Name: Kshitij Hundre
Div: D15C
Roll No:18

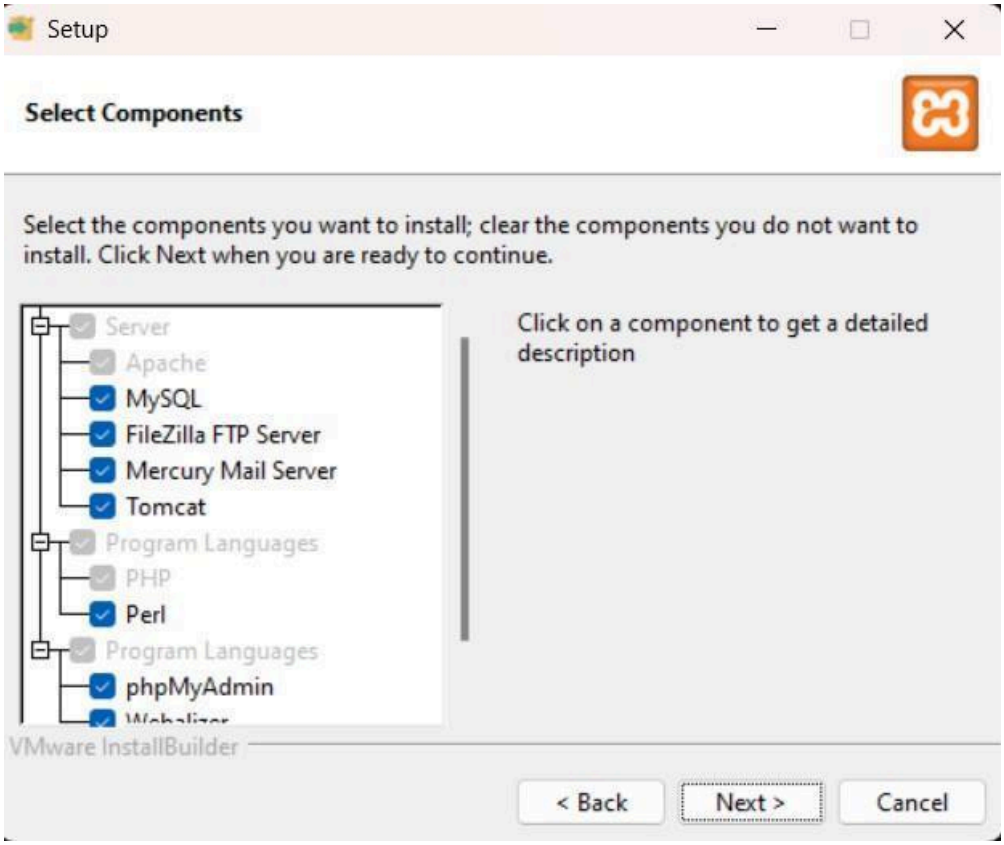
Exp 1 :Static Hosting:

1) On local server (XAMPP)

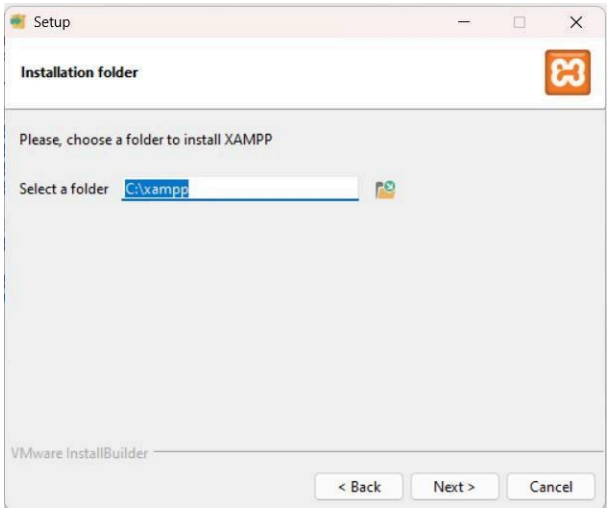
Step 1: Install XAMPP from https://www.apachefriends.org/

1) Select your OS. It will automatically start downloading.

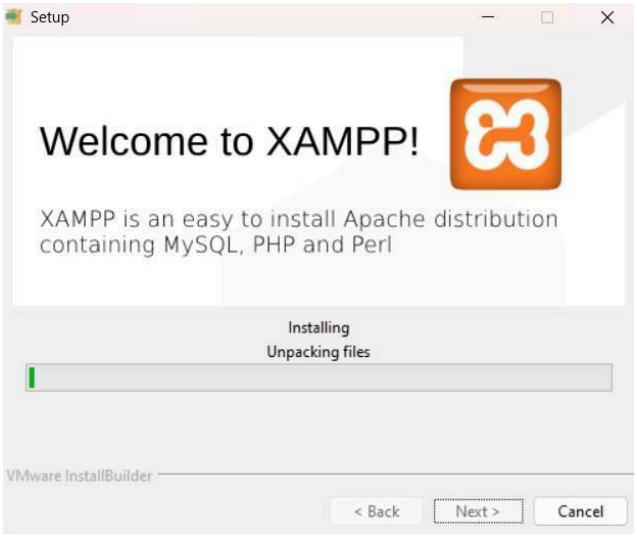
2) Open the setup file. Select all the required components and click next



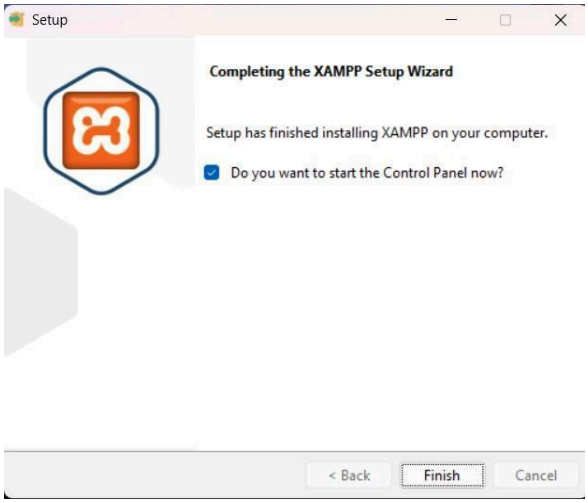
3) Choose the folder to install XAMPP in. Make sure the folder is empty. Click next



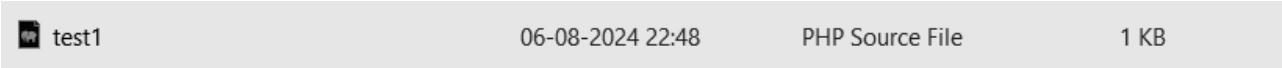
4) Select the language, click next. XAMPP starts to install



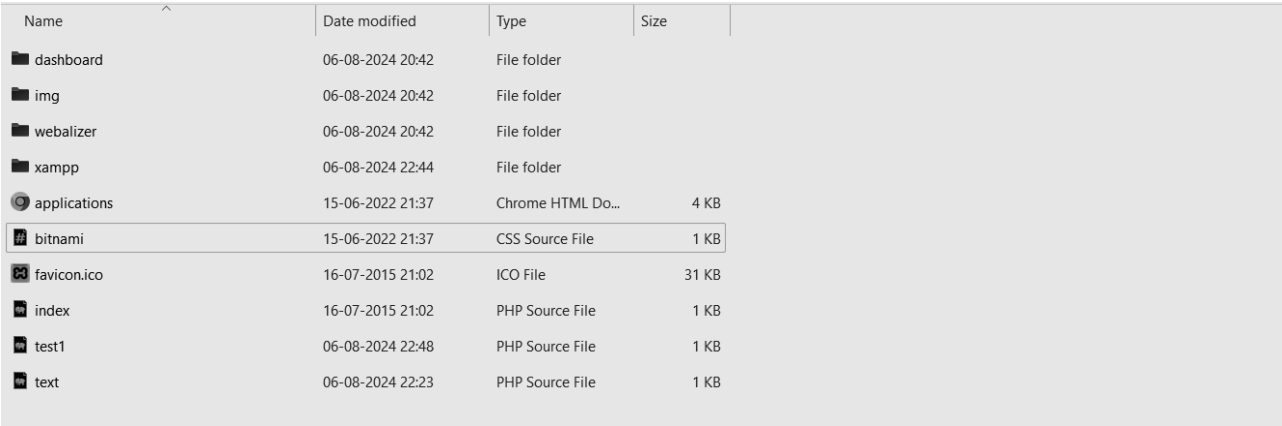
5) The installation is complete. Click Finish



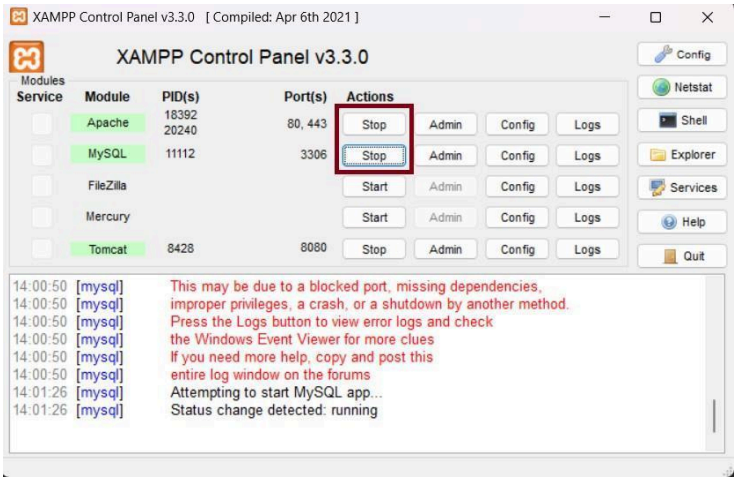
Step 2: Setup a file that is to be hosted on the server. Make sure the file has extension **.php**



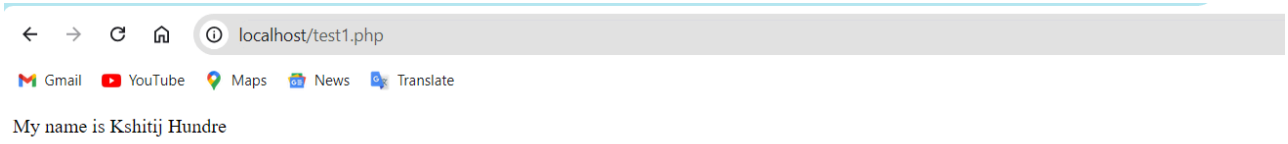
Step 3: Go to the directory where XAMPP was installed. Go to **htdocs** folder. Place your folder in this directory.



Step 4: Open XAMPP Control Panel, start the Apache service (Required) and mySQL service (if needed)

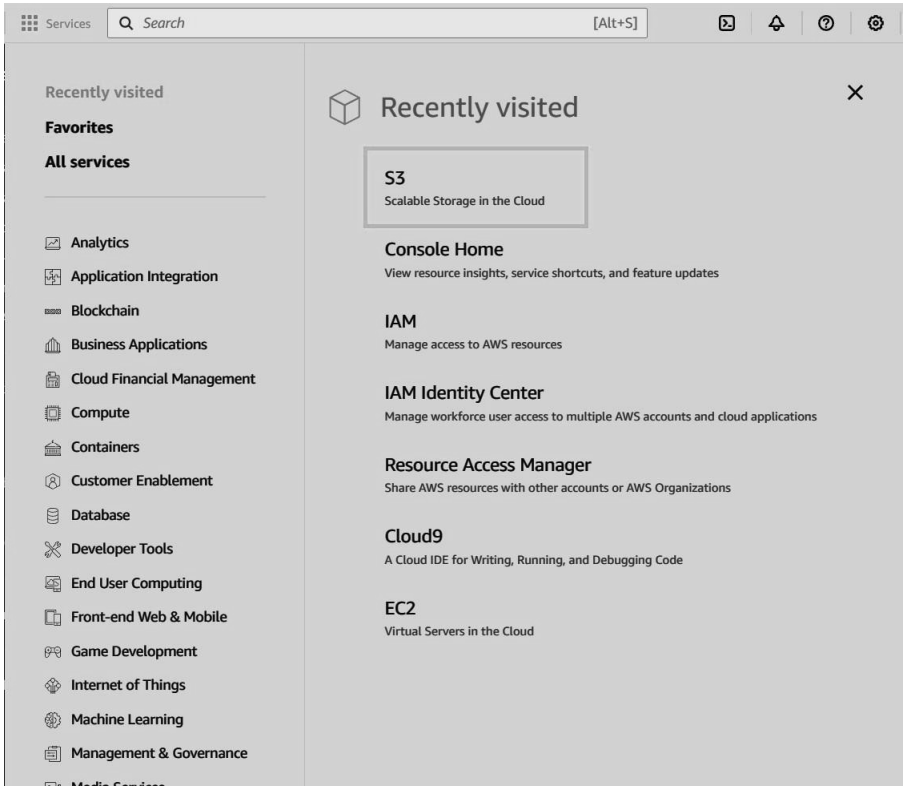


Step 5: Open your web browser. Type localhost/YOUR_FILENAME.php. This will open your website on your browser.

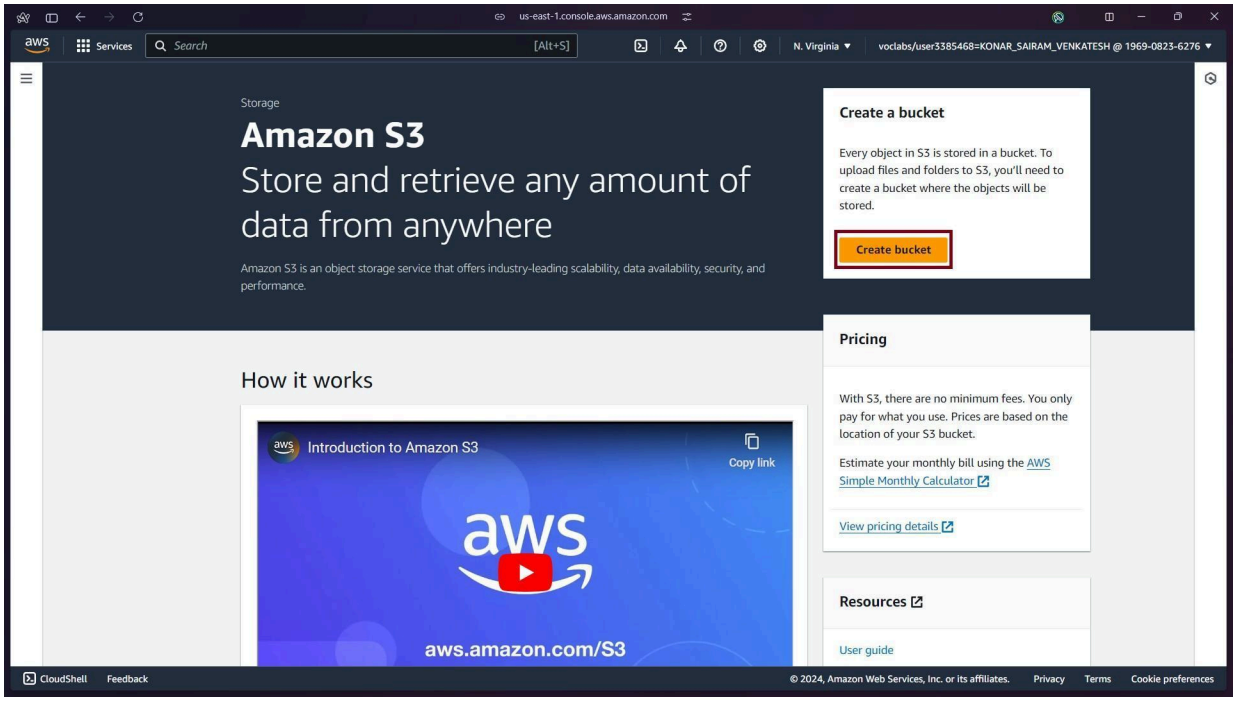


2) AWS S3

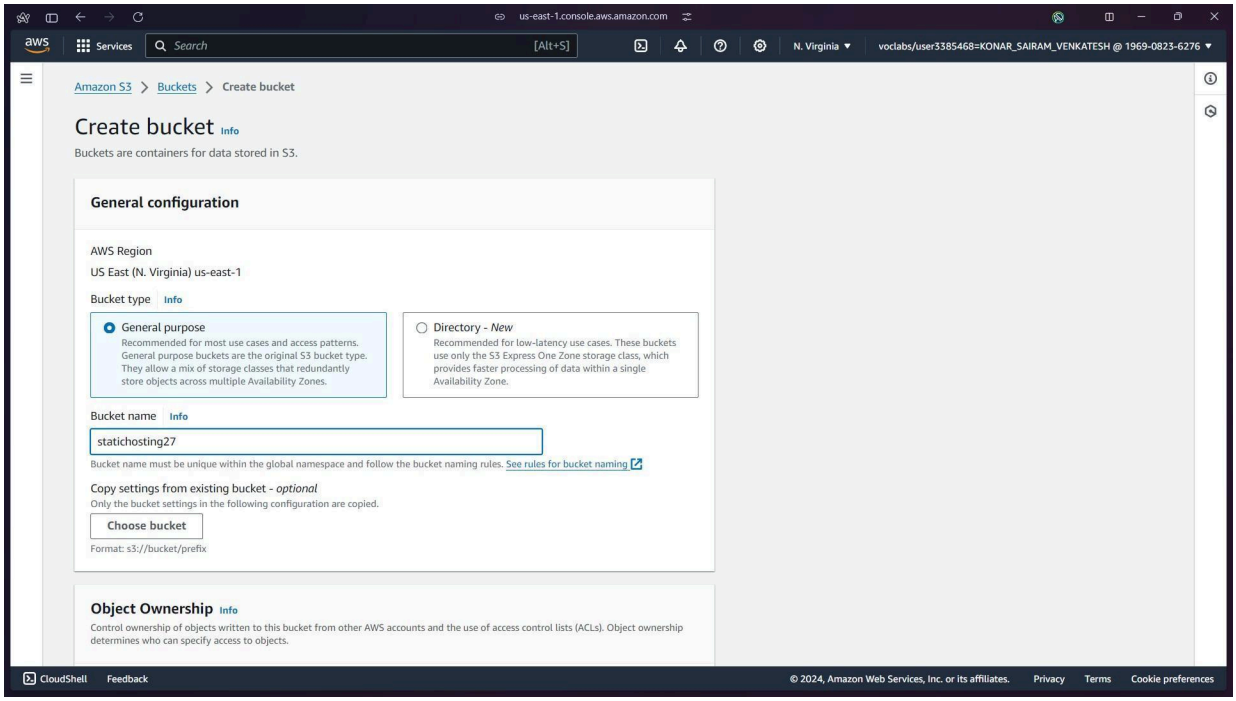
Step 1: Login to your AWS account. Go to services and open **S3.**



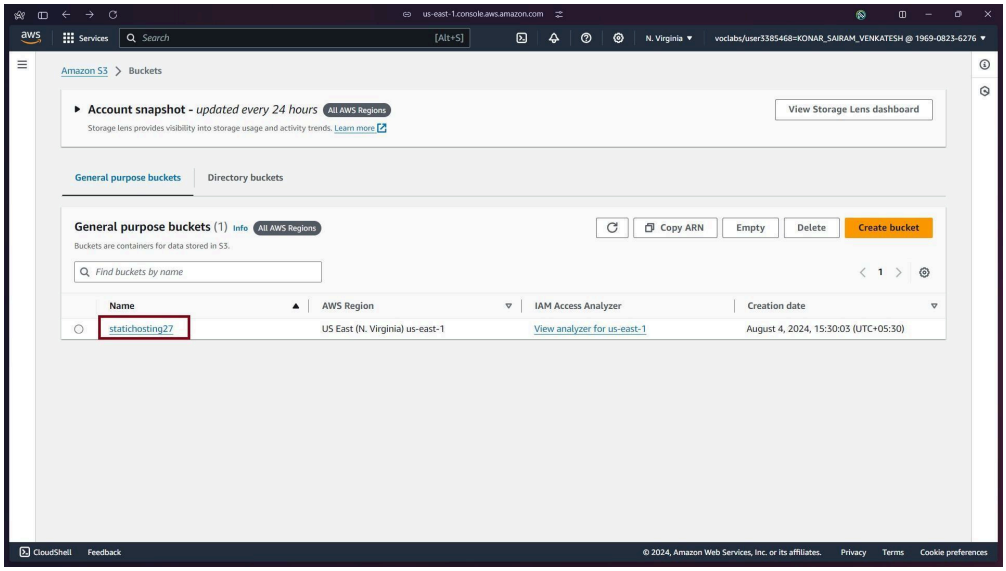
Step 2: Click on Create Bucket

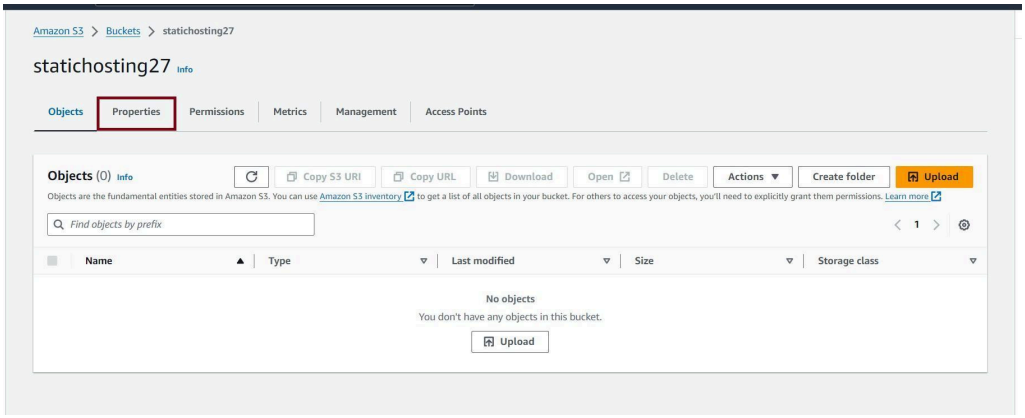


Step 3: Give a name to your bucket, keeping other options default, scroll down and click on Create Bucket

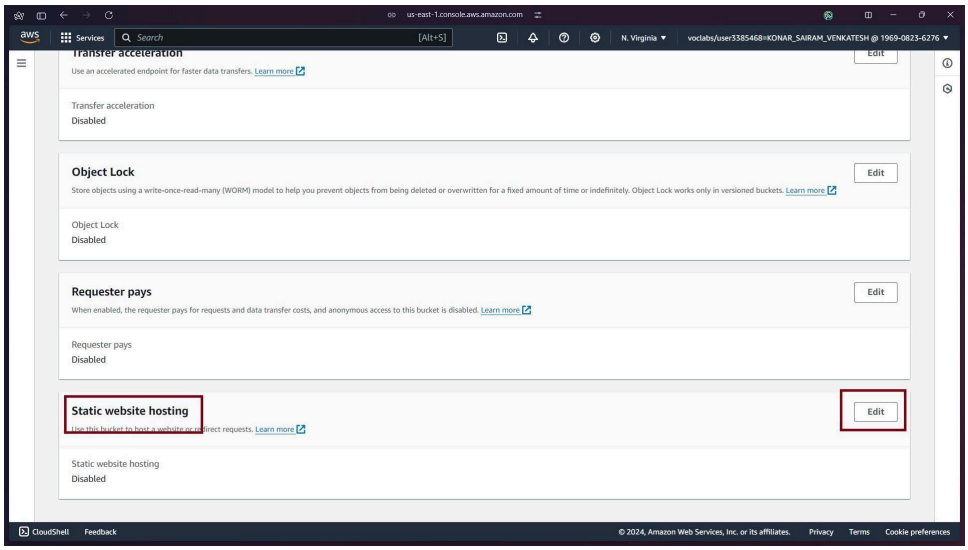


Step 4: Click on the name of your bucket and goto Properties

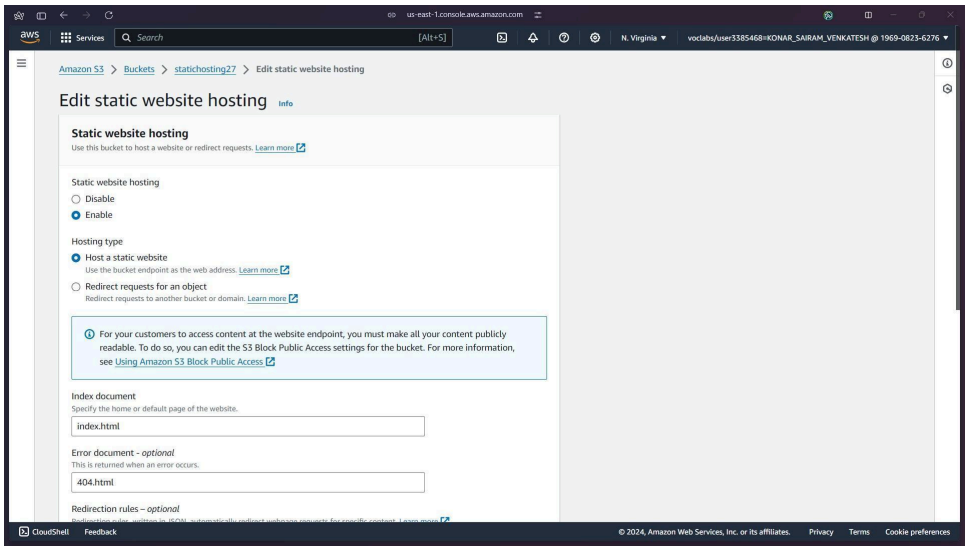




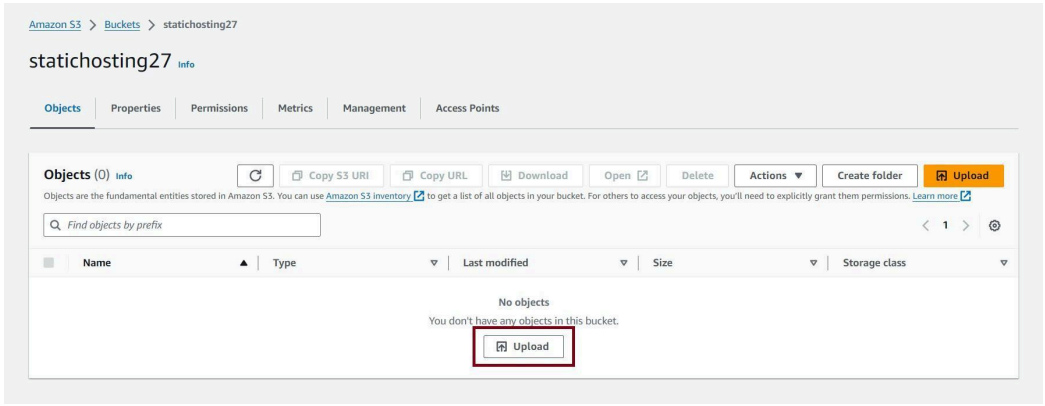
Step 5: Scroll down till you find Static website hosting, click on edit



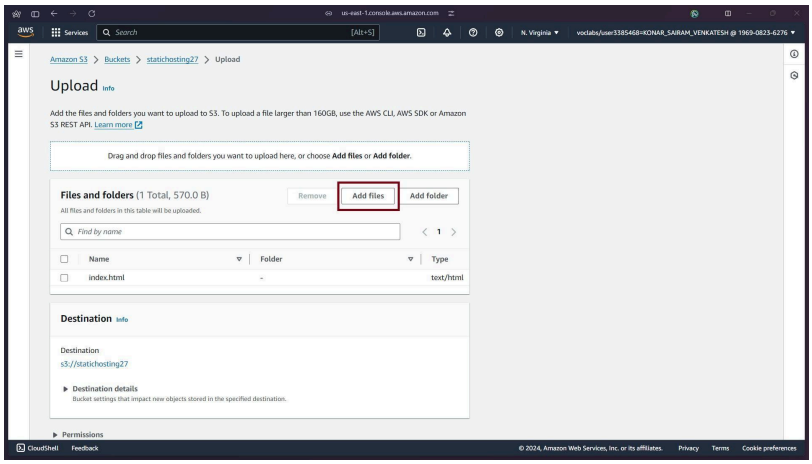
Step 6: Enable static website hosting, in Index document, write the name of your document and in error document, give name as 404.html. Save your changes.



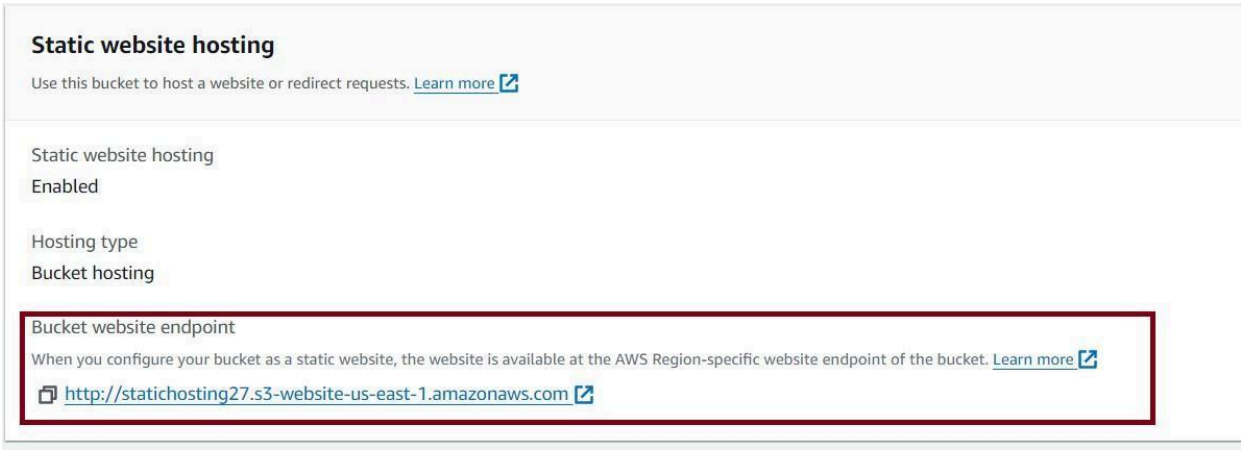
Step 7: Go to Objects tab and click on upload file.



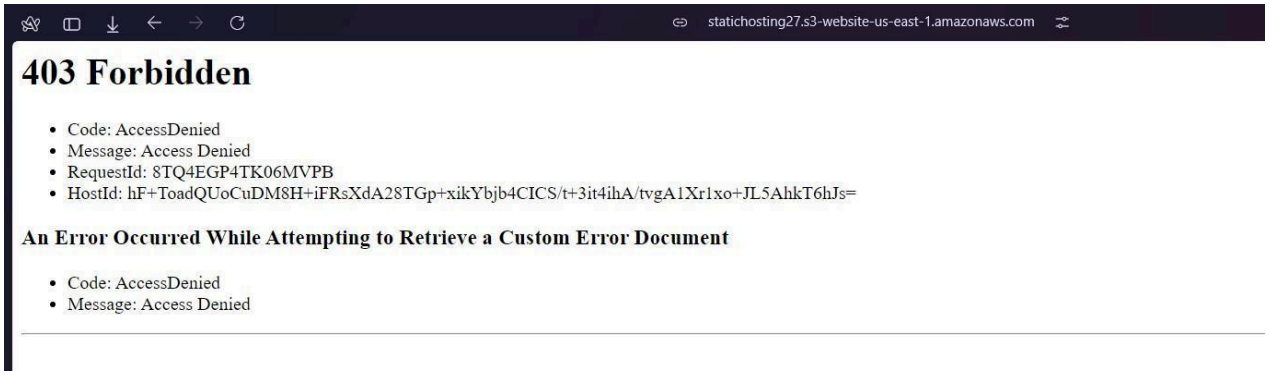
Step 8: Click on Add files. Add all the files you want to upload. Then scroll down and click on Upload



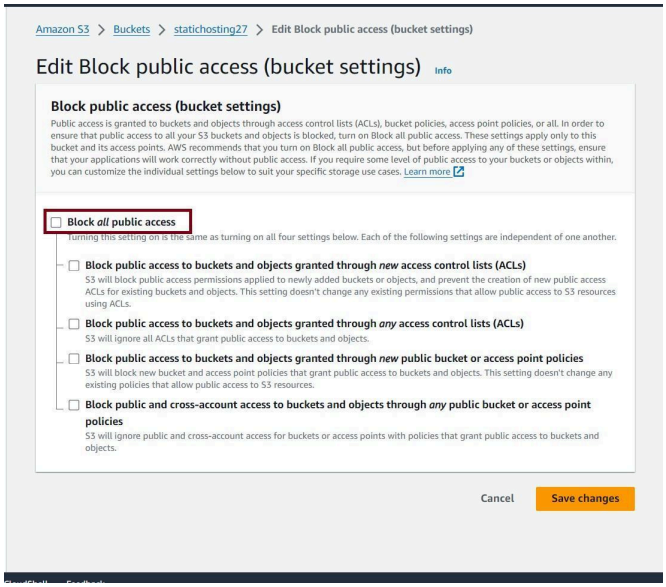
Step 9: This will take you to the Objects screen. Switch to Properties, scroll down to Static web hosting. There you would find the link (Bucket website endpoint) to your website.



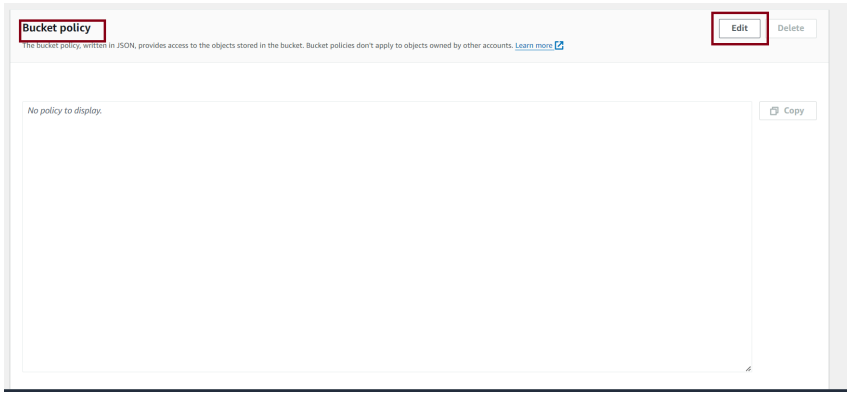
Step 10: Open the link. It will show a 403 forbidden error screen as the contents of the bucket are not available for the public users. To change this, go to Permissions tab, go to Block public access and click on edit



Step 11: Uncheck the Block all public access checkbox and click on save changes



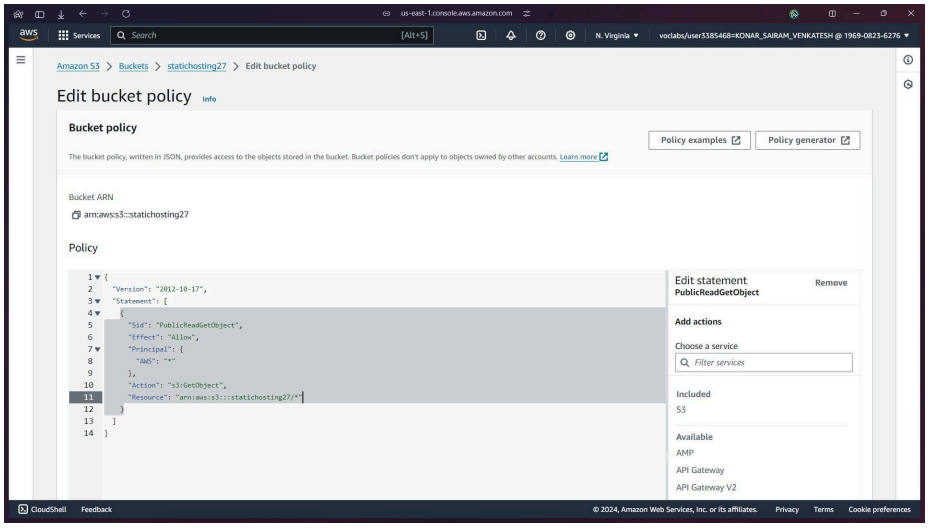
Step 12: Scroll down to bucket policy and click edit



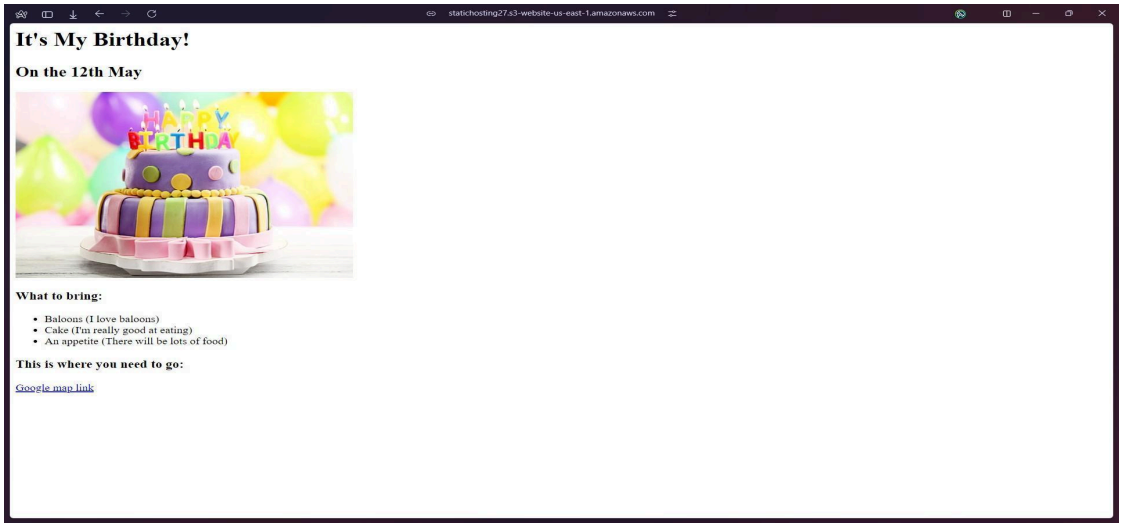
Step 13:

```
{
 "Version": "2012-10-17",
 "Statement": [
  {
   "Sid": "PublicReadGetObject",
   "Effect": "Allow",
   "Principal": {
    "AWS": "*"
   },
   "Action": "s3:GetObject",
   "Resource": "arn:aws:s3:::YOUR-BUCKET-NAME-HERE/*"
  }
 ]
}
```

Paste this code snippet in the policy textarea. Replace YOUR-BUCKET-NAME-HERE with the name you have given to your bucket. Save the changes.



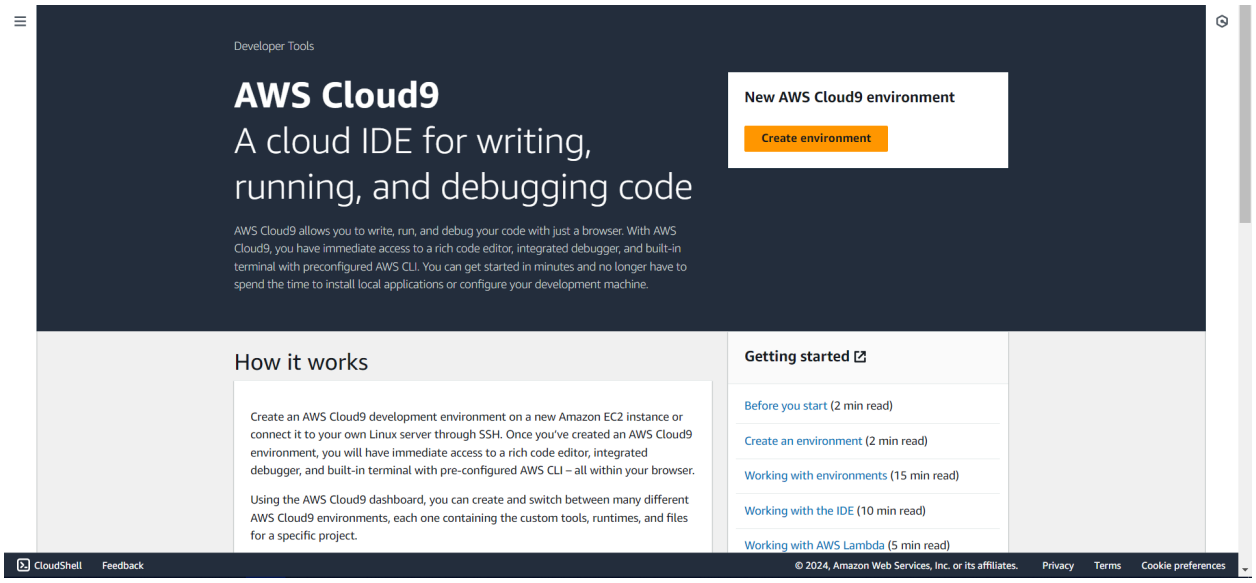
Step 14: Now reload the website. You can see your website



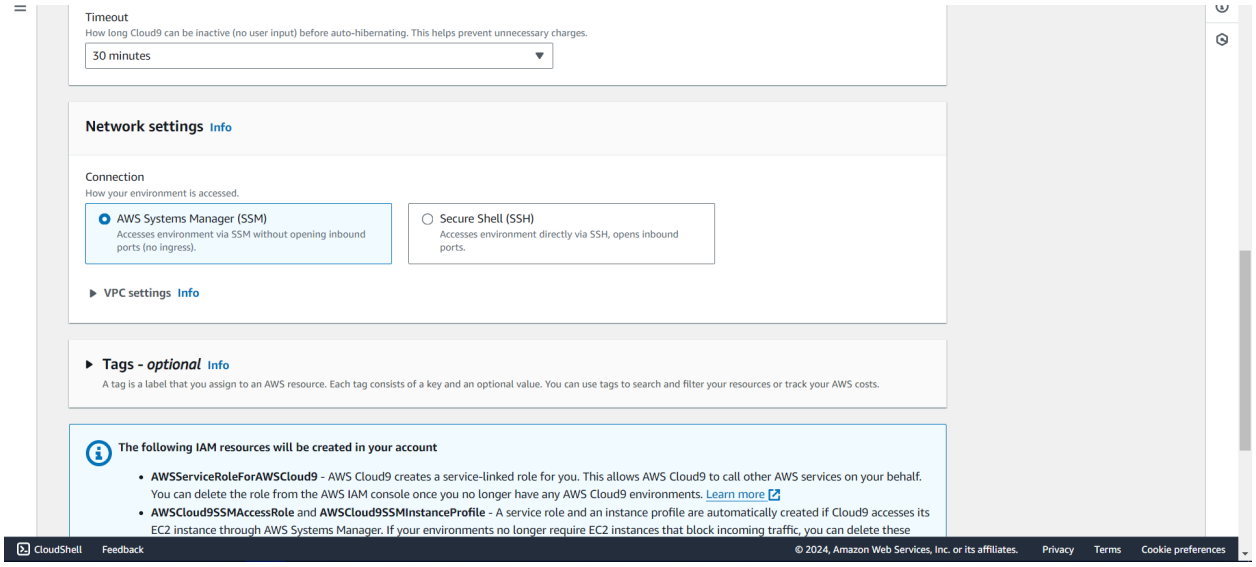
Name: Kshitij Hundre
Div: D15C
Roll No.18

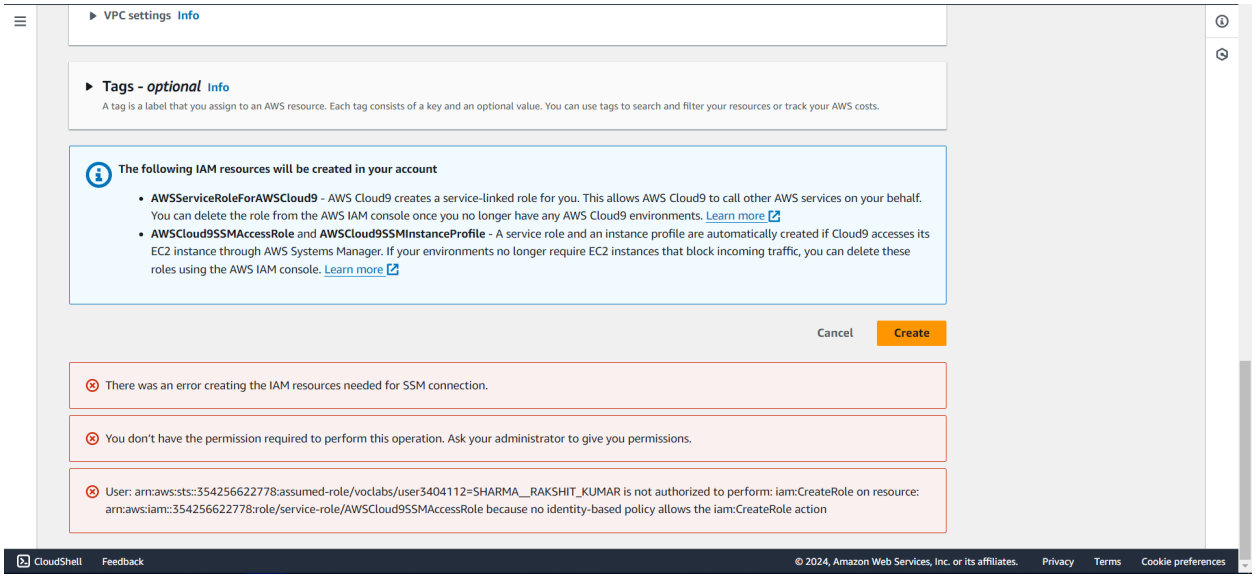
Experiment 1B: IAM and cloud9

1.Open the AWS account and search for Cloud9.

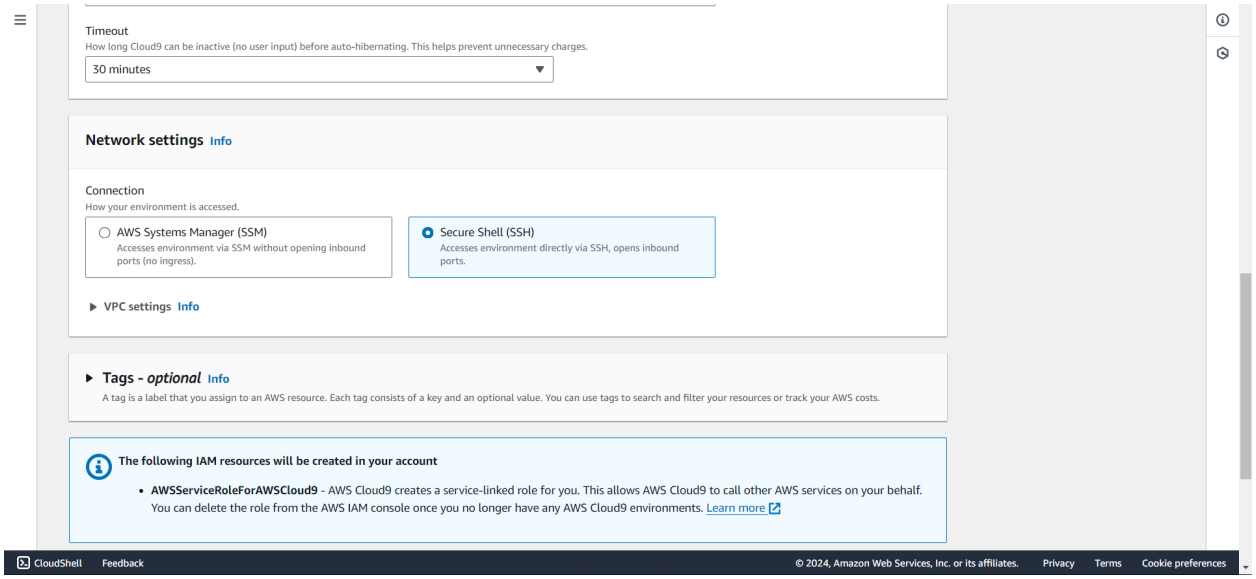


2. Enter the name and other required configuration for creating an environment. In network settings, using the AWS system manager gives an error while creating the environment

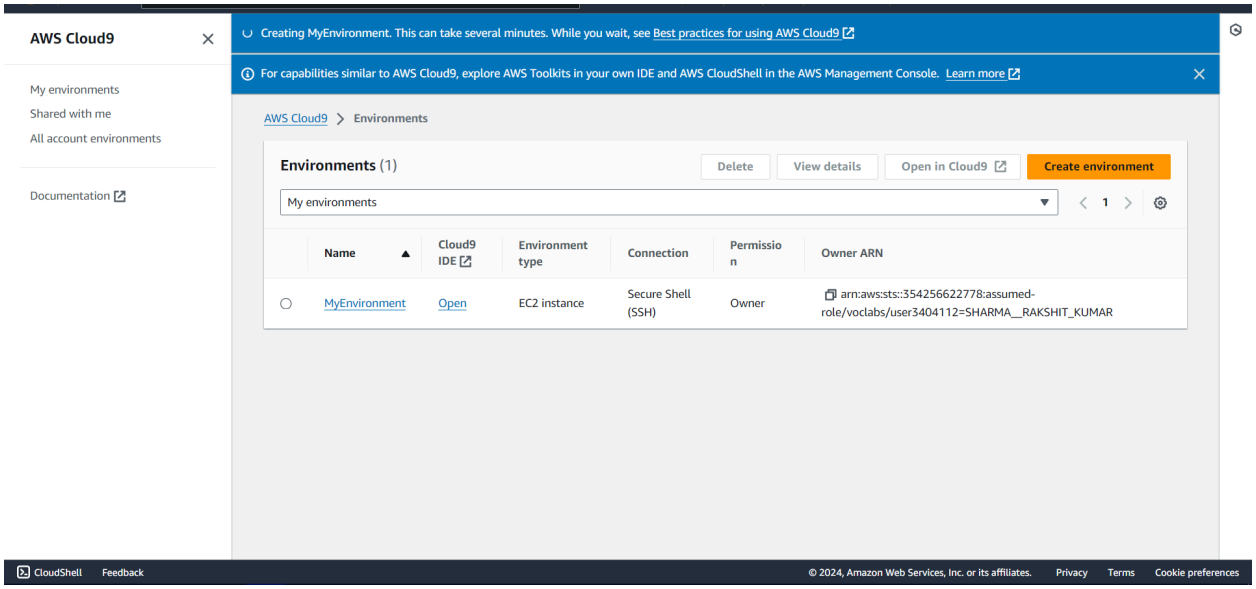




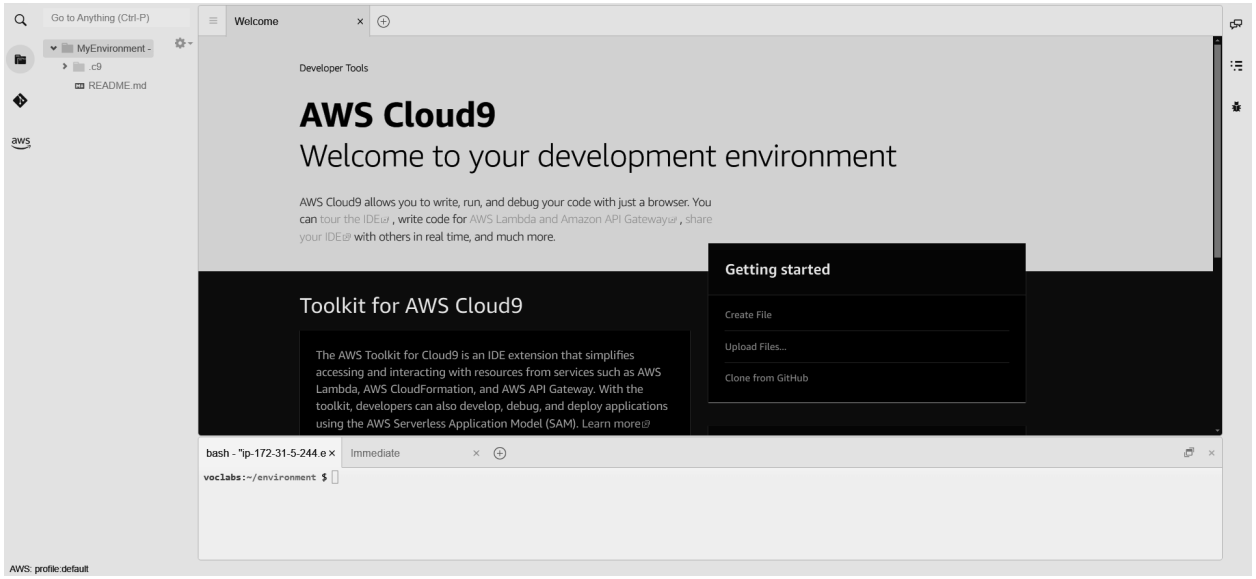
3. Use the Secure Shell option in Network settings.



