cloud environments

05/05

46

NAME: SHRAVANI RASAM
CLASS: D15A
ROLL NO: 46
ADVANCE DEVOPS ASSG 1

Q1. Use S3 bucket and host video streaming

STEP 1: Create an S3 Bucket
1. Sign into AWS management console
2. Navigate to S3:
   - In AWS management console, Select S3
3. Create a Bucket
   → Click on Create Bucket
   → Enter a unique bucket name

STEP 2: Upload video to S3 Bucket
1. Open your Bucket by clicking an bucket name you created.
2. Upload files
   → Click on upload
   → Drag and drop your files
2. Set permissions
   → For public acess under permissions, upload, check grant public read acess

STEP 3: Create a cloud front Distribution
1. Navigate to cloud front forum from AWS console
1. Click on Create distribution
   chose web as delivery method.

Teacher's Sign.: _____

3. Configure the distribution
- Origin Domain: select your S3 bucket
- Viewer protocol policy: Choose redirect HTTP to HTTPS for secure access
- Cache Behaviour setting
- Click create distribution

STEP 4: Configure cloud format for secure access
1. Creat an origin access identity (OAI)
- In cloud front origin console, go to distribution setting
- under origins and origin group click edit
- create a new origin access Identity
2. Update S3 bucket policy
- Go to your S3 bucket
- Click on permission and then bucket policy
- Add to policy to grant access to OAI

STEP 5: Access the video through cloud front
1. Get the cloud front: URL
- In cloud front URL console: Go to distribution
- Copy the domain name
2. Use the URL
- Use this URL in your webpages to stream the video.

Q2 Discuss BMW and hotstar case studies using AWS

→ OVERVIEW: BMW is leading automative manufacturer known for its luxury vehicals while Hotstar (now known Disney + Hotstar) is a popular streaming platform in India, offering a variety of content including movies, TV shows and live sports. Both companies have utilized AWS to optimize their operations

BMW's Use of AWS

1. Connected Vehicles and Data Analytics
BMW has been at the forefront of integrating technology into their vehicles. By leveraging AWS they can collect and analyze vast amount of data.
That data includes
i) Vehicle Performance Data
ii) Driver Behaviour

Benefits:
1. Predictive Maintenance: AWS enables BMW to use machine learning algorithms to predict when a vehicle needs servicing, reducing downtime.

Teacher's Sign.:

2. Enhanced Customer Insights: By analyzing driving patterns, BMW can tailor marketing strategies and develop features that resonate

2. Scalability and Cost Management
BMW utilizes AWS's scalable infrastructure to handle varying workloads, especially during product launches or events

Benefits:
1. Cost efficiency: BMW can scale resource up or down based on demand, ensuring they only pay for what they use
2. Global Reach: AWS's global infrastructure allows BMW to deploy application closer to their customer, reducing latency and improving service delivery

Hotstar's use of AWS

1. Content Delivery and Streaming Services
Hotstar relies heavily on AWS to manage its massive content library and delivery high-quality streaming experience to users

Teacher's Sign.: _____

Benefits

1. Scalability: During events like IPL user traffic can spike drastically. AWS allows Hotstar to scale resources dynamically to handle these spikes without compromising performances

2. Global Content Reach: AWS enables Hotstar to distribute content across multiple regions ensuring that users worldwide can access their services seamlessly.

2. Data Analytics for User Engagement

Hotstar leverage AWS data analytics tools to gather insights about user behaviour content preferences and viewing patterns.

Benefits:

1. Personalized Content Recommendation: By analyzing viewing habits, Hotstar can suggest relevant content to users, enhancing their viewing experience.

2. Targeted Advertising: Insights gathered from user data enables Hotstar to serve more targeted ads, increasing and revenue and improving user satisfaction.

Teacher's Sign.:

Challenges and Solutions:

While both BMW and Hotstar have seen significant benefits from using AWS, they also face challenges:

1. Data Security and Compliance: Protecting user data and adhering to regulations is paramount

Solution: Both companies utilizes AWS's robust security features such as encryption, identity and access management.

2. Cost Management: As usage scales, managing costs can become challenging.

Solution: Implementing AWS Cost Explorer and using AWS Budgets help both companies monitor and optimize their cloud expenditure

CONCLUSION:

The integration of AWS into BMW and Hotstar's operations demonstrates how cloud computing can drive innovation, improve customer experiences and enhance operational efficiency.

Q3 Why Kubernetes and advantages and disadvantages of Kubernetes. Explain how adidas uses Kubernetes.