4. Once the configuration is complete, click on create environment to create a Cloud9 environment.

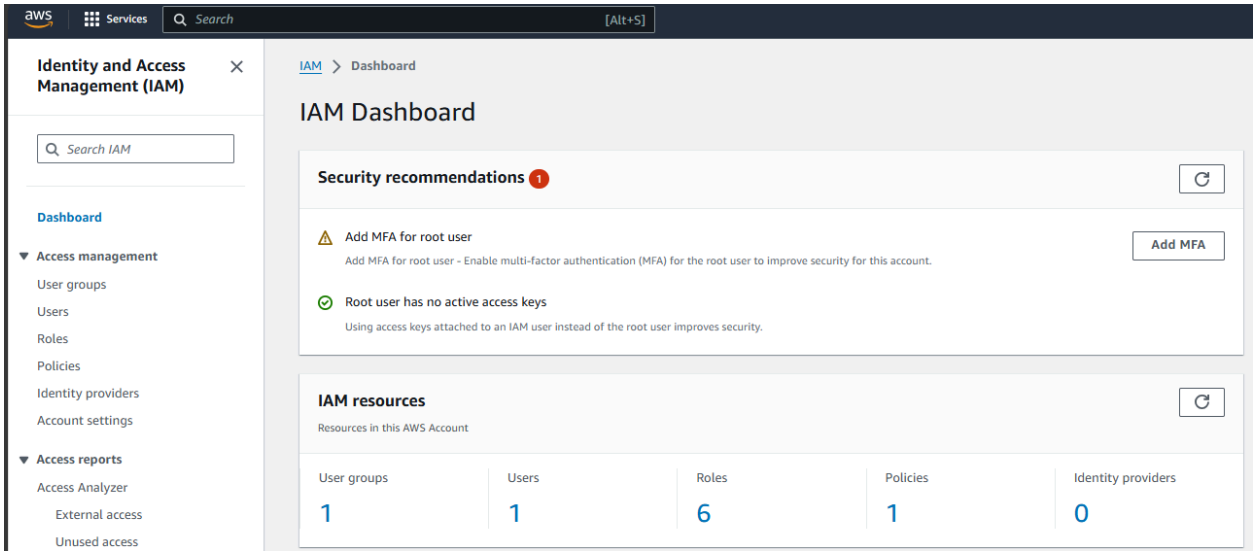


5. Cloud9 Environment.is opened when u click on the environment name

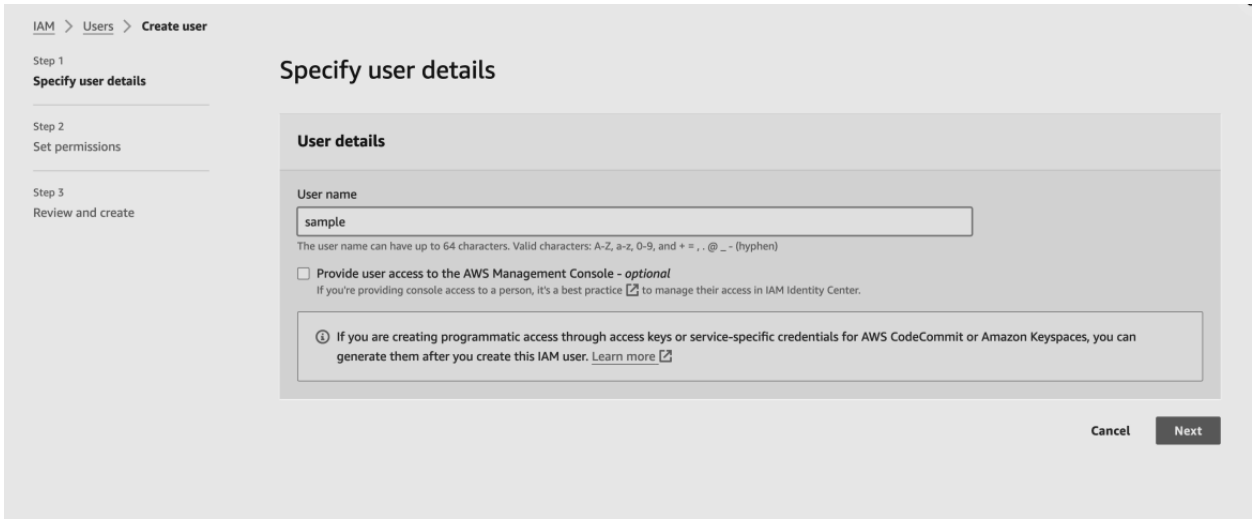


IAM user creation steps

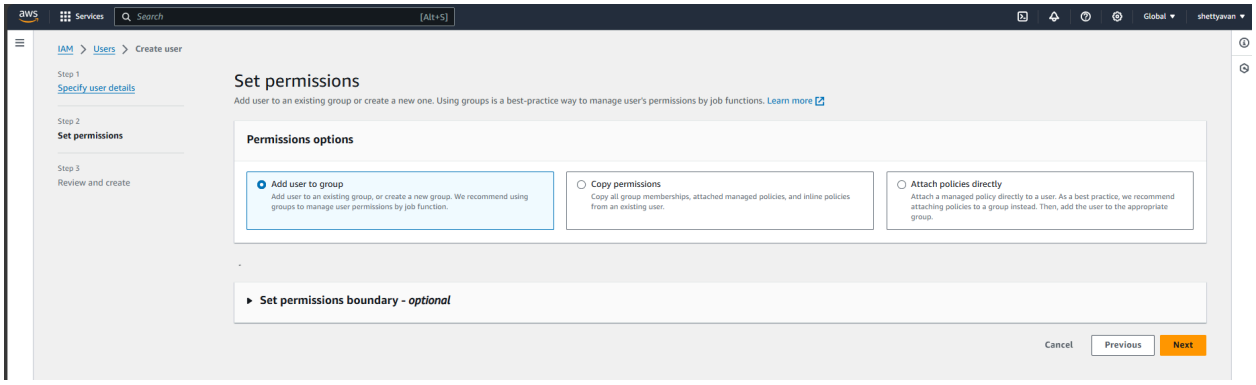
1. Open the aws account and search for IAM in service.



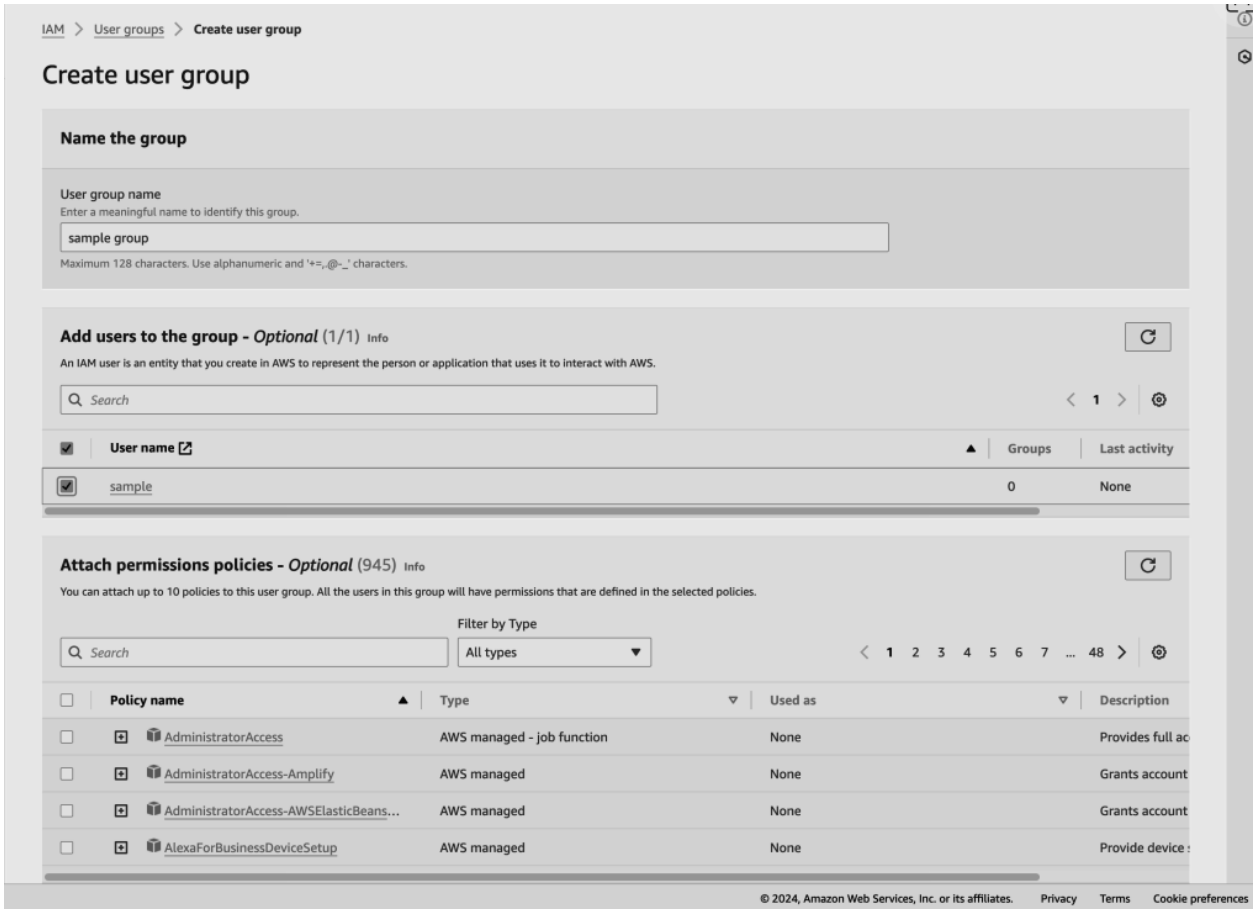
2. Select the users option from the left panel and click on create user button.Give the user name,



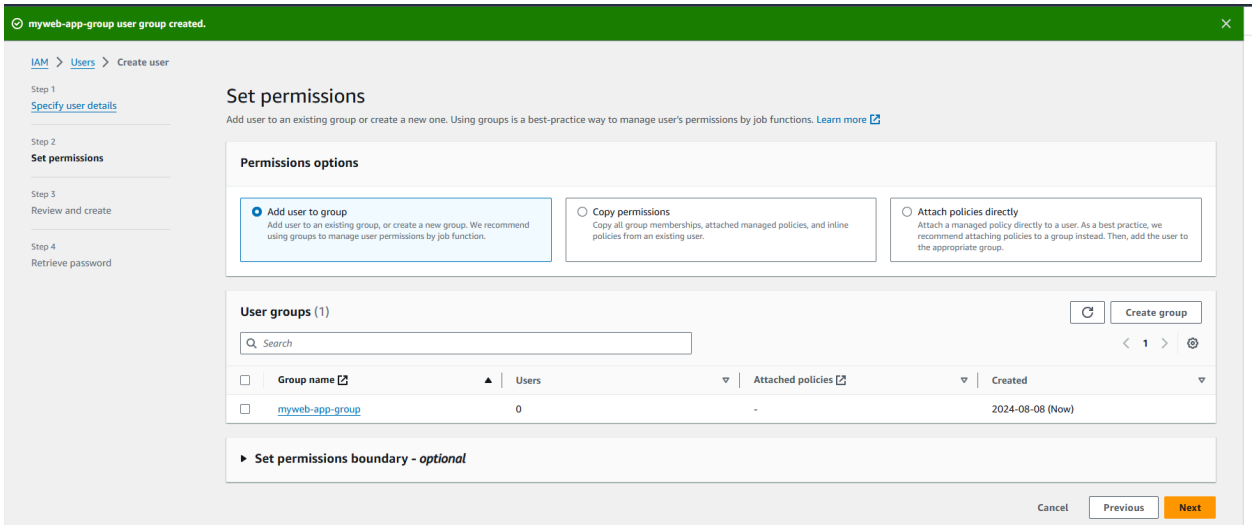
3. Click the add user option if you don't have an existing user group



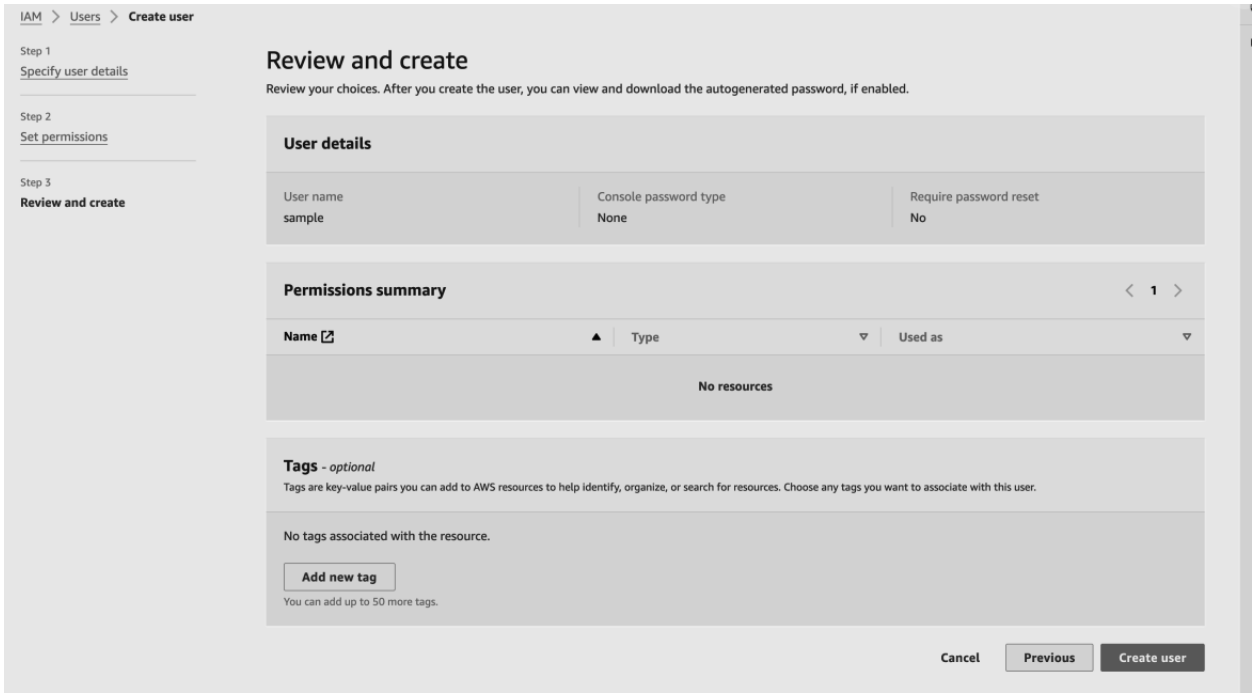
4. Give a name to your user group and check the policies if required any



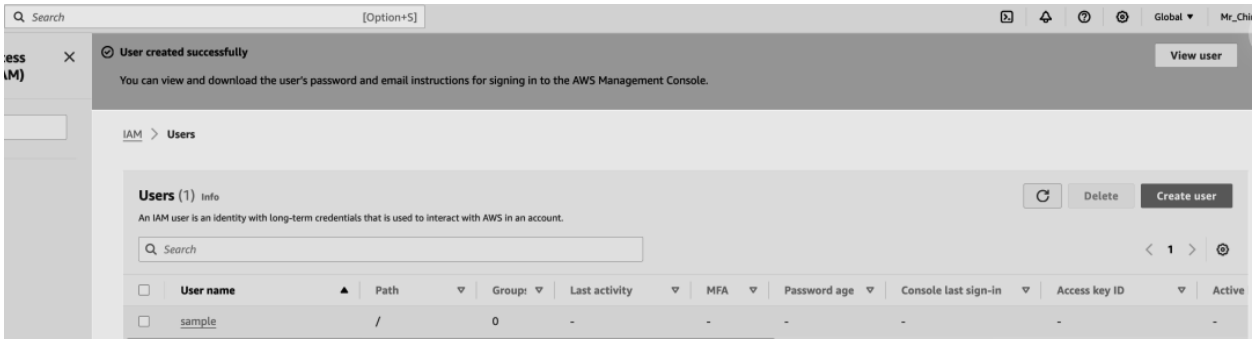
5. Once the user group is created select the name and click next to create your user



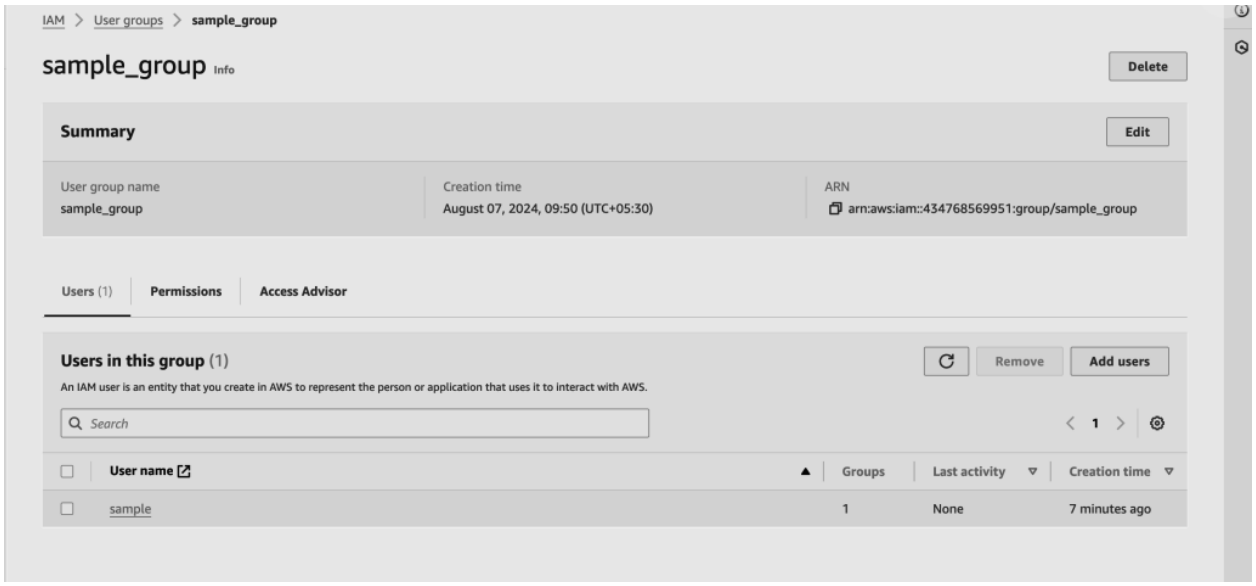
6. Review the configuration details and check if you have missed any steps and then click on ‘Create user’ button



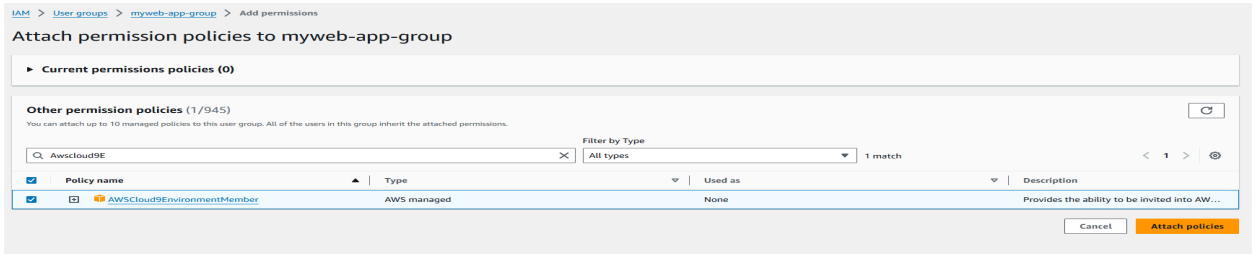
7. You will see the “user created successfully” message and incase you need then store your password by downloading the csv file



8. After creation of the user, go to the user groups tab and select the group in which the user has been added. Navigate to the permissions tab. Click on add permissions and attach policy.



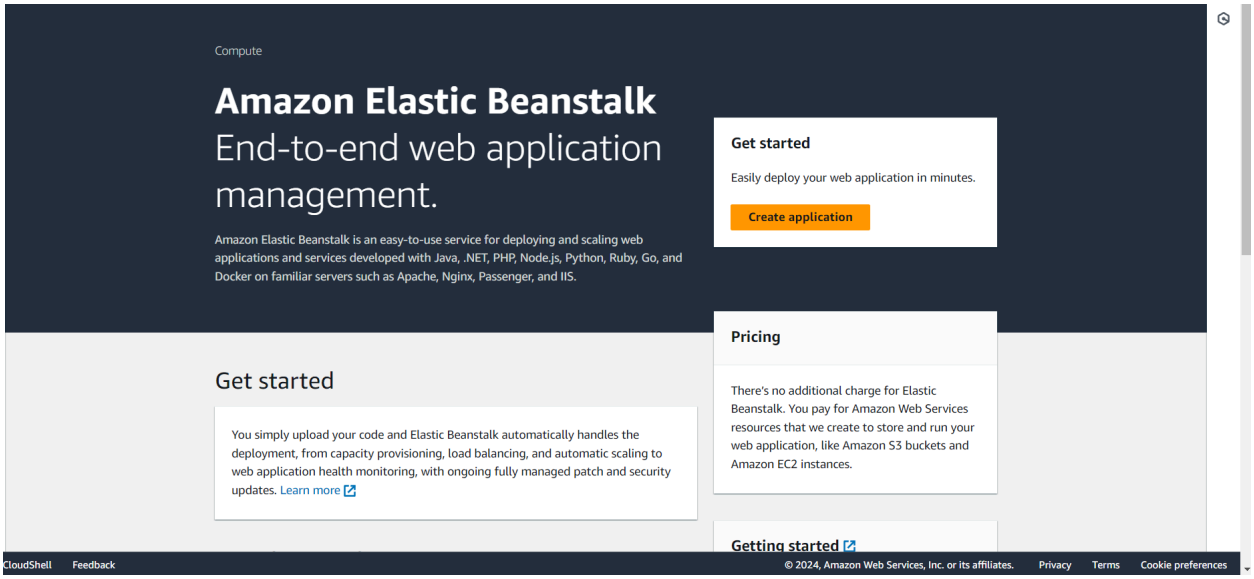
9. Search for the “AWSCloud9EnvironmentMember” policy and attach it.



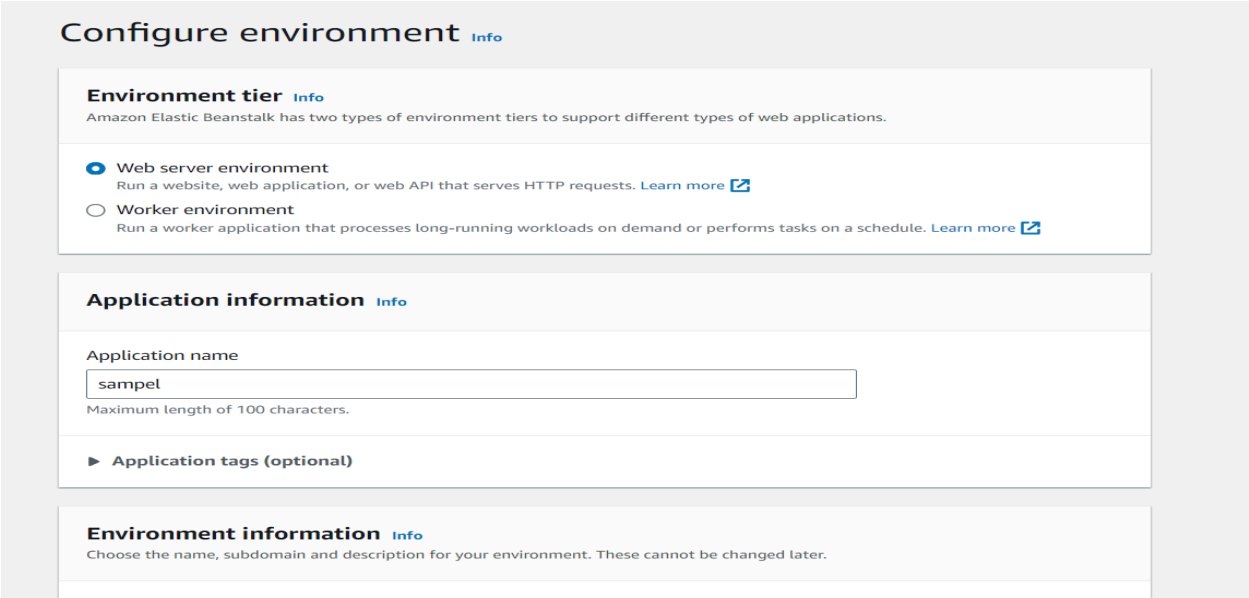
Name: Kshitij Hundre
Div: D10C
Roll No:18

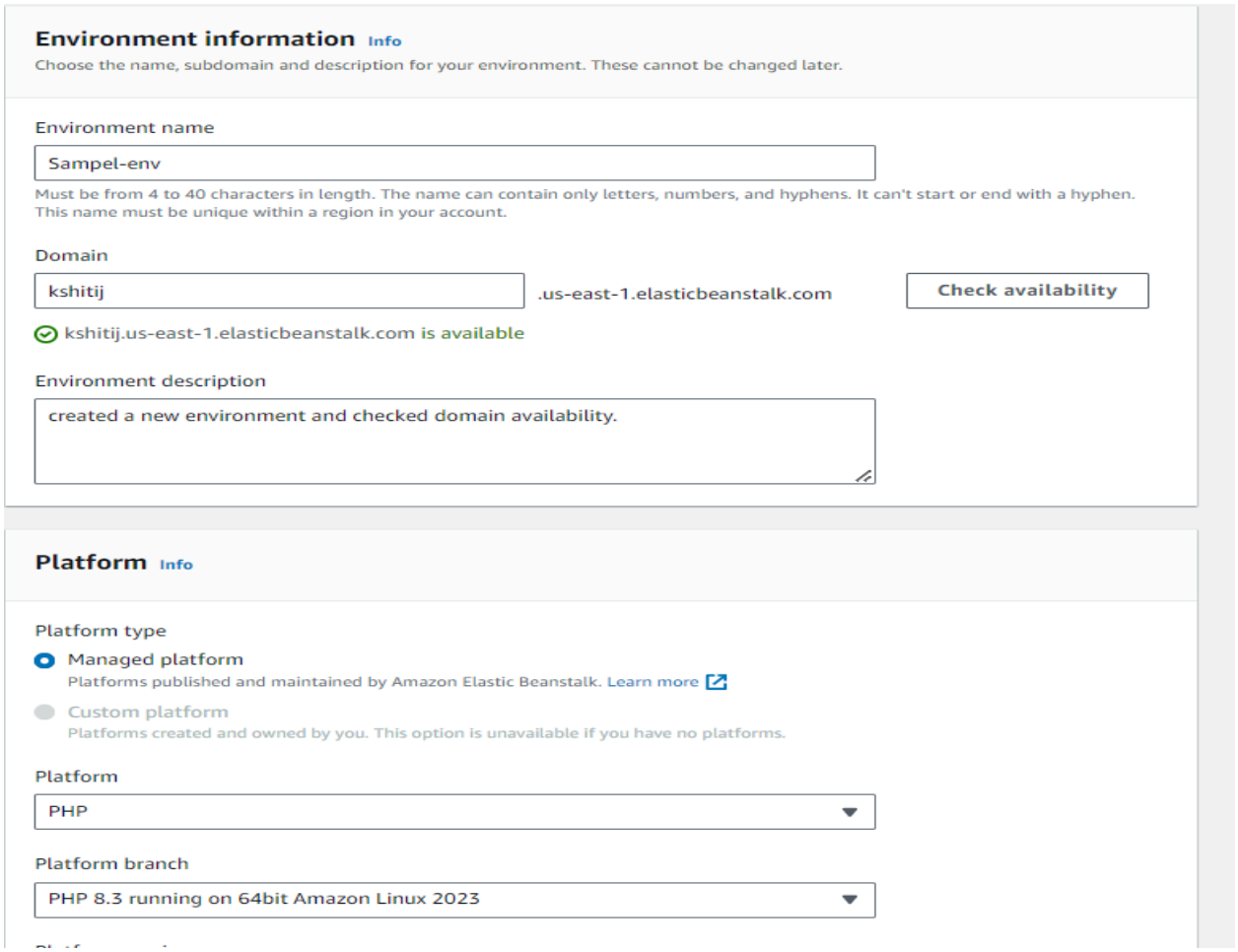
Practical No 2 : Elastic Beanstalk

1) Go to services and choose elastic Beanstalk.following page will appear.

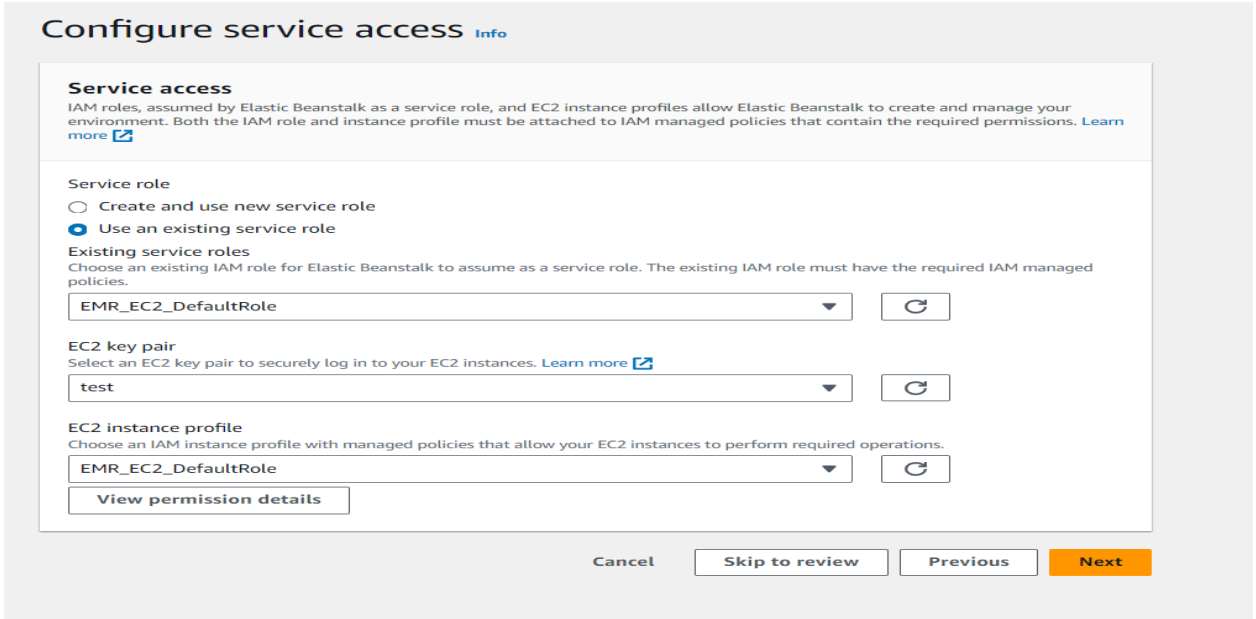


2) Configure the environment. Give the application name, check domain availability and choose PHP as platform.Then click next.

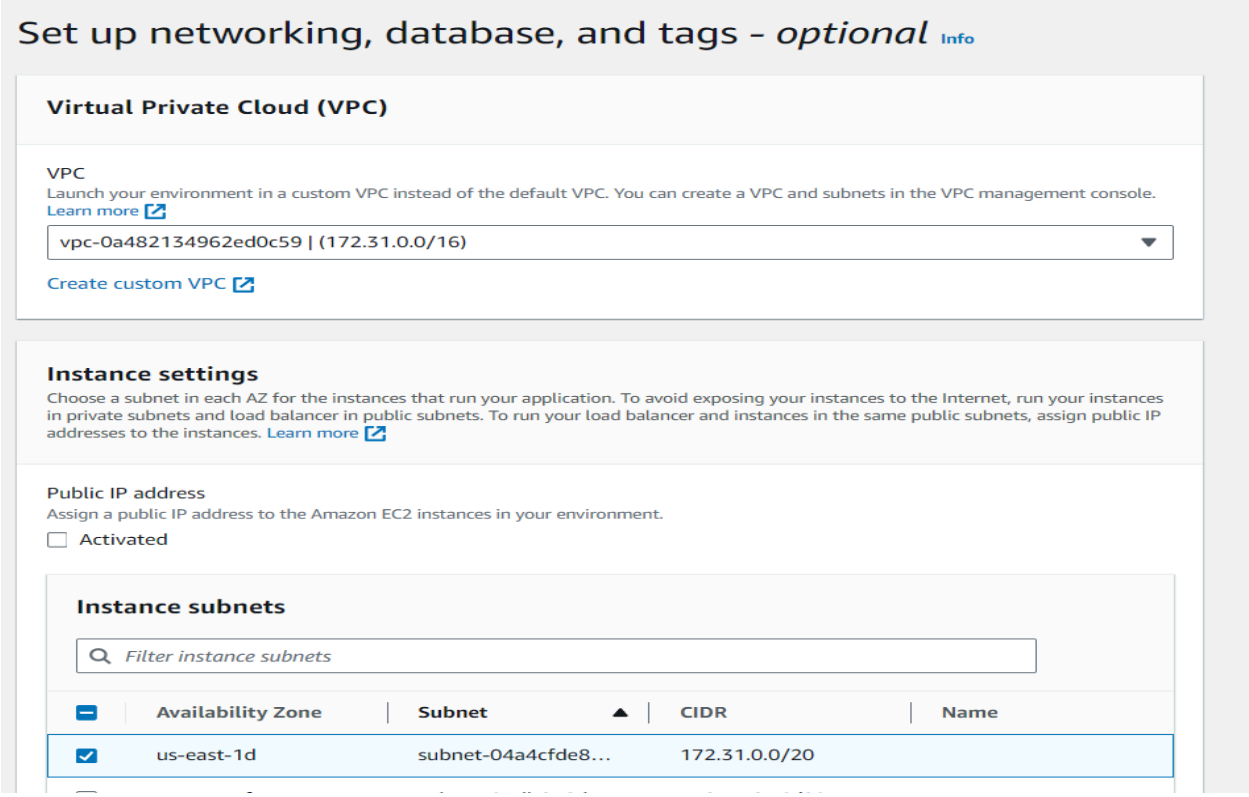




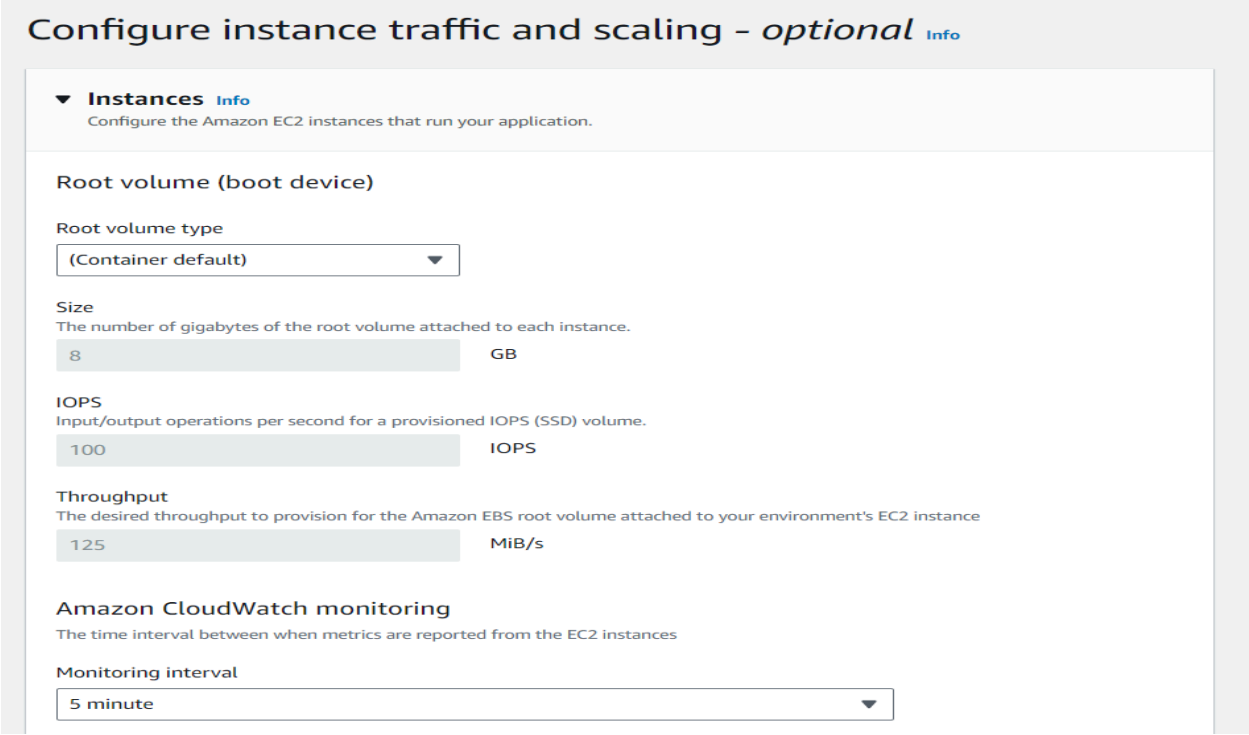
3) Configure the service access.

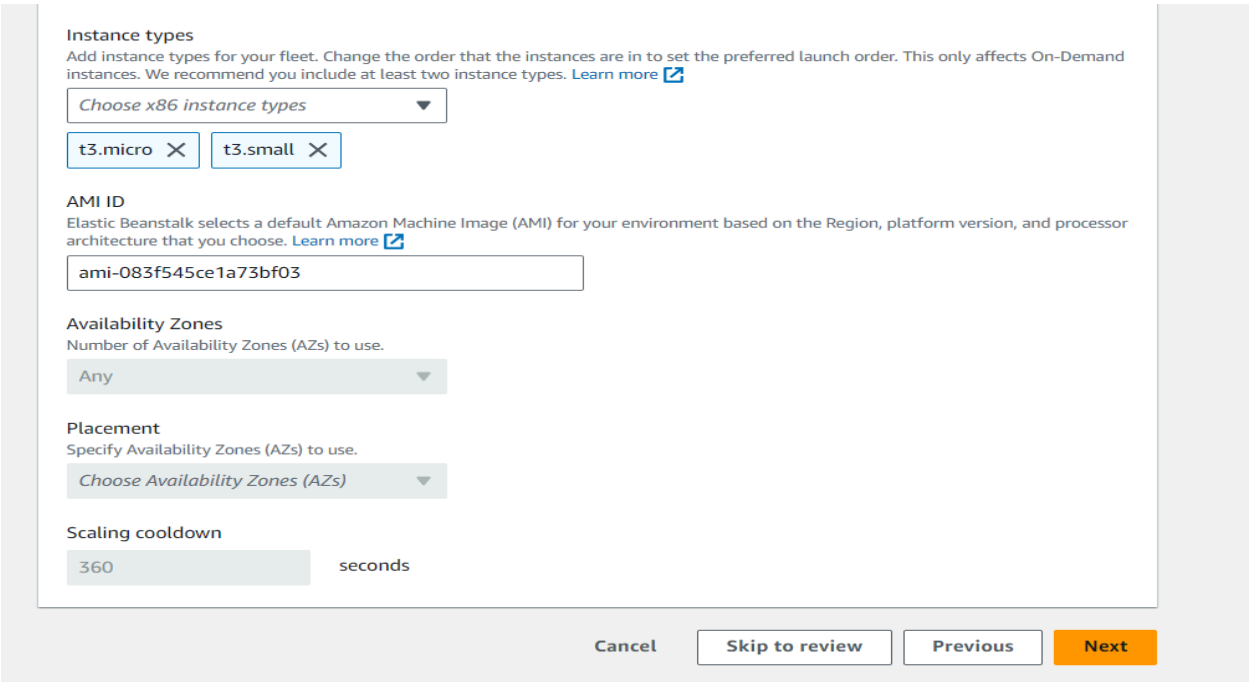


4) Choose one of the available VPC and instance subnet.Click next.

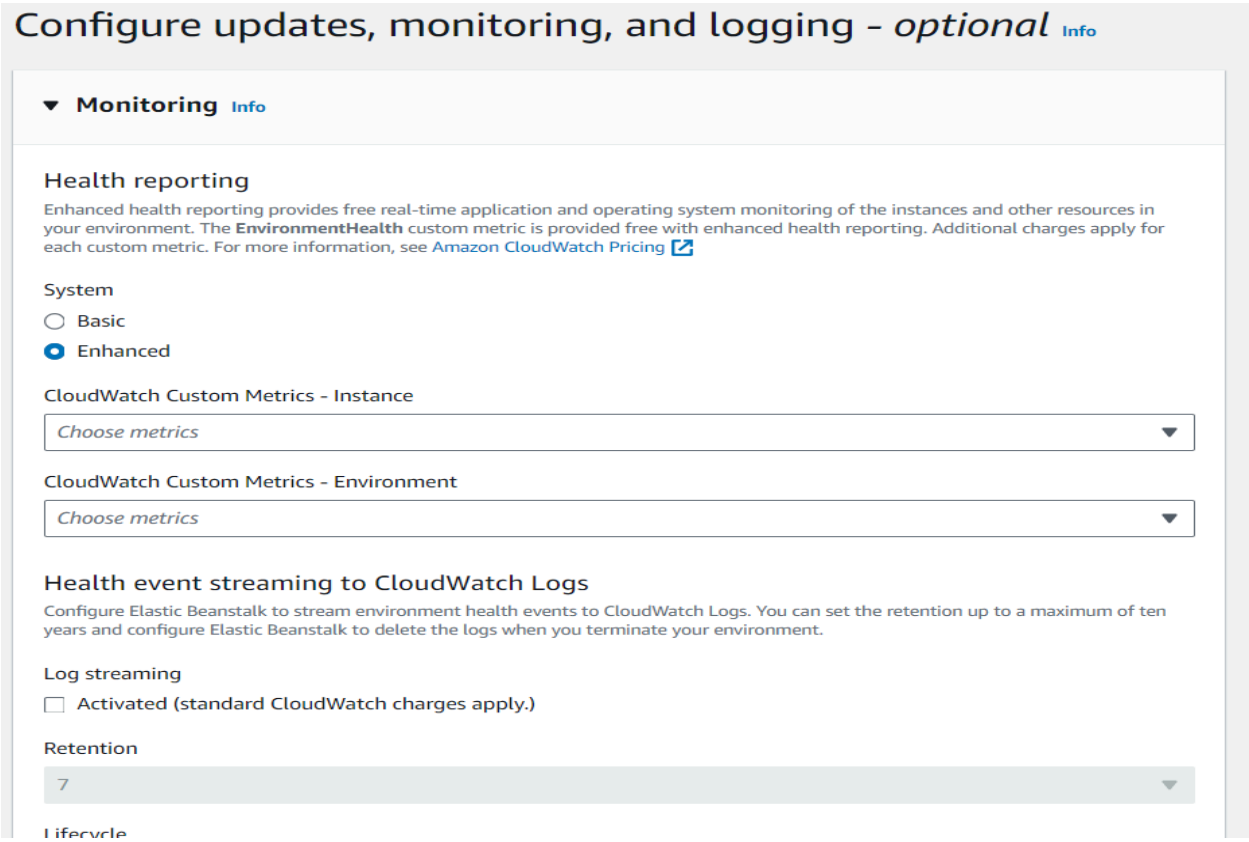


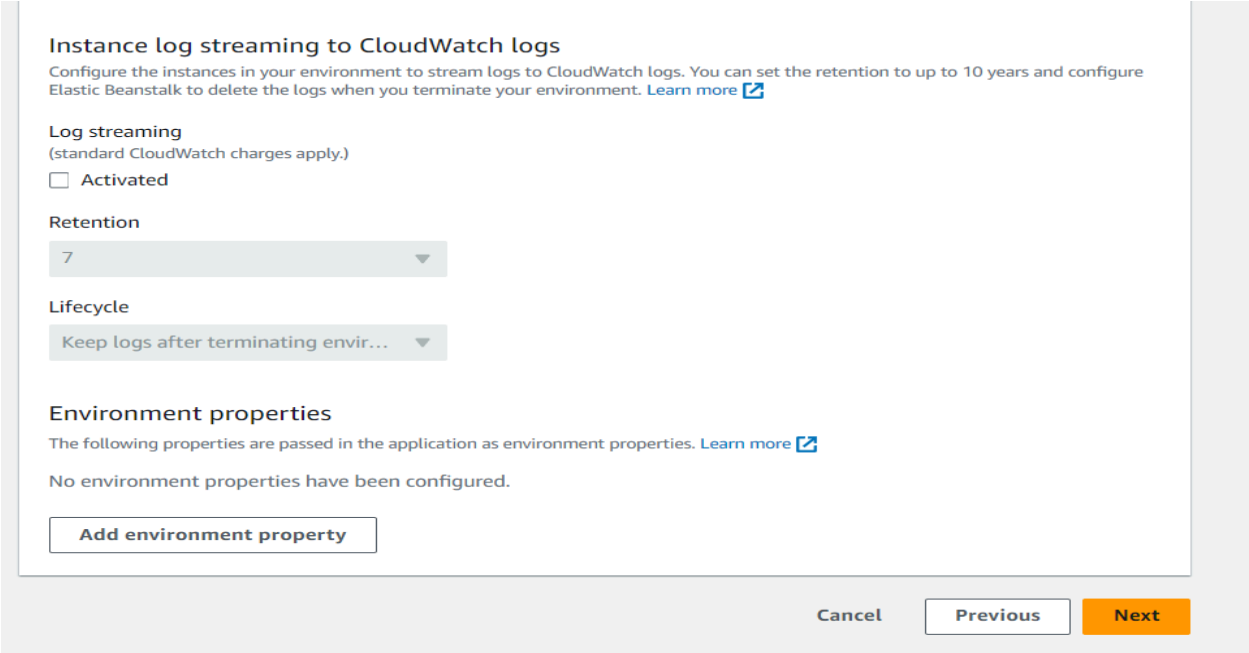
5) Configure instance traffic and scaling. Keep all the options as default.



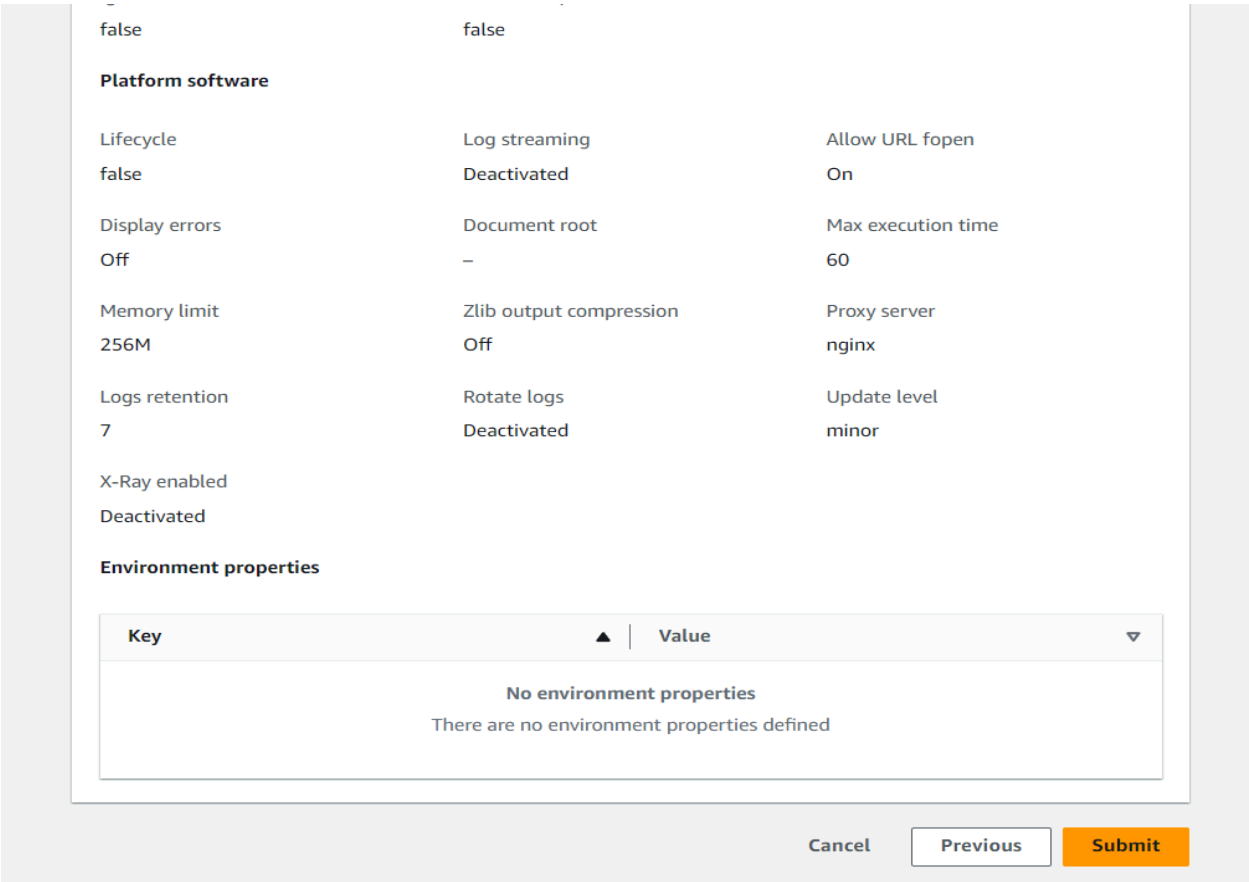


6) Configure updates, monitoring, and logging.Keep everything as default and click next.

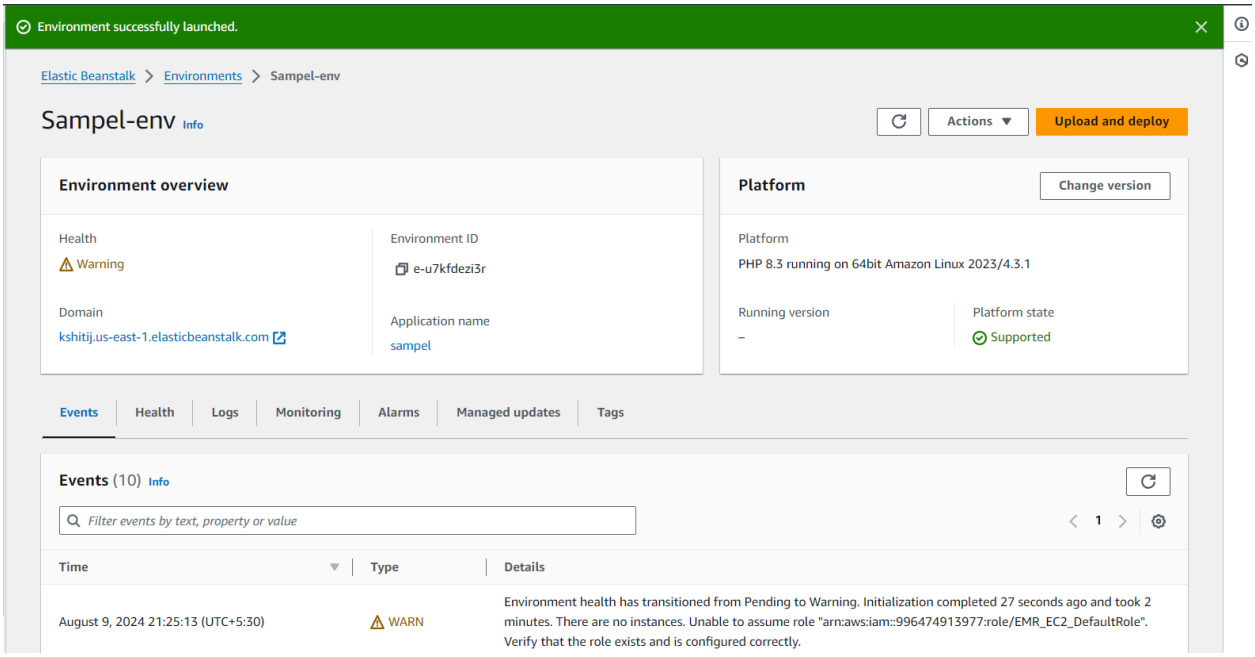




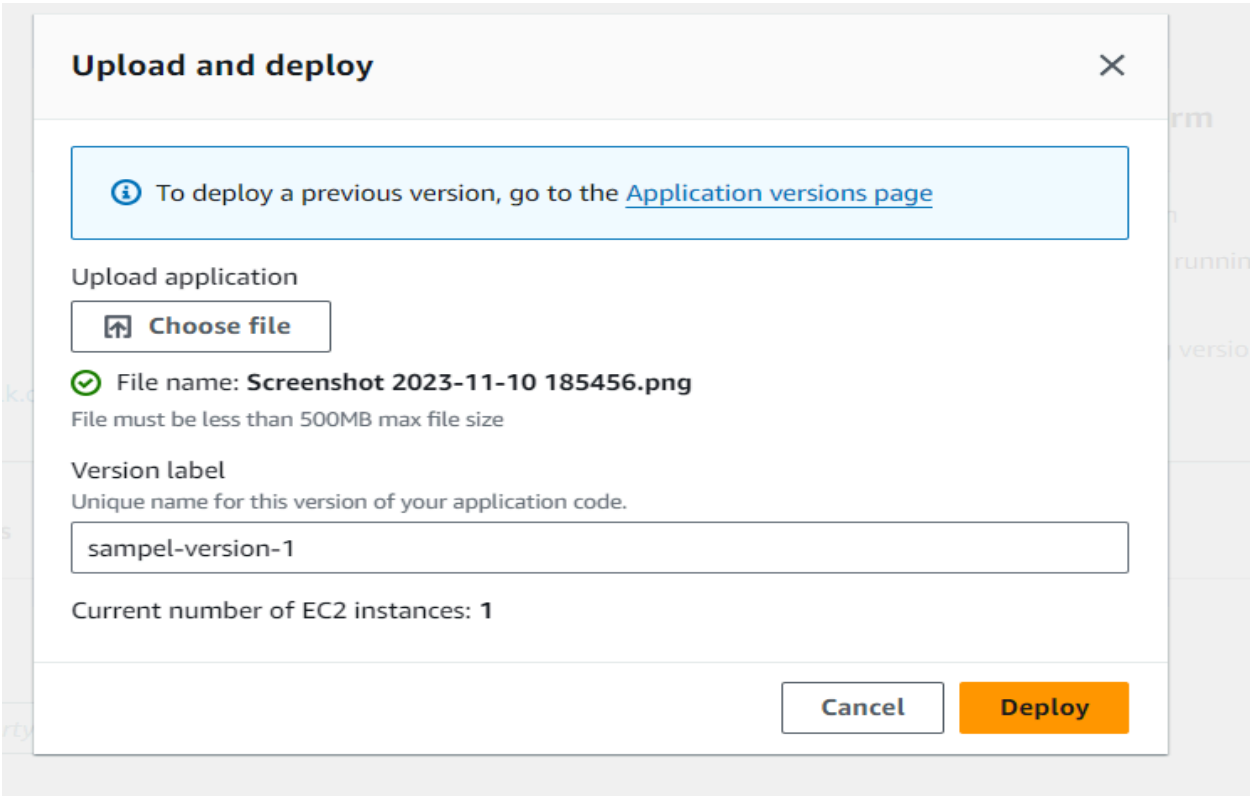
7) In the review section,click submit.

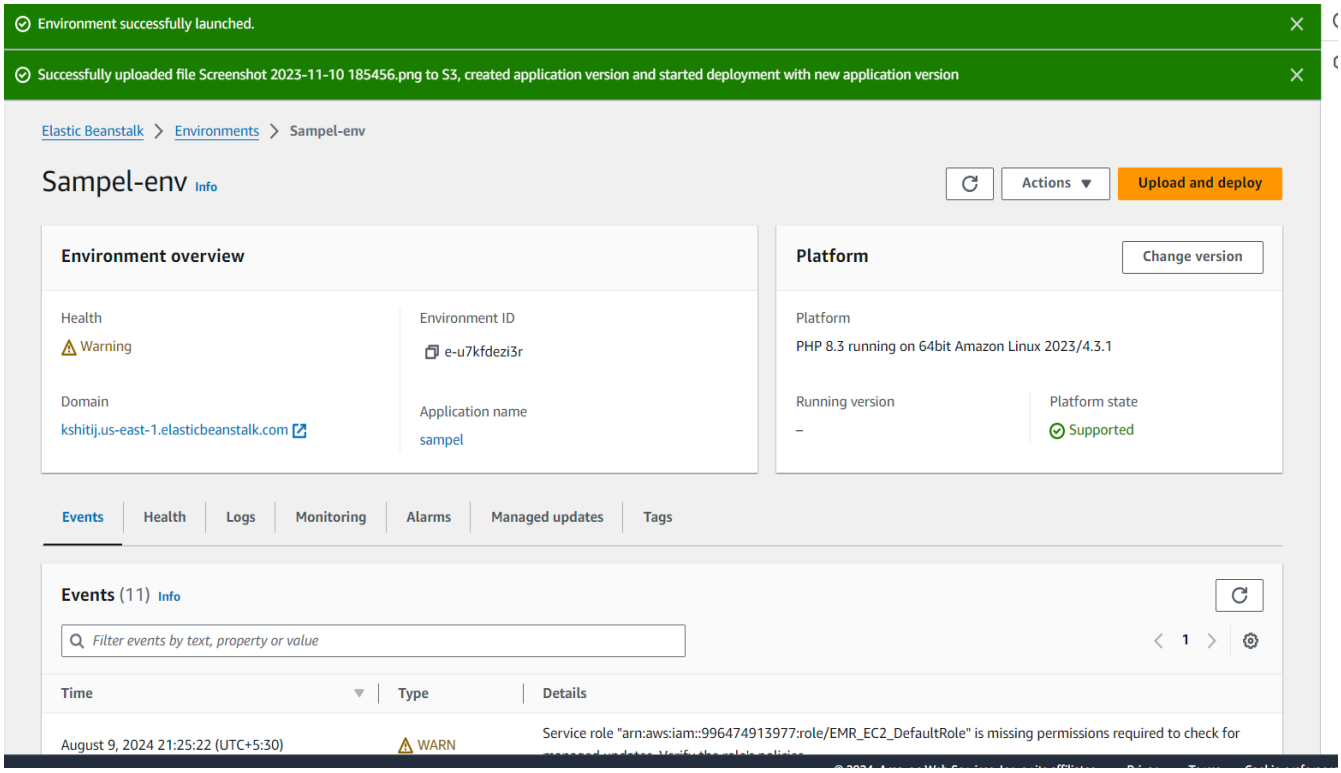


8) Environment has been created successfully.



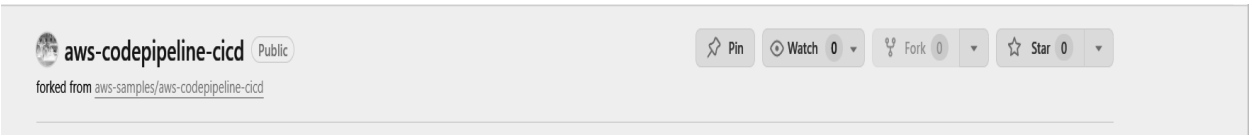
9) Deploy something on the recently created environment.



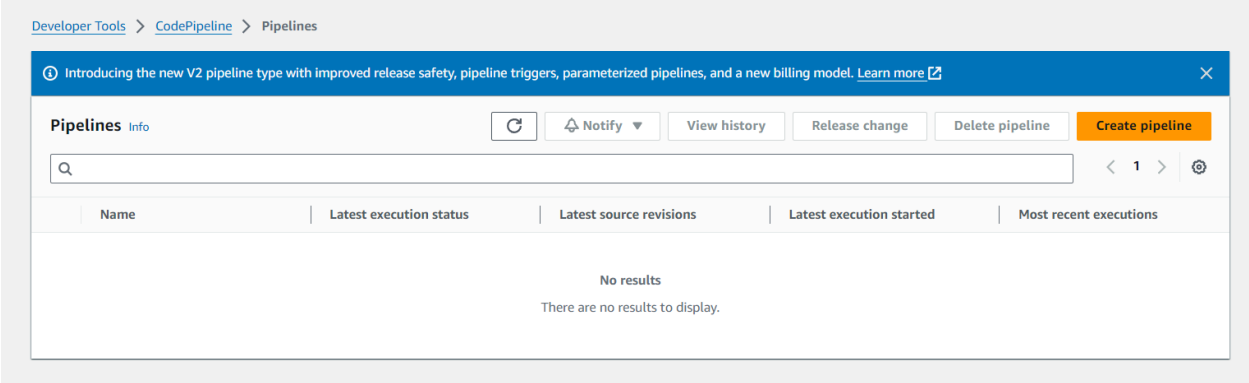

Environment successfully launched.
Successfully uploaded file Screenshot 2023-11-10 185456.png to S3, created application version and started deployment with new application version
Elastic Beanstalk > Environments > Sampel-env
Sampel-env Info
Actions
Upload and deploy
Environment overview
Health
Warning
Environment ID
e-u7kfdezi3r
Domain
kshitij.us-east-1.elasticbeanstalk.com
Application name
sampel
Platform
Change version
Platform
PHP 8.3 running on 64bit Amazon Linux 2023/4.3.1
Running version
–
Platform state
Supported
Events
Health
Logs
Monitoring
Alarms
Managed updates
Tags
Events (11) Info
Filter events by text, property or value
1
Time
Type
Details
August 9, 2024 21:25:22 (UTC+5:30)
WARN
Service role "arn:aws:iam::996474913977:role/EMR_EC2_DefaultRole" is missing permissions required to check for

PIpeline Creation:

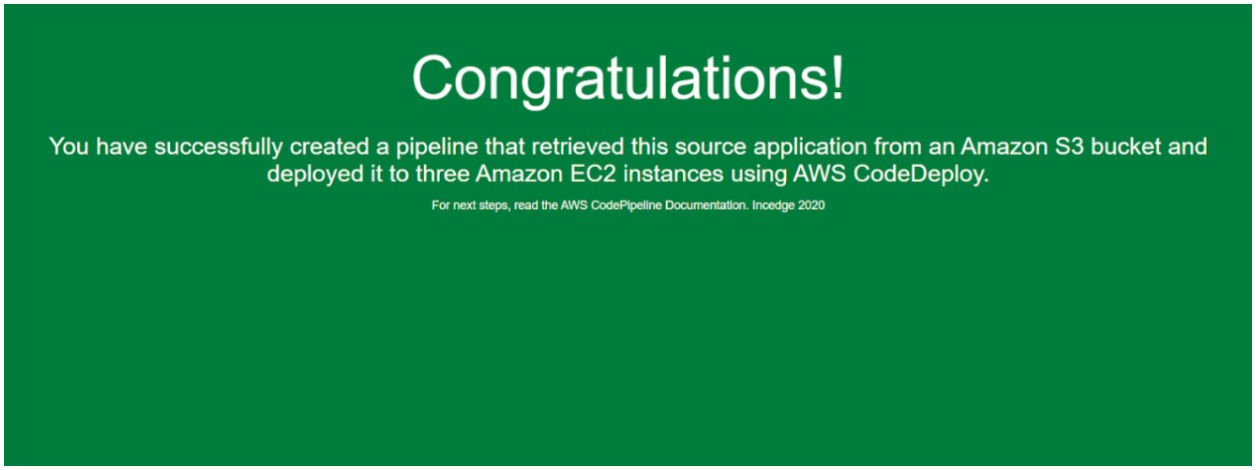
1) Fork a github repository. This forked repository will act as source for your code pipeline.



2) Go to developer tools and select CodePipeline and create a new pipeline



3) Create a pipeline:



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.

firstpipeline

No more than 100 characters

Pipeline type

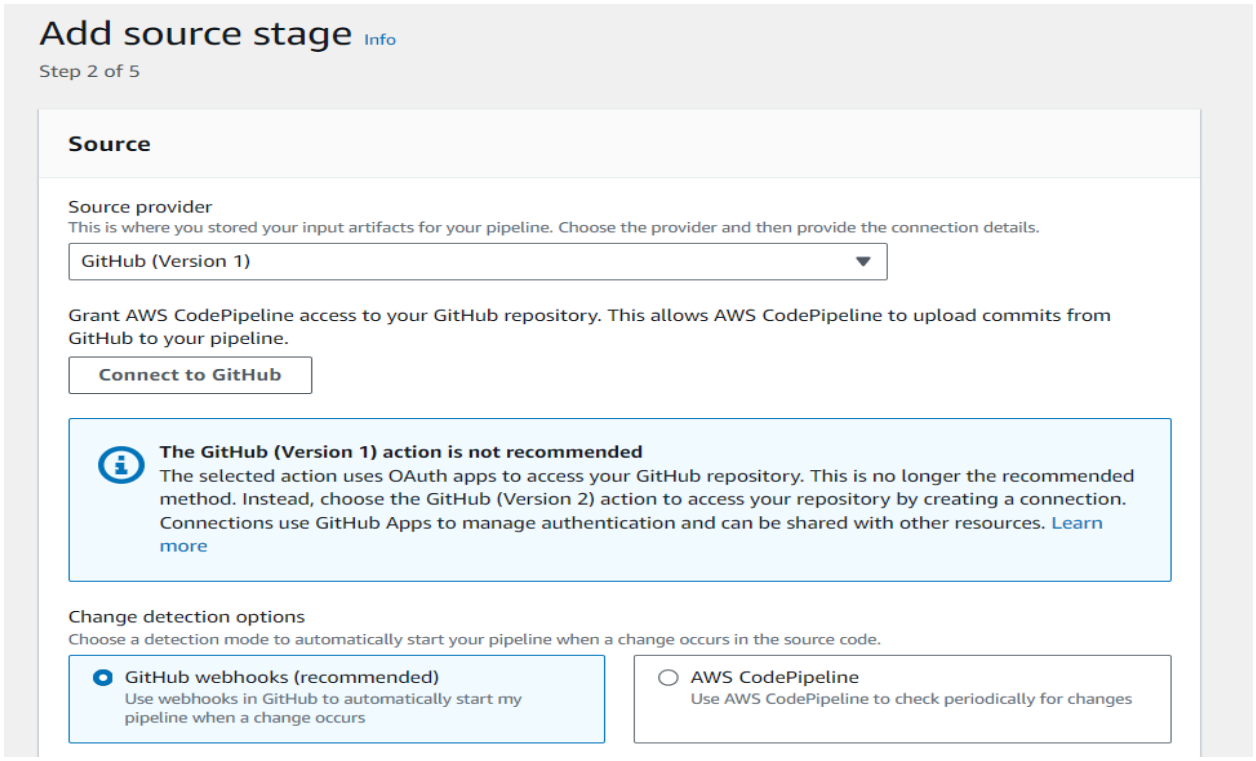
You can no longer create V1 pipelines through the console. We recommend you use the V2 pipeline type with improved release safety, pipeline triggers, parameterized pipelines, and a new billing model.

Execution mode

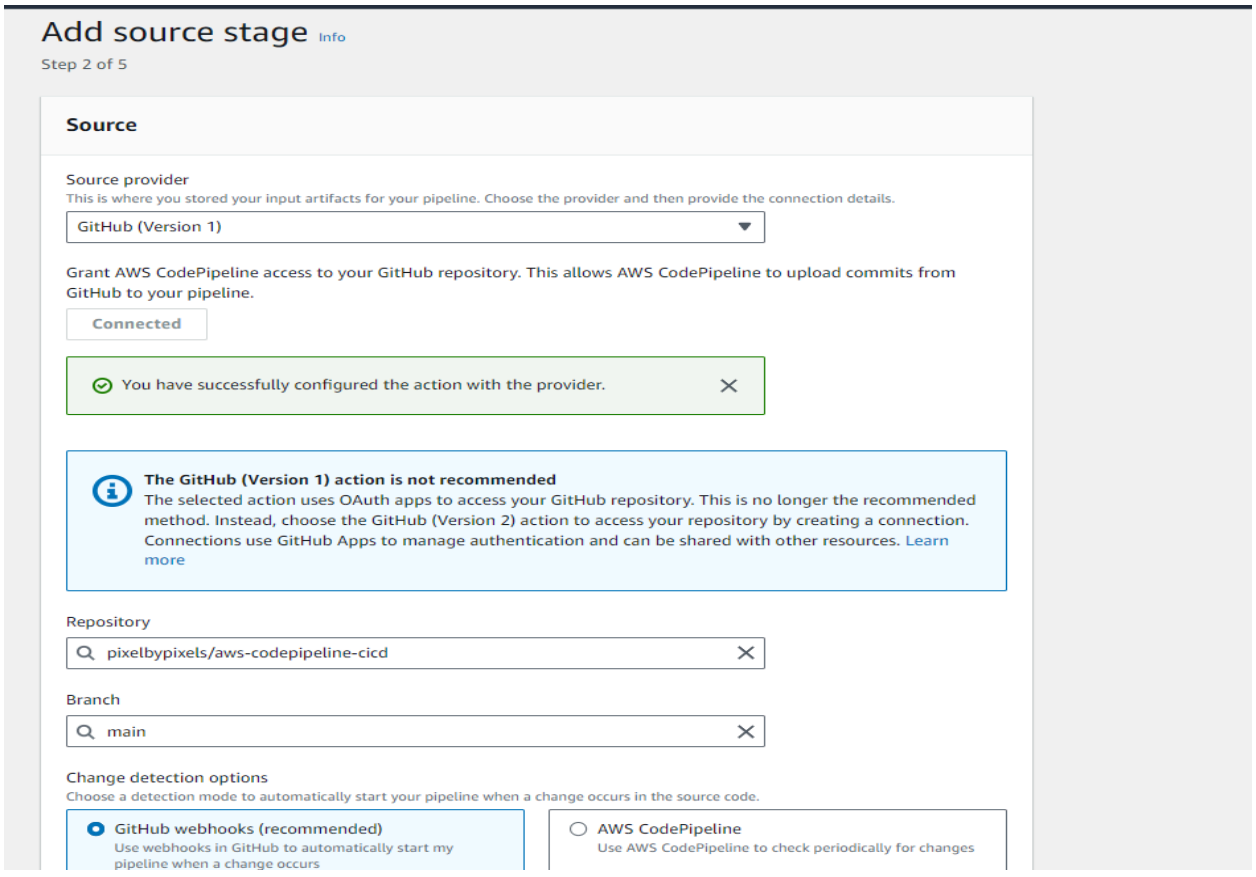
Choose the execution mode for your pipeline. This determines how the pipeline is run.

- ○ 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)

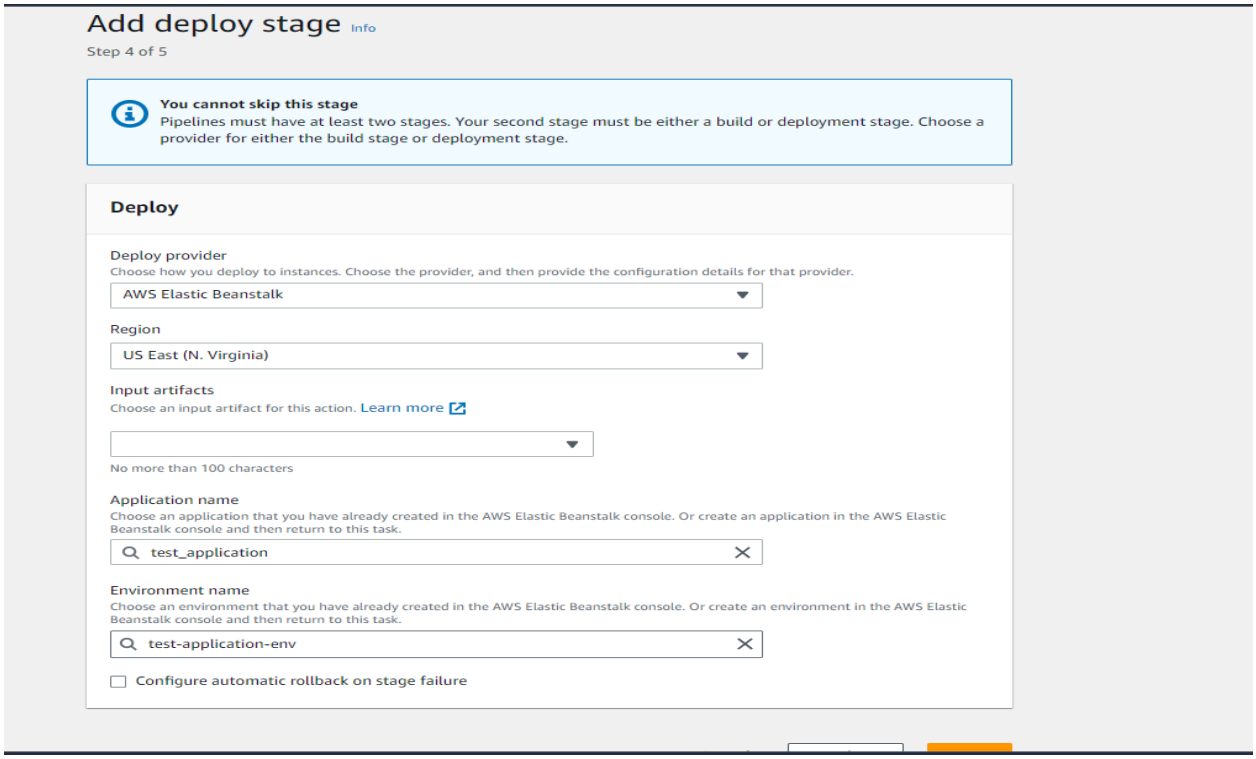
4)


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.
Connect to GitHub
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
Change detection options
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

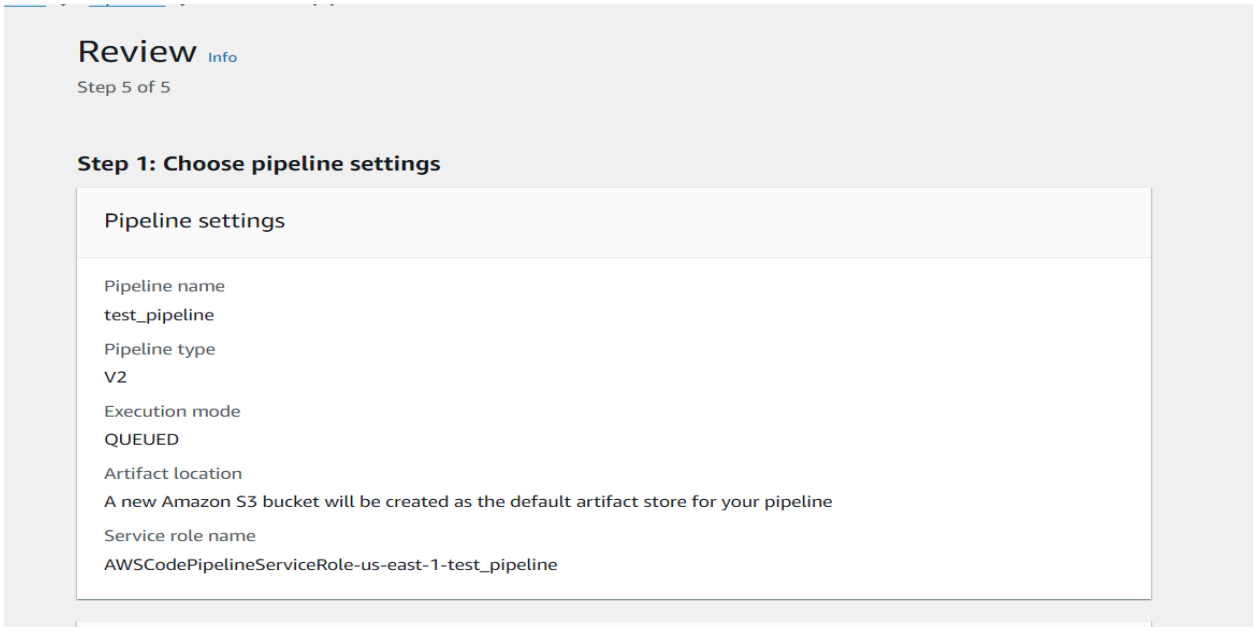
s


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
pixelbypixels/aws-codepipeline-cicd
Branch
main
Change detection options
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

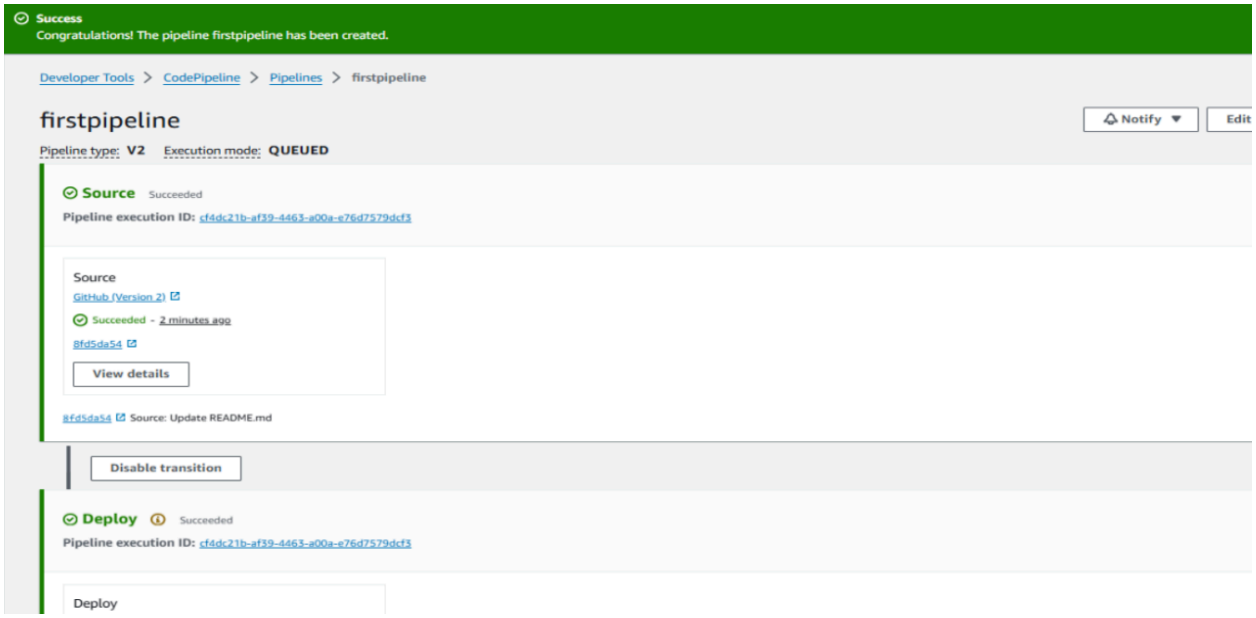
5) Go to the deploy stage and ensure the following settings.



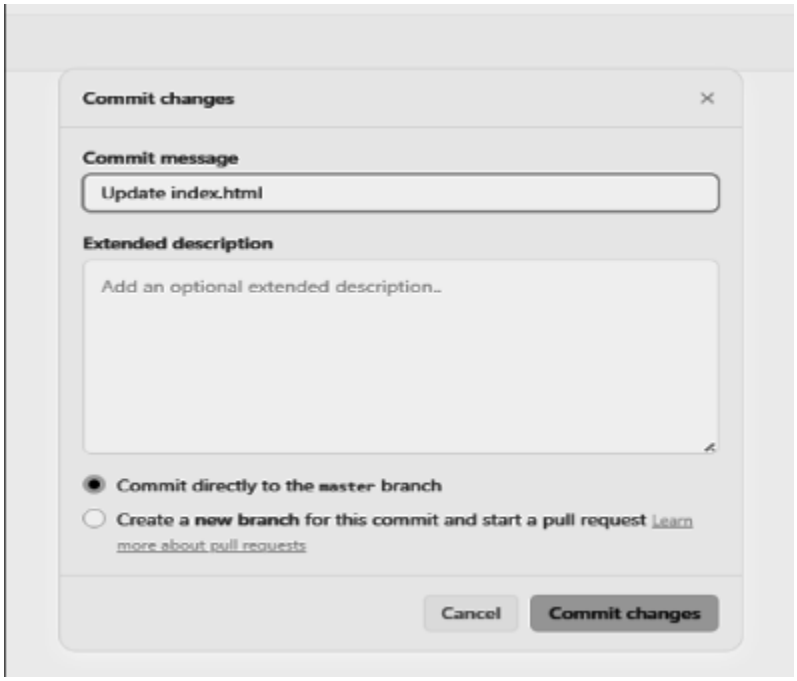
6) review the pipeline settings.



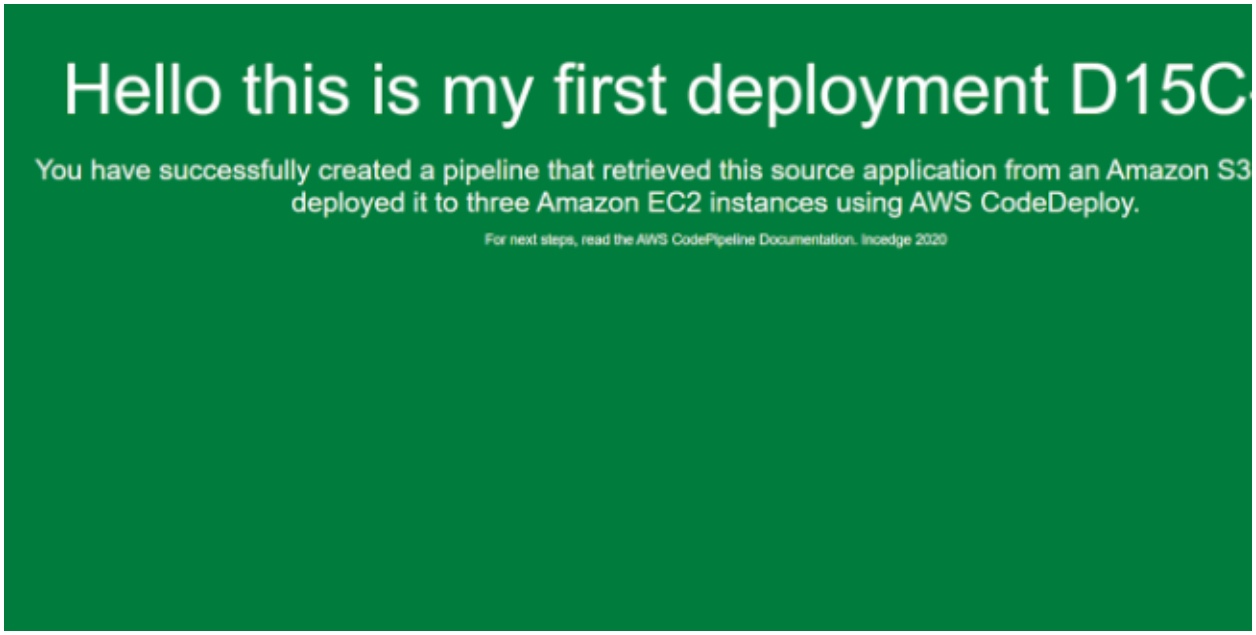
7) Then go ahead and check the URL provided in the EBS environment.



8) Go to the repository and make the changes in the index.html file and commit them

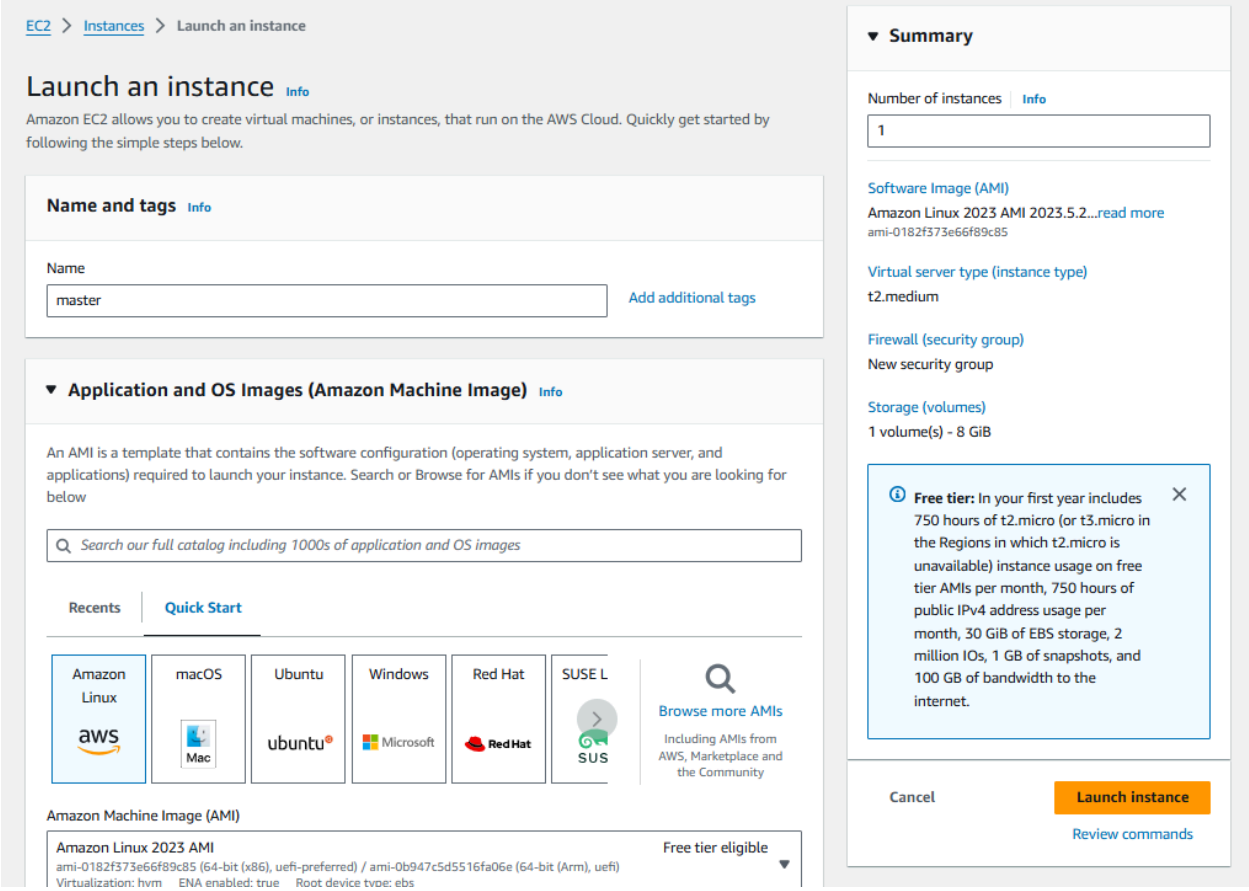


9)The changes that are committed can be noticed in the source panel in real time and to view the changes check the url (refresh it) and you can view the changes once the deployment section shows success.

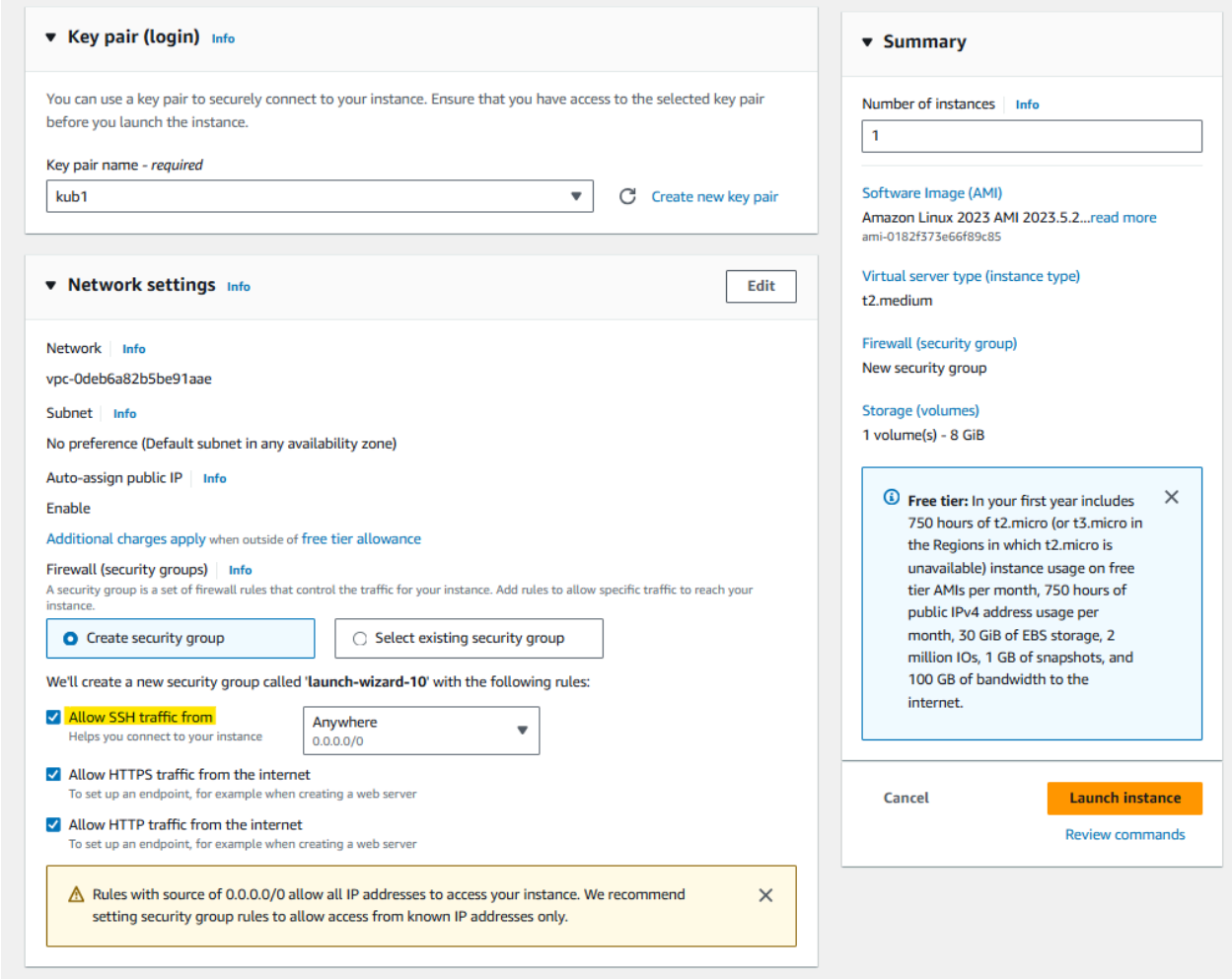


Aim: To understand the Kubernetes Cluster Architecture, install and Spin Up a Kubernetes Cluster on Linux Machines/Cloud

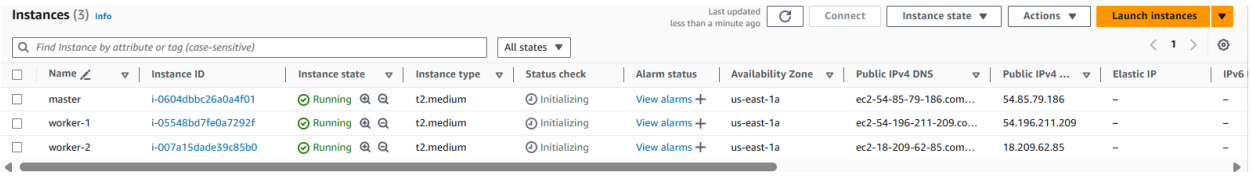
1. Create 3 EC-2 instances with all running on Amazon Linux as OS with inbound SSH allowed and the proper key



To efficient run kubernetes cluster select instance type of at least t2.medium as kubernetes recommends at least 2 vCPU to run smoothly



In this way create 3 instances namely master, worker-1 and worker-2



2. SSH into all 3 machines each in separate terminal
   a. You can do it through the aws console directly

```
C:\Users\Avan>ssh -i "C:\Users\Avan\Downloads\kub1.pem" ec2-user@3.85.237.93
The authenticity of host '3.85.237.93 (3.85.237.93)' can't be established.
ED25519 key fingerprint is SHA256:Nz2iC26abFyhATf/8i4F0IgWmDoxTXBbzY9/NkMwYyM.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '3.85.237.93' (ED25519) to the list of known hosts.
   ,     #_
   ~\_  ####_        Amazon Linux 2023
  ~~  \_#####\
  ~~     \###|
  ~~       \#/ ___   https://aws.amazon.com/linux/amazon-linux-2023
   ~~       V~' '->
    ~~~         /
      ~~._.   _/
         _/ _/
       _/m/'
Last login: Thu Sep 12 13:11:49 2024 from 18.206.107.29
```

Or

b. Locate your key from the Downloads folder and open it in cmd and paste this command

ssh -i <-your-key->.pem ec2-user<ip-address of instance>

```
C:\Users\Avan\Downloads>ssh -i C:\Users\Avan\Downloads\kub1.pem ec2-user@54.85.79.186
The authenticity of host '54.85.79.186 (54.85.79.186)' can't be established.
ED25519 key fingerprint is SHA256:ffz946cqlxbNvBsPqtNcxLlfXmhW8VJhyLD4n4jStto.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '54.85.79.186' (ED25519) to the list of known hosts.
   ,     #_
   ~\_  ####_        Amazon Linux 2023
  ~~  \_#####\
  ~~     \###|
  ~~       \#/ ___   https://aws.amazon.com/linux/amazon-linux-2023
   ~~       V~' '->
    ~~~         /
      ~~._.   _/
         _/ _/
       _/m/'
Last login: Sat Sep 14 06:03:27 2024 from 18.206.107.27
[ec2-user@ip-172-31-20-75 ~]$
```

With this you can continue your commands through local terminal

3. From now on, until mentioned, perform these steps on all 3 machines.

Install Docker

sudo yum install docker -y

```
[ec2-user@ip-172-31-31-212 ~]$ sudo yum install docker -y
Last metadata expiration check: 0:33:43 ago on Thu Sep 12 13:11:13 2024.
Dependencies resolved.
================================================================================
 Package                    Architecture          Version
================================================================================
Installing:
 docker                     x86_64                25.0.6-1.amzn2023.0.2
Installing dependencies:
 containerd                 x86_64                1.7.20-1.amzn2023.0.1
 iptables-libs              x86_64                1.8.8-3.amzn2023.0.2
 iptables-nft               x86_64                1.8.8-3.amzn2023.0.2
 libcgroup                  x86_64                3.0-1.amzn2023.0.1
 libnetfilter_conntrack     x86_64                1.0.8-2.amzn2023.0.2
 libnfnetlink               x86_64                1.0.1-19.amzn2023.0.2
 libnftnl                   x86_64                1.2.2-2.amzn2023.0.2
 pigz                       x86_64                2.5-1.amzn2023.0.3
 runc                       x86_64                1.1.13-1.amzn2023.0.1

Transaction Summary
```

Then, configure cgroup in a daemon.json file by using following commands

- cd /etc/docker
- cat <<EOF | sudo tee /etc/docker/daemon.json
  {
  "exec-opts": ["native.cgroupdriver=systemd"],
  "log-driver": "json-file",
  "log-opts": {
  "max-size": "100m"
  },
  "storage-driver": "overlay2"
  }
  EOF

```
[ec2-user@ip-172-31-20-75 ~]$ cd /etc/docker
[ec2-user@ip-172-31-20-75 docker]$ cat <<EOF | sudo tee /etc/docker/daemon.json
{
"exec-opts": ["native.cgroupdriver=systemd"],
"log-driver": "json-file",
"log-opts": {
"max-size": "100m"
},
"storage-driver": "overlay2"
}
EOF
{
"exec-opts": ["native.cgroupdriver=systemd"],
"log-driver": "json-file",
"log-opts": {
"max-size": "100m"
},
"storage-driver": "overlay2"
}
[ec2-user@ip-172-31-20-75 docker]$ ls
daemon.json  kubectl
```

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

```
[ec2-user@ip-172-31-31-212 docker]$ sudo systemctl enable docker
Created symlink /etc/systemd/system/multi-user.target.wants/docker.service → /usr/lib/systemd/system/docker.service.
[ec2-user@ip-172-31-31-212 docker]$ sudo systemctl daemon-reload
[ec2-user@ip-172-31-31-212 docker]$ sudo systemctl restart docker
[ec2-user@ip-172-31-31-212 docker]$ docker -v
Docker version 25.0.5, build 5dc9bcc
[ec2-user@ip-172-31-31-212 docker]$
```

4. **Install Kubernetes on all 3 machines**

   SELinux needs to be disabled before configuring kubelet

- sudo setenforce 0
- sudo sed -i 's/^SELINUX=enforcing$/SELINUX=permissive/' /etc/selinux/config

```
[ec2-user@ip-172-31-26-2 docker]$ sudo setenforce 0
[ec2-user@ip-172-31-26-2 docker]$ sudo sed -i 's/^SELINUX=enforcing$/SELINUX=permissive/' /etc/selinux/config
[ec2-user@ip-172-31-26-2 docker]$
```

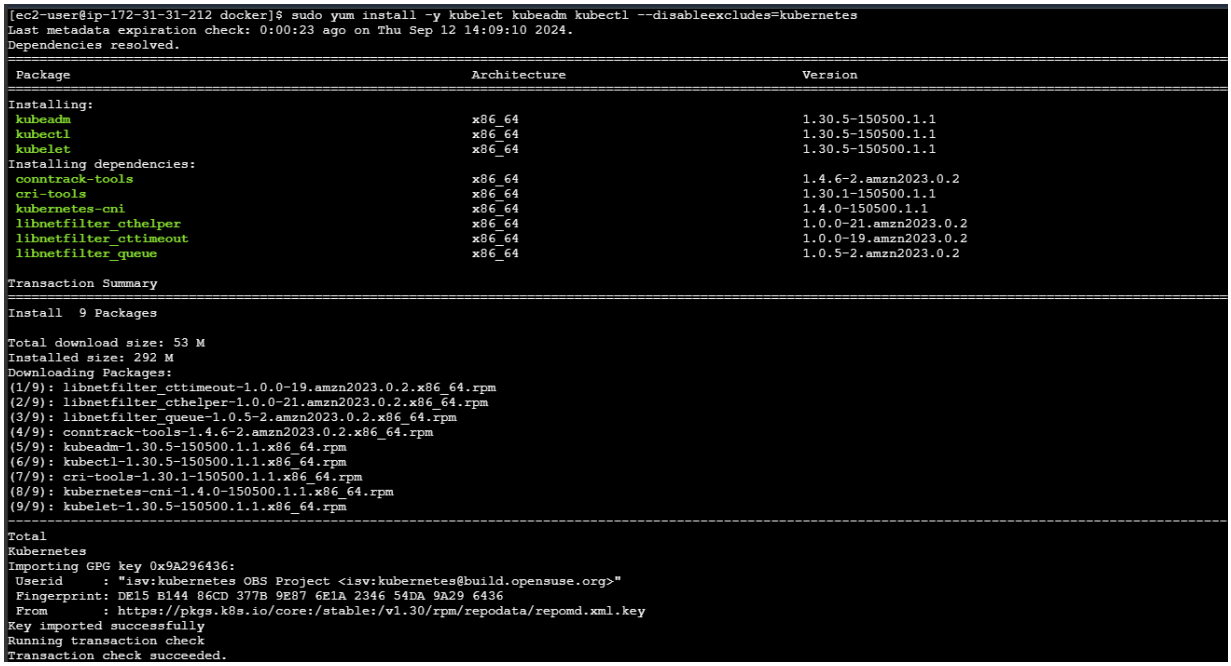
Add kubernetes repository (paste in terminal)

```
cat <<EOF | sudo tee /etc/yum.repos.d/kubernetes.repo
[kubernetes]
name=Kubernetes
baseurl=https://pkgs.k8s.io/core:/stable:/v1.30/rpm/
enabled=1
gpgcheck=1
gpgkey=https://pkgs.k8s.io/core:/stable:/v1.30/rpm/repodata/repomd.xml.key
exclude=kubelet kubeadm kubectl cri-tools kubernetes-cni
EOF
```

Type following commands to install set of kubernetes packages:

- sudo yum update

- sudo yum install -y kubelet kubeadm kubectl --disableexcludes=kubernetes



```
[ec2-user@ip-172-31-31-212 docker]$ sudo yum install -y kubelet kubeadm kubectl --disableexcludes=kubernetes
Last metadata expiration check: 0:00:23 ago on Thu Sep 12 14:09:10 2024.
Dependencies resolved.
================================================================================
 Package                 Architecture            Version
================================================================================
Installing:
 kubeadm                 x86_64                  1.30.5-150500.1.1
 kubectl                 x86_64                  1.30.5-150500.1.1
 kubelet                 x86_64                  1.30.5-150500.1.1
Installing dependencies:
 conntrack-tools         x86_64                  1.4.6-2.amzn2023.0.2
 cri-tools               x86_64                  1.30.1-150500.1.1
 kubernetes-cni          x86_64                  1.4.0-150500.1.1
 libnetfilter_cthelper   x86_64                  1.0.0-21.amzn2023.0.2
 libnetfilter_cttimeout  x86_64                  1.0.0-19.amzn2023.0.2
 libnetfilter_queue      x86_64                  1.0.5-2.amzn2023.0.2

Transaction Summary
================================================================================
Install  9 Packages

Total download size: 53 M
Installed size: 292 M
Downloading Packages:
(1/9): libnetfilter_cttimeout-1.0.0-19.amzn2023.0.2.x86_64.rpm
(2/9): libnetfilter_cthelper-1.0.0-21.amzn2023.0.2.x86_64.rpm
(3/9): libnetfilter_queue-1.0.5-2.amzn2023.0.2.x86_64.rpm
(4/9): conntrack-tools-1.4.6-2.amzn2023.0.2.x86_64.rpm
(5/9): kubeadm-1.30.5-150500.1.1.x86_64.rpm
(6/9): kubectl-1.30.5-150500.1.1.x86_64.rpm
(7/9): cri-tools-1.30.1-150500.1.1.x86_64.rpm
(8/9): kubernetes-cni-1.4.0-150500.1.1.x86_64.rpm
(9/9): kubelet-1.30.5-150500.1.1.x86_64.rpm
--------------------------------------------------------------------------------
Total
Kubernetes
Importing GPG key 0x9A296436:
 Userid     : "isv:kubernetes OBS Project <isv:kubernetes@build.opensuse.org>"
 Fingerprint: DE15 B144 86CD 377B 9E87 6E1A 2346 54DA 9A29 6436
 From       : https://pkgs.k8s.io/core:/stable:/v1.30/rpm/repodata/repomd.xml.key
Key imported successfully
Running transaction check
Transaction check succeeded.
```

After installing Kubernetes, we need to configure internet options to allow bridging.

- sudo swapoff -a
- echo "net.bridge.bridge-nf-call-iptables=1" | sudo tee -a /etc/sysctl.conf
- sudo sysctl -p

5. **Perform this ONLY on the Master machine**

   Initialize kubernetes by typing below command

   - sudo kubeadm init --pod-network-cidr=10.244.0.0/16 --ignore-preflight-errors=all

```
[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 172.31.26.2:6443 --token 6cjsz0.8ei243v0zn9k7erg \
        --discovery-token-ca-cert-hash sha256:abd917ec30e12c5616bf647a3d174bef3d271e92c30b8f2f7768cfb3181341d4
[ec2-user@ip-172-31-26-2 docker]$
```

Copy the mkdir and chown commands from the top and execute them

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

Copy this join link and save it in clipboard (copy from your output as it different for each master instance)

Example :

```
kubeadm join 172.31.20.75:6443 --token 66kg9u.2bc0kze31hrwbzvr \
        --discovery-token-ca-cert-hash
sha256:5e478da328b199e17d9b5da68e78bc9a6daab2043b05860552f4c184a7b3cb66
```

Then, add a common networking plugin called flamel file as mentioned in the code.

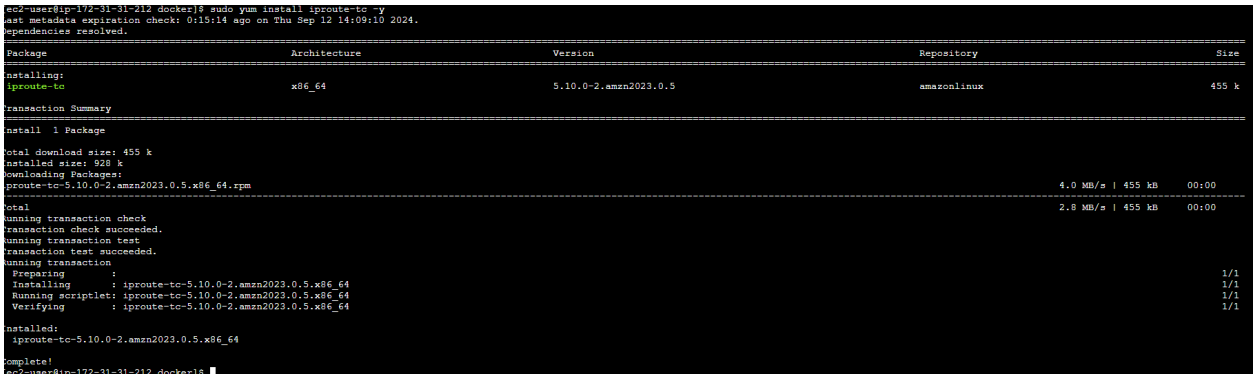
Command:

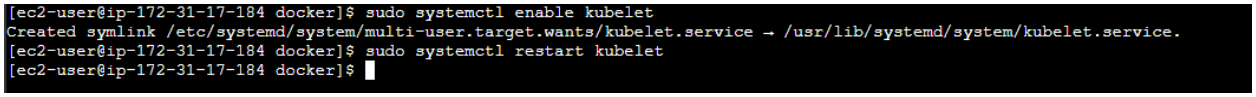
```
kubectl apply -f
https://raw.githubusercontent.com/coreos/flannel/master/Documentation/kube-flannel.yml
```

```
[ec2-user@ip-172-31-26-2 docker]$ 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
```

6. **Perform this ONLY on the worker machines**

   Paste the below command on all 2 worker machines
   - sudo yum install iproute-tc -y
   - sudo systemctl enable kubelet
   - sudo systemctl restart kubelet





Now paste the hash that yo copied in these worker note to connect to master cluster

- kubeadm join 172.31.20.75:6443 --token 66kg9u.2bc0kze31hrwbzvr \
  --discovery-token-ca-cert-hash sha256:5e478da328b199e17d9b5da68e78bc9a6daab2043b05860552f4c184a7b3cb66

Now we can see in the master/control node of kubernetes that worker nodes are connected by this command

- **watch kubectl get nodes**

(in the master node instance)

<u>Errors faced during the execution</u> :

1. In the end kubelet might not respond or the connectivity of nodes to master might not happen
2. You can see this error

3. Try to restart the kubelet from worker instance and try the commands again

Conclusion :

In this experiment, we set out to deploy Kubernetes on Docker by connecting a master node to two worker nodes. We encountered several issues, starting with misconfigured SSH inbound rules, which were resolved by correctly enabling the necessary access rules. It became evident that using t2.medium or t3 instances was crucial to provide adequate resources for running Kubernetes efficiently. However, despite these adjustments, the worker nodes were unable to join the cluster. While the master node was successfully initialized, the issue seemed to lie in the worker nodes, possibly due to misconfigurations in the kubelet setup or networking challenges. This included the worker nodes being unable to communicate with the master node's API server, which might have been caused by incorrect firewall settings, missing API server certificates, or errors during the kubeadm join process on the worker nodes.