- Kubun
- KUBERNETES:

Kubernetes is an open-source container orchestration platform designed to automate the deployment, scaling and management of containerized applications. Originally developed by Google, it has become a standard for managing cloud-native applications.

ADVANTAGES:

1. Automated Deployment and Scaling: Facilitates easy deployment and scaling of applications based on demand.

2. Self-Healing: Automatically restarts or replace failed containers ensuring high availability.

3. Load Balancing: Distributes traffic across containers for optimal resource use.

4. Declarative Configuration: Allows users to define application state using YAML

5. Multi-Cloud Support - Can run on various cloud platforms and on-premises

DISADVANTAGES

1. Complexity: Steep learning curve and operational challenges, especially for beginners

2. Resource Overhead: Can require significant computational resources

3. Networking Challenges: Configuring networking can be complex

4. Frequent Updates: Rapid evolution can lead to compatibility issues

USE OF KUBERNETES in Adidas

1. Microservice Architecture:
Adidas has adopted a microservices architecture to enable agility and faster delivery of features. Kubernetes allows Adidas to manage these microservices. Each Microservice can be developed, deployed and scaled independently

2. Scalability and Performance:
During high-traffic events (like product launches or major sales), Adidas can use Kubernetes to scale its applications automatically based on user demand.

3. Continuous Deployment and Integration
Adidas employs CI/CD pipeline, and Kubernetes plays a crucial role in this process. By integrating Kubernetes with their CI/CD tools, Adidas can automate testing and deployment, ensuring the new features and updates

4. DevOps and Collaboration
Kubernetes supports a DevOps culture at Adidas, allowing development and operations teams to work closly together.

5. Cost Management:
Kubernetes helps Adidas optimize resource usage allowing them to allocate computing resources more efficiently. This can lead to significant cost savings, especially when operating in cloud environments where usage pricing is common.

Teacher's Sign.:

Q4 What are Nagios and explain how Nagios are used in it Services?

→ NAGIOS:

Nagios is an open-source monitoring system that enables organizations to monitor their IT infrastructure, including servers, networks and applications. It provides real-time insights into the status of various components.

KEY FEATURES:

1. Host and Service Management:
Nagios monitors the overall availability and performance of hosts (servers, devices) and services (applications, processes)

2. Alerts and Notification:
Sends alerts via email or SMS when issues arise, ensuring prompt attention

3. Customizable Dashboards: Offers visual dashboard to represent the health of IT resources at a glance

4. Extensibility: Supports plugins that can extend its functionality, allowing monitoring of custom applications or services.

Teacher's Sign.: _____

Use of NAGIOS in E-Services:

1. Infrastructure Monitoring:
E-service providers use Nagios to monitor servers and networks devices, ensuring uptime and reliability.

2. Application Performance Monitoring:
Nagios can monitor the performance of web applications, databases and other services

3. Proactive Issue Resolution:
By setting thresholds for various metrics, Nagios can alert administrators before issues become critical.

4. Capacity Planning:
Historical data collected by Nagios helps e-service providers understand usage patterns and plan for further growth. This is essential for scaling resources effectively as user demand increases.

Teacher's Sign.: _____

NAME: SHRAVANI RASAM
DISA 46

62 1

ADVANCE DEVOPS
ASSIGNMENT 2.

Create a REST API with the serverless framework

The serverless framework helps you deploy applications to cloud providers like AWS using a simplified configuration

1) Install Serverless Framework: Ensure you have Node.js installed, then install the serverless framework using npm install -g serverless

2) Set up AWS credentials: Serverless uses AWS Lambda and API Gateway so configure your AWS console aws configure

3) Create your AWS Access Key, Secret key region etc.

4) Create a New Service: Create a new project using a Node.js template
serverless create --template aws-nodejs --path rest-api

This creates a basic serverless service with a structure including serverless.yml and handler.js file for code

5) Define the REST API in serverless.yml:
Edit the serverless.yml to define the REST API endpoints:
For eg: service: rest-api-services
provider:
name: aws
runtime: nodejs 14.x
functions:
hello:
handler: handler.hello
events:
- http:
path: hello
method: get

The handler.js file would contain the logic

6) Deploy the service: Deploy the API to AWS Lambda and API gateway using: serverless deploy

This deploys infrastructure and you'll get a URL to access API

Teacher's Sign.:

7) Testing : Use the URL provided to access your REST API. You can test it with tools like postman or simply via your browser.