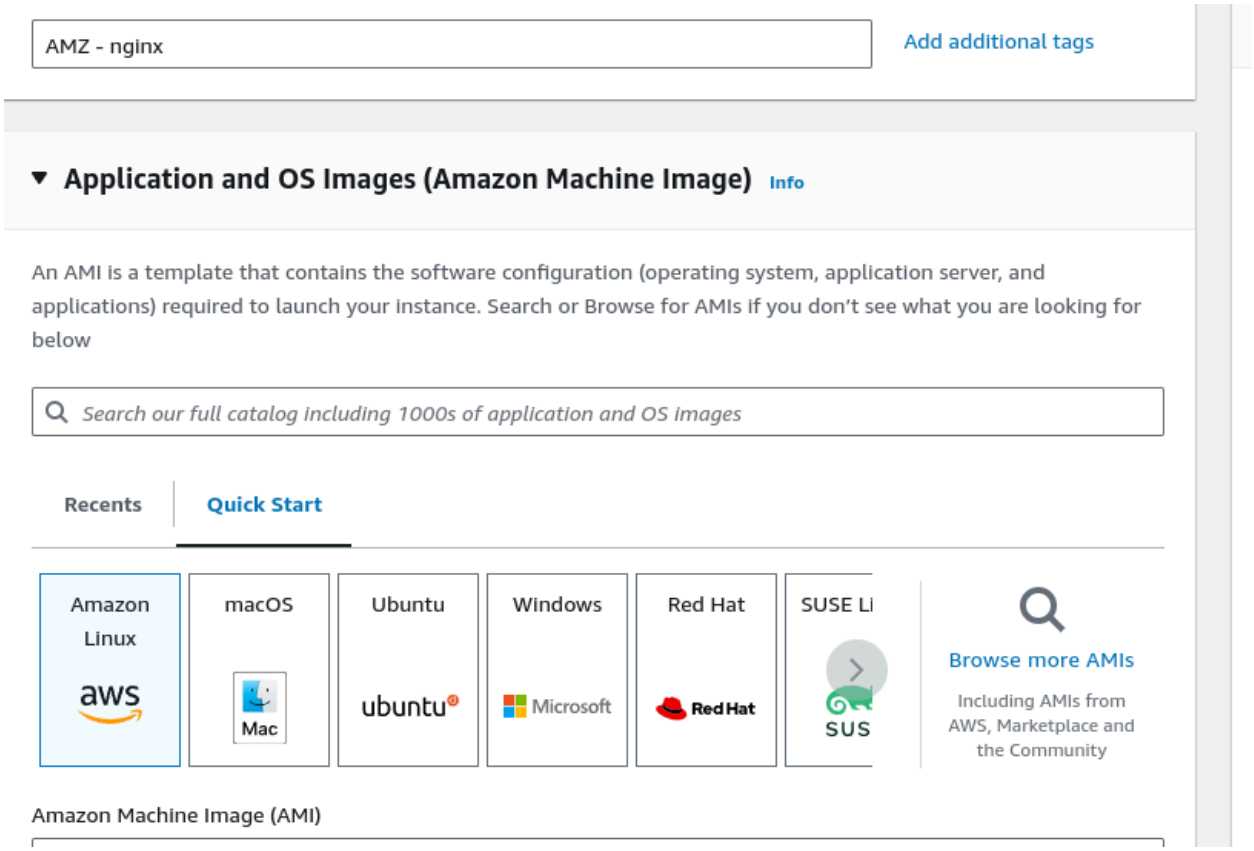
Name: Kshitij Hundre **Div:D15C** **Roll No:18**

EXPERIMENT-4 ADV-DEVOPS

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

STEPS:

1. Select Amazon linux as OS image (You can use any but then modify commands accordingly)



2. Make ssh connection in terminal
   Note: If you have directly made connection through browser then skip this part

Name: Kshitij Hundre **Div:D15C** **Roll No:18**

```
quantum@machine ~/Downloads ssh -i "ec-2-ubuntu.pem" ec2-user@ec2-54-162-208-25.compute-1.amazonaws.com
The authenticity of host 'ec2-54-162-208-25.compute-1.amazonaws.com (54.162.208.25)' can't be established.
ED25519 key fingerprint is SHA256:lBrdxB+9Hn3kWlOYZWmzh1wg/R4s+e7QDAMBZJPvf/E.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-54-162-208-25.compute-1.amazonaws.com' (ED25519) to the list of known hosts.
   ,     #_
   ~\_  ####_        Amazon Linux 2
  ~~  \_#####\
  ~~     \###|       AL2 End of Life is 2025-06-30.
  ~~       \#/ ___
   ~~       V~' '->
    ~~~         /    A newer version of Amazon Linux is available!
      ~~._.   _/
         _/ _/       Amazon Linux 2023, GA and supported until 2028-03-15.
       _/m/'           https://aws.amazon.com/linux/amazon-linux-2023/

[ec2-user@ip-172-31-31-63 ~]$ sudo yum update
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
No packages marked for update
[ec2-user@ip-172-31-31-63 ~]$ sudo yum upgrade
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
No packages marked for update
[ec2-user@ip-172-31-31-63 ~]$
```

3. **Install Docker**

sudo dnf update
sudo dnf install docker
sudo systemctl enable docker
sudo systemctl start docker

To test whether docker is successfully running, use command **sudo docker run hello-world**

```
[ec2-user@ip-172-31-24-190 ~]$ sudo docker run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
c1ec31eb5944: Pull complete
Digest: sha256:91fb4b041da273d5a3273b6d587d62d518300a6ad268b28628f74997b93171b2
Status: Downloaded newer image for hello-world:latest

Hello from Docker!
This message shows that your installation appears to be working correctly.
```

Then, configure cgroup in a daemon.json file.This allows kubernetes to manage host more efficiently

```
cd /etc/docker

cat <<EOF | sudo tee /etc/docker/daemon.json
{
"exec-opts": ["native.cgroupdriver=systemd"]
}
EOF
sudo systemctl daemon-reload
sudo systemctl restart docker
```

4. Install Kubernetes

Note: I'm directly installing binary package you may install from package repository of your distribution

Install CNI plugins (required for most pod network):

```
CNI_PLUGINS_VERSION="v1.3.0"
ARCH="amd64"
DEST="/opt/cni/bin"
sudo mkdir -p "$DEST"
curl -L
"https://github.com/containernetworking/plugins/releases/download/${CNI_PLUGINS_V
ERSION}/cni-plugins-linux-${ARCH}-${CNI_PLUGINS_VERSION}.tgz" | sudo tar -C
"$DEST" -xz
```

Define the directory to download command files:

```
DOWNLOAD_DIR="/usr/local/bin"
sudo mkdir -p "$DOWNLOAD_DIR"
```

Optionally install crictl (required for interaction with the Container Runtime Interface (CRI), optional for kubeadm):

```
CRICTL_VERSION="v1.31.0"
ARCH="amd64"
```

curl -L "https://github.com/kubernetes-sigs/cri-tools/releases/download/${CRICTL_VERSION}/crictl-${CRICTL_VERSION}-linux-${ARCH}.tar.gz" | sudo tar -C $DOWNLOAD_DIR -xz

Install `kubeadm`, `kubelet` and add a `kubelet` systemd service:

RELEASE="$(curl -sSL https://dl.k8s.io/release/stable.txt)"
ARCH="amd64"
cd $DOWNLOAD_DIR
sudo curl -L --remote-name-all https://dl.k8s.io/release/${RELEASE}/bin/linux/${ARCH}/{kubeadm,kubelet}
sudo chmod +x {kubeadm,kubelet}

RELEASE_VERSION="v0.16.2"
curl -sSL "https://raw.githubusercontent.com/kubernetes/release/${RELEASE_VERSION}/cmd/krel/templates/latest/kubelet/kubelet.service" | sed "s:/usr/bin:${DOWNLOAD_DIR}:g" | sudo tee /usr/lib/systemd/system/kubelet.service
sudo mkdir -p /usr/lib/systemd/system/kubelet.service.d
curl -sSL "https://raw.githubusercontent.com/kubernetes/release/${RELEASE_VERSION}/cmd/krel/templates/latest/kubeadm/10-kubeadm.conf" | sed "s:/usr/bin:${DOWNLOAD_DIR}:g" | sudo tee /usr/lib/systemd/system/kubelet.service.d/10-kubeadm.conf

Now we need to install kubectl

```
Set up repository:

cat <<EOF | sudo tee /etc/yum.repos.d/kubernetes.repo
[kubernetes]
name=Kubernetes
baseurl=https://pkgs.k8s.io/core:/stable:/v1.31/rpm/
enabled=1
gpgcheck=1
gpgkey=https://pkgs.k8s.io/core:/stable:/v1.31/rpm/repodata/repomd.xml.key
EOF

sudo yum install -y kubectl
```

Name: Kshitij Hundre **Div:D15C** **Roll No:18**

```
ec2-user@ip-172-31-24-190 ~ $ kubectl version
Client Version: v1.31.1
Kustomize Version: v5.4.2
```

We have installed successfully installed kubernetes

After installing Kubernetes, we need to configure internet options to allow bridging.

```
sudo swapoff -a
echo "net.bridge.bridge-nf-call-iptables=1" | sudo tee -a /etc/sysctl.conf
sudo sysctl -p
```

```
[root@ip-172-31-24-190 bin]# sudo swapoff -a
echo "net.bridge.bridge-nf-call-iptables=1" | sudo tee -a /etc/sysctl.conf
sudo sysctl -p
net.bridge.bridge-nf-call-iptables=1
net.bridge.bridge-nf-call-iptables = 1
```

Disable SELINUX

Type **sudo nano /etc/selinux/config** and set the value of **SELINUX=disabled** instead of **SELINUX=permissive**
Save the file by pressing ctrl+o then press enter then press ctrl+x

```
# This file controls the state of SELinux on the system.
# SELINUX= can take one of these three values:
#     enforcing - SELinux security policy is enforced.
#     permissive - SELinux prints warnings instead of enforcing.
#     disabled - No SELinux policy is loaded.
# See also:
# https://docs.fedoraproject.org/en-US/quick-docs/getting-started-with-selinux/#getting-started-with-selinux-selinux-states-and-modes
#
# NOTE: In earlier Fedora kernel builds, SELINUX=disabled would also
# fully disable SELinux during boot. If you need a system with SELinux
# fully disabled instead of SELinux running with no policy loaded, you
# need to pass selinux=0 to the kernel command line. You can use grubby
# to persistently set the bootloader to boot with selinux=0:
#
#    grubby --update-kernel ALL --args selinux=0
#
# To revert back to SELinux enabled:
#
#    grubby --update-kernel ALL --remove-args selinux
#
SELINUX=disabled
# SELINUXTYPE= can take one of these three values:
#     targeted - Targeted processes are protected,
#     minimum - Modification of targeted policy. Only selected processes are protected.
#     mls - Multi Level Security protection.
SELINUXTYPE=targeted
```

Then reboot the system using **sudo reboot**
After rebooting we need to make ssh connection with machine after it gets disconnected

Now if we type command **sestatus,** then it show disabled

```
ec2-user@ip-172-31-24-190 ~ $ sestatus
SELinux status:                 disabled
```

5. Initialize the Kubecluster

Install packages socat and iproute-tc and conntrack to avoid prelight errors
sudo dnf install socat iproute-tc conntrack-tools -y

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

Name: Kshitij Hundre **Div:D15C** **Roll No:18**

```
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 172.31.24.190:6443 --token xsbsq1.6ro11sawnvttbsvu \
        --discovery-token-ca-cert-hash sha256:10d2b67f4f4749b51854065a554c74e6a956e4782d9ab4bb79b8591648b3edef
ec2-user@ip-172-31-24-190 ~ $ kubectl get nodes
```

Copy the mkdir and chown commands from the top and execute them

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

sudo systemctl restart kubelet

Then, 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

```
ec2-user@ip-172-31-24-190 ~ $ 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
```

Now type **kubectl get nodes**

```
The connection to the server 172.31.24.190:6443 was refused - did you specify the right host or port?
ec2-user@ip-172-31-24-190 ~ $ kubectl get nodes
The connection to the server 172.31.24.190:6443 was refused - did you specify the right host or port?
ec2-user@ip-172-31-24-190 ~ $ kubectl get nodes
^[[AError from server (Forbidden): nodes is forbidden: User "kubernetes-admin" cannot list resource "nodes" in
ec2-user@ip-172-31-24-190 ~ $ kubectl get nodes
NAME                            STATUS   ROLES           AGE   VERSION
ip-172-31-24-190.ec2.internal   Ready    control-plane   34m   v1.31.0
ec2-user@ip-172-31-24-190 ~ $ kubectl get nodes
NAME                            STATUS   ROLES           AGE   VERSION
ip-172-31-24-190.ec2.internal   Ready    control-plane   34m   v1.31.0
ec2-user@ip-172-31-24-190 ~ $
```

Note: If any time of get error of connection refused just restart the kubelet service (sudo systemctl restart kubelet)

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

```
ec2-user@ip-172-31-24-190 ~ $ kubectl apply -f https://k8s.io/examples/application/deployment.yaml
deployment.apps/nginx-deployment created
```

Use ‘kubectl get pods’ to verify if the deployment was properly created and the pod is working correctly.

```
ec2-user@ip-172-31-24-190 ~ $ kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
nginx-deployment-d556bf558-mwd8p    0/1     Pending   0          7s
nginx-deployment-d556bf558-zc25s    0/1     Pending   0          7s
```

As we can see our pods are in pending state
On checking logs to we came to know the pods are in tainted state (using command **kubectl describe pod nginx-deployment-d556bf558-mwd8p**)

```
Events:
  Type     Reason            Age   From               Message
  ----     ------            ----  ----               -------
  Warning  FailedScheduling  56s   default-scheduler  0/1 nodes are available: 1 node(s) had untolerated taint {node-role.kubernetes.io/control-plane: }. preemption: 0/1 nodes are available:
```

To make pods untainted
Type kubectl get nodes to see name of node

```
ec2-user@ip-172-31-24-190 ~ $ kubectl get nodes
NAME                              STATUS   ROLES           AGE   VERSION
ip-172-31-24-190.ec2.internal   Ready    control-plane   43m   v1.31.0
ec2-user@ip-172-31-24-190 ~ $
 3  1:ec2-user@ip-172-31-24-190:~*  2:~/Downloads-  3:~/Downloads#  4:~/Downloads
```

Copy the name of the node (ip-172-31-24-190.ec2.internal)

Then type command **kubectl taint nodes <NODE_NAME> - -all**

Name: Kshitij Hundre **Div:D15C** **Roll No:18**

In my case **kubectl taint nodes ip-172-31-24-190.ec2.internal node-role.kubernetes.io/control-plane-**

```
ec2-user@ip-172-31-24-190 ~ $ kubectl taint nodes ip-172-31-24-190.ec2.internal node-role.kubernetes.io/control-plane-
node/ip-172-31-24-190.ec2.internal untainted
```

After executing above command, check again status of pods if still pending then restart kubelet wait for 1-2 minutes and check again

```
ec2-user@ip-172-31-24-190 ~ $ kubectl get pods
NAME                                READY   STATUS    RESTARTS      AGE
nginx-deployment-d556bf558-mwd8p    1/1     Running   2 (73s ago)   12m
nginx-deployment-d556bf558-zc25s    1/1     Running   2 (73s ago)   12m
```

As we can see our pods are running

Lastly, port forward the deployment to your localhost so that you can view it.

kubectl port-forward <POD_NAME> 8080:80
In my case : **kubectl port-forward nginx-deployment-d556bf558-mwd8p 8080:80**

Note: if you are getting connection refused error then restart kubelet

```
ec2-user@ip-172-31-24-190 ~ $  kubectl port-forward  nginx-deployment-d556bf558-mwd8p 8080:80
Forwarding from 127.0.0.1:8080 -> 80
Forwarding from [::1]:8080 -> 80
```

As port forwarding is active so we cannot type other commands.
Open new terminal window and make ssh connection to same machine OR we can open instance of same machine in new browser tab
And type command **curl --head http://127.0.0.1:8080**

Name: Kshitij Hundre **Div:D15C** **Roll No:18**

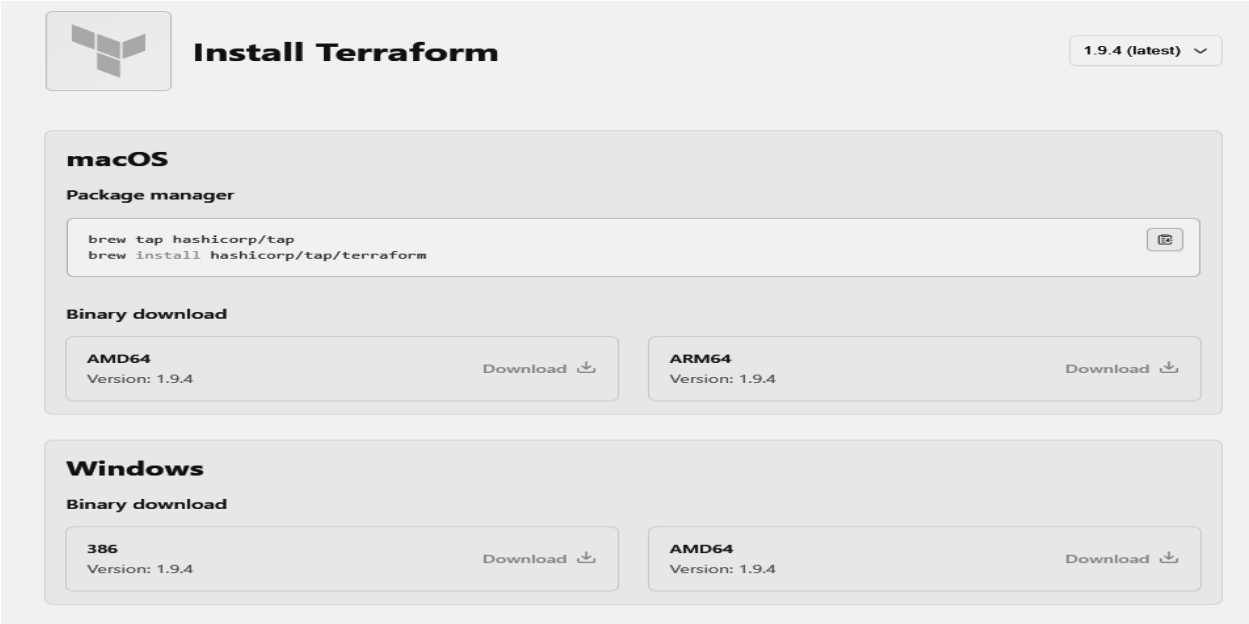
```
ec2-user@ip-172-31-24-190 ~ $ curl --head http://127.0.0.1:8080
HTTP/1.1 200 OK
Server: nginx/1.14.2
Date: Sat, 14 Sep 2024 06:54:21 GMT
Content-Type: text/html
Content-Length: 612
Last-Modified: Tue, 04 Dec 2018 14:44:49 GMT
Connection: keep-alive
ETag: "5c0692e1-264"
Accept-Ranges: bytes

ec2-user@ip-172-31-24-190 ~ $
 3  1:ec2-user@ip-172-31-24-190:~#-  2:ec2-user@ip-172-31-24-190:~*  3:~/Downloads
```

Response status 200 (OK) indicates that our nginx server is running successfully on kubernetes

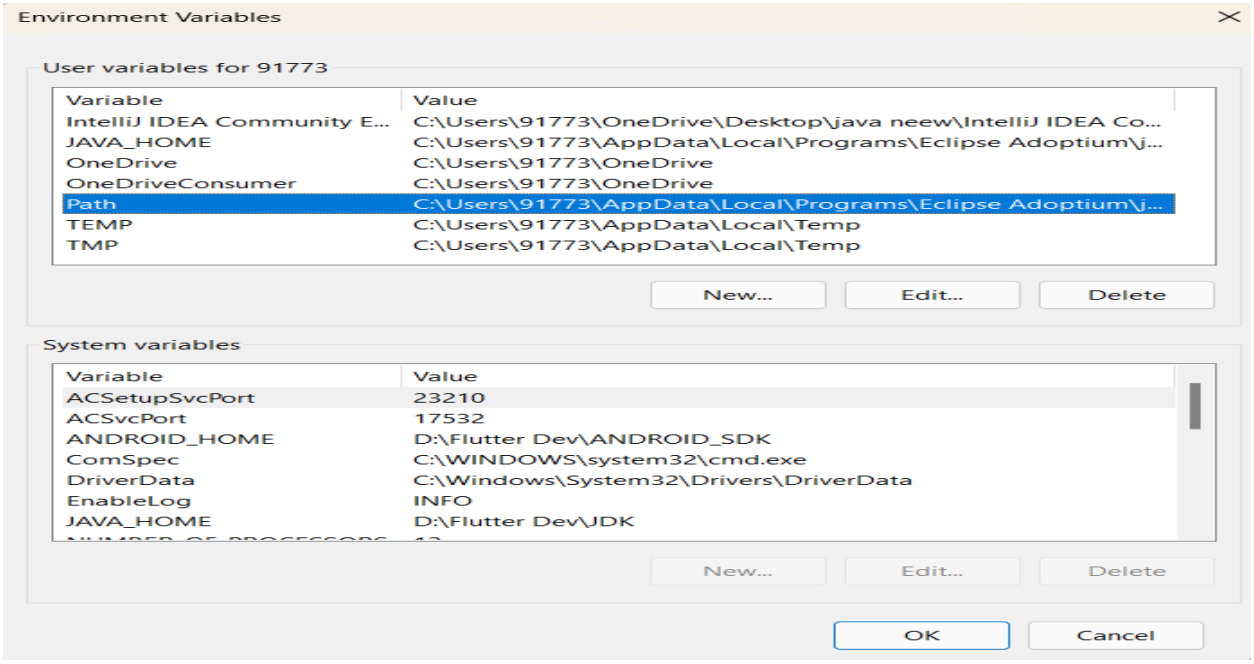
Conclusion: We started by installing and setting up Docker and Kubernetes. Initially, there were issues with the Kubernetes API server, but restarting the kubelet service resolved them. The pods weren't running at first due to node tainting, which we corrected by untainting the nodes. After addressing all the errors, we successfully deployed the NGINX server pods, which are now accessible via the forwarded port. The NGINX server can be accessed either through different terminals or by running the port forwarding process in the background, achieved by appending `&` to the command.

Name: Kshitij Hundre
Div: D15C
Roll No:18

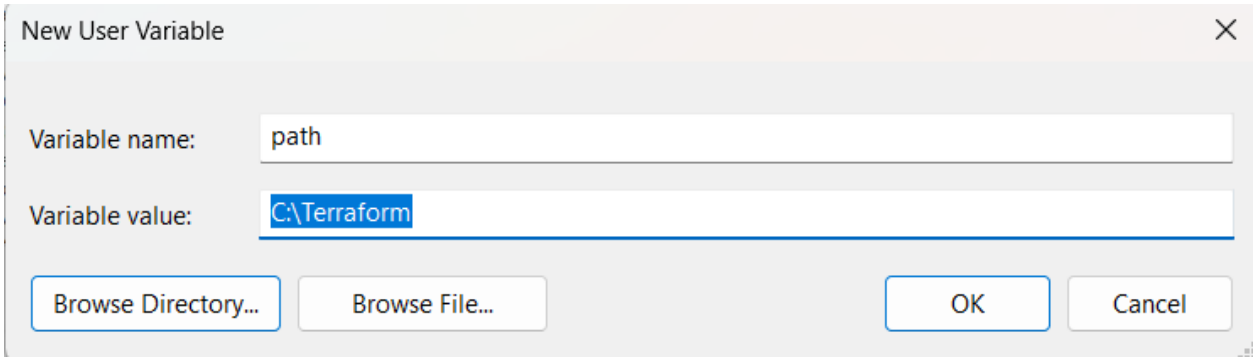
Step 1) Install Terraform



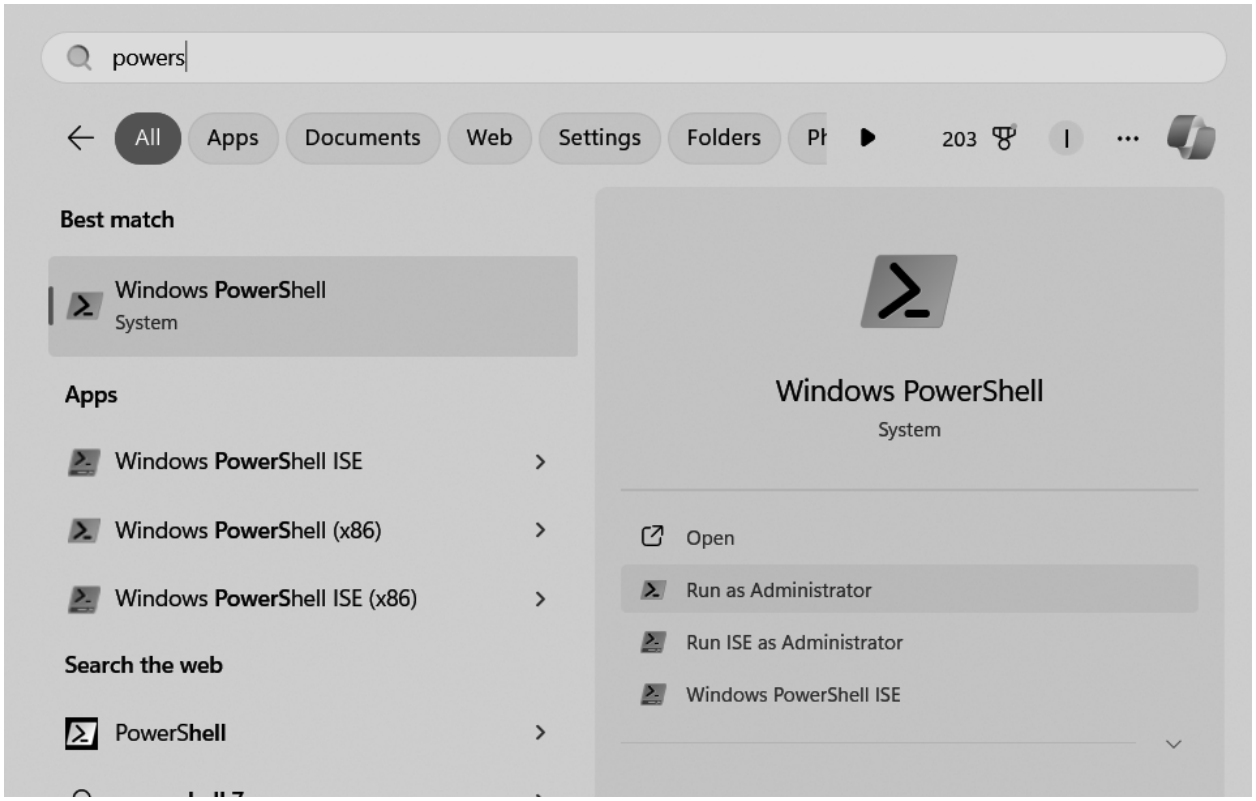
Step 2) Setup path in environment variables.



Step 3) Select the C drive Terraform folder as variable value.



Step 4) Open Windows powershell as Administrator.



Step 5) Run the Terraform command in powershell.

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

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

PS C:\WINDOWS\system32> 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
  version       Show the current Terraform version
  workspace     Workspace management

Global options (use these before the subcommand, if any):
  -chdir=DIR    Switch to a different working directory before executing the
                given subcommand.
  -help         Show this help output, or the help for a specified subcommand.
  -version      An alias for the "version" subcommand.
PS C:\WINDOWS\system32>
```

Name: KSHITIJ HUNDRE
Div: D15C
Roll NO:18

Experiment No.: 6

Implementation:

A. Creating docker image using terraform

Prerequisite:
1) Download and Install Docker Desktop from https://www.docker.com/

Step 1: Check the docker functionality

```
PS C:\Users\91773> docker

Usage:  docker [OPTIONS] COMMAND

A self-sufficient runtime for containers

Common Commands:
  run         Create and run a new container from an image
  exec        Execute a command in a running container
  ps          List containers
  build       Build an image from a Dockerfile
  pull        Download an image from a registry
  push        Upload an image to a registry
  images      List images
  login       Log in to a registry
  logout      Log out from a registry
  search      Search Docker Hub for images
  version     Show the Docker version information
  info        Display system-wide information

Management Commands:
  builder     Manage builds
  buildx*     Docker Buildx
  compose*    Docker Compose
  container   Manage containers
  context     Manage contexts
  debug*      Get a shell into any image or container
  desktop*    Docker Desktop commands (Alpha)
  dev*        Docker Dev Environments
```

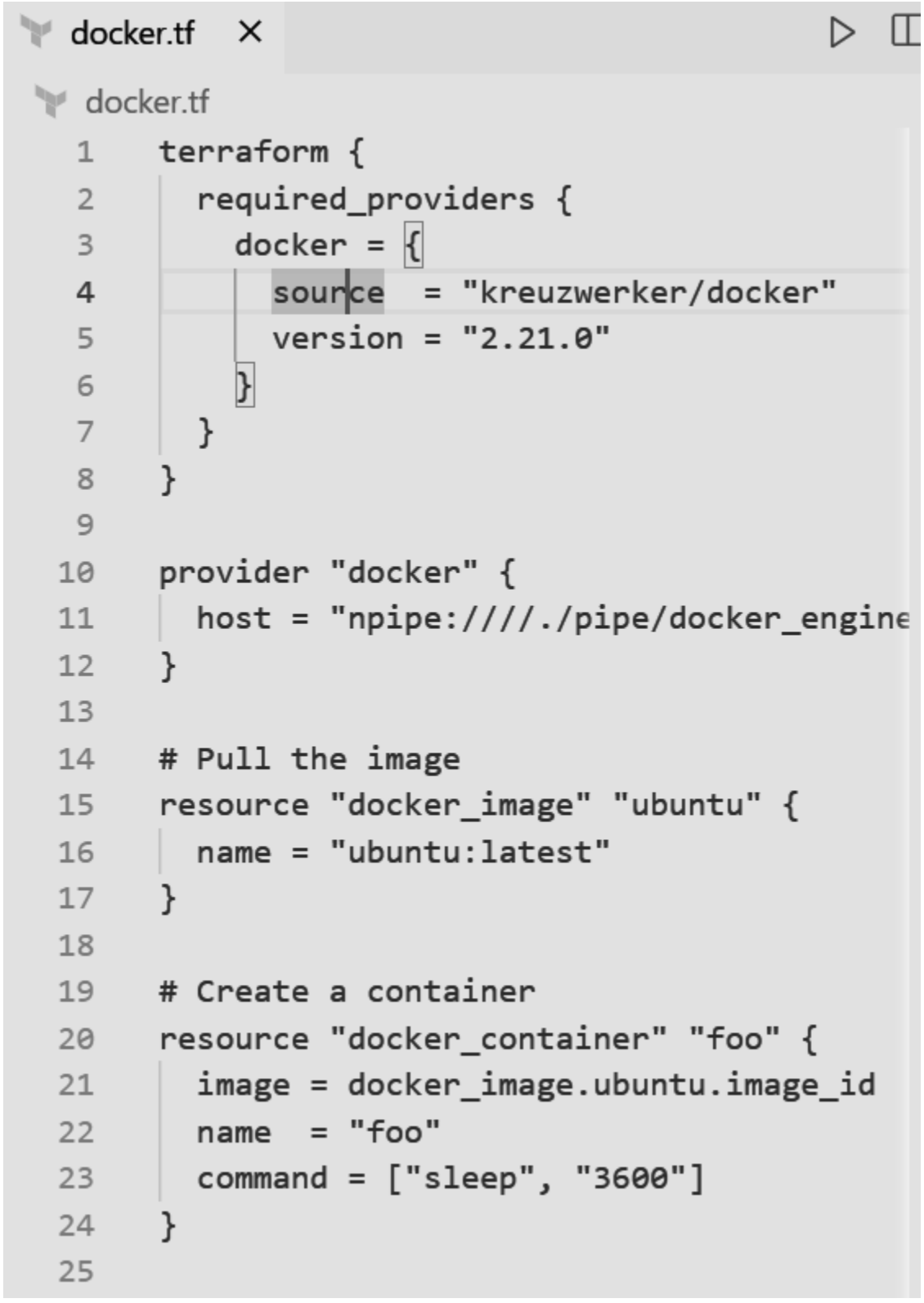
```
PS C:\Users\91773> docker --version
Docker version 27.0.3, build 7d4bcd8
PS C:\Users\91773>
```

Now, create a folder named ‘Terraform Scripts’ in which we save our different types of scripts which will be further used in this experiment.

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.

Script:

```
terraform {
  required_providers {
    docker = {
      source  = "kreuzwerker/docker"
      version = "2.21.0"
    }
  }
}

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

# Pull the image
resource "docker_image" "ubuntu" {
  name = "ubuntu:latest"
}

# Create a container
resource "docker_container" "foo" {
  image = docker_image.ubuntu.image_id
  name  = "foo"
  command = ["sleep", "3600"]
}
```


docker.tf
docker.tf
terraform {
required_providers {
docker = {
source = "kreuzwerker/docker"
version = "2.21.0"
}
}
}
provider "docker" {
host = "npipe:////./pipe/docker_engine
}
# Pull the image
resource "docker_image" "ubuntu" {
name = "ubuntu:latest"
}
# Create a container
resource "docker_container" "foo" {
image = docker_image.ubuntu.image_id
name = "foo"
command = ["sleep", "3600"]
}

Step 3: Execute Terraform Init command to initialize the resources

```
PS C:\Users\91773\Desktop\College Resources\TerraformScripts\Docker> terraform init
Initializing the backend...
Initializing provider plugins...
- Finding kreuzwerker/docker versions matching "2.21.0"...
- Installing kreuzwerker/docker v2.21.0...
- Installed kreuzwerker/docker v2.21.0 (self-signed, key ID BD080C4571C6104C)
Partner and community providers are signed by their developers.
If you'd like to know more about provider signing, you can read about it here:
https://www.terraform.io/docs/cli/plugins/signing.html
Terraform has created a lock file .terraform.lock.hcl to record the provider
selections it made above. Include this file in your version control repository
so that Terraform can guarantee to make the same selections by default when
you run "terraform init" in the future.

Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.

If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
```

Step 4: Execute Terraform plan to see the available resources

```
PS C:\Users\91773\Desktop\College Resources\TerraformScripts\Docker> terraform plan

Terraform used the selected providers to generate the following execution plan. Resource actions are indicate
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)
```

```
      + security_opts    = (known after apply)
      + shm_size         = (known after apply)
      + start            = true
      + stdin_open       = false
      + stop_signal      = (known after apply)
      + stop_timeout     = (known after apply)
      + tty              = false

      + 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.
```

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"**

```
PS C:\Users\91773\Desktop\College Resources\TerraformScripts\Docker> terraform apply
docker_image.ubuntu: Refreshing state... [id=sha256:edbfe74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761
tu:latest]

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated
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          = [
          + "sleep",
          + "3600",
        ]
```

Docker images, Before Executing Apply step:

```
PS C:\Users\91773\Desktop\College Resources\TerraformScripts\Docker> docker images
REPOSITORY   TAG       IMAGE ID   CREATED   SIZE
```

Docker images, After Executing Apply step:

```
Apply complete! Resources: 1 added, 0 changed, 0 destroyed.
PS C:\Users\91773\Desktop\College Resources\TerraformScripts\Docker> docker images
REPOSITORY   TAG       IMAGE ID       CREATED       SIZE
ubuntu       latest    edbfe74c41f8   2 weeks ago   78.1MB
PS C:\Users\91773\Desktop\College Resources\TerraformScripts\Docker>
```

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

```
PS C:\Users\91773\Desktop\College Resources\TerraformScripts\Docker> terraform destroy
docker_image.ubuntu: Refreshing state... [id=sha256:edbfe74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad76
tu:latest]
docker_container.foo: Refreshing state... [id=f03a28e4658896c23c9992f7a98eb1011befc7d014e997ea9fc6372da70b790

Terraform used the selected providers to generate the following execution plan. Resource actions are indicate
following symbols:
  - destroy

Terraform will perform the following actions:

  # docker_container.foo will be destroyed
  - resource "docker_container" "foo" {
      - attach            = false -> null
      - command           = [
          - "sleep",
          - "3600",
        ] -> null
      - cpu_shares        = 0 -> null
```

```
  # docker_image.ubuntu will be destroyed
  - resource "docker_image" "ubuntu" {
      - id          = "sha256:edbfe74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598aubuntu:latest"
      - image_id    = "sha256:edbfe74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598a" -> null
      - latest      = "sha256:edbfe74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598a" -> null
      - name        = "ubuntu:latest" -> null
      - repo_digest = "ubuntu@sha256:8a37d68f4f73ebf3d4efafbcf66379bf3728902a8038616808f04e34a9ab63ee" -> nul
    }

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

Do you really want to destroy all resources?
  Terraform will destroy all your managed infrastructure, as shown above.
  There is no undo. Only 'yes' will be accepted to confirm.

  Enter a value: yes

docker_container.foo: Destroying... [id=f03a28e4658896c23c9992f7a98eb1011befc7d014e997ea9fc6372da70b7903]
docker_container.foo: Destruction complete after 0s
docker_image.ubuntu: Destroying... [id=sha256:edbfe74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598
est]
docker_image.ubuntu: Destruction complete after 1s

Destroy complete! Resources: 2 destroyed.
```

Docker images After Executing Destroy step

```
PS C:\Users\91773\Desktop\College Resources\TerraformScripts\Docker> docker im
REPOSITORY   TAG       IMAGE ID   CREATED   SIZE
```

Name:Kshitij Hundre Div:D15C Roll No.18

Adv DevOps Practical 7

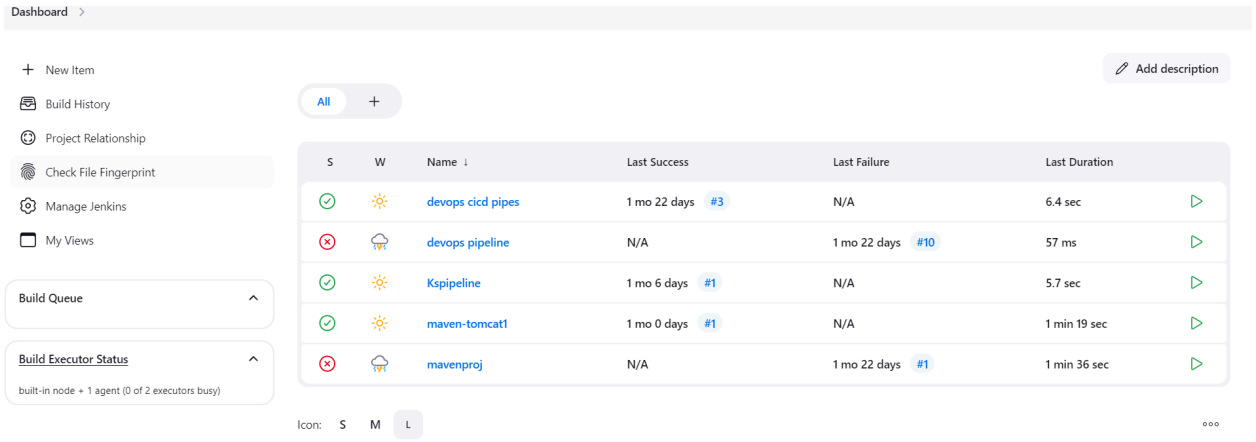
Aim: To understand Static Analysis SAST process and learn to integrate Jenkins SAST to SonarQube/GitLab.

Integrating Jenkins with SonarQube:

- Jenkins installed
- Docker Installed (for SonarQube)
- SonarQube Docker Image

Steps to integrate Jenkins with SonarQube

1. Open up Jenkins Dashboard on localhost, port 8090 or whichever port it is at for you.



Name:Kshitij Hundre Div:D15C Roll No.18

2. Run SonarQube in a Docker container using this command -

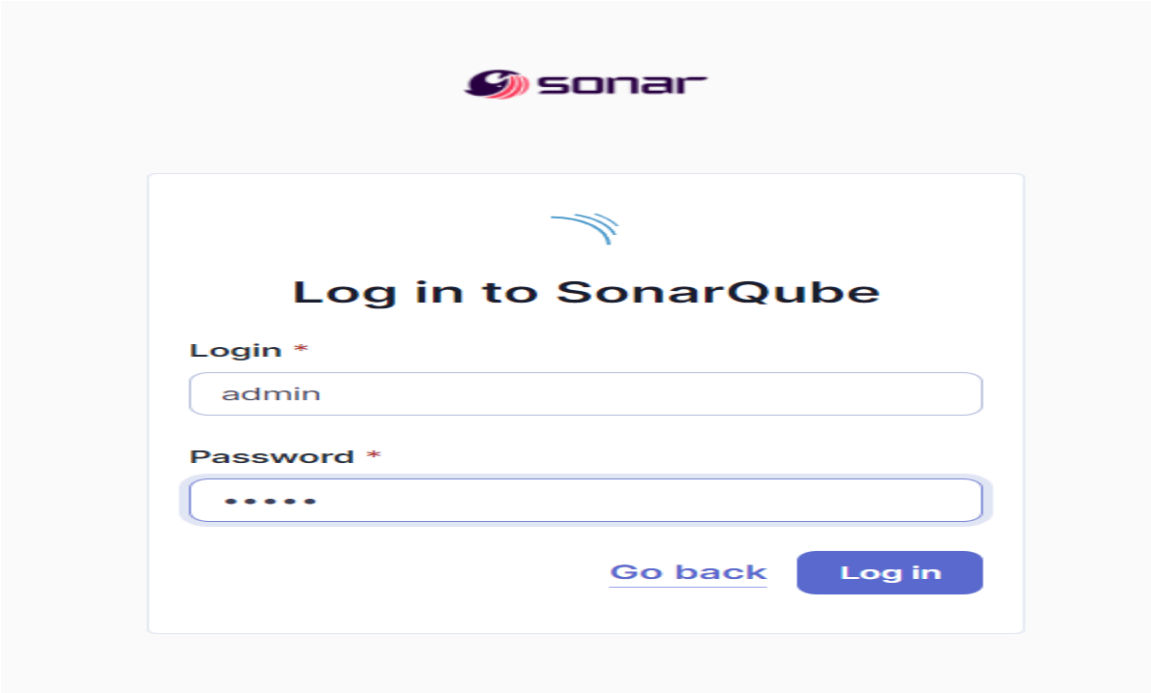
docker run -d --name sonarqube -e SONAR_ES_BOOTSTRAP_CHECKS_DISABLE=true -p 9000:9000 sonarqube:latest

----------Warning: run below command only once

```
PS C:\Users\91773\Desktop\College Resources\Exp7 adv devops> docker run -d --name sonarqube -e SONAR_ES_BOOTSTRAP_CHECKS
_DISABLE=true -p 9000:9000 sonarqube:latest
Unable to find image 'sonarqube:latest' locally
latest: Pulling from library/sonarqube
7478e0ac0f23: Pull complete
90a925ab929a: Pull complete
7d9a34308537: Pull complete
80338217a4ab: Pull complete
1a5fd5c7e184: Pull complete
7b87d6fa783d: Pull complete
bd819c9b5ead: Pull complete
4f4fb700ef54: Pull complete
Digest: sha256:72e9feec71242af83faf65f95a40d5e3bb2822a6c3b2cda8568790f3d31aecde
Status: Downloaded newer image for sonarqube:latest
77e678cded2ef5f989912d3d9e6991dd548eac03faa1eed68dd906614be53acc
PS C:\Users\91773\Desktop\College Resources\Exp7 adv devops>
```

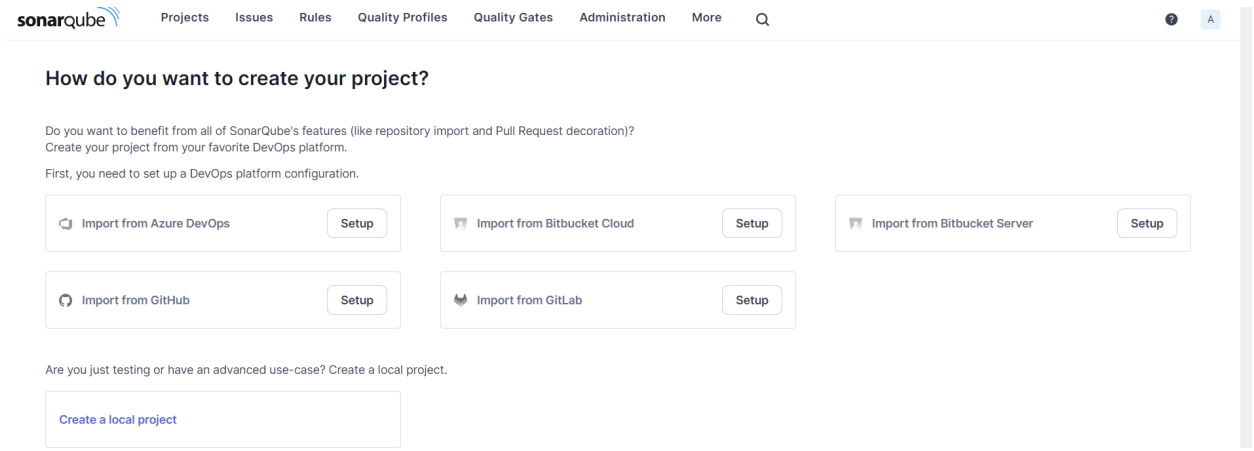
3. Once the container is up and running, you can check the status of SonarQube at

localhost port 9000.

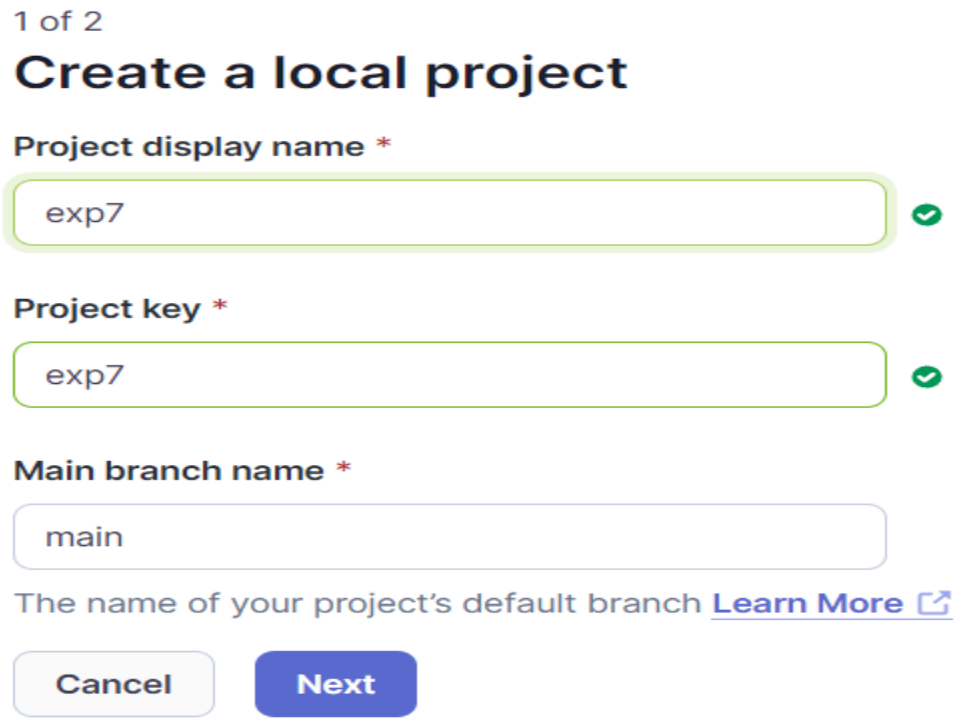


Name:Kshitij Hundre Div:D15C Roll No.18

4. Login to SonarQube using username admin and password admin.



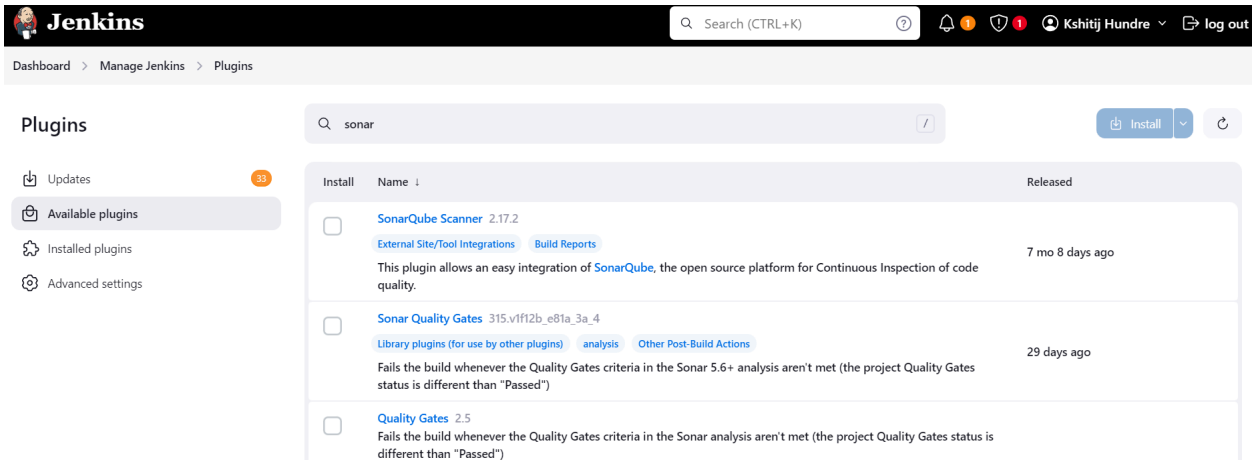
5. Create a manual project in SonarQube with the name sonarqube



Name:Kshitij Hundre Div:D15C Roll No.18

Setup the project and come back to Jenkins Dashboard.

Go to Manage Jenkins and search for SonarQube Scanner for Jenkins and install it.



6. Under Jenkins ‘Manage Jenkins’ then go to ‘system’, scroll and look for **SonarQube Servers** and enter the details.

Enter the Server Authentication token if needed.

In SonarQube installations: Under **Name** add <project name of sonarqube> for me

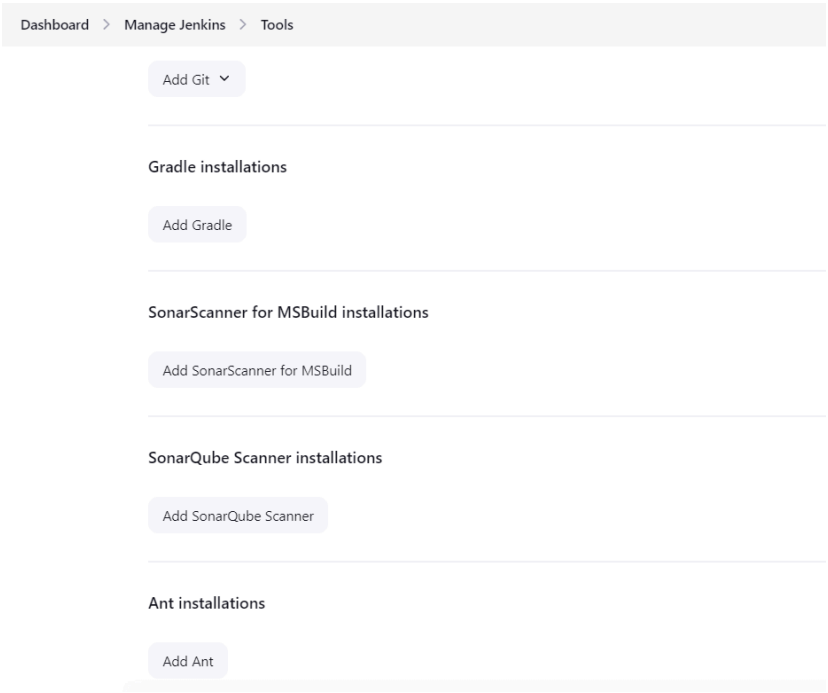
sahilexp7

In **Server URL** Default is http://localhost:9000

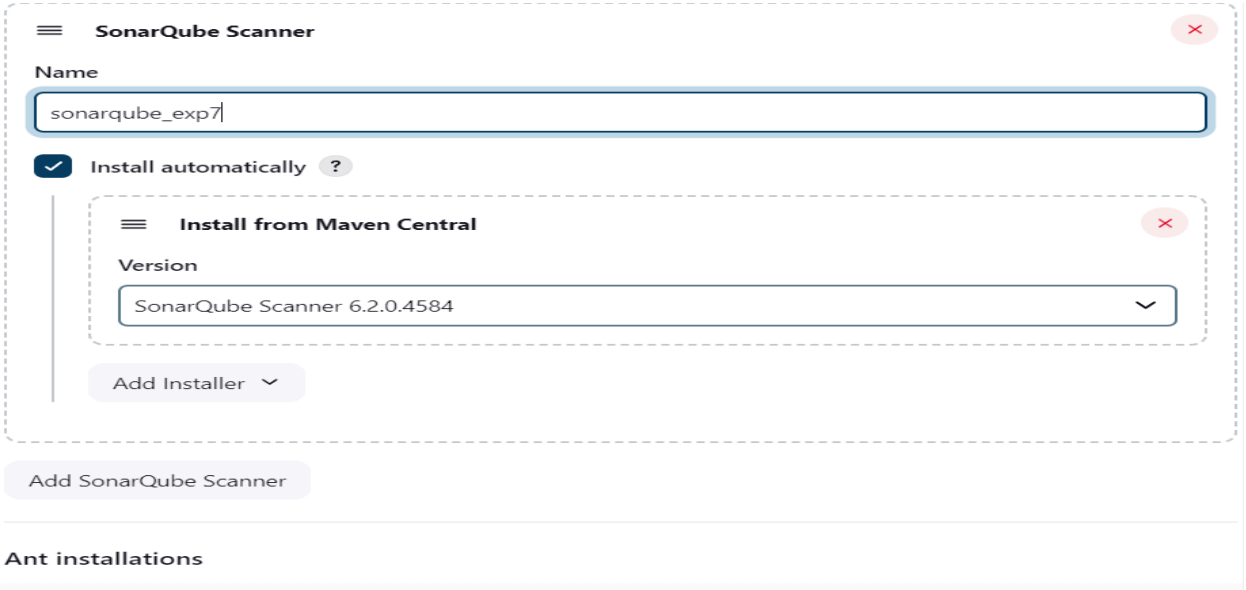
Name:Kshitij Hundre Div:D15C Roll No.18

7. Search for SonarQube Scanner under Global Tool Configuration. Choose the latest configuration and choose Install automatically.

Dashboard > Manage Jenkins > Tools

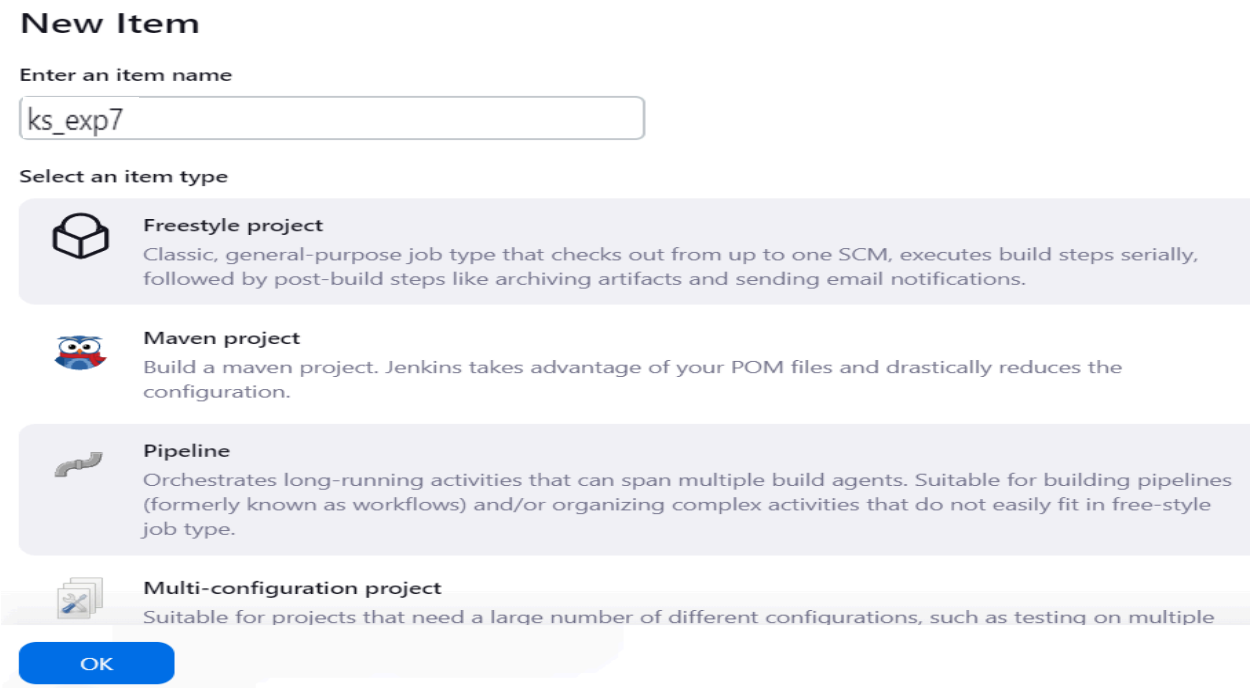


Check the “Install automatically” option. → Under name any name as identifier → Check the “Install automatically” option.



Name:Kshitij Hundre Div:D15C Roll No.18

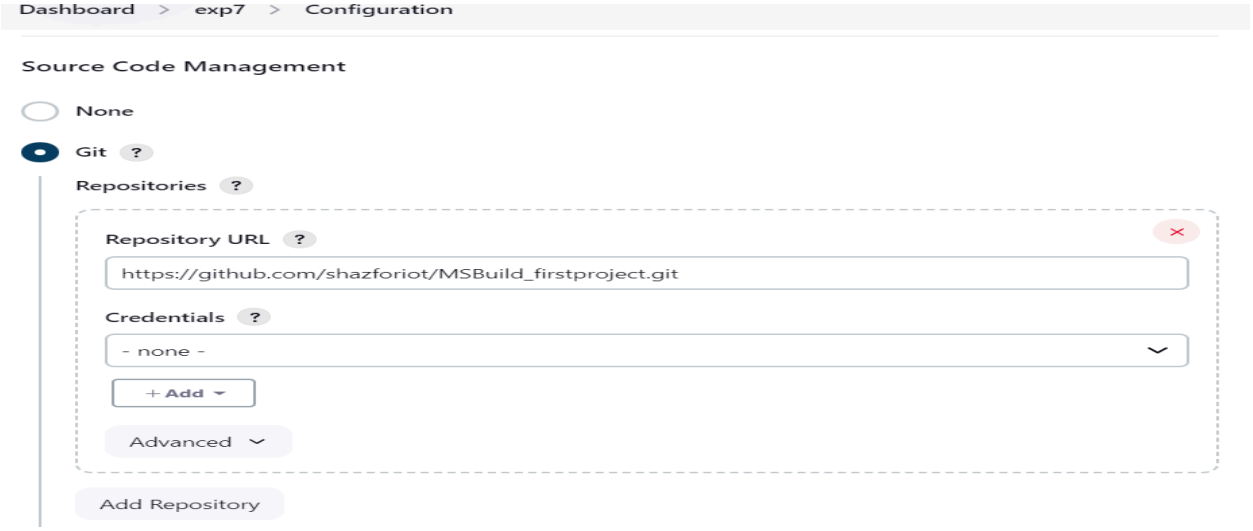
8. After the configuration, create a New Item in Jenkins, choose a freestyle project.ks



9. Choose this GitHub repository in Source Code Management.

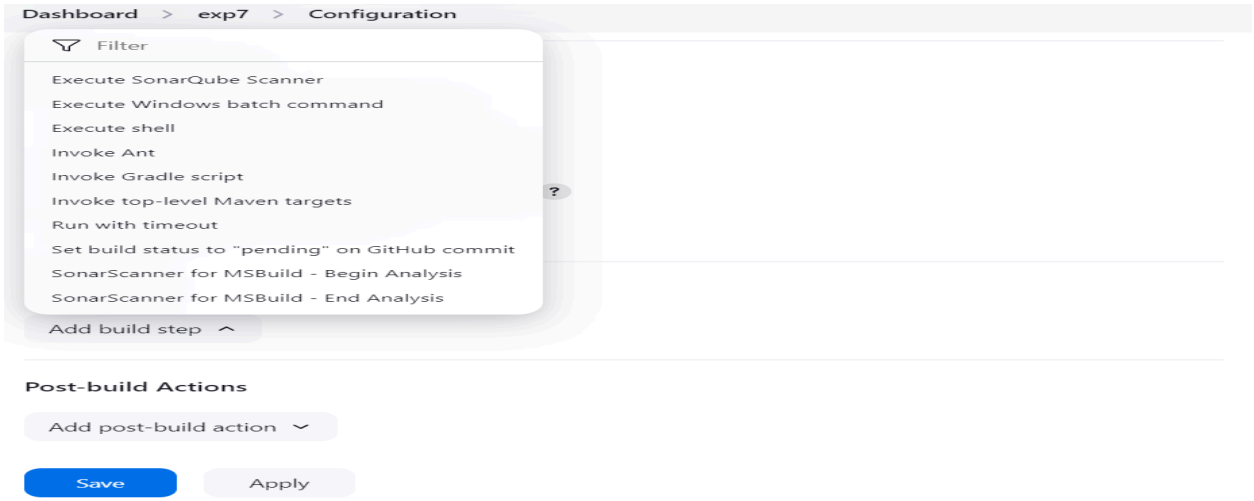
https://github.com/shazforiot/MSBuild_firstproject.git

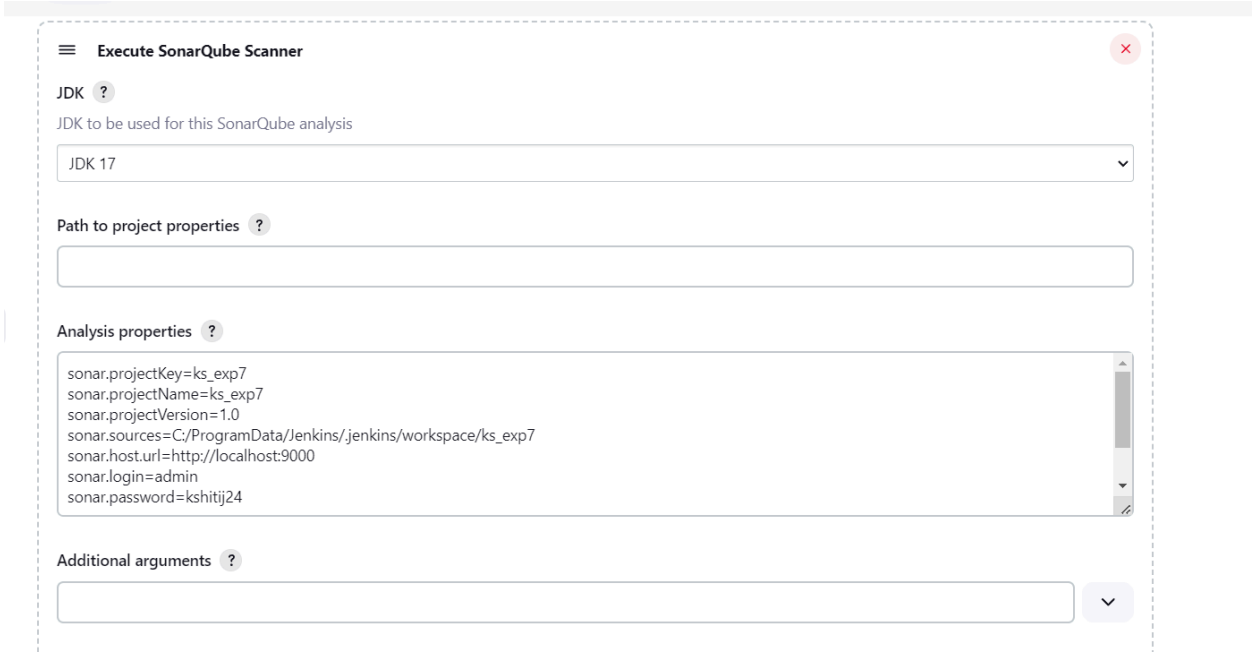
It is a sample hello-world project with no vulnerabilities and issues, just to test the integration.



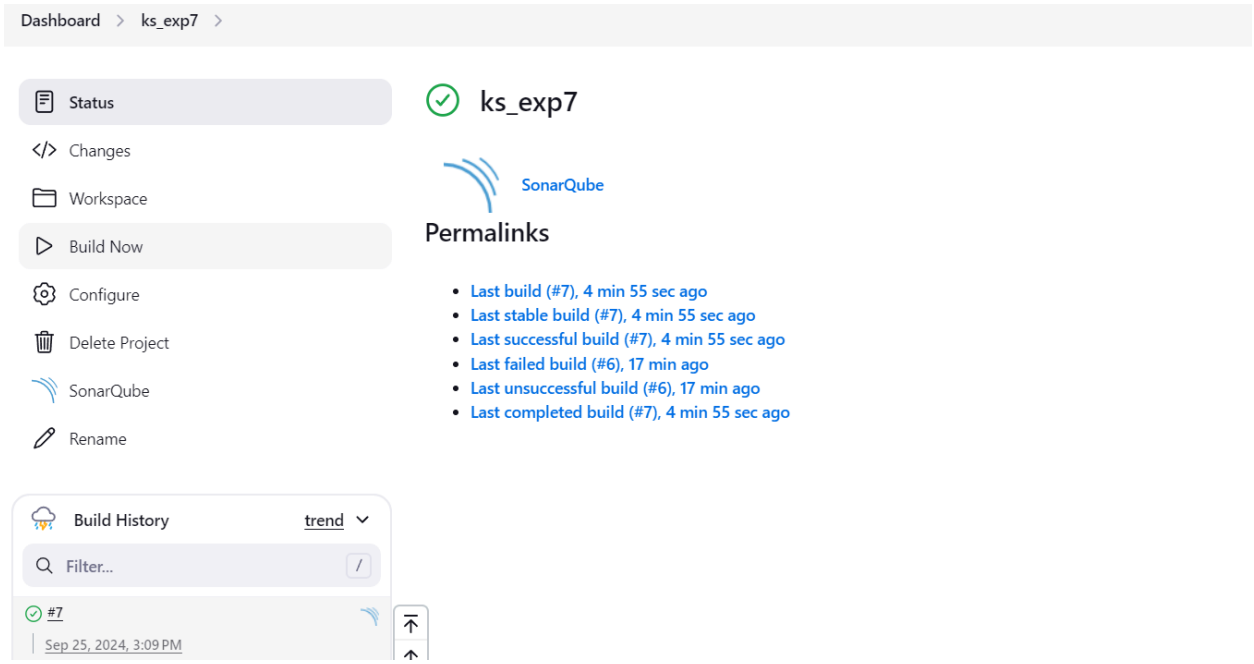
Name:Kshitij Hundre Div:D15C Roll No.18

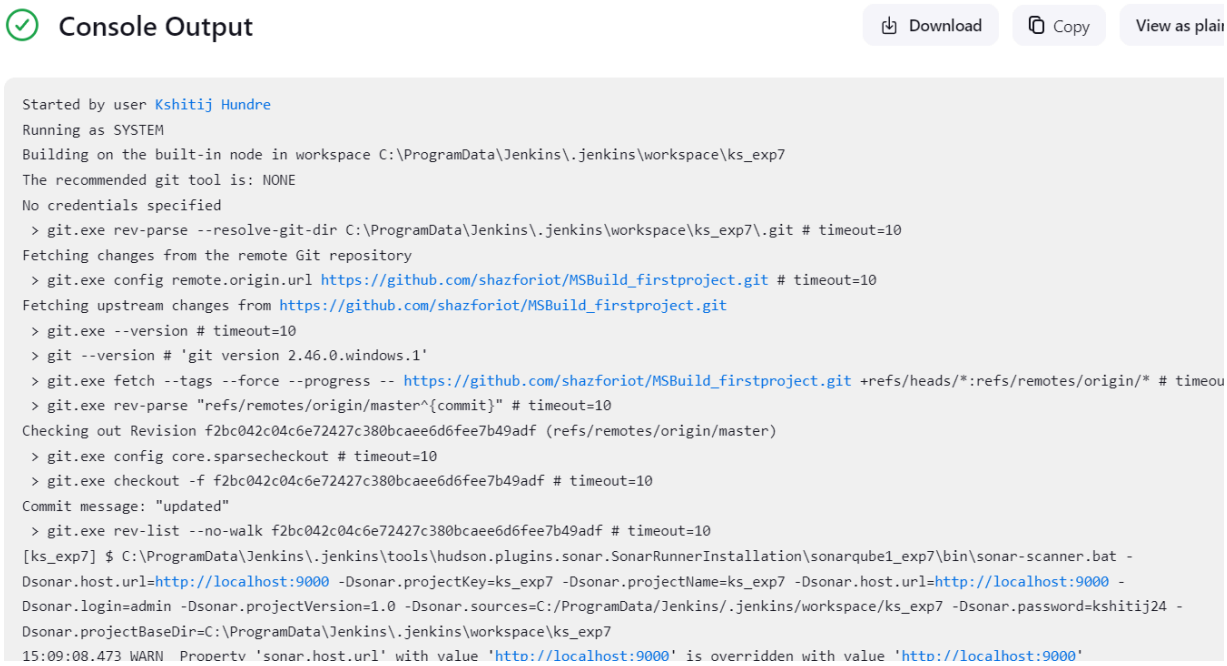
10. Under **Select project → Configuration → Build steps → Execute SonarQube Scanner**, enter these Analysis properties. Mention the SonarQube Project Key, Login, Password, Source path and Host URL.





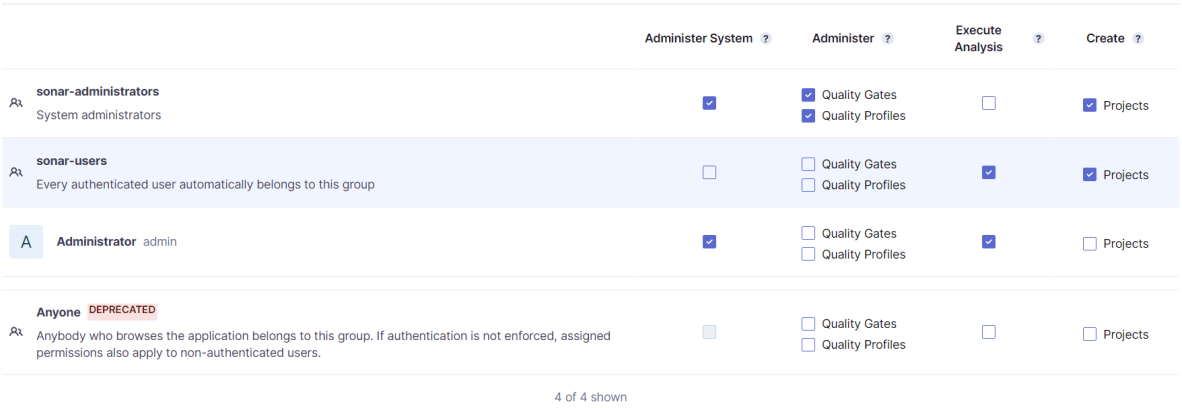
Name:Kshitij Hundre Div:D15C Roll No.18



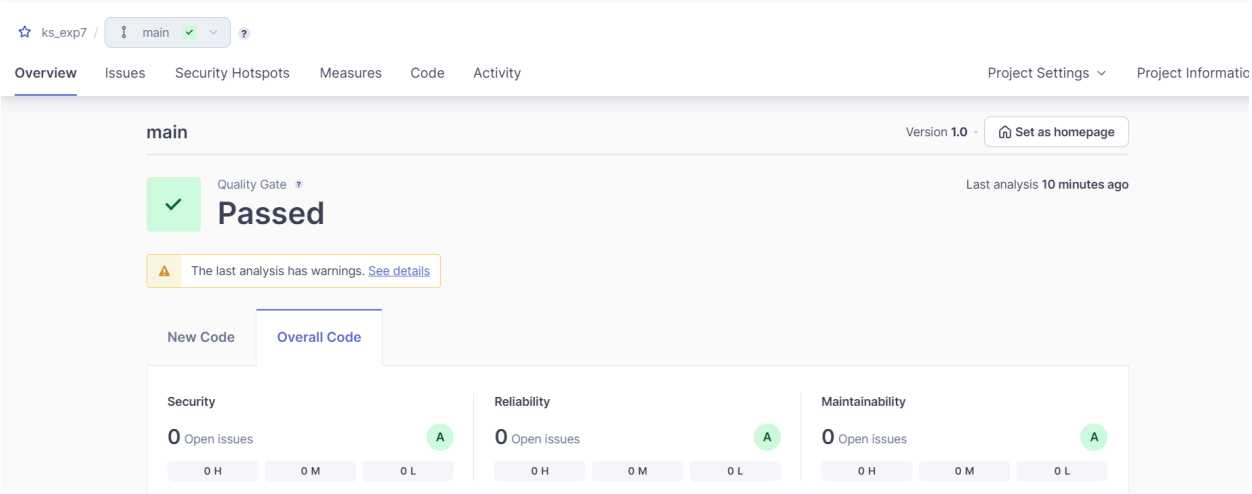


11. Go to http://localhost:9000/<user_name>/permissions and allow Execute Permissions to the Admin user.

Name:Kshitij Hundre Div:D15C Roll No.18



13. Once the build is complete, check project on SonarQube



In this way, we have integrated Jenkins with SonarQube for SAST.

Conclusion:

In this project, we integrated Jenkins with SonarQube for automated static application security testing (SAST). We set up SonarQube using Docker, configured Jenkins with the necessary plugins and authentication, and linked it to a GitHub repository. The SonarQube scanner was added as a build step, enabling continuous code analysis for vulnerabilities, code smells, and quality issues, ensuring automated reporting and continuous code quality improvement.

Expt No. 08 Advanced DevOps Lab

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:

What is SAST?

Static application security testing (SAST), or static analysis, is a testing methodology that analyzes source code to find security vulnerabilities that make your organization's applications susceptible to attack. SAST scans an application before the code is compiled. It's also known as white box testing.

What problems does SAST solve?

SAST takes place very early in the software development life cycle (SDLC) as it does not require a working application and can take place without code being executed. It helps developers identify vulnerabilities in the initial stages of development and quickly resolve issues without breaking builds or passing on vulnerabilities to the final release of the application.

SAST tools give developers real-time feedback as they code, helping them fix issues before they pass the code to the next phase of the SDLC. This prevents security-related issues from being considered an afterthought. SAST tools also provide graphical representations of the issues found, from source to sink. These help you navigate the code easier. Some tools point out the exact location of vulnerabilities and highlight the risky code. Tools can also provide in-depth guidance on how to fix issues and the best place in the code to fix them, without requiring deep security domain expertise.

It's important to note that SAST tools must be run on the application on a regular basis, such as during daily/monthly builds, every time code is checked in, or during a code release.

Why is SAST important?

Developers dramatically outnumber security staff. It can be challenging for an organization to find the resources to perform code reviews on even a fraction of its applications. A key strength of SAST tools is the ability to analyze 100% of the codebase. Additionally, they are much faster

than manual secure code reviews performed by humans. These tools can scan millions of lines of code in a matter of minutes. SAST tools automatically identify critical vulnerabilities—such as buffer overflows, SQL injection, cross-site scripting, and others—with high confidence.

What is a CI/CD Pipeline?

CI/CD pipeline refers to the Continuous Integration/Continuous Delivery pipeline. Before we dive deep into this segment, let's first understand what is meant by the term 'pipeline'?

A pipeline is a concept that introduces a series of events or tasks that are connected in a sequence to make quick software releases. For example, there is a task, that task has got five different stages, and each stage has got some steps. All the steps in phase one have to be completed, to mark the latter stage to be complete.

Now, consider the CI/CD pipeline as the backbone of the DevOps approach. This Pipeline is responsible for building codes, running tests, and deploying new software versions. The Pipeline executes the job in a defined manner by first coding it and then structuring it inside several blocks that may include several steps or tasks.

What is SonarQube?

SonarQube is an open-source platform developed by SonarSource for continuous inspection of code quality. Sonar does static code analysis, which provides a detailed report of bugs, code smells, vulnerabilities, code duplications.
It supports 25+ major programming languages through built-in rulesets and can also be extended with various plugins.

Benefits of SonarQube

- **Sustainability -** Reduces complexity, possible vulnerabilities, and code duplications, optimising the life of applications.
- **Increase productivity -** Reduces the scale, cost of maintenance, and risk of the application; as such, it removes the need to spend more time changing the code
- **Quality code -** Code quality control is an inseparable part of the process of software development.
- **Detect Errors -** Detects errors in the code and alerts developers to fix them automatically before submitting them for output.
- **Increase consistency -** Determines where the code criteria are breached and enhances the quality
- **Business scaling -** No restriction on the number of projects to be evaluated
- **Enhance developer skills -** Regular feedback on quality problems helps developers to improve their coding skills

Integrating Jenkins with SonarQube:

Prerequisites:

- Jenkins installed
- Docker Installed (for SonarQube)
- SonarQube Docker Image

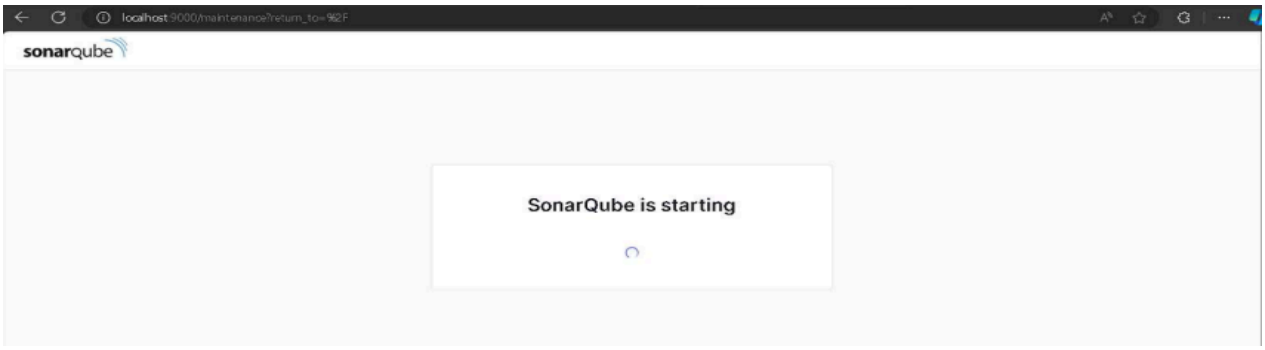
Steps to create a Jenkins CI/CD Pipeline and use SonarQube to perform SAST

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

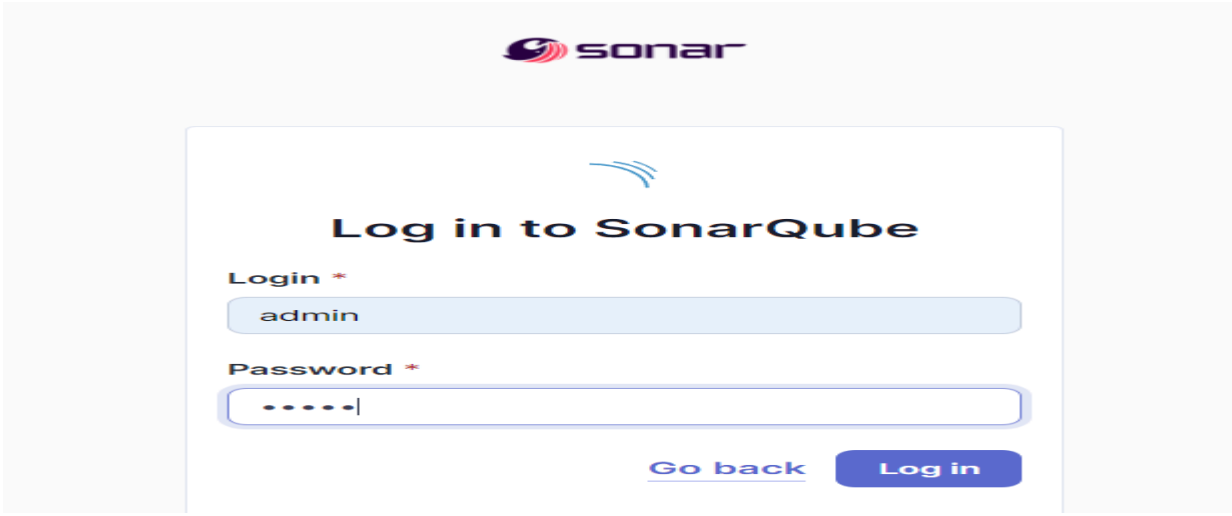
2. Run SonarQube in a Docker container using this command -

```
PS C:\Users\91773\Desktop\College Resources\Advdevops Exp8> docker run -d --name sonarqube2 -e SONAR_ES_BOOTSTRAP_CHECKS
_DISABLE=true -p 9000:9000 sonarqube:latest
71fc67f0b15baa5be5bdcdd66966938e18682683d020beadcbc909dd027cfe7a
PS C:\Users\91773\Desktop\College Resources\Advdevops Exp8>
```

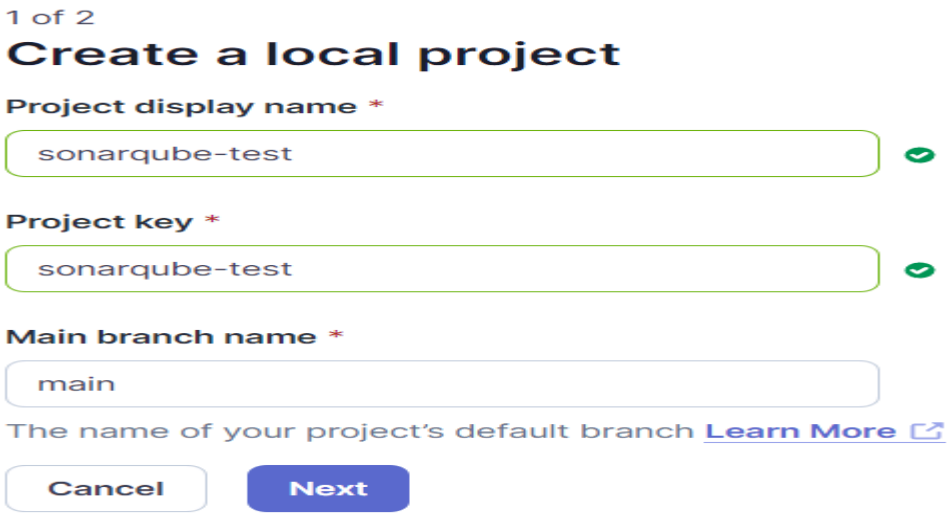
3. Once the container is up and running, you can check the status of SonarQube at localhost port 9000.



4. Login to SonarQube using username *admin* and password *admin*.

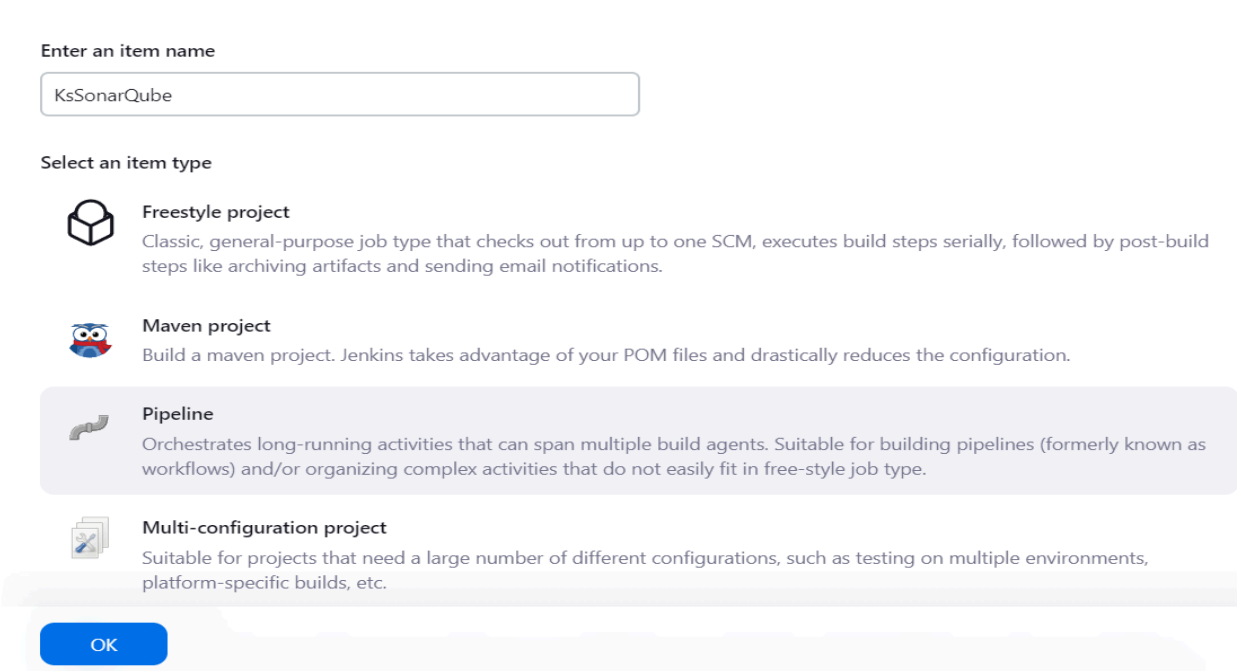


5. Create a manual project in SonarQube with the name **sonarqube-test**



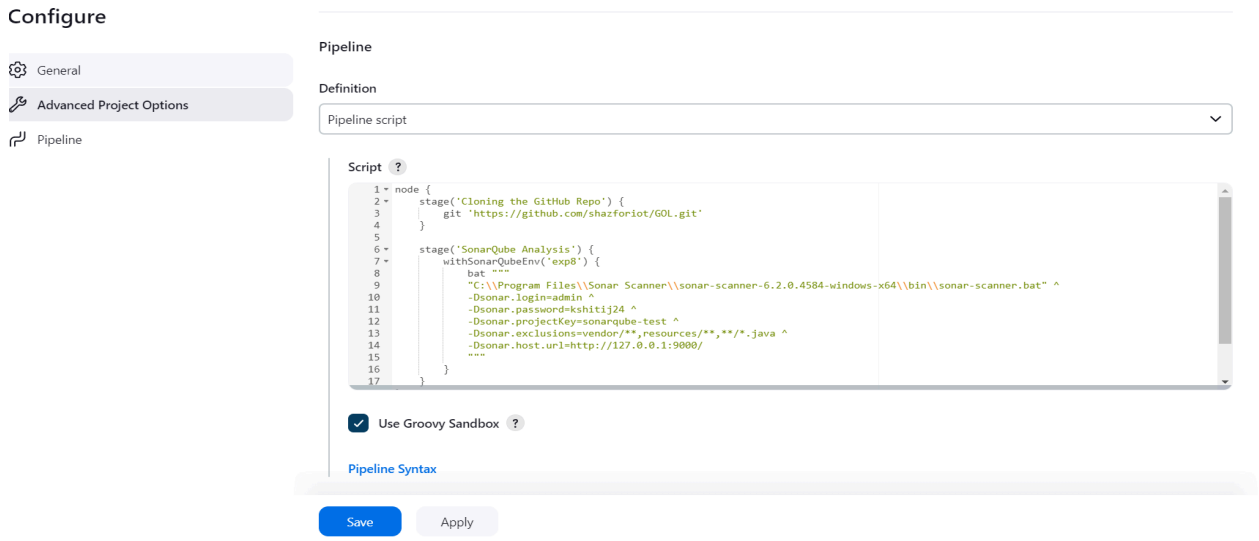
Setup the project and come back to Jenkins Dashboard.

6. Create a New Item in Jenkins, choose **Pipeline**.



7. Under Pipeline Script, enter the following -

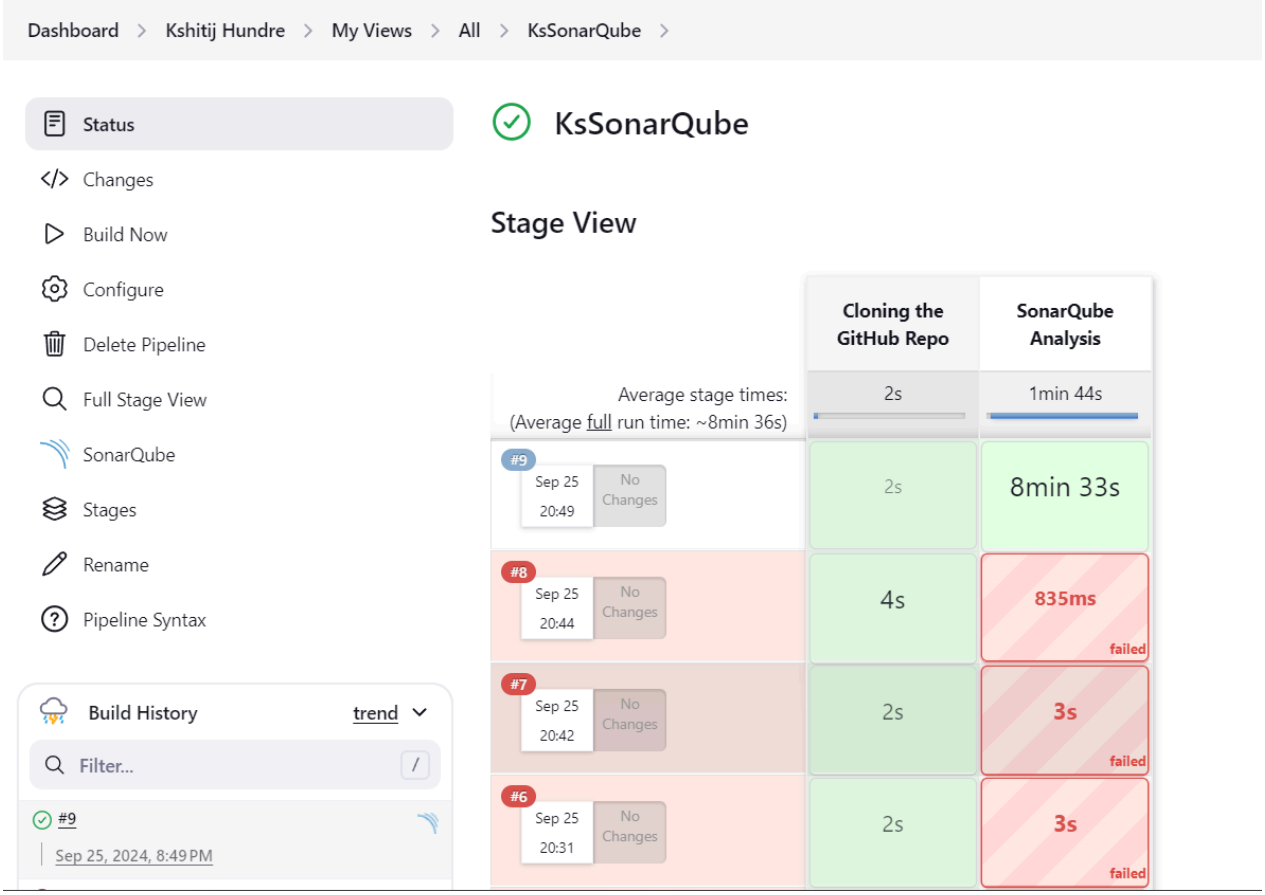
```
node {
  stage('Cloning the GitHub Repo') {
    git 'https://github.com/shazforiot/GOL.git'
  }
  stage('SonarQube analysis') {
    withSonarQubeEnv('sonarqube') {
      sh "<PATH_TO_SONARQUBE_FOLDER>//bin//sonar-scanner \
      -D sonar.login=<SonarQube_USERNAME> \
      -D sonar.password=<SonarQube_PASSWORD> \
      -D sonar.projectKey=<Project_KEY> \
-D sonar.exclusions=vendor/**,resources/**,**/*.java \
      -D sonar.host.url=http://127.0.0.1:9000/"
    }
  }
}
```

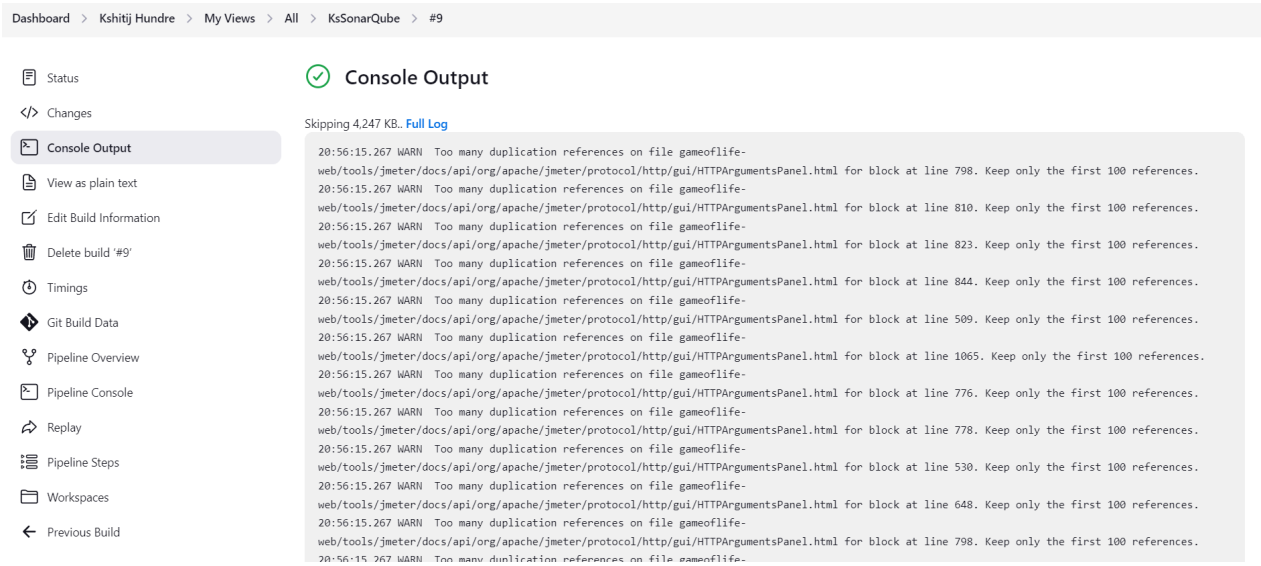


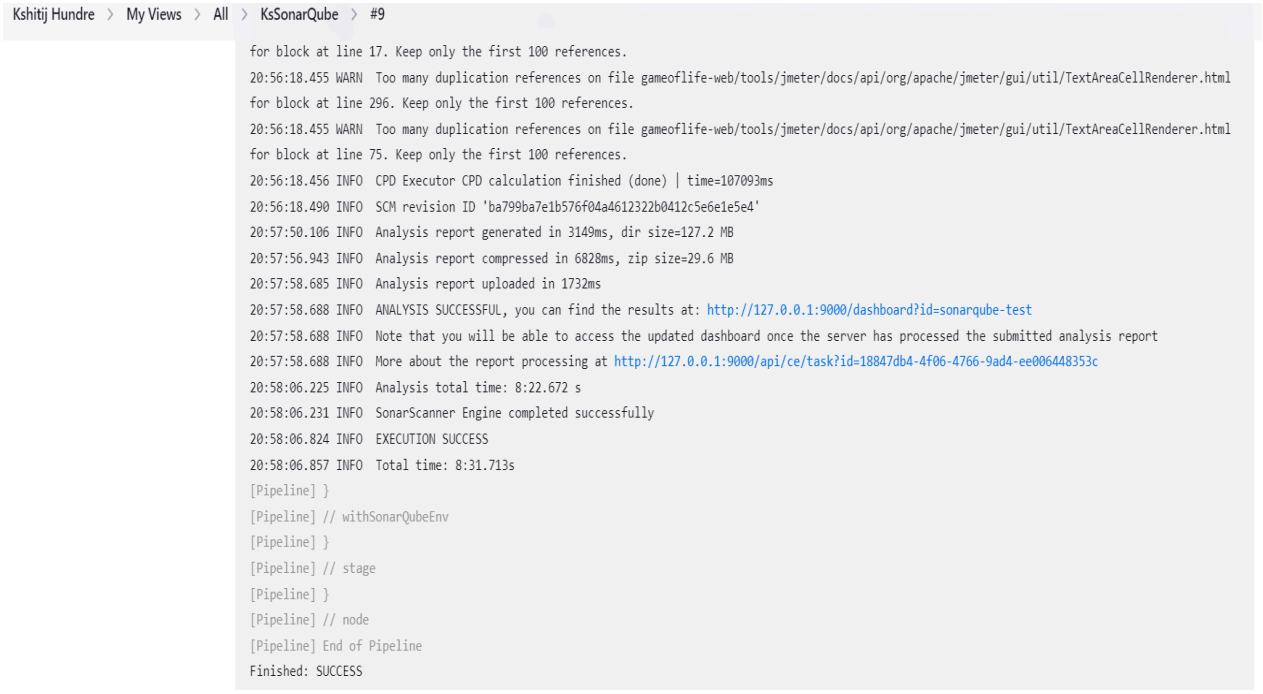
It is a java sample project which has a lot of repetitions and issues that will be detected by SonarQube.

8. Run The Build.

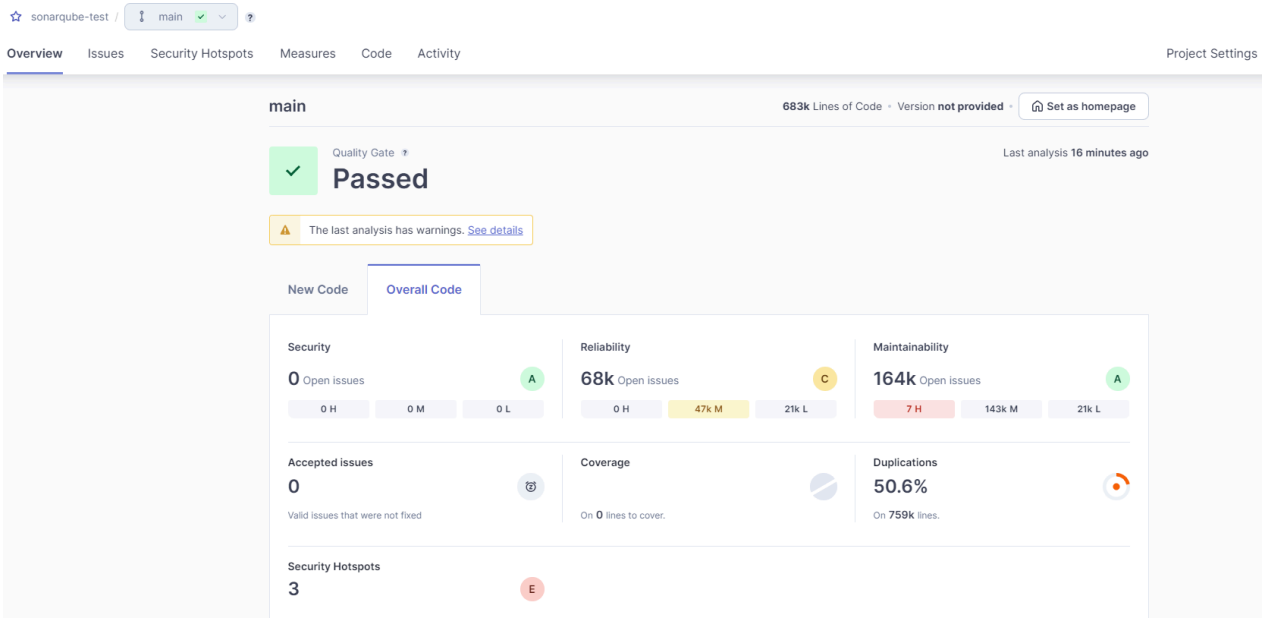
9. Check the console output once the build is complete.


Dashboard > Kshitij Hundre > My Views > All > KsSonarQube >
Status
Changes
Build Now
Configure
Delete Pipeline
Full Stage View
SonarQube
Stages
Rename
Pipeline Syntax
Build History
trend
Filter...
#9
Sep 25, 2024, 8:49 PM
KsSonarQube
Stage View
Cloning the GitHub Repo
SonarQube Analysis
Average stage times:
(Average full run time: ~8min 36s)
2s
1min 44s
#9
Sep 25
20:49
No Changes
2s
8min 33s
#8
Sep 25
20:44
No Changes
4s
835ms
failed
#7
Sep 25
20:42
No Changes
2s
3s
failed
#6
Sep 25
20:31
No Changes
2s
3s
failed


Dashboard > Kshitij Hundre > My Views > All > KsSonarQube > #9
Status
Changes
Console Output
View as plain text
Edit Build Information
Delete build '#9'
Timings
Git Build Data
Pipeline Overview
Pipeline Console
Replay
Pipeline Steps
Workspaces
Previous Build
Console Output
Skipping 4,247 KB.. Full Log
20:56:15.267 WARN Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/protocol/http/gui/HTTPArgumentsPanel.html for block at line 798. Keep only the first 100 references.
20:56:15.267 WARN Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/protocol/http/gui/HTTPArgumentsPanel.html for block at line 810. Keep only the first 100 references.
20:56:15.267 WARN Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/protocol/http/gui/HTTPArgumentsPanel.html for block at line 823. Keep only the first 100 references.
20:56:15.267 WARN Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/protocol/http/gui/HTTPArgumentsPanel.html for block at line 844. Keep only the first 100 references.
20:56:15.267 WARN Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/protocol/http/gui/HTTPArgumentsPanel.html for block at line 509. Keep only the first 100 references.
20:56:15.267 WARN Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/protocol/http/gui/HTTPArgumentsPanel.html for block at line 1065. Keep only the first 100 references.
20:56:15.267 WARN Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/protocol/http/gui/HTTPArgumentsPanel.html for block at line 776. Keep only the first 100 references.
20:56:15.267 WARN Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/protocol/http/gui/HTTPArgumentsPanel.html for block at line 778. Keep only the first 100 references.
20:56:15.267 WARN Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/protocol/http/gui/HTTPArgumentsPanel.html for block at line 530. Keep only the first 100 references.
20:56:15.267 WARN Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/protocol/http/gui/HTTPArgumentsPanel.html for block at line 648. Keep only the first 100 references.
20:56:15.267 WARN Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/protocol/http/gui/HTTPArgumentsPanel.html for block at line 798. Keep only the first 100 references.

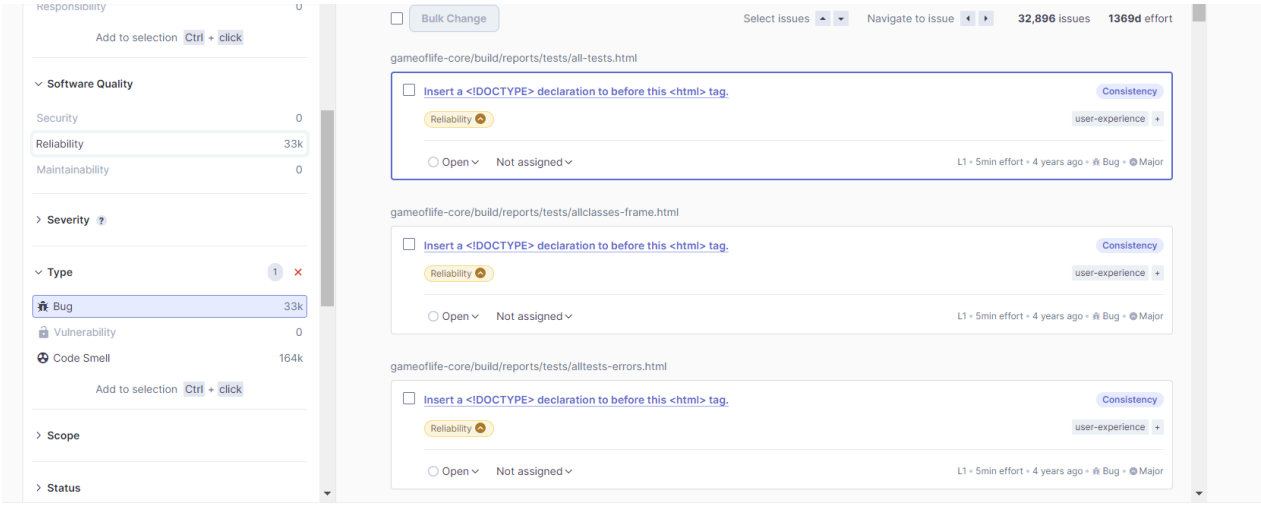


10. After that, check the project in SonarQube.

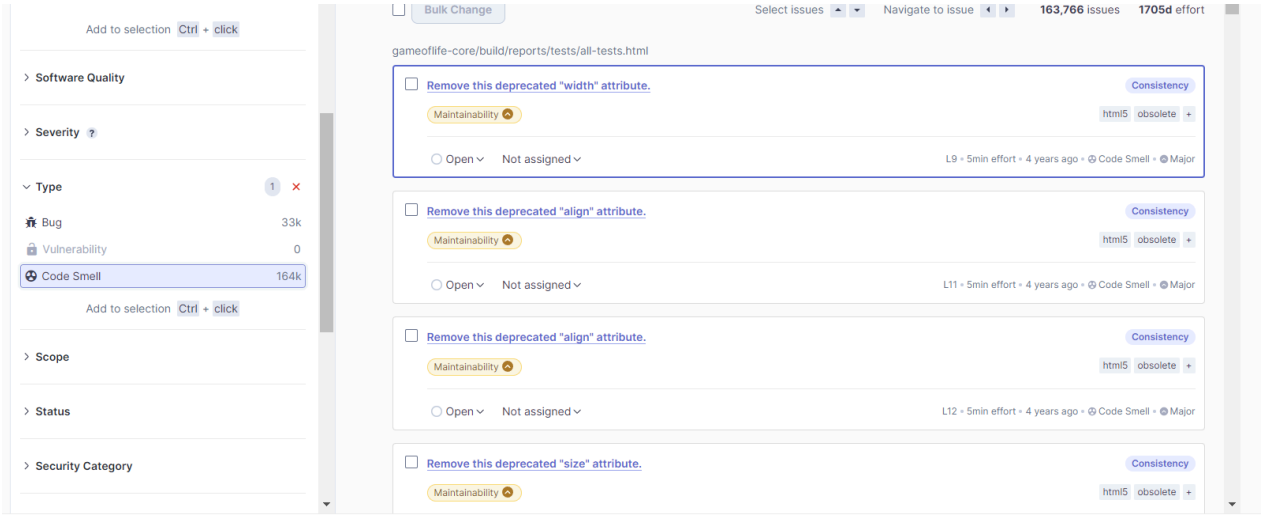


Under different tabs, check all different issues with the code.

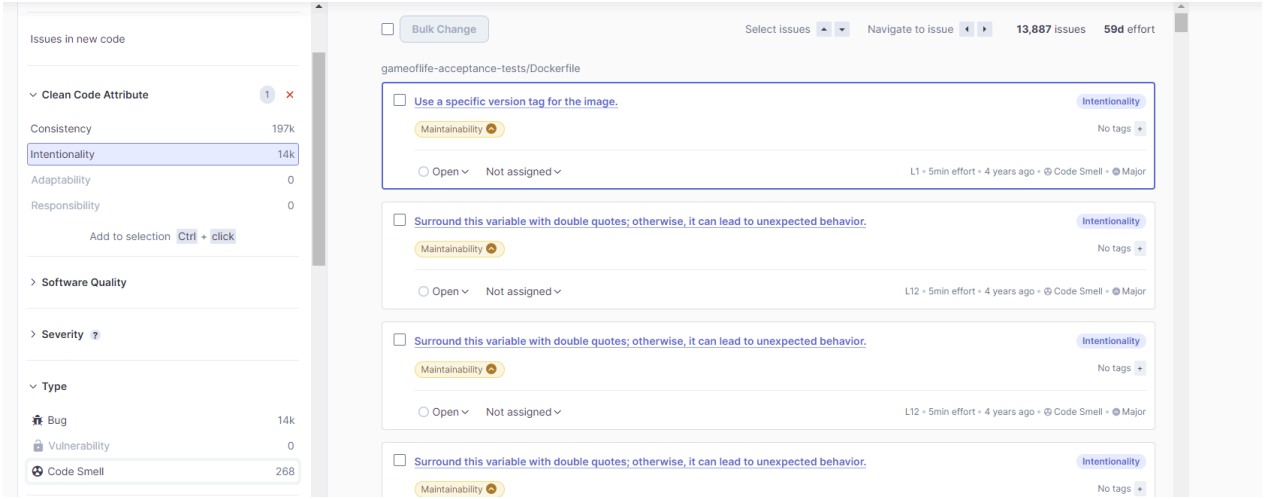
11. **Bugs**



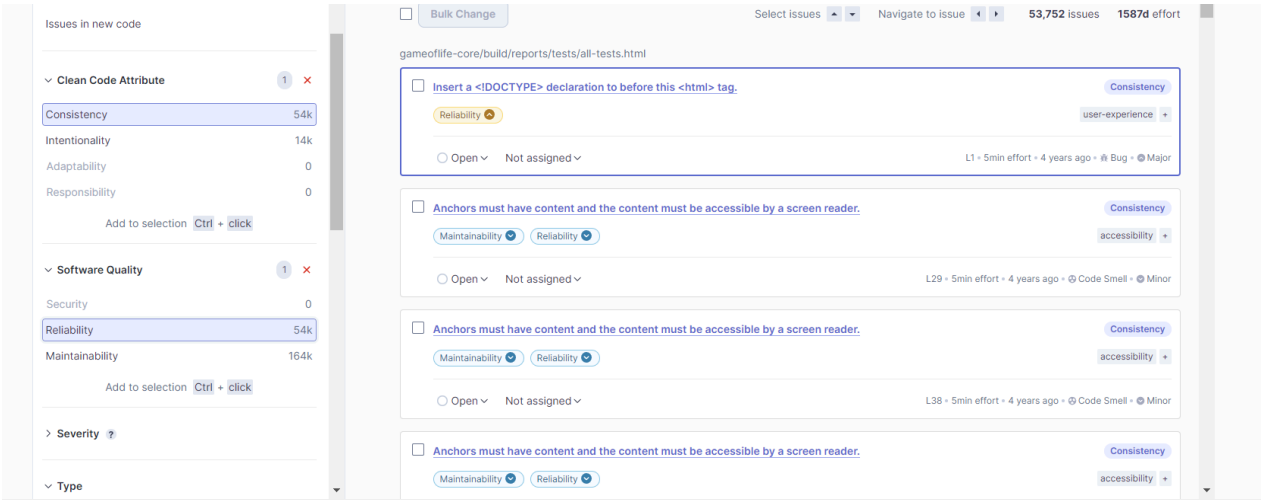
Code Smells



Intentional issues



Reliabilities issue



Duplicates

In this way, we have created a CI/CD Pipeline with Jenkins and integrated it with SonarQube to find issues in the code like bugs, code smells, duplicates, cyclomatic complexities, etc.

Conclusion:

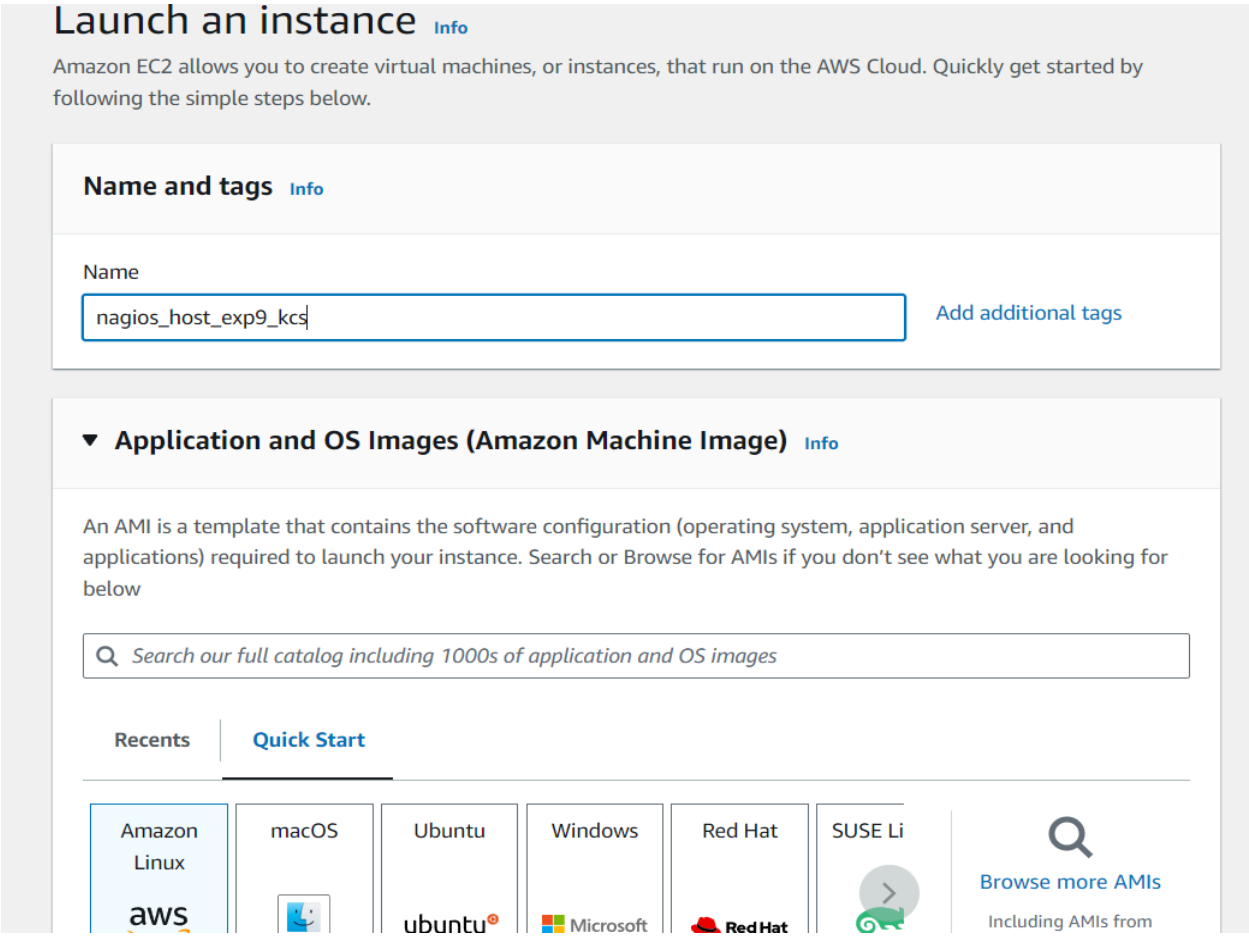
In this experiment, we integrated Jenkins with SonarQube to enable automated code quality checks within our CI/CD pipeline. We started by deploying SonarQube using Docker, setting up a project, and configuring it to analyze code quality. Next, we configured Jenkins by installing the SonarQube Scanner plugin, adding SonarQube server details, and setting up the scanner tool. We then developed a Jenkins pipeline to automate the process of cloning a GitHub repository and running SonarQube analysis on the code. This integration helps ensure continuous monitoring of code quality, detecting issues such as bugs, code smells, and security vulnerabilities throughout the development process.

Name : Kshitij Hundre
Div: D15C
Roll No:18

Adv-Devops Exp 9

Steps to perform the experiment:

Step1) Create an EC2 instance.keep the settings as default.



Create a new key pair login and save the downloaded file in a folder of your local desktop.

Also create a new security group. In my case its name will **'launch-wizard-10'.**
Later we will edit rules of this security group.

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*

nagios_exp9 | Create new key pair

Network settings Info

Edit

Network | Info

vpc-0a482134962ed0c59

Subnet | Info

No preference (Default subnet in any availability zone)

Auto-assign public IP | Info

Enable

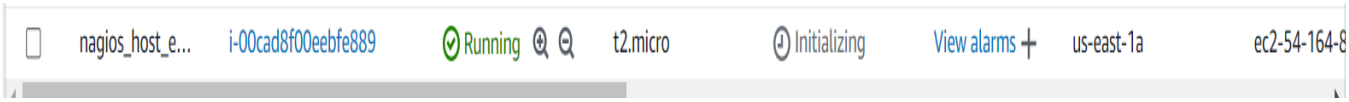
Additional charges apply when outside of free tier allowance

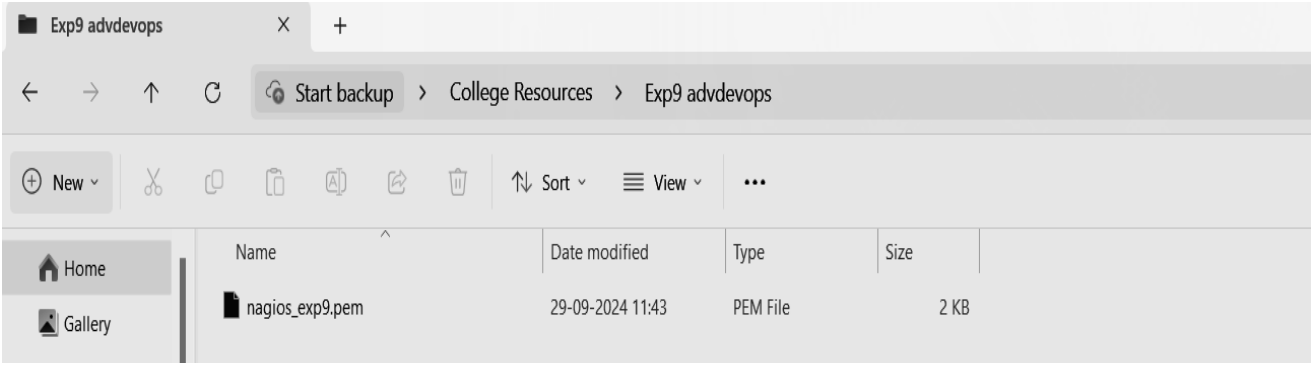
Firewall (security groups) | Info

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.

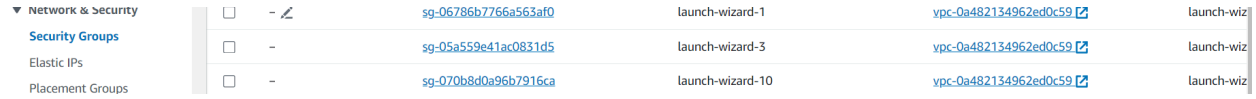
(•) Create security group ( ) Select existing security group

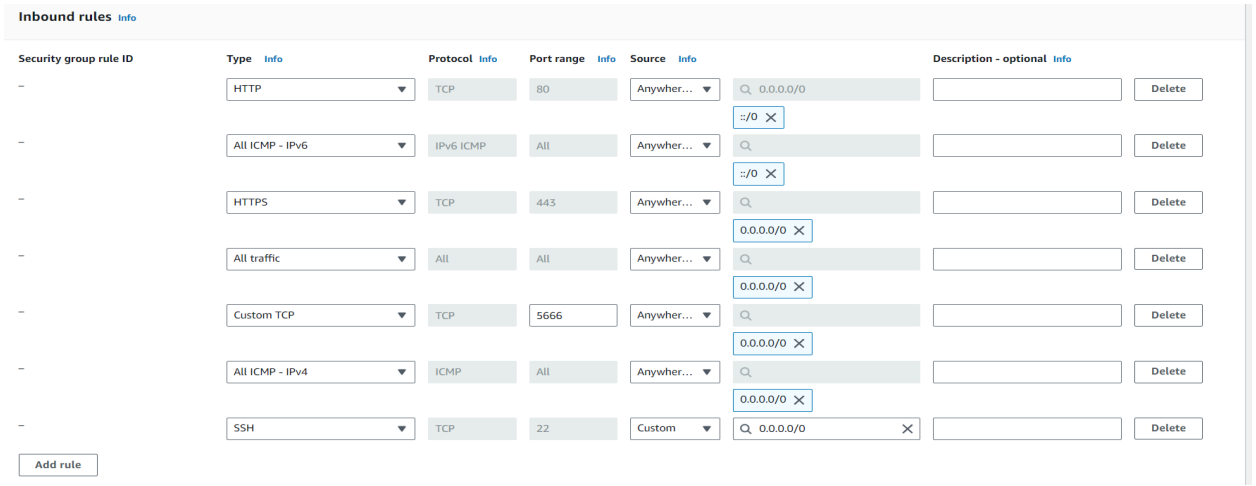
We'll create a new security group called '**launch-wizard-10**' with the following rules:

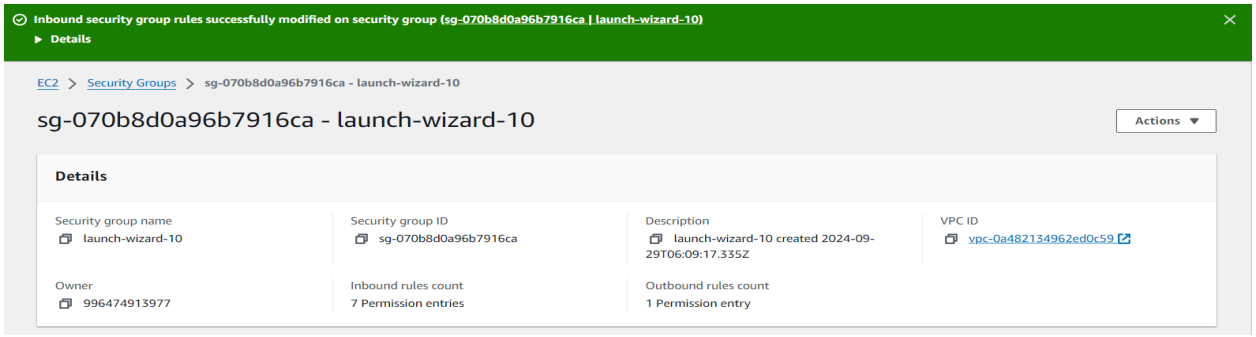




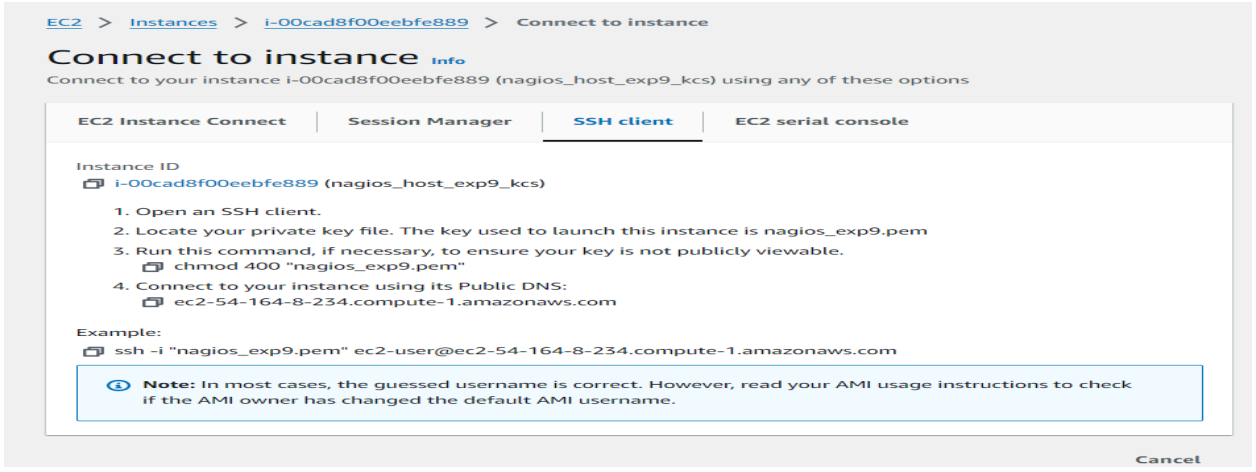
Now to edit security groups,select your security group and click on edit inbound rules.Add these security rules.







now navigate to instances, click on the instance which was created earlier and click on connect. now copy the ssh command and just replace the .pem file with its actual location in your computer.



ssh -i "nagios_exp9.pem" ec2-user@ec2-54-164-8-234.compute-1.amazonaws.com…paste this command in terminal..just replace your.pem file path.