2 Case Study for Sonarqube

- SonarQube helps to automatically review your code for bugs, vulnerabilities and code smells. The steps below cover Java, Python and Node.js analysis

1) Create a SonarQube Profile:
Go to SonarQube cloud and create an account
You can link this account to your github profile to analyze code directly from repositories
Create a project in Sonar Cloud and connect it with your github repository

2) Analyze code on Sonar Cloud:
For your Github repository, configure it with Sonar cloud. This can be done using CI pipelines (Eg. Github Actions) or manually

Teacher's Sign.: _____

Example of Github Actions YAML for Sonar Cloud

```
name: Sonar Cloud
on:
  push:
    branches:
      - main
jobs:
  sonarcloud:
    runs-on : ubuntu-latest
    steps:
      - uses : actions/checkout@v1
      - name : SonarCloud Scan
        uses : SonarSource/sonarcloud-github-
                                    action@v.14
        with:
          args:>
            -Dsonar.projectkey = my-project-key

            -Dsonar.organization = my-org

            -Dsonar.host.url = https://sonarcloud.io
```

Teacher's Sign.:

3. Sonarlint setup in Java IDE:
Install Sonarlint in intelliJ IDE or Eclipse from the plugin market place

Configure it to link with your sonar cloud account for continuous quality checks

Once installed SonarLint will analyze your Java code locally for issues

4. Python Project Analysis:

1) Create a Sonar-project.properties file in the root of your python project.

Sonar project key = mypython project.
sonar.organization = my-org
Sonar.sources = .

Run the analysis using SonarScanner
sonar-scanner

5) Nodejs Project Analysis:

For a Node.js project the steps are similar to python. configure a sonar-project.properties file and scan the project using sonar scanner

Teacher's Sign.: _____

Example: sonar - project - properties

sonar.projectKey = my-nodejs-project
sonar.sources =
sonar.exclusions = node_modules/**,**/*.test.js

Run sonar-scanner to analyze your Node.js project.

Analyze Results:
Just like with Python, the results of the analysis will be uploaded to Sonar Cloud. The report can be accessed from the dashboard which will highlight issues such as missing semicolons, unused variables and more.

Key features:

1) Sonar cloud allows you to integrate with Github easily and provides a cloud based dashboard for viewing the quality of your project.

2) Sonarlist helps developers fix issues in real time as they write code, making it easier to catch issues before they are committed

Teacher's Sign.:

3) For Python and Node.js the sonar-project properties file is essential for configuring the analysis and the sonar-scanner tool helps run the analysis locally

83. Let a large organization, your centralized operations team get many repetitive infrastructure request. You can use Terraform to built a '" self-serve" infrastructure model that lets product teams manage their own infrastructure independently. You can create and use Terraform modules that codify the standards for deploying and managing services in your organization, allowing teams to efficiently deploy services in compliance with your organizations practices. Terraform cloud can also integrate with ticketing systems like Service-now to automatically generate new infrastructure requests

ns- The goal of this task is to use Terraform to create a reusable infrastructure model enabling teams to deploy and manage their own resources without involving central operations

Teacher's Sign.: _____

1. Understand the self serve infrastructure model:

In large organizations product teams often requests infrastructure resources from a central ops team. This can be repetitive and time-consuming.

By using Terraform you can create reusable modules that product teams can use independently to deploy their own infrastructure based on organization standards.

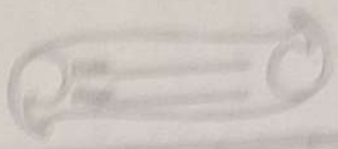
2. Creating Terraform Modules:

Modules allow you to reuse Terraform code- create a module that defines a common infrastructure component, such as an EC2 instance.

Teams can use this module in Terraform configurations

```
module "ec2" {
    source = "./ec2-instance"
    ami_id = "ami-0cbcd1234"
    instance_name = "team-app-instance"
}
```

Teacher's Sign.:



5) Automating with Terraform Cloud and Ticketing Systems

Terraform Cloud allows you to collaborate on infrastructure deployments. It can be integrated with a ticketing system like service now to automate infrastructure requests

This integration ensures that infrastructure is deployed in compliance with the organization security and governance standards

Key Tools to Use:

1) Serverless framework: for deploying REST API's

2) SonarQube / SonarCloud: for code quality analysis

3) SonarLint in IntelliJ / Eclipse for on-the-fly Java Analysis

4) Terraform: for infrastructure automation and self-service infrastructure.

Teacher's Sign.: _____