```
PS C:\Users\91773\Desktop\College Resources\Exp9 advdevops> ssh -i "C:\Users\91773\Desktop\College Resources\Exp9 advdevops\nagios_exp9.pem" ec2-user@ec2-54-164-8-234.compute-1.amazonaws.com
The authenticity of host 'ec2-54-164-8-234.compute-1.amazonaws.com (54.164.8.234)' can't be established.
ED25519 key fingerprint is SHA256:scPnbaAMCV+FHEbUBCnwZdAjd6MQStqfeD/Rh06wDSY.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-54-164-8-234.compute-1.amazonaws.com' (ED25519) to the list of known hosts.
   ,     #_
   ~\_  ####_        Amazon Linux 2023
  ~~  \_#####\
  ~~     \###|
  ~~       \#/ ___   https://aws.amazon.com/linux/amazon-linux-2023
   ~~       V~' '->
    ~~~         /
      ~~._.   _/
         _/ _/
       _/m/'
```

sudo yum update

```
[ec2-user@ip-172-31-86-195 ~]$ sudo yum update
Last metadata expiration check: 0:32:14 ago on Sun Sep 29 06:16:51 2024.
Dependencies resolved.
Nothing to do.
Complete!
[ec2-user@ip-172-31-86-195 ~]$
```

sudo yum install httpd php
Select y when asked i prompt.

```
[ec2-user@ip-172-31-86-195 ~]$ sudo yum install httpd php
Last metadata expiration check: 0:33:12 ago on Sun Sep 29 06:16:51 2024.
Dependencies resolved.
================================================================================================================================
 Package                         Architecture       Version                              Repository                Size
================================================================================================================================
Installing:
 httpd                           x86_64             2.4.62-1.amzn2023                    amazonlinux               48 k
 php8.3                          x86_64             8.3.10-1.amzn2023.0.1                amazonlinux               10 k
Installing dependencies:
 apr                             x86_64             1.7.2-2.amzn2023.0.2                 amazonlinux              129 k
 apr-util                        x86_64             1.6.3-1.amzn2023.0.1                 amazonlinux               98 k
 generic-logos-httpd             noarch             18.0.0-12.amzn2023.0.3               amazonlinux               19 k
 httpd-core                      x86_64             2.4.62-1.amzn2023                    amazonlinux              1.4 M
 httpd-filesystem                noarch             2.4.62-1.amzn2023                    amazonlinux               14 k
 httpd-tools                     x86_64             2.4.62-1.amzn2023                    amazonlinux               81 k
 libbrotli                       x86_64             1.0.9-4.amzn2023.0.2                 amazonlinux              315 k
 libsodium                       x86_64             1.0.19-4.amzn2023                    amazonlinux              176 k
 libxslt                         x86_64             1.1.34-5.amzn2023.0.2                amazonlinux              241 k
 mailcap                         noarch             2.1.49-3.amzn2023.0.3                amazonlinux               33 k
 nginx-filesystem                noarch             1:1.24.0-1.amzn2023.0.4              amazonlinux              9.8 k
 php8.3-cli                      x86_64             8.3.10-1.amzn2023.0.1                amazonlinux              3.7 M
 php8.3-common                   x86_64             8.3.10-1.amzn2023.0.1                amazonlinux              737 k
 php8.3-process                  x86_64             8.3.10-1.amzn2023.0.1                amazonlinux               45 k
 php8.3-xml                      x86_64             8.3.10-1.amzn2023.0.1                amazonlinux              154 k
Installing weak dependencies:
 apr-util-openssl                x86_64             1.6.3-1.amzn2023.0.1                 amazonlinux               17 k
```

sudo yum install gcc glibc glibc-common

```
[ec2-user@ip-172-31-86-195 ~]$ sudo yum install gcc glibc glibc-common
Last metadata expiration check: 0:35:10 ago on Sun Sep 29 06:16:51 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       Version                                 Repository                 Size
================================================================================================================================
Installing:
 gcc                          x86_64             11.4.1-2.amzn2023.0.2                   amazonlinux                32 M
Installing dependencies:
 annobin-docs                 noarch             10.93-1.amzn2023.0.1                    amazonlinux                92 k
 annobin-plugin-gcc           x86_64             10.93-1.amzn2023.0.1                    amazonlinux               887 k
 cpp                          x86_64             11.4.1-2.amzn2023.0.2                   amazonlinux                10 M
 gc                           x86_64             8.0.4-5.amzn2023.0.2                    amazonlinux               105 k
 glibc-devel                  x86_64             2.34-52.amzn2023.0.11                   amazonlinux                27 k
 glibc-headers-x86            noarch             2.34-52.amzn2023.0.11                   amazonlinux               427 k
 guile22                      x86_64             2.2.7-2.amzn2023.0.3                    amazonlinux               6.4 M
 kernel-headers               x86_64             6.1.109-118.189.amzn2023                amazonlinux               1.4 M
 libmpc                       x86_64             1.2.1-2.amzn2023.0.2                    amazonlinux                62 k
 libtool-ltdl                 x86_64             2.4.7-1.amzn2023.0.3                    amazonlinux                38 k
 libxcrypt-devel              x86_64             4.4.33-7.amzn2023                       amazonlinux                32 k
 make                         x86_64             1:4.3-5.amzn2023.0.2                    amazonlinux               534 k
```

sudo yum install gd gd-devel

```
[ec2-user@ip-172-31-86-195 ~]$ sudo yum install gd gd-devel
Last metadata expiration check: 0:36:28 ago on Sun Sep 29 06:16:51 2024.
Dependencies resolved.
================================================================================================================================
 Package                      Architecture       Version                                 Repository                 Size
================================================================================================================================
Installing:
 gd                           x86_64             2.3.3-5.amzn2023.0.3                    amazonlinux               139 k
 gd-devel                     x86_64             2.3.3-5.amzn2023.0.3                    amazonlinux                38 k
Installing dependencies:
 brotli                       x86_64             1.0.9-4.amzn2023.0.2                    amazonlinux               314 k
 brotli-devel                 x86_64             1.0.9-4.amzn2023.0.2                    amazonlinux                31 k
 bzip2-devel                  x86_64             1.0.8-6.amzn2023.0.2                    amazonlinux               214 k
 cairo                        x86_64             1.17.6-2.amzn2023.0.1                   amazonlinux               684 k
 cmake-filesystem             x86_64             3.22.2-1.amzn2023.0.4                   amazonlinux                16 k
 fontconfig                   x86_64             2.13.94-2.amzn2023.0.2                  amazonlinux               273 k
 fontconfig-devel             x86_64             2.13.94-2.amzn2023.0.2                  amazonlinux               128 k
 fonts-filesystem             noarch             1:2.0.5-12.amzn2023.0.2                 amazonlinux               9.5 k
 freetype                     x86_64             2.13.2-5.amzn2023.0.1                   amazonlinux               423 k
 freetype-devel               x86_64             2.13.2-5.amzn2023.0.1                   amazonlinux               912 k
 glib2-devel                  x86_64             2.74.7-689.amzn2023.0.2                 amazonlinux               486 k
 google-noto-fonts-common     noarch             20201206-2.amzn2023.0.2                 amazonlinux                15 k
 google-noto-sans-vf-fonts    noarch             20201206-2.amzn2023.0.2                 amazonlinux               492 k
 graphite2                    x86_64             1.3.14-7.amzn2023.0.2                   amazonlinux                97 k
 graphite2-devel              x86_64             1.3.14-7.amzn2023.0.2                   amazonlinux                21 k
```

sudo adduser -m nagios
sudo passwd nagios

```
[ec2-user@ip-172-31-86-195 ~]$ sudo adduser -m nagios
sudo passwd nagios
Changing password for user nagios.
New password:
Retype new password:
passwd: all authentication tokens updated successfully.
```

sudo groupadd nagcmd

```
[ec2-user@ip-172-31-86-195 ~]$ sudo groupadd nagcmd
[ec2-user@ip-172-31-86-195 ~]$
```

sudo usermod -a -G nagcmd nagios
sudo usermod -a -G nagcmd apache

```
[ec2-user@ip-172-31-86-195 ~]$ sudo usermod -a -G nagcmd nagios
sudo usermod -a -G nagcmd apache
[ec2-user@ip-172-31-86-195 ~]$
```

mkdir ~/downloads
cd ~/downloads

```
[ec2-user@ip-172-31-86-195 ~]$ mkdir ~/downloads
cd ~/downloads
[ec2-user@ip-172-31-86-195 downloads]$
```

wget https://assets.nagios.com/downloads/nagioscore/releases/nagios-4.5.5.tar.gz

```
[ec2-user@ip-172-31-86-195 downloads]$ wget https://assets.nagios.com/downloads/nagioscore/releases/nagios-4.5.5.tar.gz
--2024-09-29 07:22:16--  https://assets.nagios.com/downloads/nagioscore/releases/nagios-4.5.5.tar.gz
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: 2065473 (2.0M) [application/x-gzip]
Saving to: ‘nagios-4.5.5.tar.gz’

nagios-4.5.5.tar.gz           100%[=================================================>]   1.97M  6.21MB/s    in 0.3s

2024-09-29 07:22:17 (6.21 MB/s) - ‘nagios-4.5.5.tar.gz’ saved [2065473/2065473]

[ec2-user@ip-172-31-86-195 downloads]$
```

wget https://nagios-plugins.org/download/nagios-plugins-2.4.11.tar.gz

```
[ec2-user@ip-172-31-86-195 downloads]$ wget https://nagios-plugins.org/download/nagios-plugins-2.4.11.tar.gz
--2024-09-29 07:23:16--  https://nagios-plugins.org/download/nagios-plugins-2.4.11.tar.gz
Resolving nagios-plugins.org (nagios-plugins.org)... 45.56.123.251
Connecting to nagios-plugins.org (nagios-plugins.org)|45.56.123.251|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 2753049 (2.6M) [application/x-gzip]
Saving to: ‘nagios-plugins-2.4.11.tar.gz’

nagios-plugins-2.4.11.tar.gz  100%[=================================================>]   2.62M  6.58MB/s    in 0.4s

2024-09-29 07:23:16 (6.58 MB/s) - ‘nagios-plugins-2.4.11.tar.gz’ saved [2753049/2753049]
```

tar zxvf nagios-4.5.5.tar.gz

```
[ec2-user@ip-172-31-86-195 downloads]$ tar zxvf nagios-4.5.5.tar.gz
nagios-4.5.5/
nagios-4.5.5/.github/
nagios-4.5.5/.github/workflows/
nagios-4.5.5/.github/workflows/test.yml
nagios-4.5.5/.gitignore
nagios-4.5.5/CONTRIBUTING.md
nagios-4.5.5/Changelog
nagios-4.5.5/INSTALLING
nagios-4.5.5/LEGAL
nagios-4.5.5/LICENSE
nagios-4.5.5/Makefile.in
nagios-4.5.5/README.md
nagios-4.5.5/THANKS
nagios-4.5.5/UPGRADING
```

Now we have to first navigate to the nagios-4.5.5 folder in downloads.

- **commands to enter:**

  ls (verify whether nagios-4.5.5 exists).Then go inside nagios 4.5.5 using cd.

```
[ec2-user@ip-172-31-86-195 downloads]$ ls
nagios-4.5.5  nagios-4.5.5.tar.gz  nagios-plugins-2.4.11.tar.gz
[ec2-user@ip-172-31-86-195 downloads]$ cd nagios-4.5.5
[ec2-user@ip-172-31-86-195 nagios-4.5.5]$
```

we now have to install openssl dev library

The OpenSSL development library, or openssl-devel contains include files that help develop applications that use cryptographic algorithms and protocols

commands to enter:

sudo yum install openssl-devel

```
[ec2-user@ip-172-31-86-195 nagios-4.5.5]$ sudo yum install openssl-devel
Last metadata expiration check: 1:13:15 ago on Sun Sep 29 06:16:51 2024.
Dependencies resolved.
================================================================================================================================
 Package                    Architecture         Version                              Repository                  Size
================================================================================================================================
Installing:
 openssl-devel              x86_64               1:3.0.8-1.amzn2023.0.14              amazonlinux                3.0 M

Transaction Summary
================================================================================================================================
Install  1 Package

Total download size: 3.0 M
Installed size: 4.7 M
Is this ok [y/N]: y
Downloading Packages:
openssl-devel-3.0.8-1.amzn2023.0.14.x86_64.rpm                                   31 MB/s | 3.0 MB     00:00
--------------------------------------------------------------------------------------------------------------------------------
Total                                                                            22 MB/s | 3.0 MB     00:00
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
  Preparing        :                                                                                              1/1
  Installing       : openssl-devel-1:3.0.8-1.amzn2023.0.14.x86_64                                                 1/1
  Running scriptlet: openssl-devel-1:3.0.8-1.amzn2023.0.14.x86_64                                                 1/1
  Verifying        : openssl-devel-1:3.0.8-1.amzn2023.0.14.x86_64                                                 1/1

Installed:
  openssl-devel-1:3.0.8-1.amzn2023.0.14.x86_64

Complete!
```

Then finally we can run the commands like usual.

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

```
[ec2-user@ip-172-31-86-195 nagios-4.5.5]$ ./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 the compiler supports GNU C... yes
checking whether gcc accepts -g... yes
checking for gcc option to enable C11 features... none needed
checking whether make sets $(MAKE)... yes
checking whether ln -s works... yes
checking for strip... /usr/bin/strip
checking for sys/wait.h that is POSIX.1 compatible... yes
checking for stdio.h... yes
checking for stdlib.h... yes
checking for string.h... yes
checking for inttypes.h... yes
checking for stdint.h... yes
checking for strings.h... yes
```

make all

```
[ec2-user@ip-172-31-86-195 nagios-4.5.5]$ make all
cd ./base && make
make[1]: Entering directory '/home/ec2-user/downloads/nagios-4.5.5/base'
gcc -Wall -I.. -I. -I../lib -I../include  -I../include -I..  -g -O2  -DHAVE_CONFIG_H -DNSCORE -c -o nagios.o ./nagios.c
gcc -Wall -I.. -I. -I../lib -I../include  -I../include -I..  -g -O2  -DHAVE_CONFIG_H -DNSCORE -c -o broker.o broker.c
gcc -Wall -I.. -I. -I../lib -I../include  -I../include -I..  -g -O2  -DHAVE_CONFIG_H -DNSCORE -c -o nebmods.o nebmods.c
gcc -Wall -I.. -I. -I../lib -I../include  -I../include -I..  -g -O2  -DHAVE_CONFIG_H -DNSCORE -c -o ../common/shared.o ../common/shared.c
gcc -Wall -I.. -I. -I../lib -I../include  -I../include -I..  -g -O2  -DHAVE_CONFIG_H -DNSCORE -c -o query-handler.o query-handler.c
gcc -Wall -I.. -I. -I../lib -I../include  -I../include -I..  -g -O2  -DHAVE_CONFIG_H -DNSCORE -c -o workers.o workers.c
```

```
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

************************************************************

Enjoy.
```

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

```
[ec2-user@ip-172-31-86-195 nagios-4.5.5]$ sudo make install
sudo make install-init
sudo make install-config
sudo make install-commandmode
cd ./base && make install
make[1]: Entering directory '/home/ec2-user/downloads/nagios-4.5.5/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-4.5.5/base'
cd ./cgi && make install
make[1]: Entering directory '/home/ec2-user/downloads/nagios-4.5.5/cgi'
```

Now the next command will take us to nano editor:
sudo nano /usr/local/nagios/etc/objects/contacts.cfg

```
  GNU nano 5.8                                  /usr/local/nagios/etc/objects/contacts.cfg
##############################################################################
# CONTACTS.CFG - SAMPLE CONTACT/CONTACTGROUP DEFINITIONS
#
#
# NOTES: This config file provides you with some example contact and contact
#        group definitions that you can reference in host and service
#        definitions.
#
#        You don't need to keep these definitions in a separate file from your
#        other object definitions.  This has been done just to make things
#        easier to understand.
#
##############################################################################



##############################################################################
#
# CONTACTS
#
##############################################################################

# 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                   nagios@localhost ; <****** CHANGE THIS TO YOUR EMAIL ADDRESS ******
}



##############################################################################

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

Change your email

```
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                   2022.kshitij.hundre@ves.ac.in ; <****** CHANGE THIS TO YOUR EMAIL ADDRESS ******
}



##############################################################################

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

Press crtl + O and enter
Press crtl + X

```
[ec2-user@ip-172-31-86-195 nagios-4.5.5]$ sudo nano /usr/local/nagios/etc/objects/contacts.cfg
[ec2-user@ip-172-31-86-195 nagios-4.5.5]$
```

sudo make install-webconf

```
[ec2-user@ip-172-31-86-195 nagios-4.5.5]$ 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 ***
```

Adding password for nagios admin

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

```
[ec2-user@ip-172-31-86-195 nagios-4.5.5]$ sudo htpasswd -c /usr/local/nagios/etc/htpasswd.users nagiosadmin
New password:
Re-type new password:
Adding password for user nagiosadmin
[ec2-user@ip-172-31-86-195 nagios-4.5.5]$
```

sudo service httpd restart

```
[ec2-user@ip-172-31-86-195 nagios-4.5.5]$ sudo service httpd restart
Redirecting to /bin/systemctl restart httpd.service
[ec2-user@ip-172-31-86-195 nagios-4.5.5]$
```

cd ~/downloads
tar zxvf nagios-plugins-2.4.11.tar.gz

```
[ec2-user@ip-172-31-86-195 nagios-4.5.5]$ cd ~/downloads
tar zxvf nagios-plugins-2.4.11.tar.gz
nagios-plugins-2.4.11/
nagios-plugins-2.4.11/build-aux/
nagios-plugins-2.4.11/build-aux/compile
nagios-plugins-2.4.11/build-aux/config.guess
nagios-plugins-2.4.11/build-aux/config.rpath
nagios-plugins-2.4.11/build-aux/config.sub
nagios-plugins-2.4.11/build-aux/install-sh
nagios-plugins-2.4.11/build-aux/ltmain.sh
nagios-plugins-2.4.11/build-aux/missing
```

cd nagios-plugins-2.4.11
./configure --with-nagios-user=nagios --with-nagios-group=nagios

```
[ec2-user@ip-172-31-86-195 downloads]$ cd nagios-plugins-2.4.11
./configure --with-nagios-user=nagios --with-nagios-group=nagios
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 make supports nested variables... yes
checking whether to enable maintainer-specific portions of Makefiles... yes
checking build system type... x86_64-pc-linux-gnu
checking host system type... x86_64-pc-linux-gnu
checking for gcc... gcc
```

sudo chkconfig --add nagios
sudo chkconfig nagios on

make
sudo make install

```
[ec2-user@ip-172-31-86-195 nagios-plugins-2.4.11]$ make
sudo make install
make  all-recursive
make[1]: Entering directory '/home/ec2-user/downloads/nagios-plugins-2.4.11'
Making all in gl
make[2]: Entering directory '/home/ec2-user/downloads/nagios-plugins-2.4.11/gl'
rm -f alloca.h-t alloca.h && \
{ echo '/* DO NOT EDIT! GENERATED AUTOMATICALLY! */'; \
  cat ./alloca.in.h; \
} > alloca.h-t && \
mv -f alloca.h-t alloca.h
rm -f c++defs.h-t c++defs.h && \
sed -n -e '/_GL_CXXDEFS/,$p' \
```

sudo chkconfig --add nagios
sudo chkconfig nagios on

```
[ec2-user@ip-172-31-86-195 nagios-plugins-2.4.11]$ sudo chkconfig --add nagios
sudo chkconfig nagios on
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/lib/systemd/system/nagios.service.
```

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

```
Running pre-flight check on configuration data...

Checking objects...
        Checked 8 services.
        Checked 1 hosts.
        Checked 1 host groups.
        Checked 0 service groups.
        Checked 1 contacts.
        Checked 1 contact groups.
        Checked 24 commands.
        Checked 5 time periods.
        Checked 0 host escalations.
        Checked 0 service escalations.
Checking for circular paths...
        Checked 1 hosts
        Checked 0 service dependencies
        Checked 0 host dependencies
        Checked 5 timeperiods
Checking global event handlers...
Checking obsessive compulsive processor commands...
Checking misc settings...

Total Warnings: 0
Total Errors:   0

Things look okay - No serious problems were detected during the pre-flight check
[ec2-user@ip-172-31-86-195 nagios-plugins-2.4.11]$
```

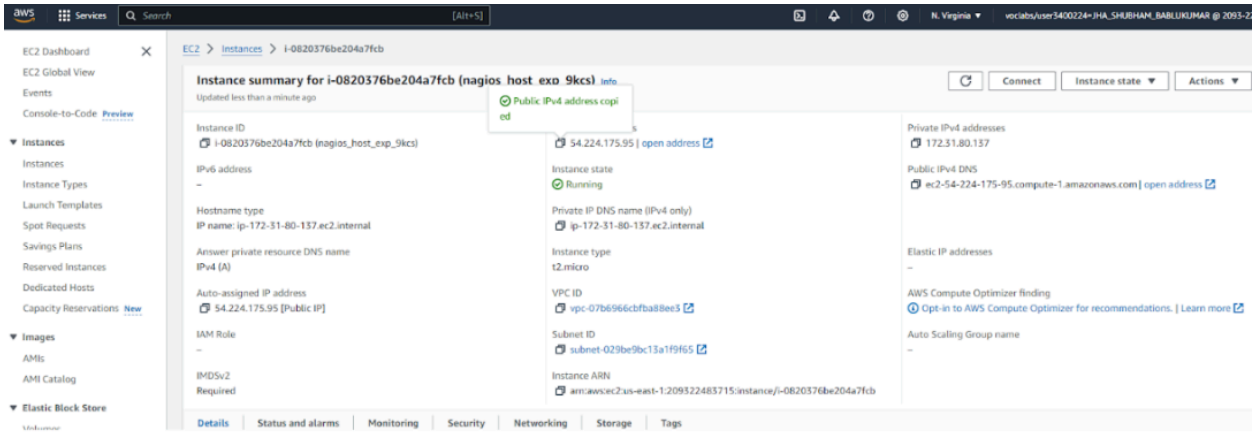
sudo service nagios start

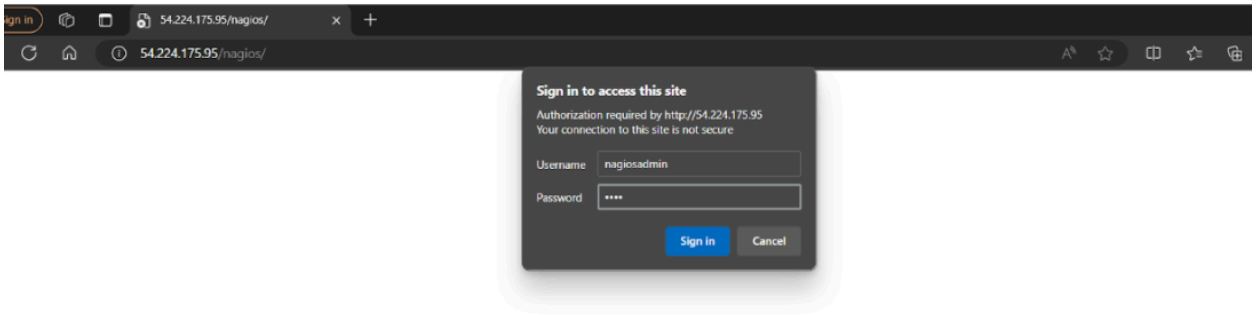
```
[ec2-user@ip-172-31-86-195 nagios-plugins-2.4.11]$ sudo service nagios start
Redirecting to /bin/systemctl start nagios.service
[ec2-user@ip-172-31-86-195 nagios-plugins-2.4.11]$
```

sudo systemctl status nagios

```
[ec2-user@ip-172-31-86-195 nagios-plugins-2.4.11]$ sudo systemctl status nagios
● nagios.service - Nagios Core 4.5.5
     Loaded: loaded (/usr/lib/systemd/system/nagios.service; enabled; preset: disabled)
     Active: active (running) since Sun 2024-09-29 08:00:47 UTC; 1min 5s ago
       Docs: https://www.nagios.org/documentation
    Process: 66625 ExecStartPre=/usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg (code=exited, status=0
    Process: 66626 ExecStart=/usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg (code=exited, status=0/SU
   Main PID: 66627 (nagios)
      Tasks: 6 (limit: 1112)
     Memory: 5.8M
        CPU: 90ms
     CGroup: /system.slice/nagios.service
             ├─66627 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg
             ├─66628 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─66629 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─66630 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─66631 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             └─66632 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg

Sep 29 08:00:47 ip-172-31-86-195.ec2.internal nagios[66627]: qh: Socket '/usr/local/nagios/var/rw/nagios.qh' successful
Sep 29 08:00:47 ip-172-31-86-195.ec2.internal nagios[66627]: qh: core query handler registered
Sep 29 08:00:47 ip-172-31-86-195.ec2.internal nagios[66627]: qh: echo service query handler registered
Sep 29 08:00:47 ip-172-31-86-195.ec2.internal nagios[66627]: qh: help for the query handler registered
Sep 29 08:00:47 ip-172-31-86-195.ec2.internal nagios[66627]: wproc: Successfully registered manager as @wproc with quer
Sep 29 08:00:47 ip-172-31-86-195.ec2.internal nagios[66627]: wproc: Registry request: name=Core Worker 66630;pid=66630
Sep 29 08:00:47 ip-172-31-86-195.ec2.internal nagios[66627]: wproc: Registry request: name=Core Worker 66631;pid=66631
Sep 29 08:00:47 ip-172-31-86-195.ec2.internal nagios[66627]: wproc: Registry request: name=Core Worker 66628;pid=66628
Sep 29 08:00:47 ip-172-31-86-195.ec2.internal nagios[66627]: wproc: Registry request: name=Core Worker 66629;pid=66629
Sep 29 08:00:48 ip-172-31-86-195.ec2.internal nagios[66627]: Successfully launched command file worker with pid 66632
lines 1-28/28 (END)
```

Now, go to EC2 instance and click on instance id. Then, click on the copy icon just before the public ip address on public IP.





Conclusion:

In this experiment, we successfully installed and configured Nagios Core, Nagios Plugins, and NRPE on a Linux machine for continuous monitoring. Nagios proves to be an essential tool in DevOps culture by detecting network and server issues in real-time, ensuring infrastructure health. Its scalability, security, and ability to send automated alerts enhance monitoring efficiency. By integrating NRPE, we extended monitoring to remote hosts, allowing proactive troubleshooting. Overall, Nagios is highly customizable with its plugin support and architecture, making it invaluable for maintaining service availability and operational stability.

Name: Kshitij Hundre
Div: D15C
Roll No:18

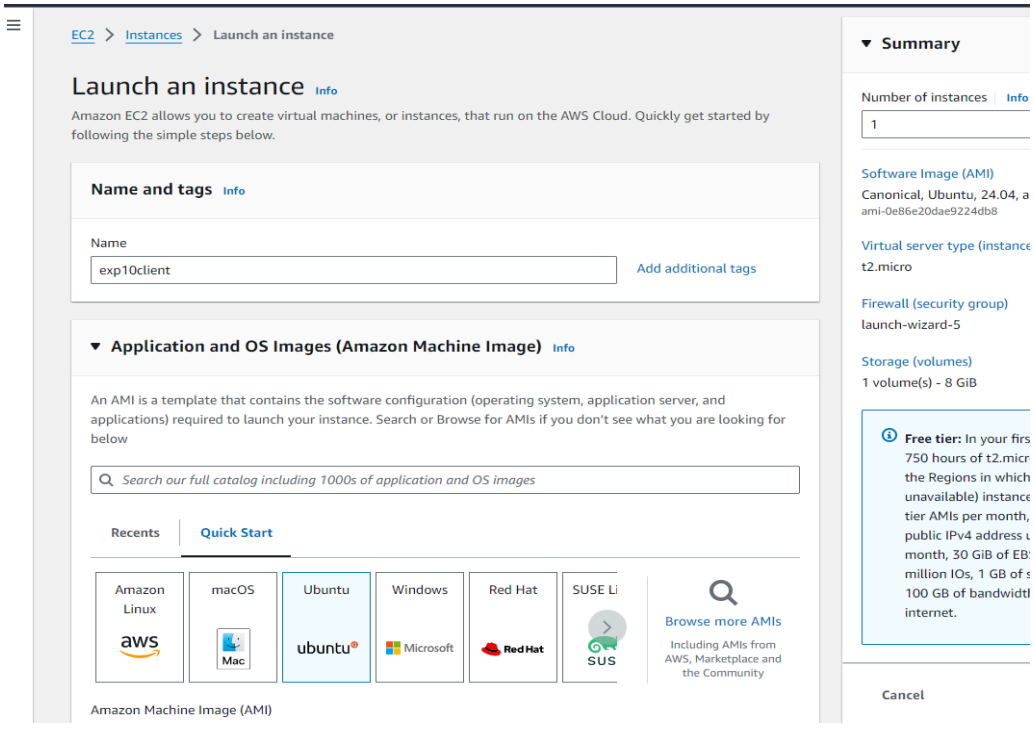
Adv Devops Exp:10

Aim :
The aim of this experiment is to set up and configure Nagios for comprehensive monitoring of ports, services, and both Windows and Linux servers. The objective is to ensure real-time monitoring, detect potential issues, and provide timely alerts for system administrators to take preventive or corrective actions, ensuring optimal system performance and uptime.

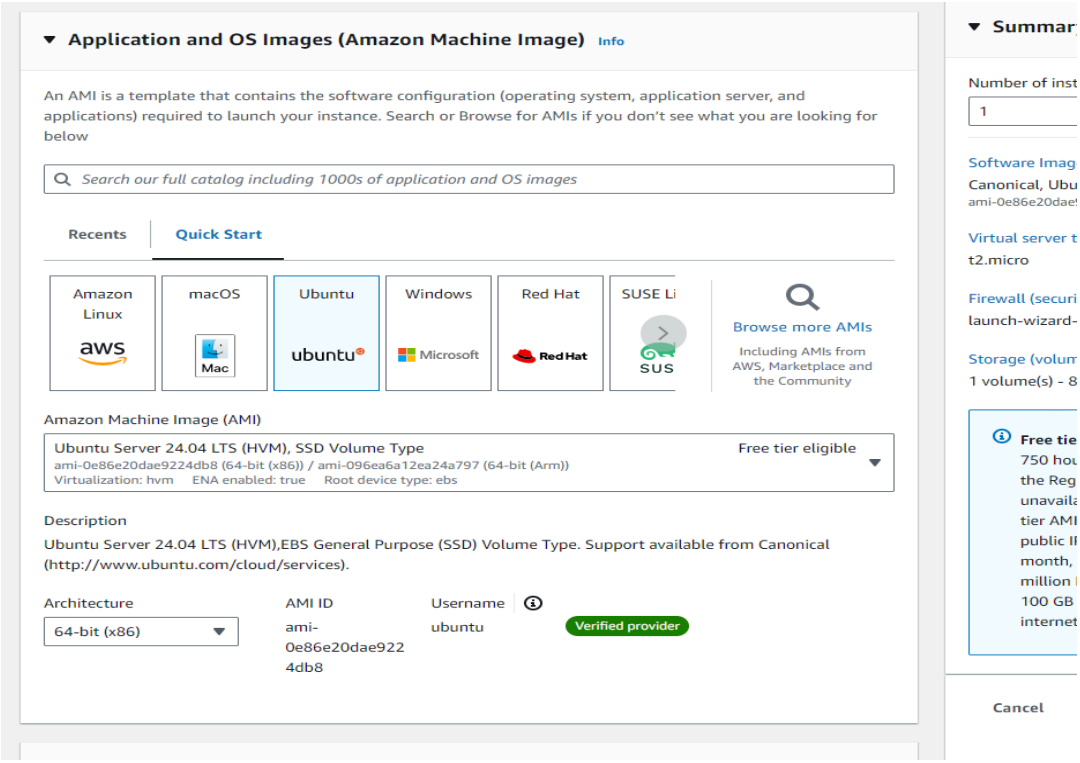
Steps

1) Launch an instance

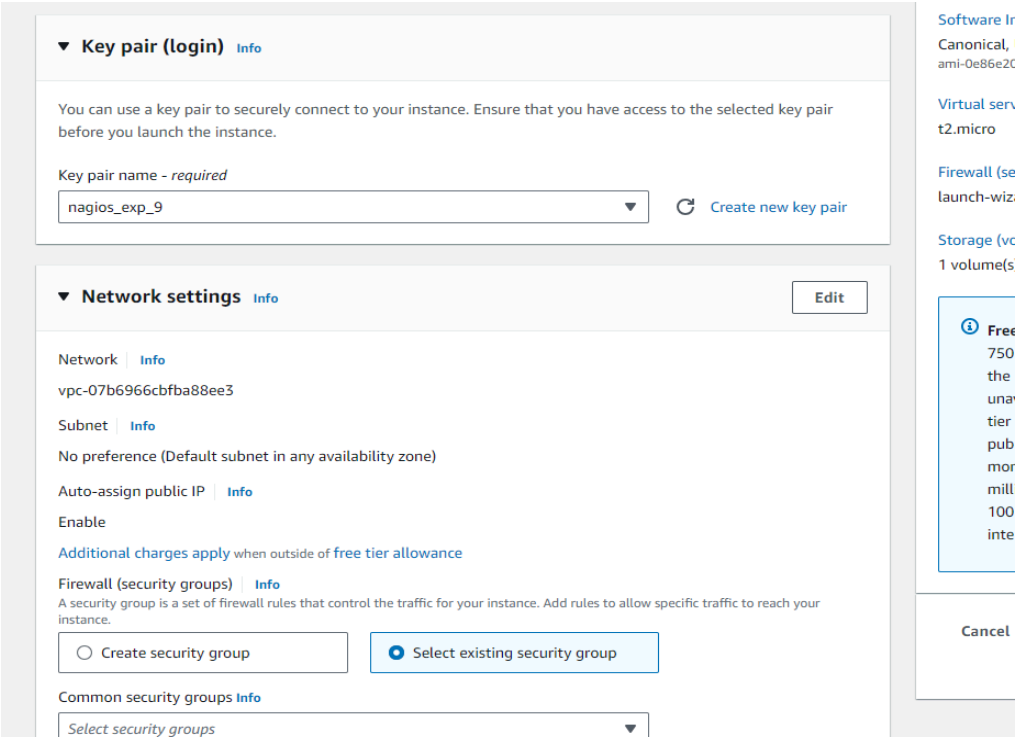
Launch an ec2 instance.
Select Ubuntu as the OS to give a meaningful name of the instance.



Select the same security group as given to the exp9 machine.

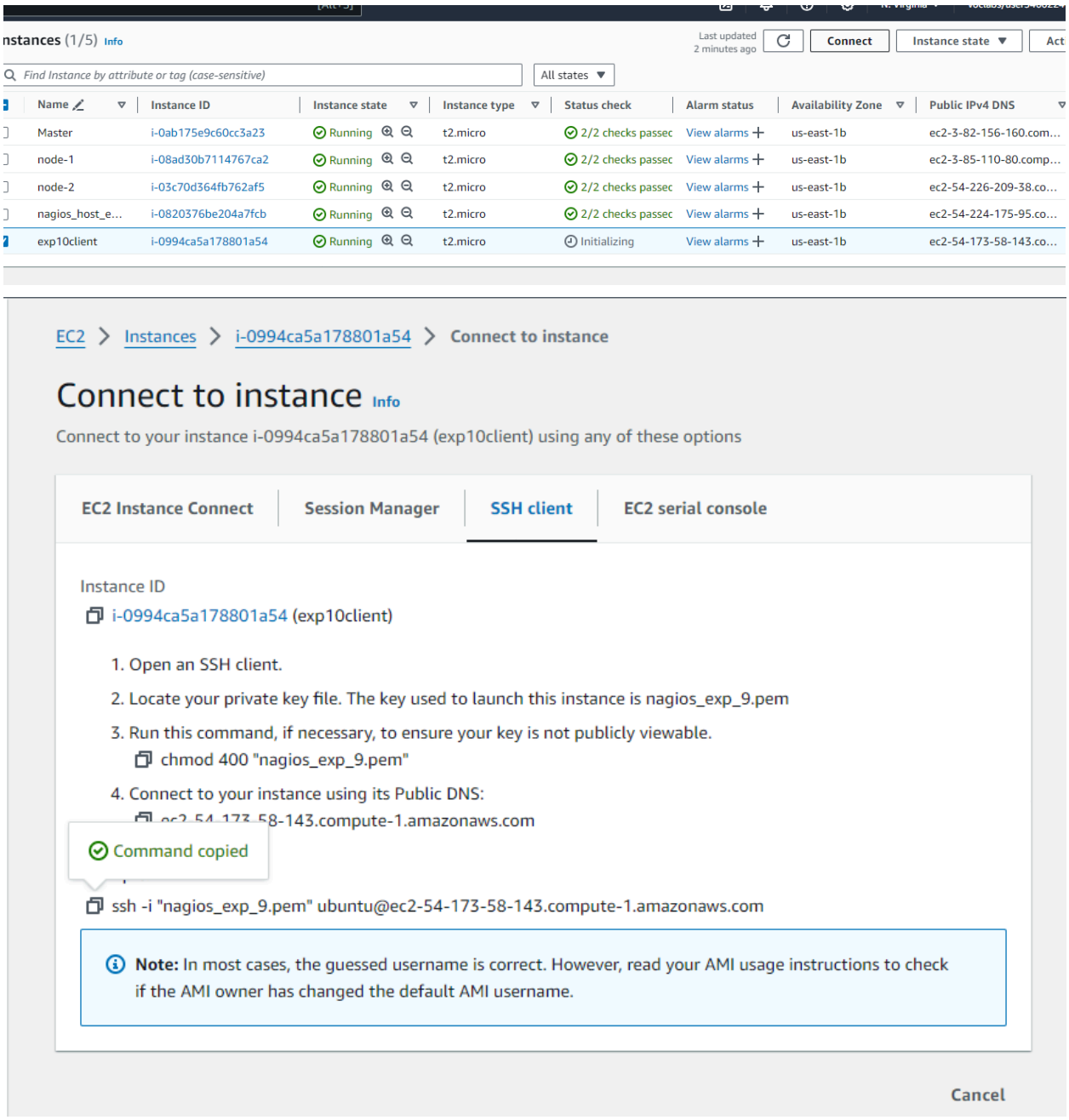


Make sure to select the same key-pair login used in the exp9 machine.

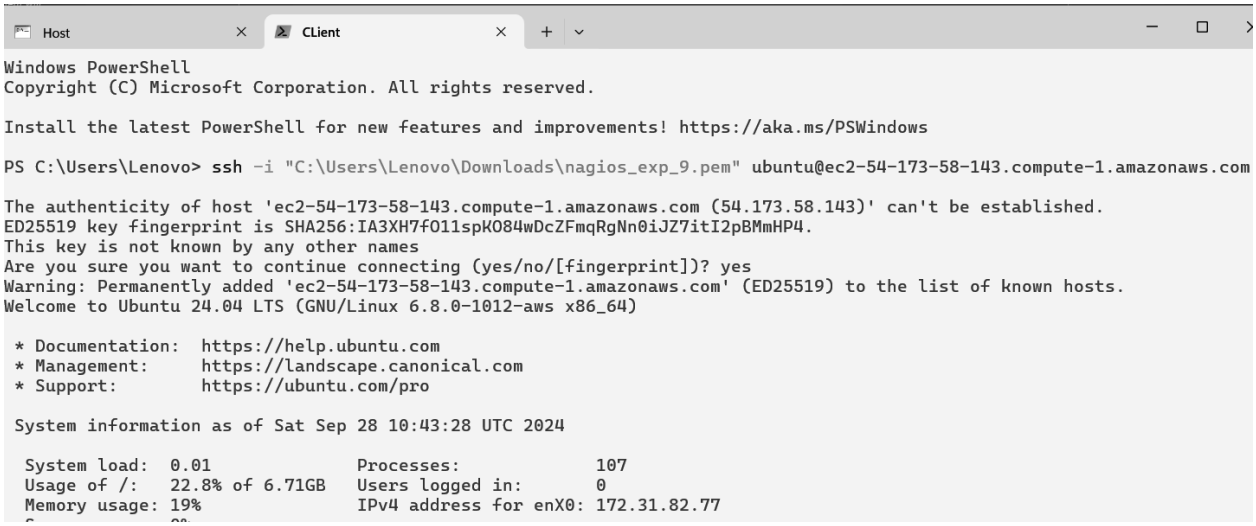


click on launch instance.

Now connect with this client machine using the ssh through your terminal(open a new terminal in your local machine and we will need both of the terminals open)



Note to change the path of the .pem file.



2) Go to nagios host machine (Host machine)

Perform the following commands

ps -ef | grep nagios

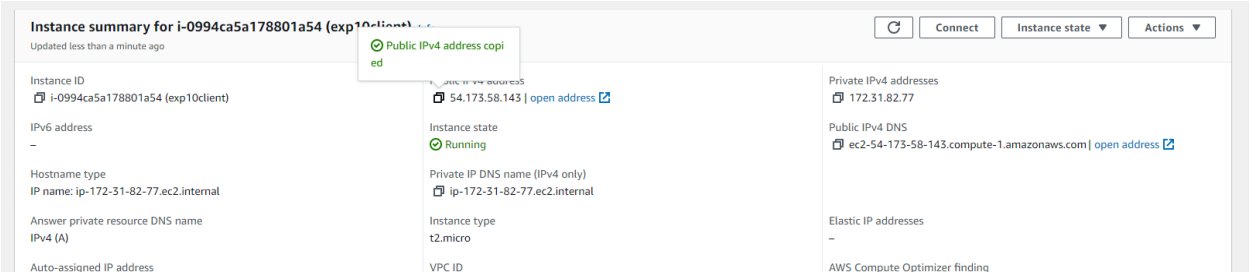
sudo su
mkdir -p /usr/local/nagios/etc/objects/monitorhosts/linuxhosts

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

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

Change hostname and alias to linuxserver
Change address to public ip address of client instance (Ubuntu instance) you can get the

ip address by clicking on the instance id on the instances section there you will get the public ipv4 address




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

# Define a host for the local machine

define host {

    use                     linux-server            ; Name of host template to use
                                                    ; This host definition will in>
                                                    ; in (or inherited by) the lin>
    host_name               linuxserver
    alias                   linuxserver
    address                 54.173.58.143
}
```

Change hostgroup_name to linux-servers1

```
# 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 host>
}
```

Change the occurrences of hostname further in the document from localhost to linuxserver
example like:

```
    host_name                       localhost
```

changed to

```
define service {

    use                     local-service           ; Name of service templat
    host_name               linuxserver
    service_description     PING
    check_command           check_ping!100.0,20%!500.0,60%
}
```

This is the last one

```
define service {

    use                     local-service           ; Name of service template to >
    host_name               linuxserver
    service_description     HTTP
    check_command           check_http
    notifications_enabled   0
```

now ctrl+O and enter to save and then ctrl+X for exiting.

Open nagios configuration file and add the line shown below
nano /usr/local/nagios/etc/nagios.cfg

```
[root@ip-172-31-80-137 ec2-user]# nano /usr/local/nagios/etc/nagios.cfg
```

##Add this line below the opened nano interface where similar lines are commented.
cfg_dir=/usr/local/nagios/etc/objects/monitorhosts/

```
 GNU nano 5.8                                                  /usr/local/nagios/etc/nagios.cfg
# These are the object configuration files in which you define hosts,
# host groups, contacts, contact groups, services, etc.
# You can split your object definitions across several config files
# if you wish (as shown below), or keep them all in a single config file.

# You can specify individual object config files as shown below:
cfg_file=/usr/local/nagios/etc/objects/commands.cfg
cfg_file=/usr/local/nagios/etc/objects/contacts.cfg
cfg_file=/usr/local/nagios/etc/objects/timeperiods.cfg
cfg_file=/usr/local/nagios/etc/objects/templates.cfg

# Definitions for monitoring the local (Linux) host
cfg_file=/usr/local/nagios/etc/objects/localhost.cfg

# Definitions for monitoring a Windows machine
#cfg_file=/usr/local/nagios/etc/objects/windows.cfg

# Definitions for monitoring a router/switch
#cfg_file=/usr/local/nagios/etc/objects/switch.cfg

# Definitions for monitoring a network printer
#cfg_file=/usr/local/nagios/etc/objects/printer.cfg


# You can also tell Nagios to process all config files (with a .cfg
# extension) in a particular directory by using the cfg_dir
# directive as shown below:

#cfg_dir=/usr/local/nagios/etc/servers
#cfg_dir=/usr/local/nagios/etc/printers
#cfg_dir=/usr/local/nagios/etc/switches
#cfg_dir=/usr/local/nagios/etc/routers
cfg_dir=/usr/local/nagios/etc/objects/monitorhosts/



# OBJECT CACHE FILE
# This option determines where object definitions are cached when
# Nagios starts/restarts.  The CGIs read object definitions from
```

ctrl+o and enter for saving and ctrl+x to exit nano editor.

Verify configuration files

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

```
[root@ip-172-31-80-137 ec2-user]# /usr/local/nagios/bin/nagios -v /usr/local/nagios
/etc/nagios.cfg

Nagios Core 4.5.5
Copyright (c) 2009-present Nagios Core Development Team and Community Contributors
Copyright (c) 1999-2009 Ethan Galstad
Last Modified: 2024-09-17
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
```

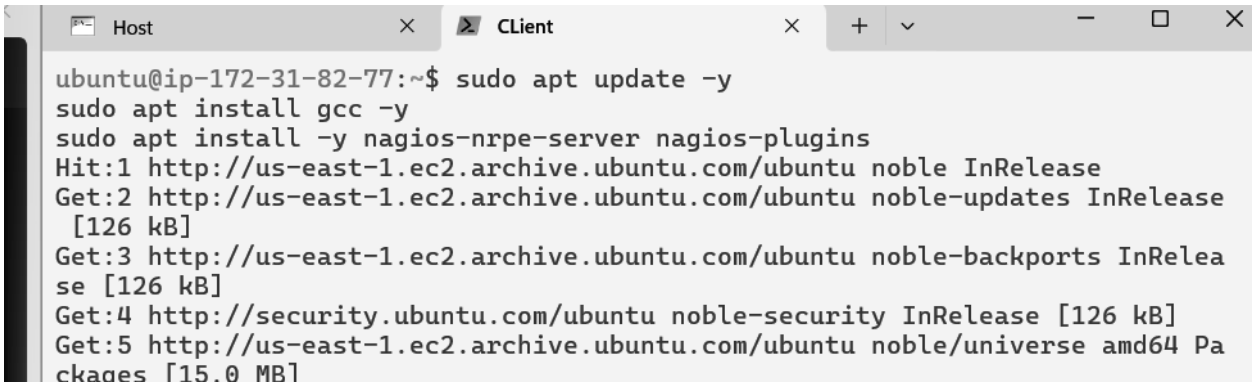
```
        Checked 0 service dependencies
        Checked 0 host dependencies
        Checked 5 timeperiods
Checking global event handlers...
Checking obsessive compulsive processor commands...
Checking misc settings...

Total Warnings: 0
Total Errors:   0

Things look okay - No serious problems were detected during the pre-flight check
[root@ip-172-31-80-137 ec2-user]#
```

Restart nagios service.
service nagios restart

```
Things look okay - No serious problems were detected during the pre-flight check
[root@ip-172-31-80-137 ec2-user]# service nagios restart
Redirecting to /bin/systemctl restart nagios.service
[root@ip-172-31-80-137 ec2-user]#
```

3) Go to client machine (ubuntu machine)

Perform the following commands
sudo apt update -y
sudo apt install gcc -y
sudo apt install -y nagios-nrpe-server nagios-plugins



```
Running kernel seems to be up-to-date.

Restarting services...

Service restarts being deferred:
 /etc/needrestart/restart.d/dbus.service
 systemctl restart getty@tty1.service
 systemctl restart networkd-dispatcher.service
 systemctl restart serial-getty@ttyS0.service
 systemctl restart systemd-logind.service
 systemctl restart unattended-upgrades.service

No containers need to be restarted.

User sessions running outdated binaries:
 ubuntu @ session #1: sshd[990,1101]
 ubuntu @ user manager service: systemd[996]

No VM guests are running outdated hypervisor (qemu) binaries on this host.
ubuntu@ip-172-31-82-77:~$
```

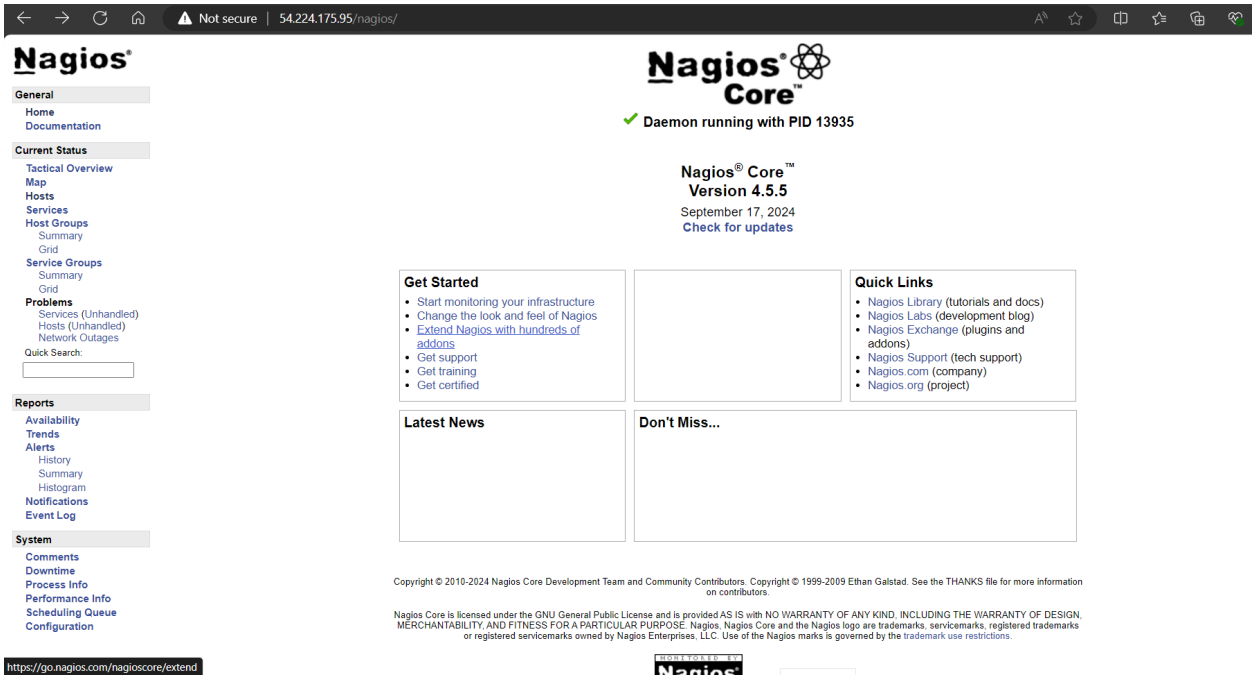
Open the nrpe.cfg file in nano editor
sudo nano /etc/nagios/nrpe.cfg

Under allowed_hosts, add the nagios host ip address (public)

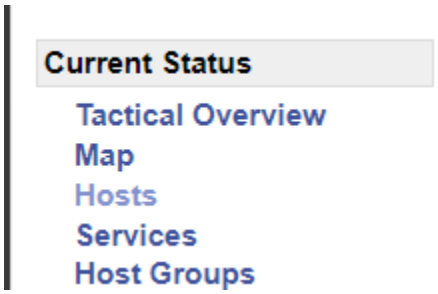
```
# You can either supply a username or a UID.
#
# NOTE: This option is ignored if NRPE is running under either inetd or xin>

nrpe_user=nagios



# NRPE GROUP
# This determines the effective group that the NRPE daemon should run as.
# You can either supply a group name or a GID.
#
# NOTE: This option is ignored if NRPE is running under either inetd or xin>

nrpe_group=nagios



# ALLOWED HOST ADDRESSES
# This is an optional comma-delimited list of IP address or hostnames
# that are allowed to talk to the NRPE daemon. Network addresses with a bit>
# (i.e. 192.168.1.0/24) are also supported. Hostname wildcards are not curr>
# 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 xin>

allowed_hosts=127.0.0.1,54.224.175.95



# COMMAND ARGUMENT PROCESSING
# This option determines whether or not the NRPE daemon will allow clients
```

again save and exit the nano editor.

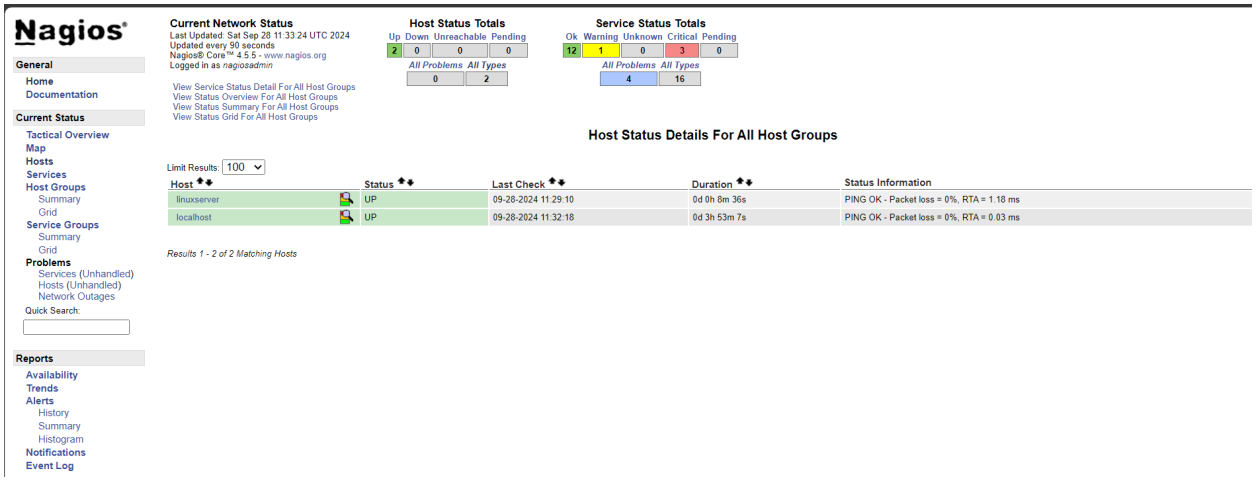
4) Go to nagios dashboard and click on hosts

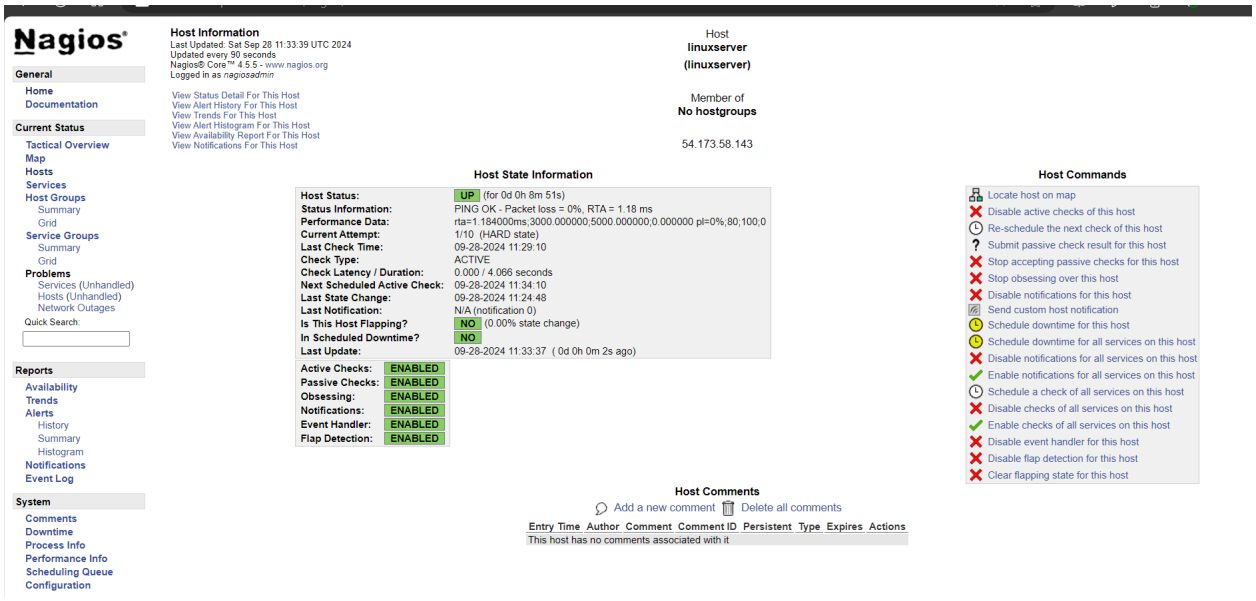


Click on hosts

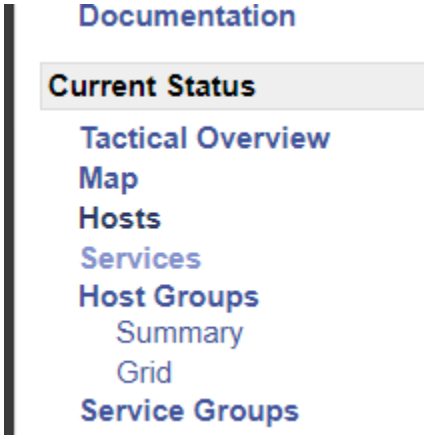


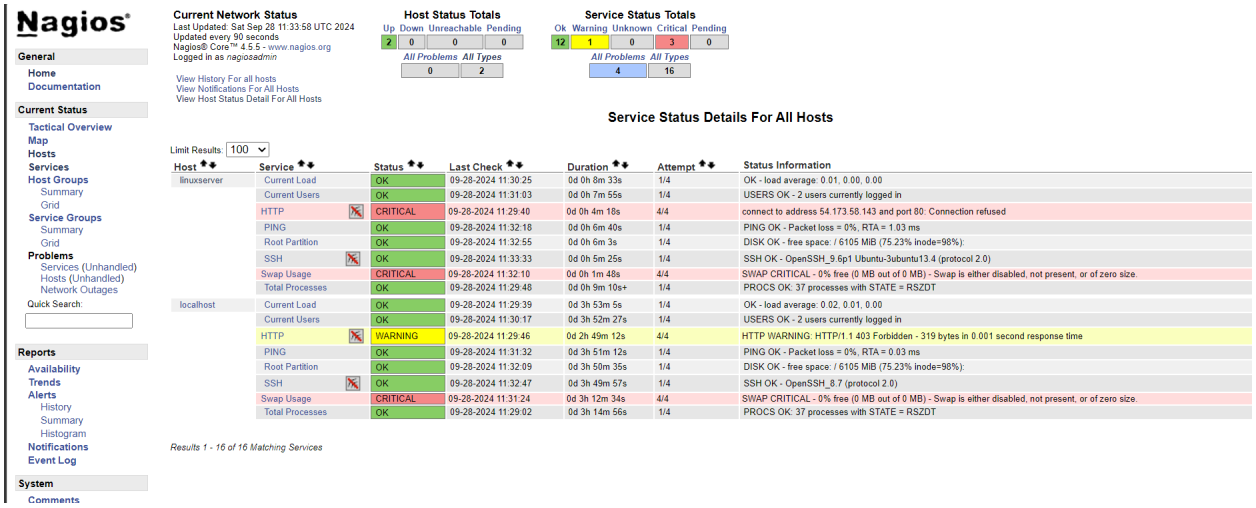
5) Click on linux server





6) Click on nagios services





Conclusion:

In this experiment, we successfully set up Nagios to monitor a remote Linux server (Ubuntu instance) from a Nagios host machine, both operating within an EC2 environment. The process involved careful configuration of both the Nagios host (referred to as the "exp9 machine") and the remote client machine, allowing for effective real-time monitoring of server performance. By launching an Ubuntu EC2 instance as the client machine, we ensured that consistent security group policies and key-pairs were maintained, enabling seamless SSH access to both machines for configuration purposes.

The configuration on the Nagios host involved creating a new directory for the remote client's monitoring configuration, where we modified the **localhost.cfg** file to reflect the IP address of the remote client, adjusted the hostgroup, and ensured proper integration with the Nagios setup.

We also verified these changes through a Nagios configuration check before restarting the service, ensuring that the Nagios dashboard was ready to monitor the remote machine.

On the client side, we installed the **nagios-nrpe-server** and **nagios-plugins**, which allowed communication between the Nagios host and client machine. Correctly configuring the **nrpe.cfg** file, particularly the **allowed_hosts** directive, was crucial in establishing connectivity. Once the configuration was complete and the Nagios service was restarted, we successfully confirmed that the remote Linux server was being monitored on the Nagios dashboard, achieving the experiment's objective.

This experiment not only highlighted the steps for setting up monitoring between a Nagios host and a remote Linux client but also demonstrated key troubleshooting techniques. By resolving common issues such as restarting the Apache server on the Nagios host, ensuring proper directory structure, and configuring the NRPE correctly on the client machine, we gained valuable insights into server monitoring and network management using Nagios. The hands-on experience provided a deeper understanding of how Nagios can be effectively used for real-world infrastructure monitoring.

.

Name: Kshitij Hundre
Div: D15C
Roll No:18

Adv 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

A fully managed, serverless computing service where you run code without provisioning or managing servers. Lambda automatically scales your application based on the number of incoming requests or events, ensuring efficient resource utilization. You are only charged for the time your code is running, with no upfront cost, making it cost-effective for on-demand workloads.

Lambda Workflow:

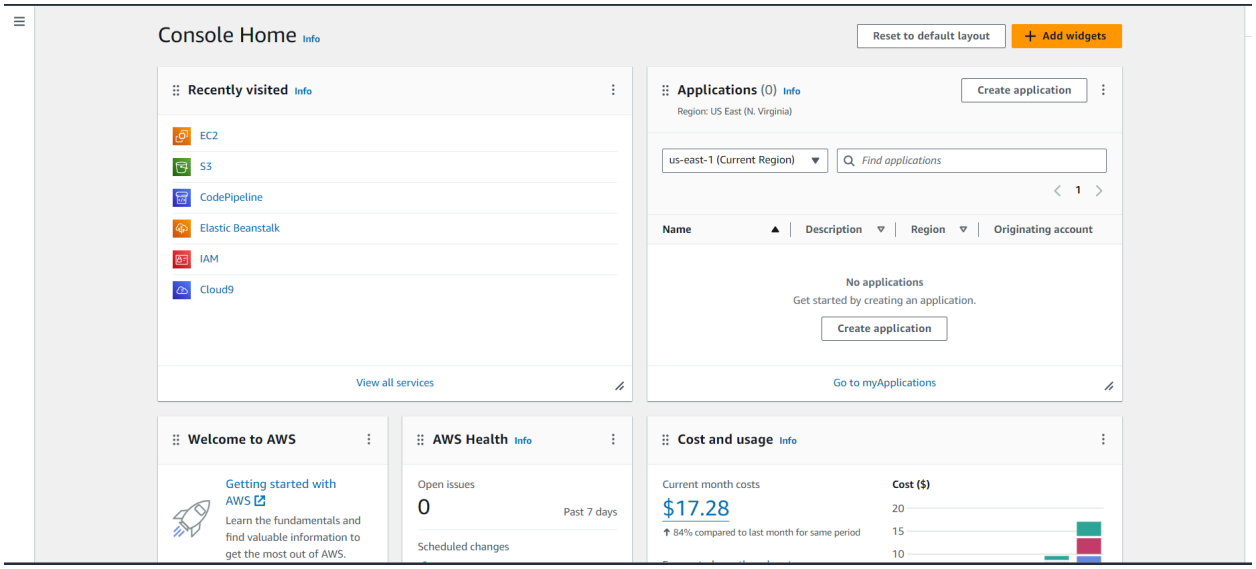
● **Create a Function**: Write the function code and define its handler (entry point). You can use the AWS Console, CLI, or upload a deployment package.
● **Set Event Sources**: Define how the function is triggered (e.g., when an object is uploaded to S3 or a DynamoDB table is updated).
● **Execution**: When triggered, Lambda runs your function, executes the logic, and automatically scales to handle the incoming event volume.
● **Scaling and Concurrency**: Lambda scales automatically by launching more instances of the function to handle simultaneous invocations. There are also options for configuring reserved concurrency to manage traffic.
● **Monitoring and Logging**: Lambda integrates with Amazon CloudWatch for logging and monitoring. Logs for each invocation are sent to CloudWatch, allowing you to track performance and troubleshoot errors.

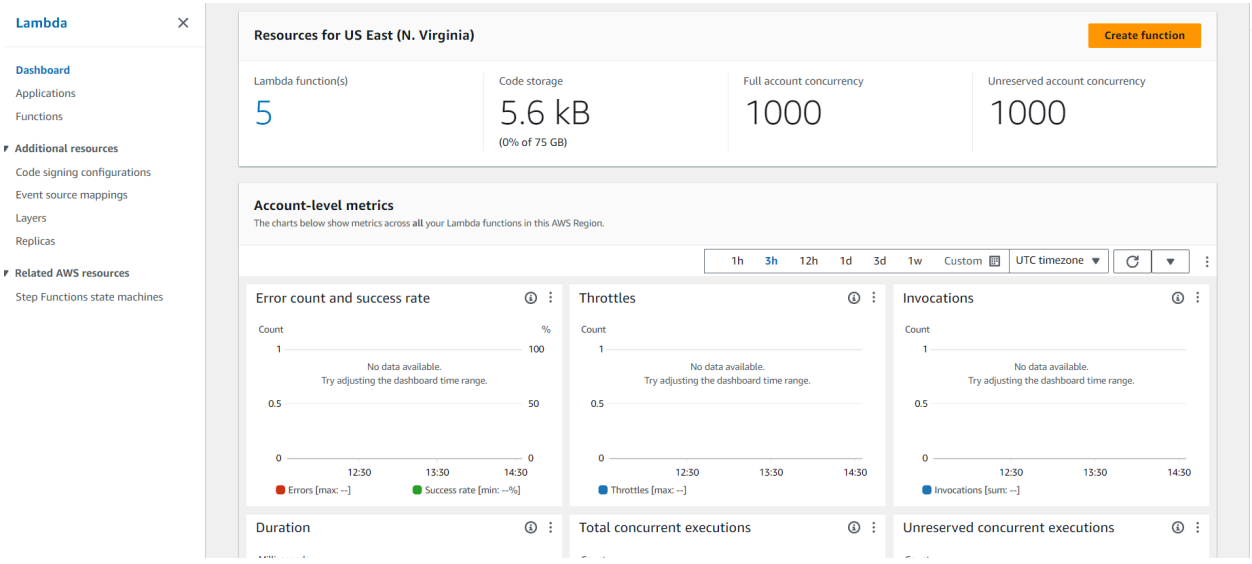
AWS Lambda Functions:

● **Python**: Great for quick development with its rich standard library and support for lightweight tasks.
● **Java**: Typically used for more complex, compute-intensive tasks. While it's robust, cold start times can be higher.
● **Node.js**: Excellent for I/O-bound tasks like handling APIs or streaming data, with fast startup times and efficient memory usage.
Prerequisites: AWS Personal/Academy Account
Prerequisites: AWS Personal/Academy Account

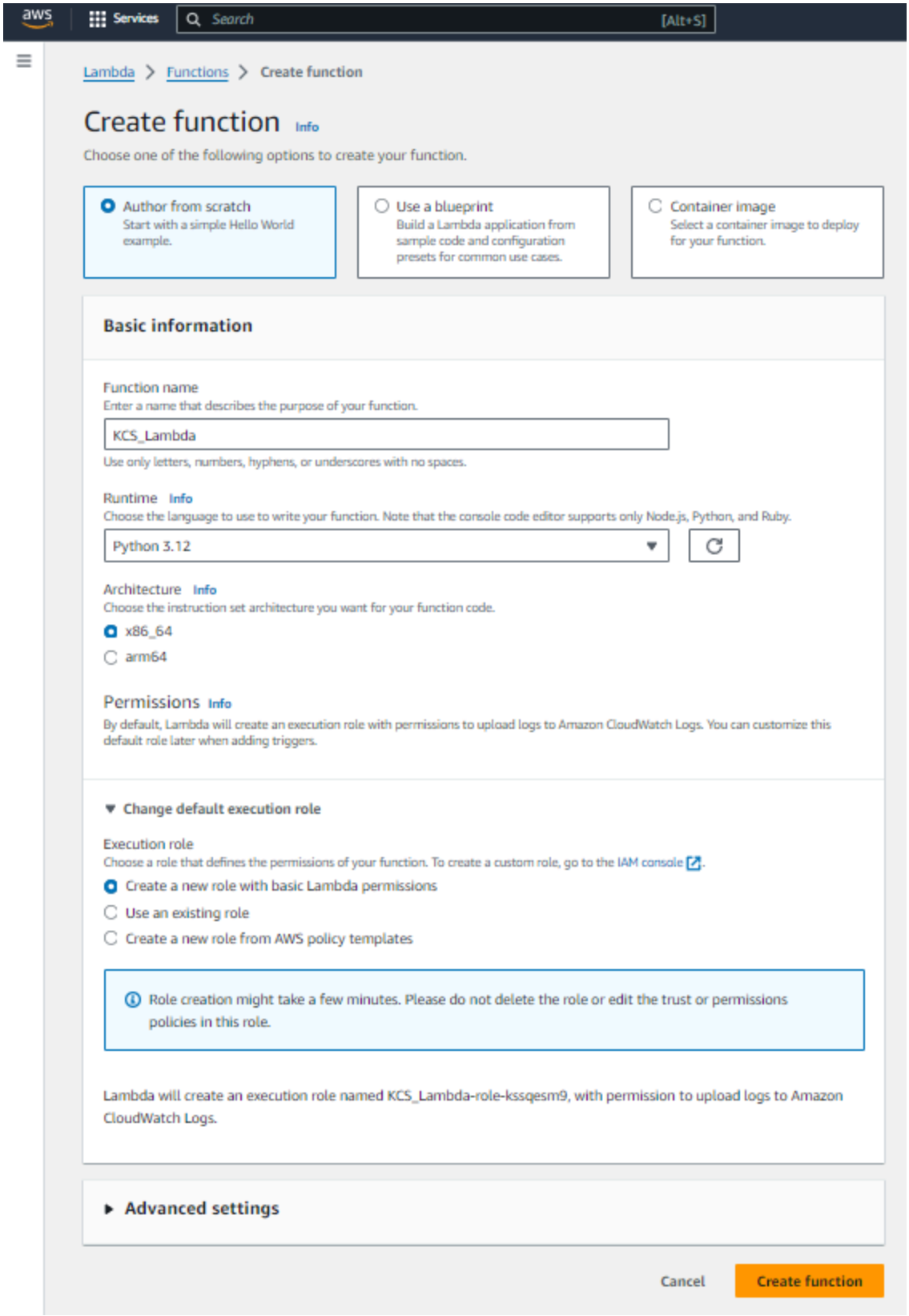
Steps To create the lambda function:

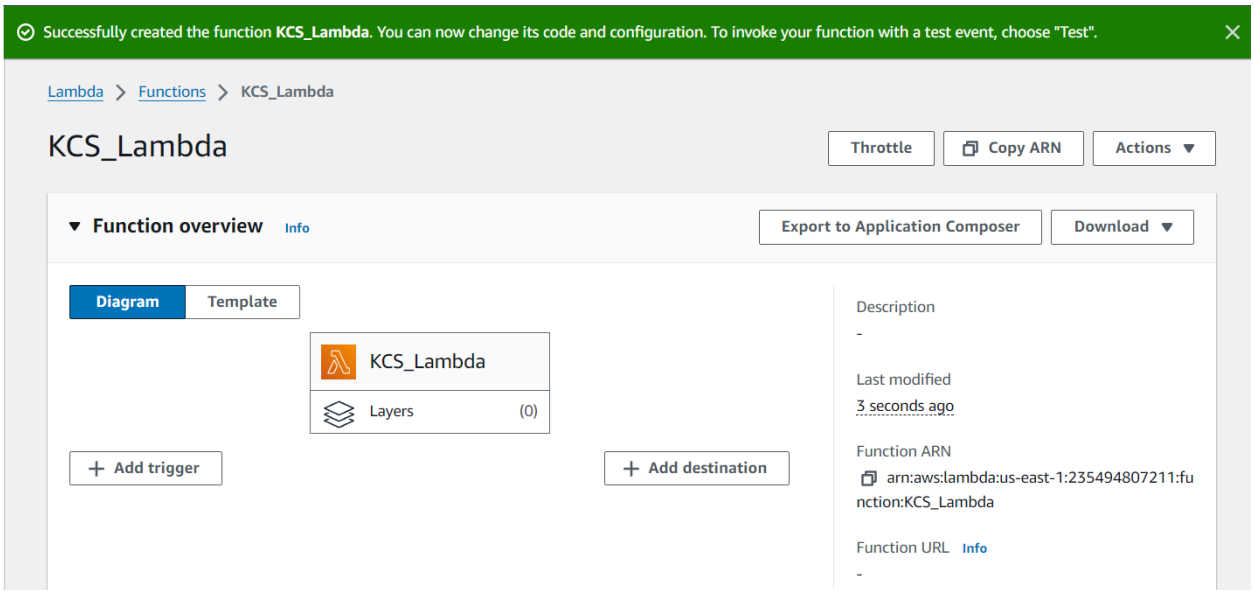
Step 1: Login to your AWS Personal/Academy Accout.Open lambda and click on create function button.

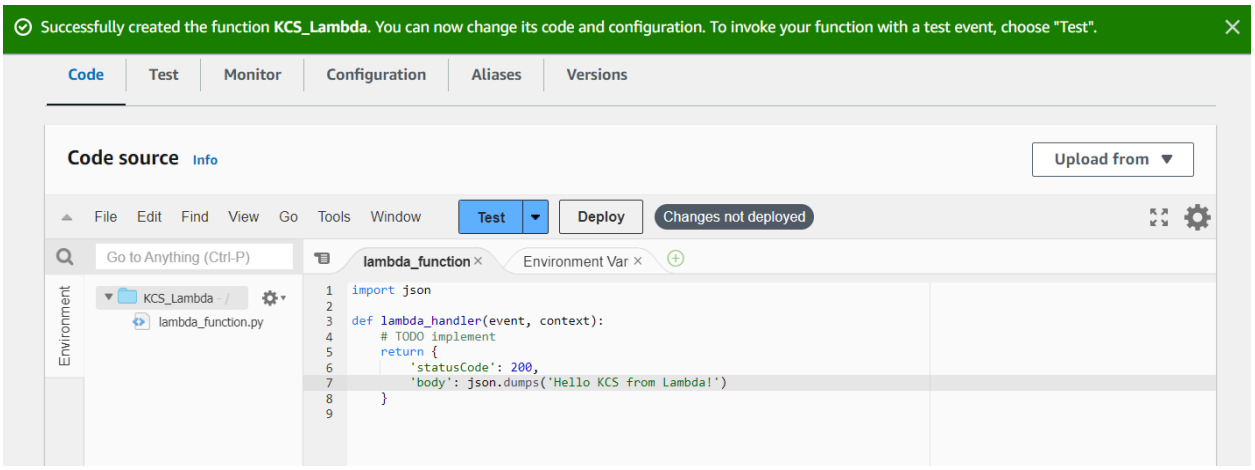




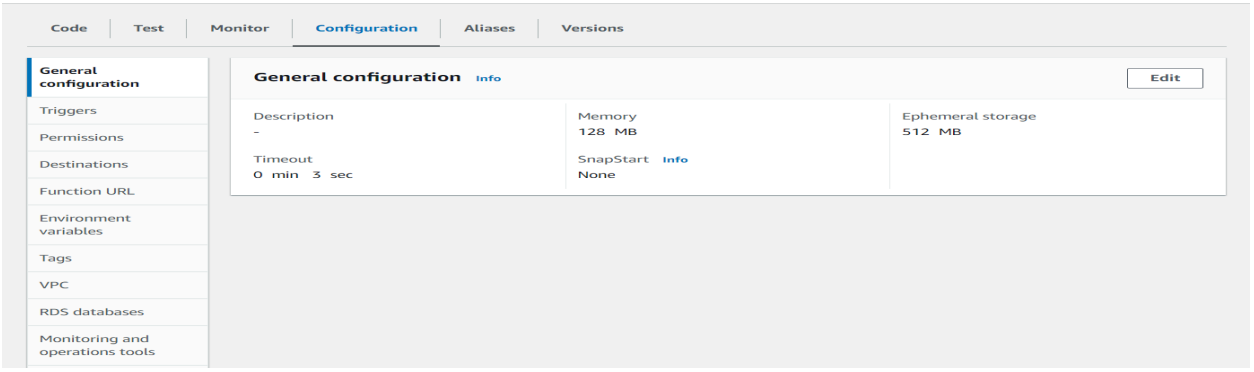
Step 2: Now Give a name to your Lambda function, Select the language to use to write your function. Note that the console code editor supports only Node.js, Python, and Ruby. So will select Python 3.12 , Architecture as x86, and Execution role to Create a new role with basic Lambda permissions.


aws
Services
Search
[Alt+S]
Lambda > Functions > Create function
Create function Info
Choose one of the following options to create your function.
Author from scratch
Start with a simple Hello World example.
Use a blueprint
Build a Lambda application from sample code and configuration presets for common use cases.
Container image
Select a container image to deploy for your function.
Basic information
Function name
Enter a name that describes the purpose of your function.
KCS_Lambda
Use only letters, numbers, hyphens, or underscores with no spaces.
Runtime Info
Choose the language to use to write your function. Note that the console code editor supports only Node.js, Python, and Ruby.
Python 3.12
Architecture Info
Choose the instruction set architecture you want for your function code.
x86_64
arm64
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
Role creation might take a few minutes. Please do not delete the role or edit the trust or permissions policies in this role.
Lambda will create an execution role named KCS_Lambda-role-kssqesm9, with permission to upload logs to Amazon CloudWatch Logs.
Advanced settings
Cancel
Create function

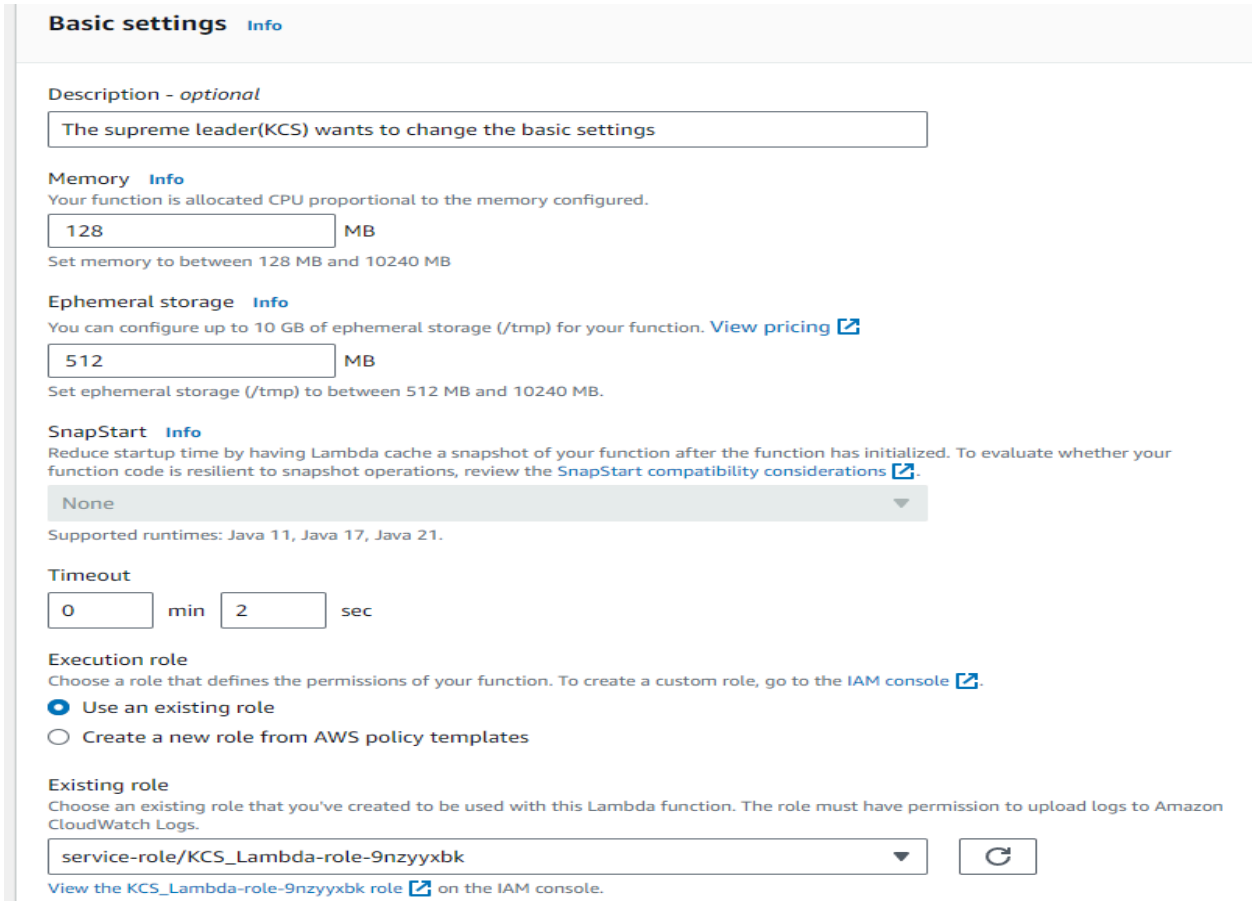


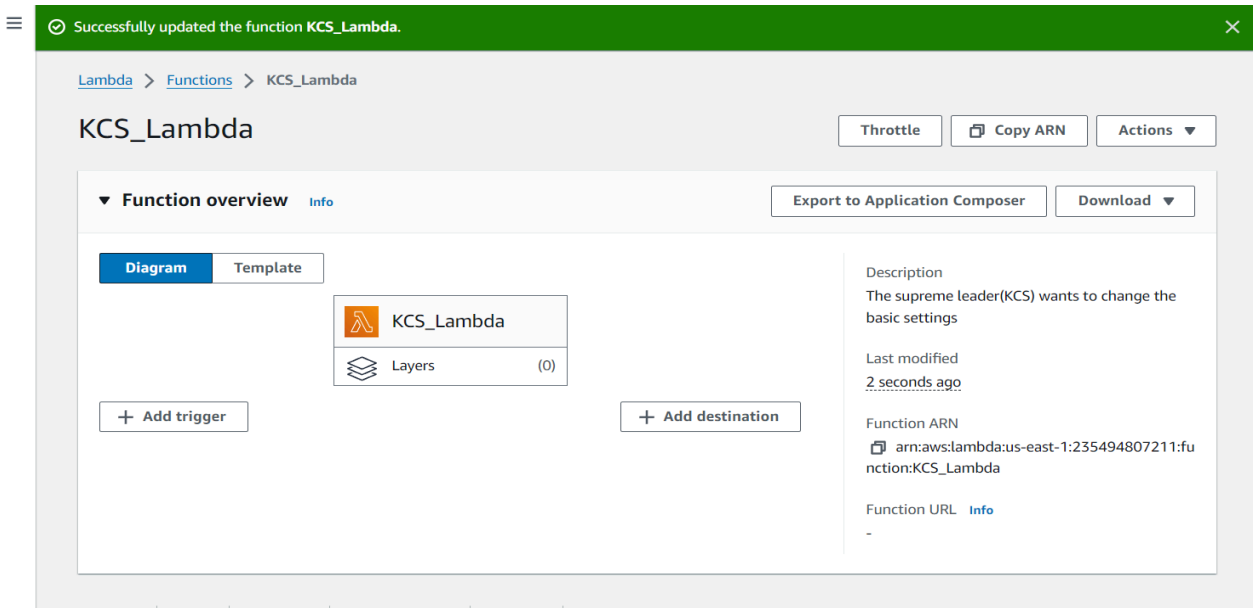


To See or Edit the basic settings go to configuration then click on edit general configuration.

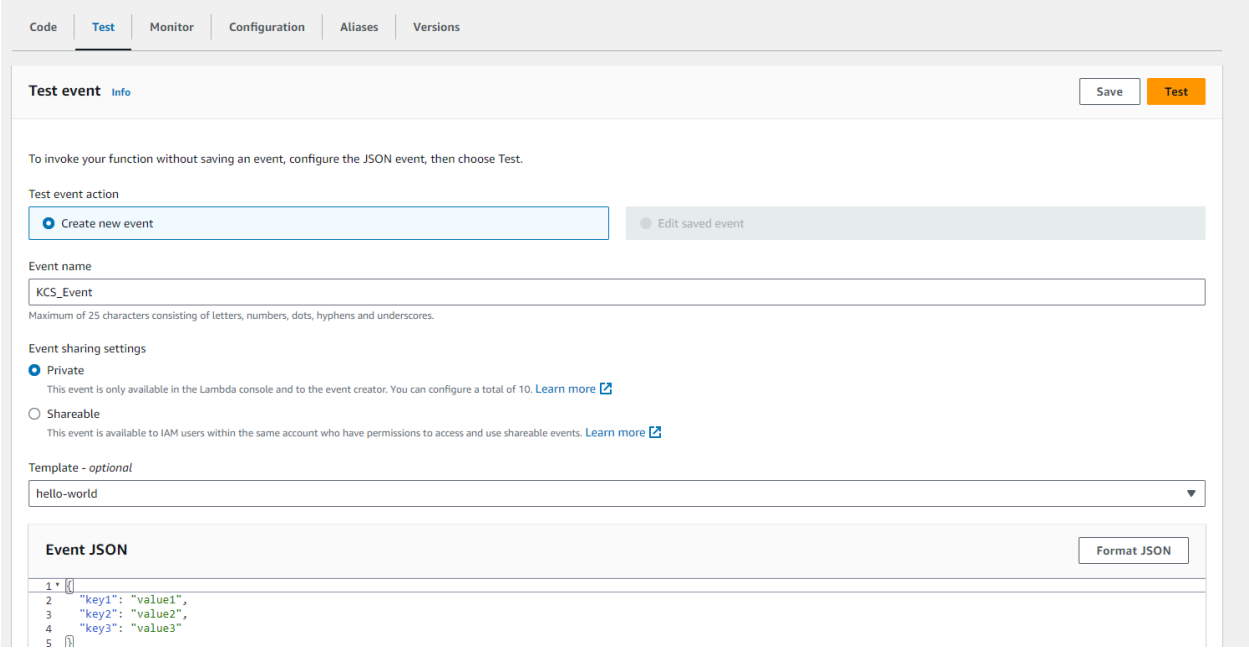


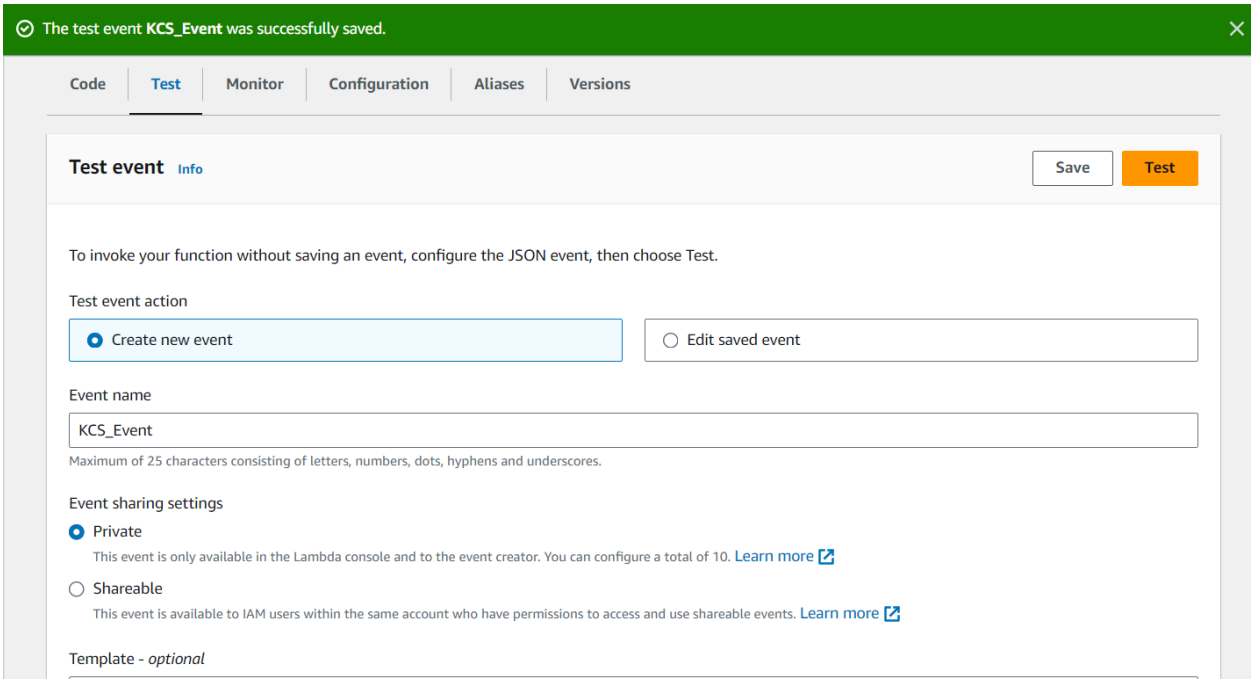
Here, you can enter a description and change Memory and Timeout. I've changed the Timeout period to 2 sec since that is sufficient for now.



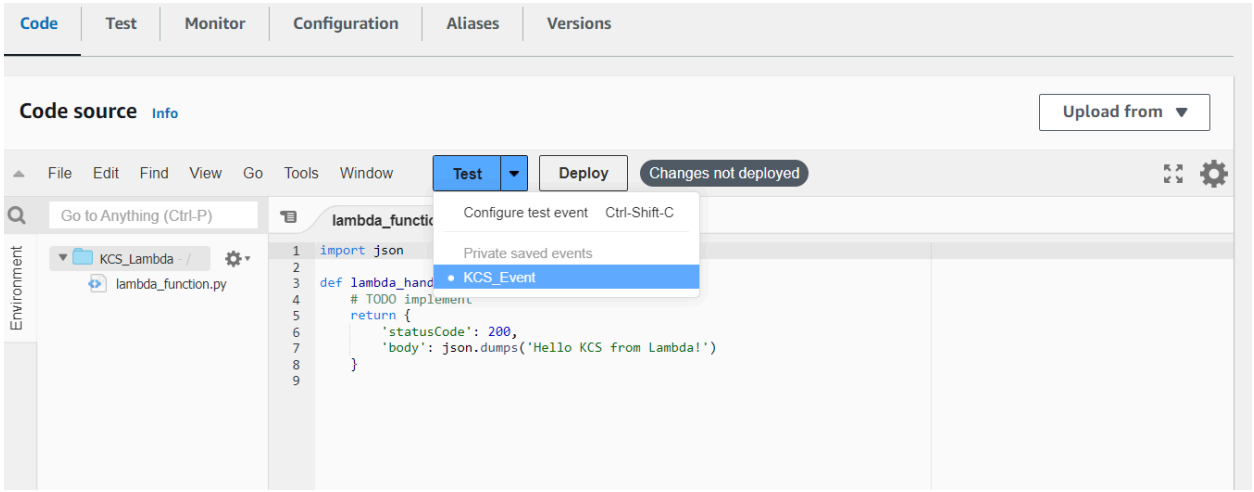


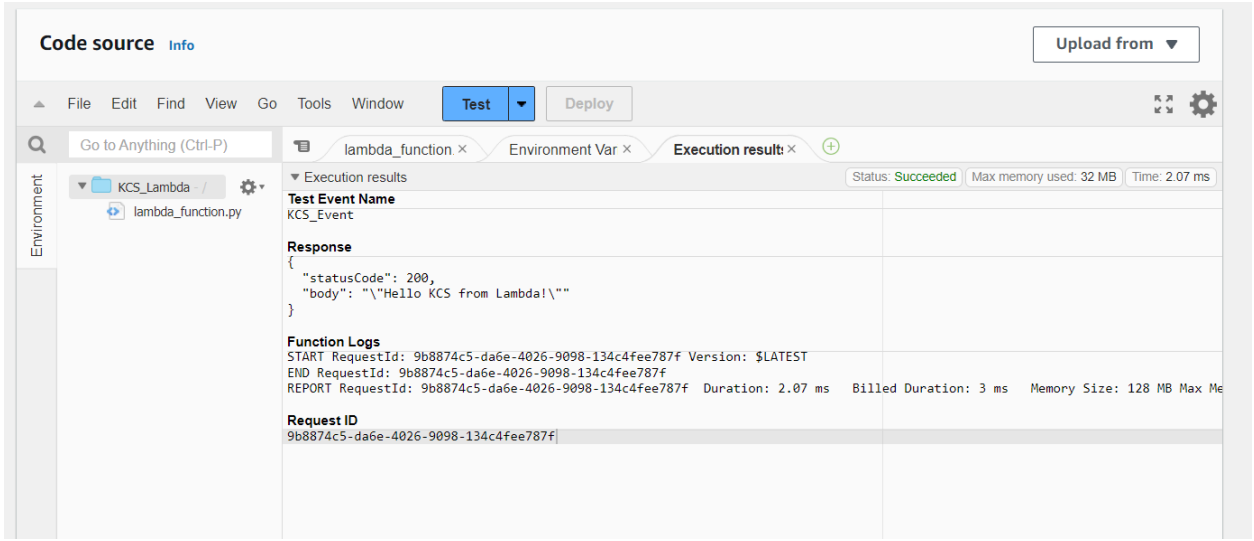
Step 3: Now Click on the Test tab then select Create a new event, give a name to the event and select Event Sharing to private, and select hello-world template.

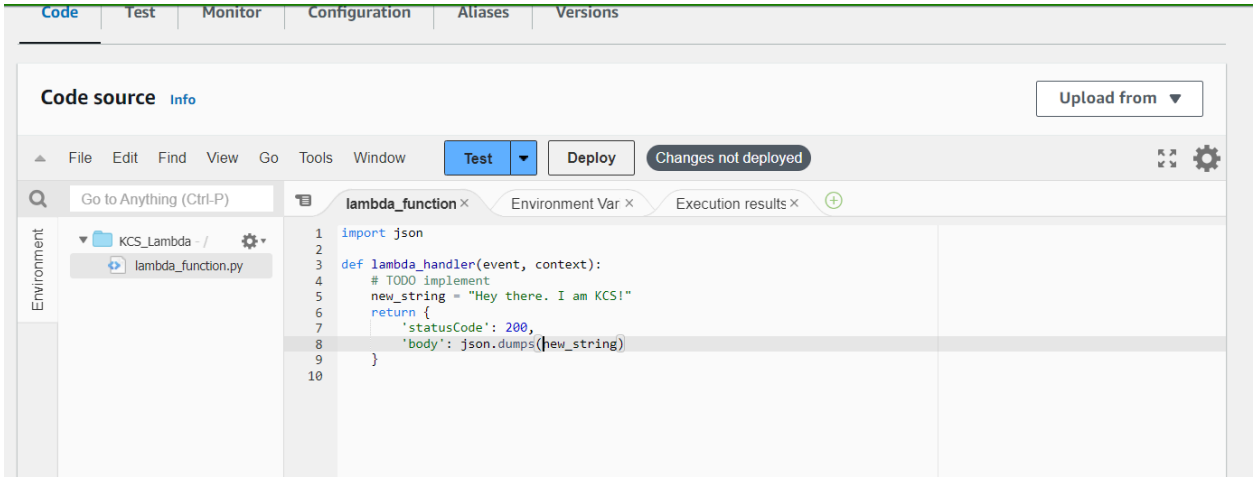




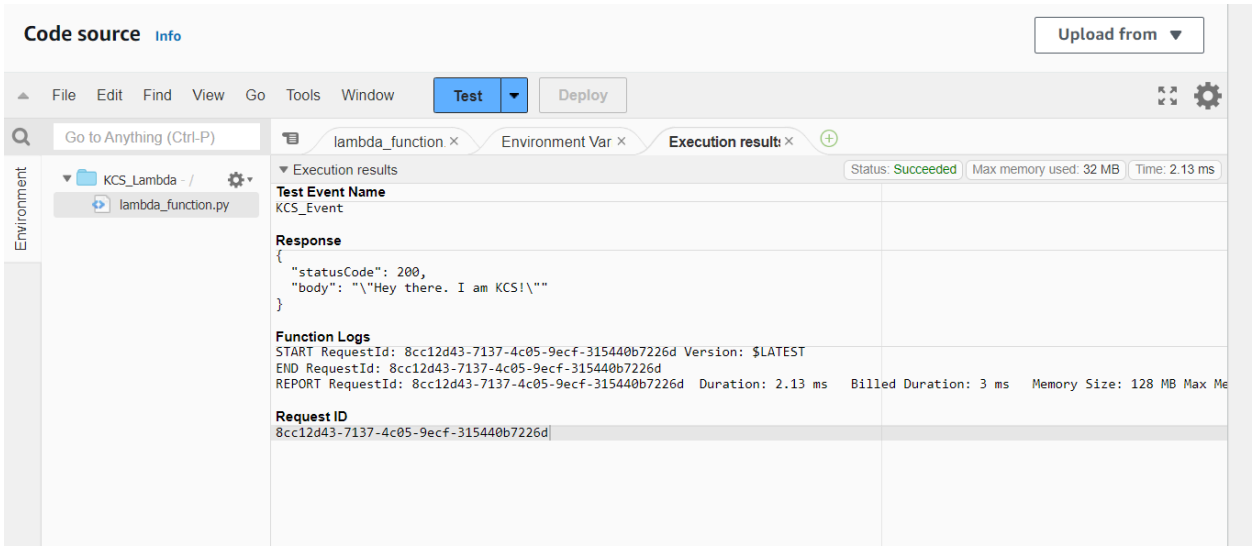
Step 4: Now In Code section select the created event from the dropdown of test then click on test . You will see the below output.







Now ctrl+s to save and click on deploy to deploy the changes



You can see the desired output.

Conclusion: In this experiment, we successfully developed an AWS Lambda function, covering the key steps involved. Starting with the Python-based setup, we configured the function's fundamental settings, including setting the timeout to 1 second. We proceeded to create a test event, deployed the function, and verified its output. Additionally, we made updates to the Lambda function's code and redeployed it, observing the real-time changes. This hands-on experience highlighted AWS Lambda's efficiency and adaptability, enabling rapid serverless application development while AWS handles infrastructure and scaling effortlessly.

Name: Kshitij Hundre Div:D15C Roll No:18

Adv 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:

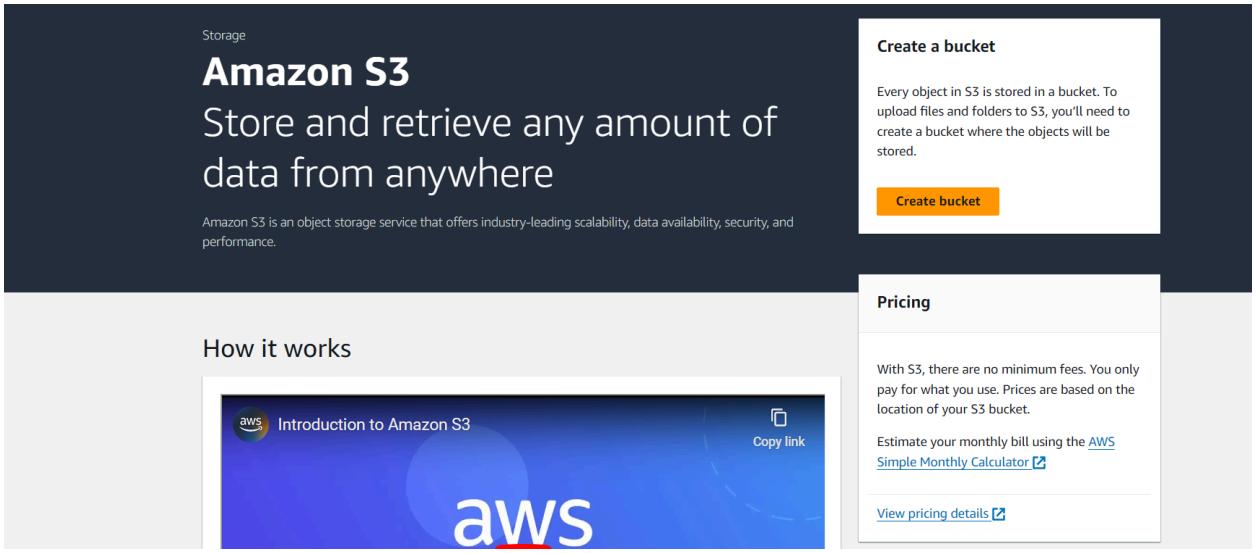
○ Upload an object (e.g., an image) to the S3 bucket to test the trigger. The Lambda function should execute and log the message “An Image has been added” in AWS CloudWatch Logs.
Prerequisites: AWS Personal Account

Prerequisites: AWS Personal Account

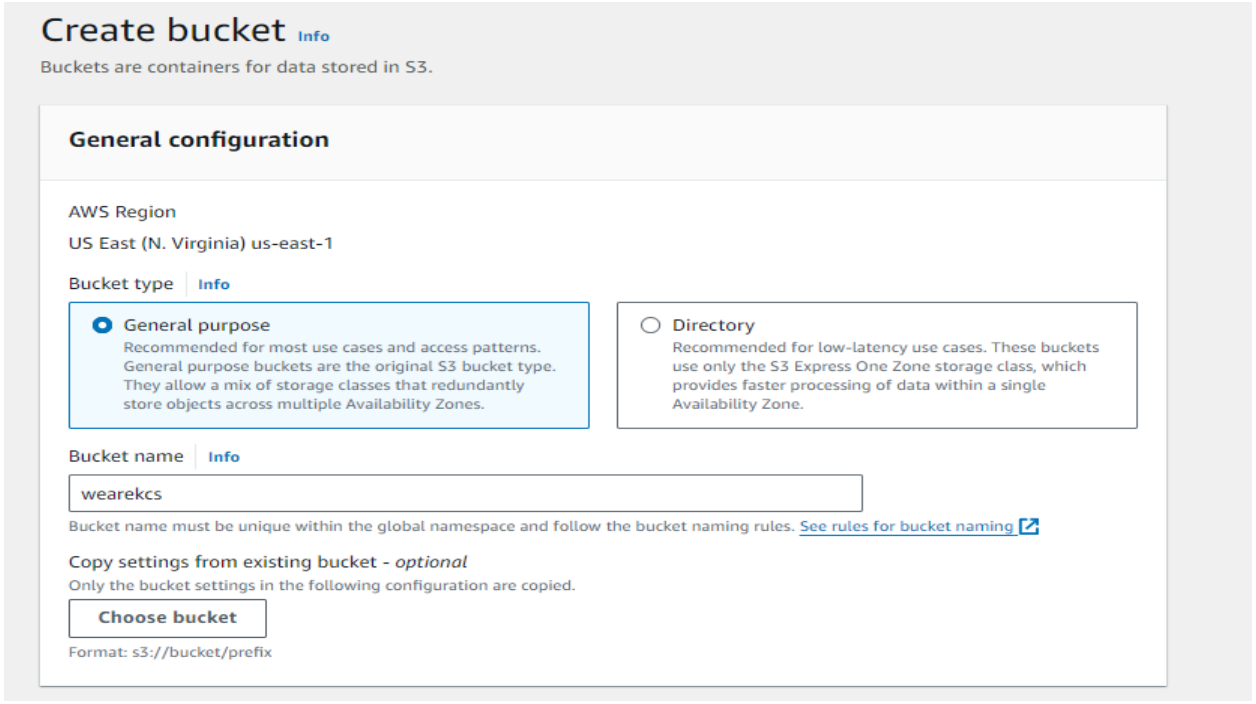
Name: Kshitij Hundre Div:D15C Roll No:18

Steps To create the lambda function:

Step 1: Login to your AWS Personal account. Now open S3 from services and click on create S3 bucket.



Step 2: Now Give a name to the Bucket, select general purpose project and deselect the Block public access and keep other this to default.



Name: Kshitij Hundre Div:D15C Roll No:18

Object Ownership Info

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.

Object Ownership
Bucket owner enforced

Block Public Access settings for this bucket

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 this bucket and its 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 this bucket 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.

⚠ **Turning off block all public access might result in this bucket and the objects within becoming**

Successfully created bucket "wearekcs"
To upload files and folders, or to configure additional bucket settings, choose **View details**.

View details

Amazon S3 > Buckets

▸ **Account snapshot** - *updated every 24 hours* All AWS Regions
Storage lens provides visibility into storage usage and activity trends. Learn more

View Storage Lens dashboard

General purpose buckets | Directory buckets

General purpose buckets (1) Info All AWS Regions
Buckets are containers for data stored in S3.

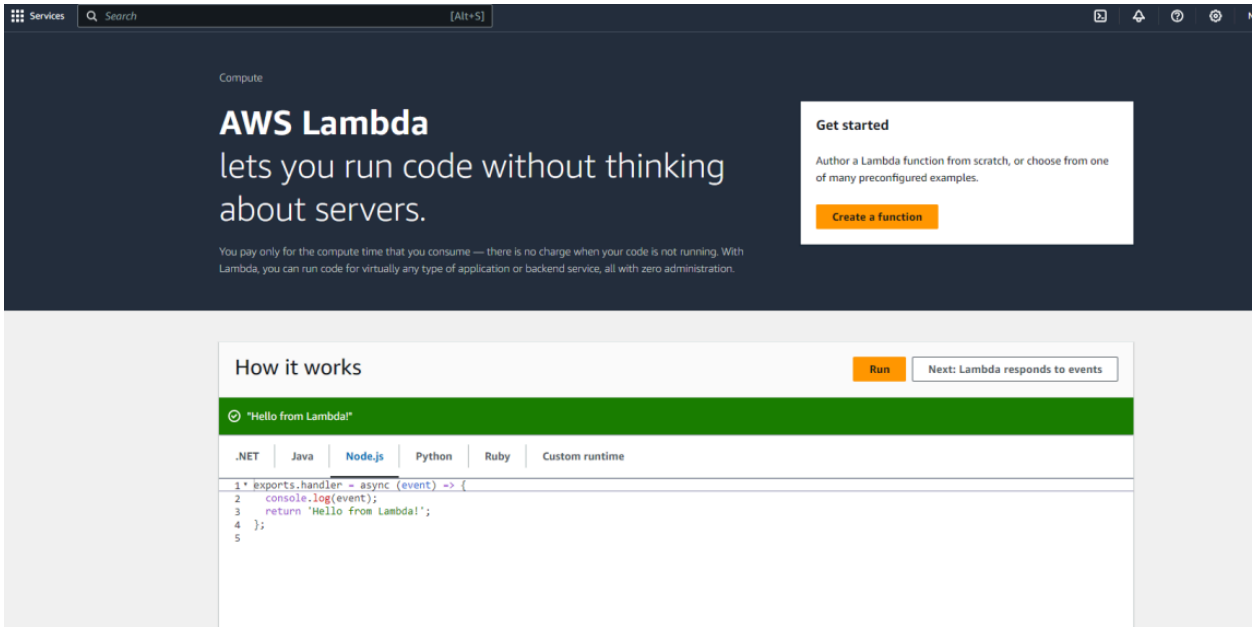
Copy ARN | Empty | Delete | Create bucket

Find buckets by name

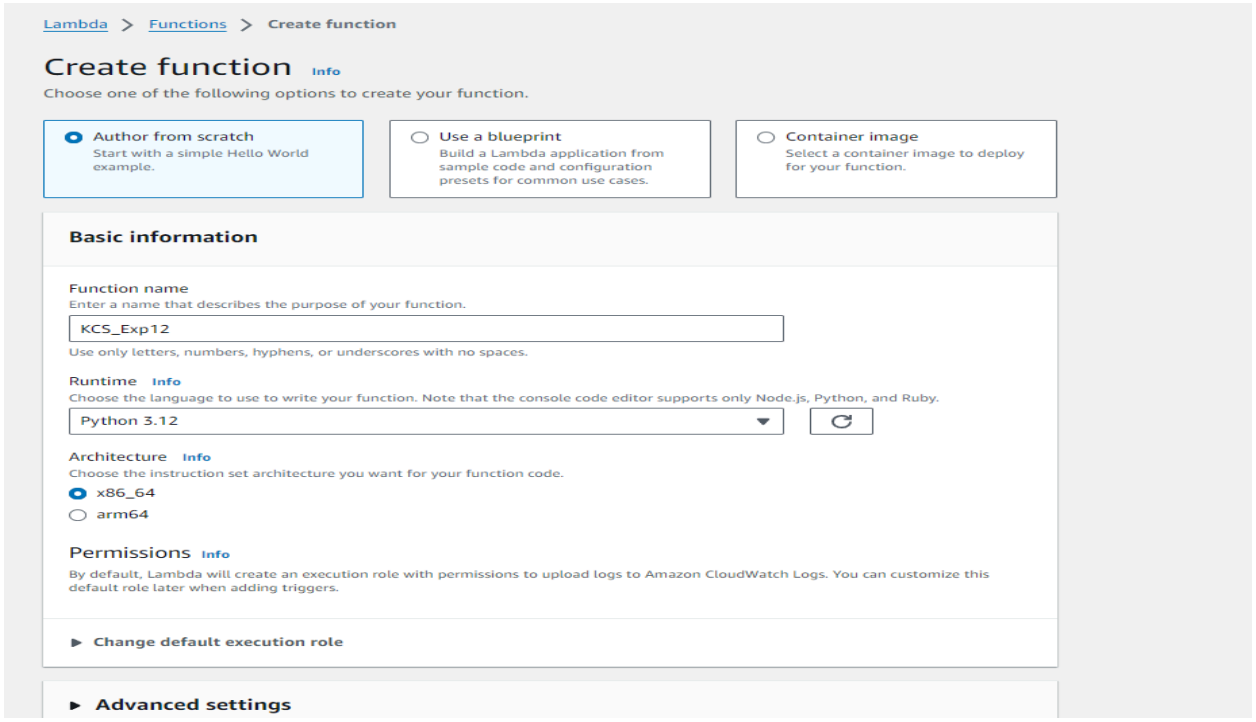
Name	AWS Region	IAM Access Analyzer	Creation date
wearekcs	US East (N. Virginia) us-east-1	View analyzer for us-east-1	October 1, 2024, 13:40:40 (UTC+05:30)

Name: Kshitij Hundre Div:D15C Roll No:18

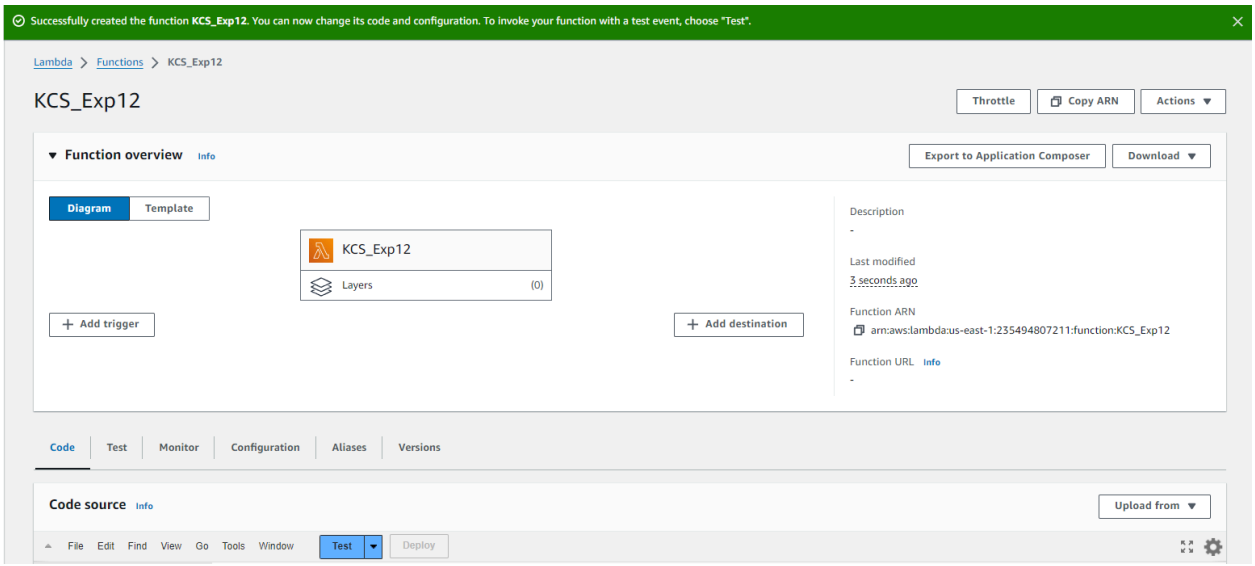
Step 3: Open lambda console and click on create function button

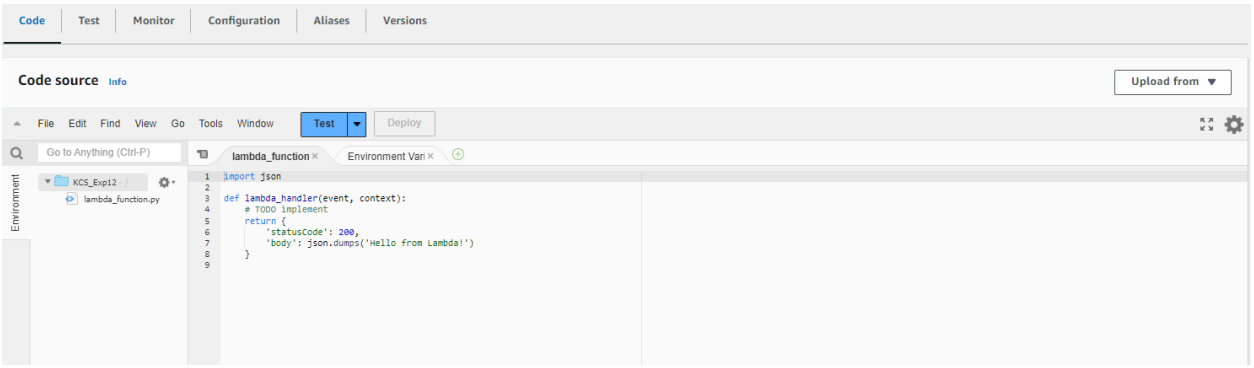


Step 4: Now Give a name to your Lambda function, Select the language to use to write your function. Note that the console code editor supports only Node.js, Python, and Ruby. So will select Python 3.12 , Architecture as x86, and Execution role to Create a new role with basic Lambda permissions.

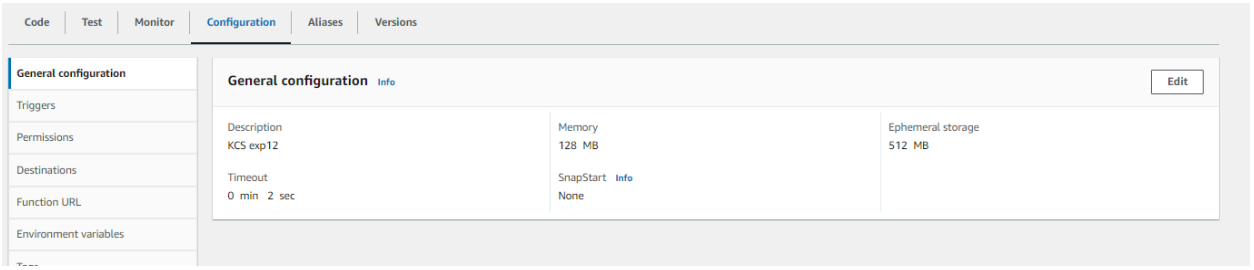


Name: Kshitij Hundre Div:D15C Roll No:18

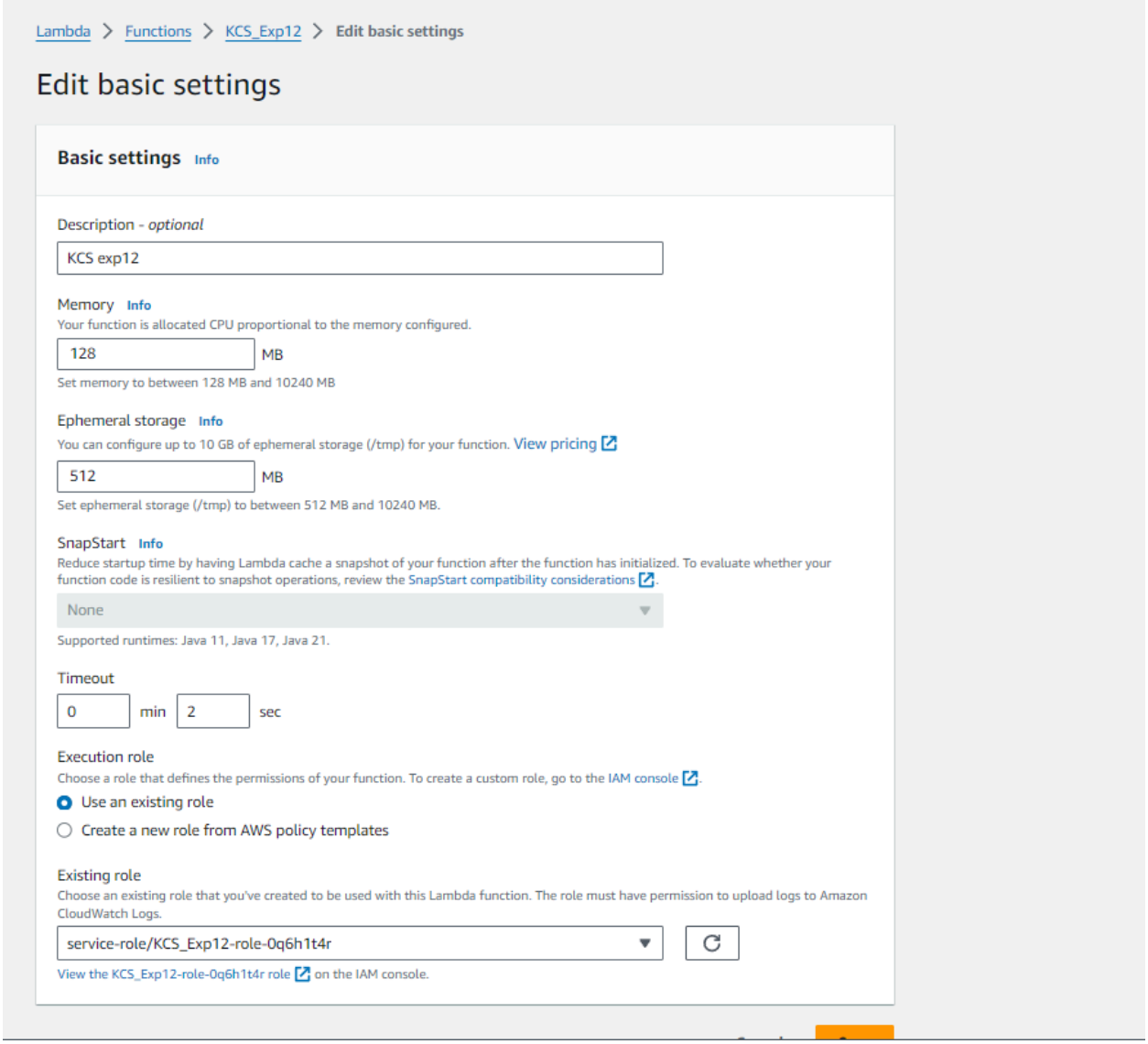




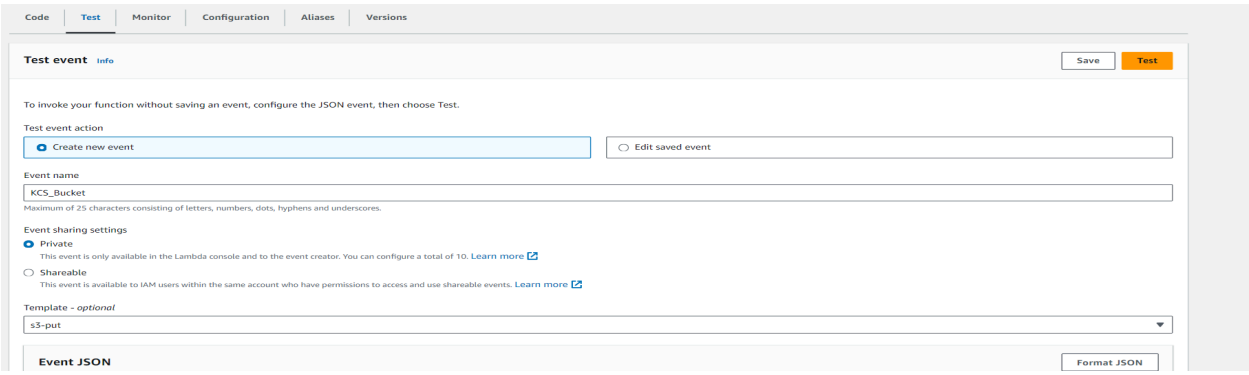
To See or Edit the basic settings go to configuration then click on edit general setting



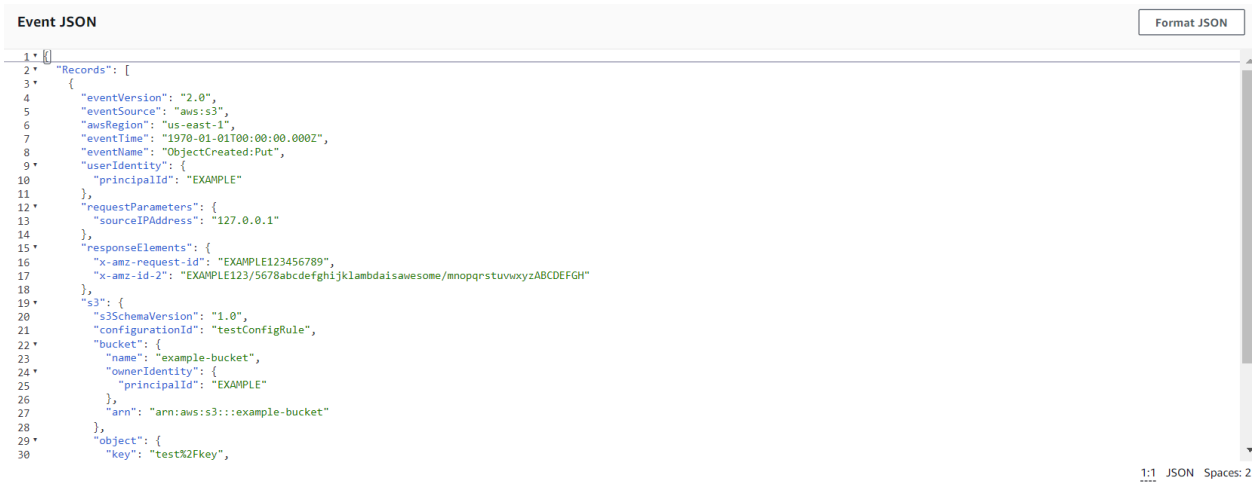
Change any setting of your choice. Here I have set a timeout of 2 secs.Then save changes



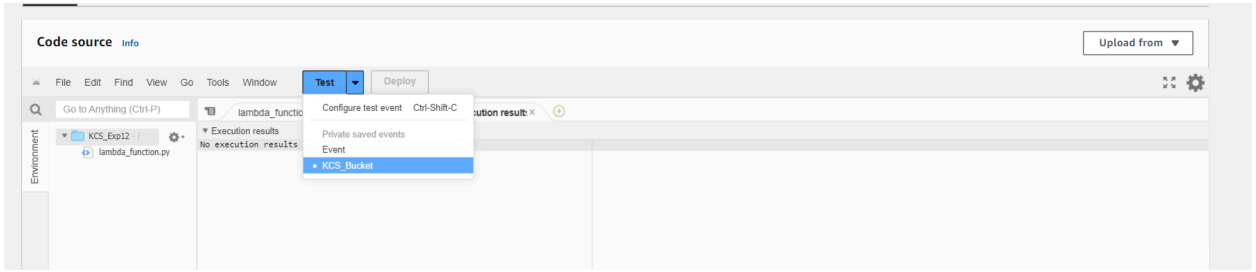
Step 5: Now Click on the Test tab then select Create a new event, give a name to the event and select Event Sharing to private, and select s3 put template.



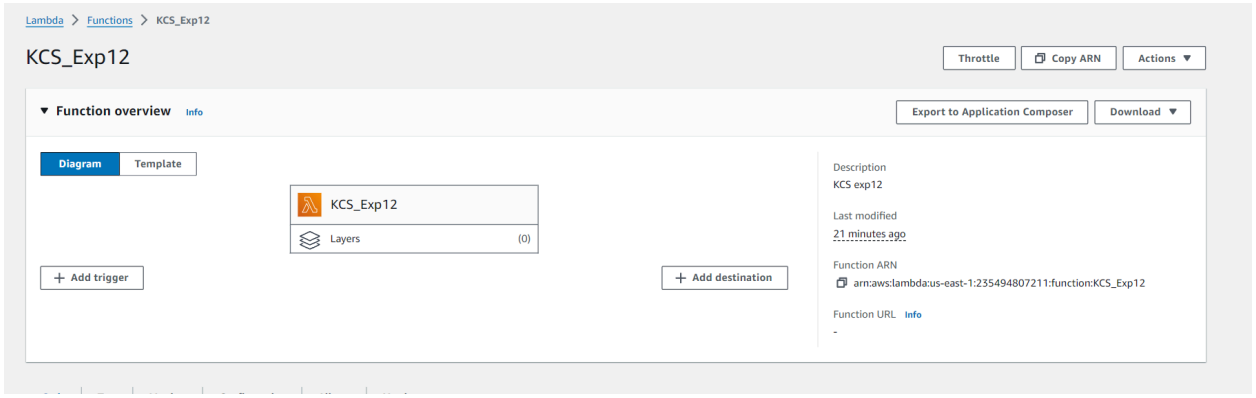
Name: Kshitij Hundre Div:D15C Roll No:18



Step 6: Now In the Code section select the created event from the dropdown .

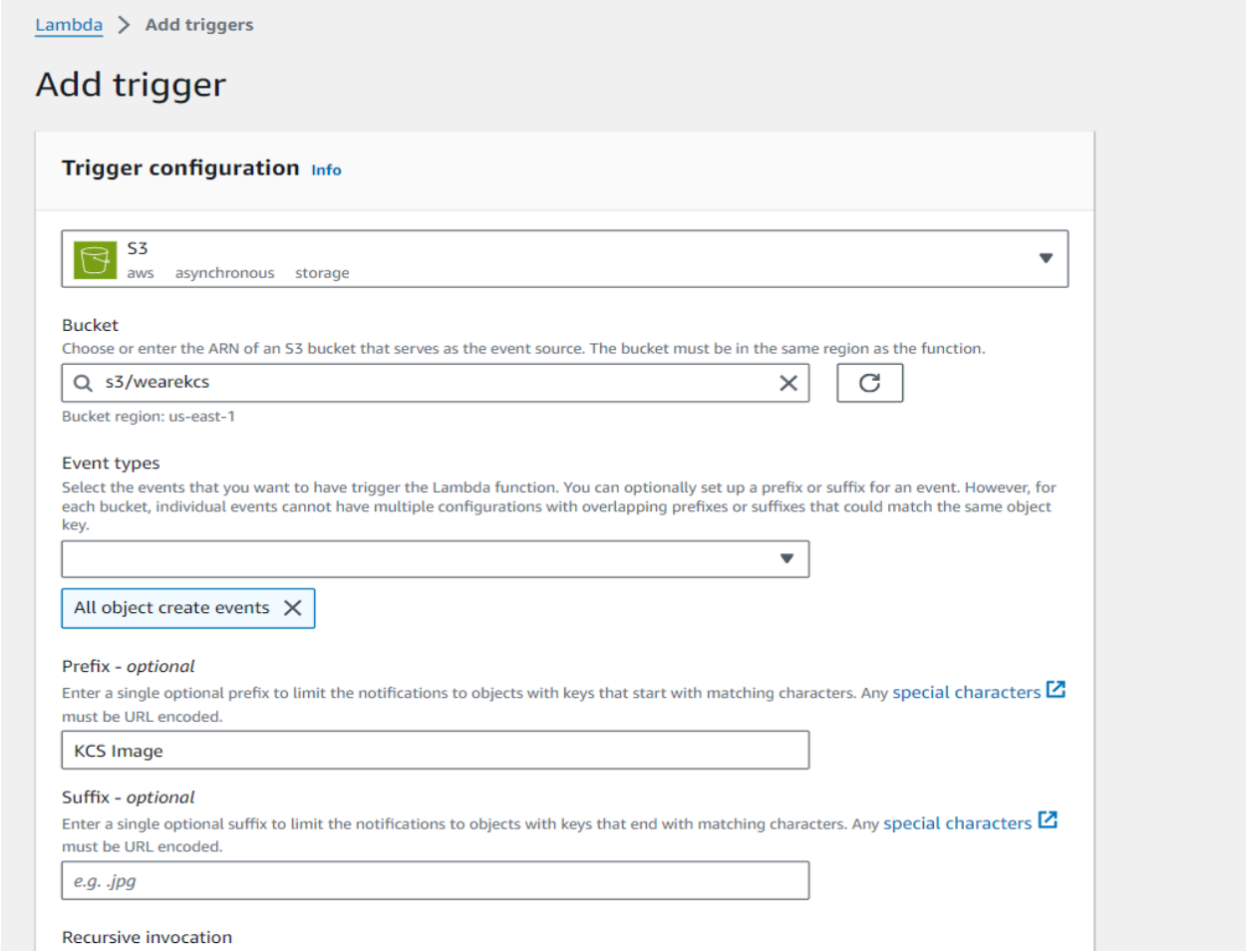


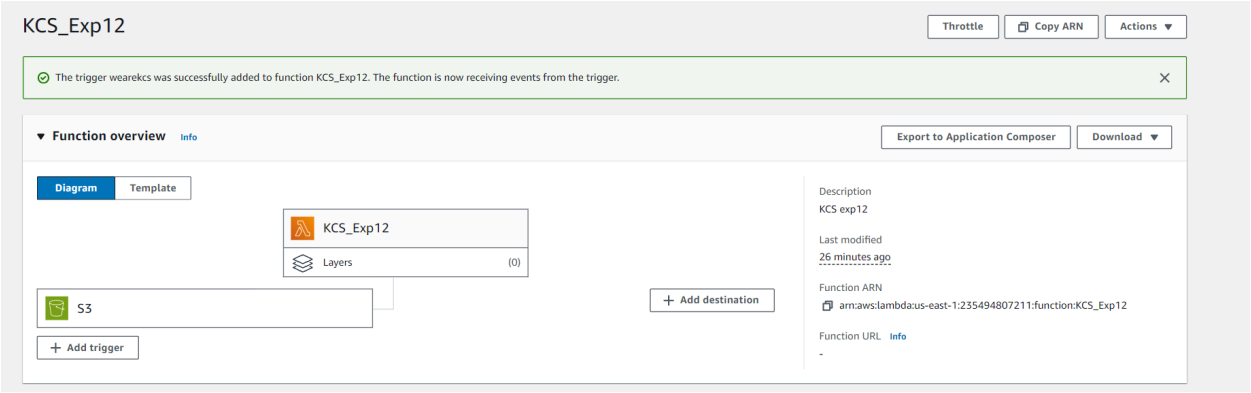
Step 7: Now In the Lambda function click on add tigger



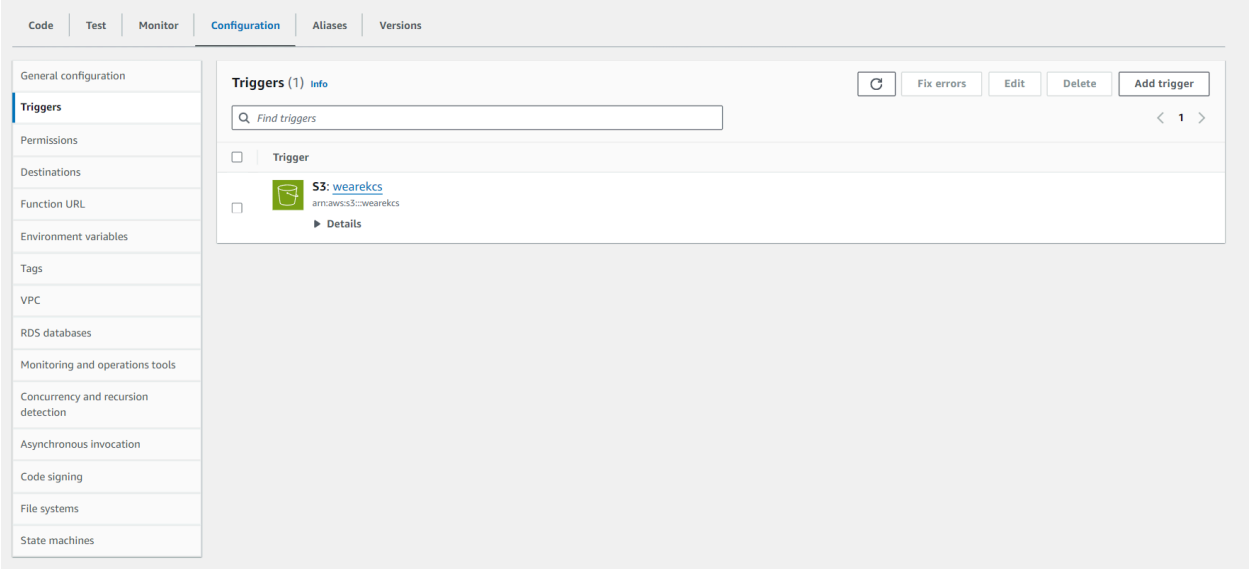
Now select the source as S3 then select the bucket name from the dropdown, keep other things to default and also you can add prefix to image.

Name: Kshitij Hundre Div:D15C Roll No:18

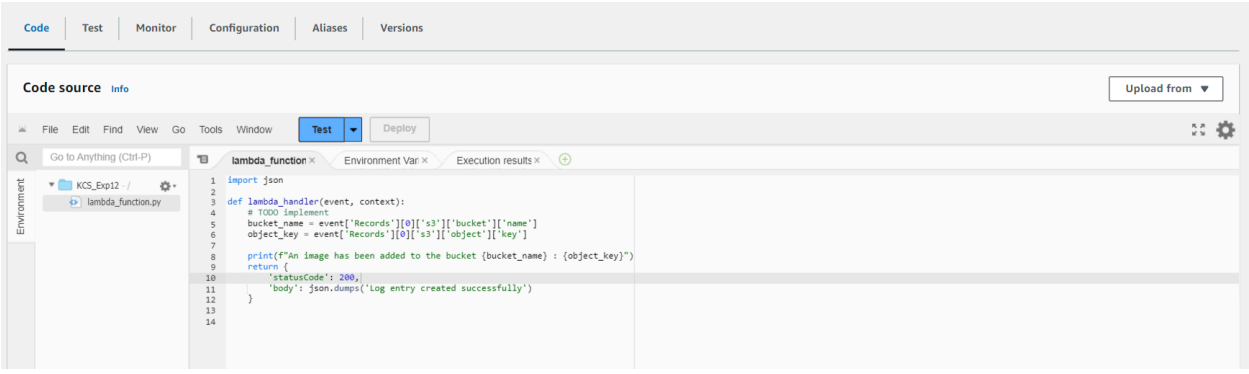


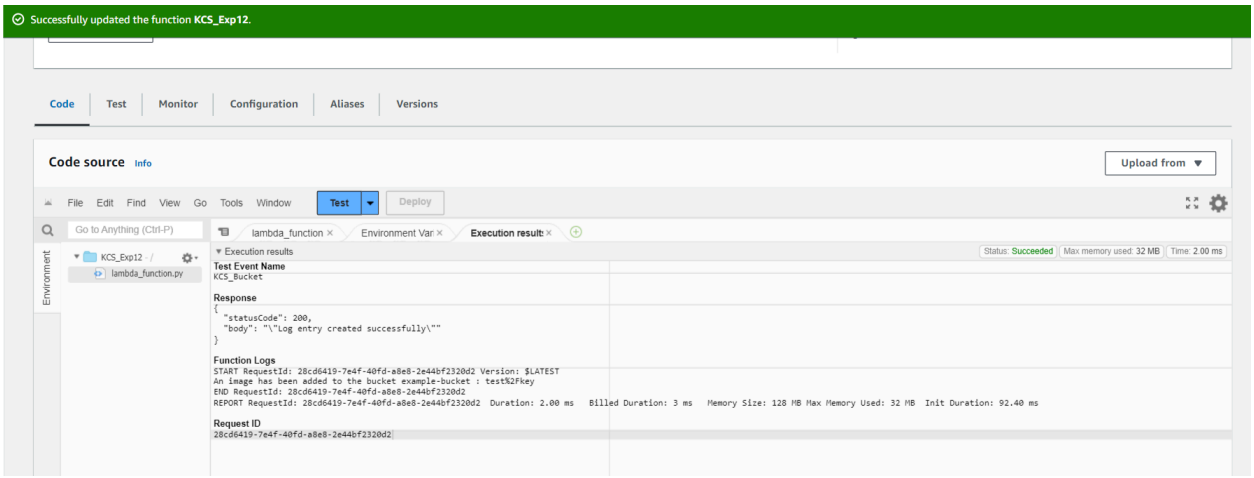


Name: Kshitij Hundre Div:D15C Roll No:18



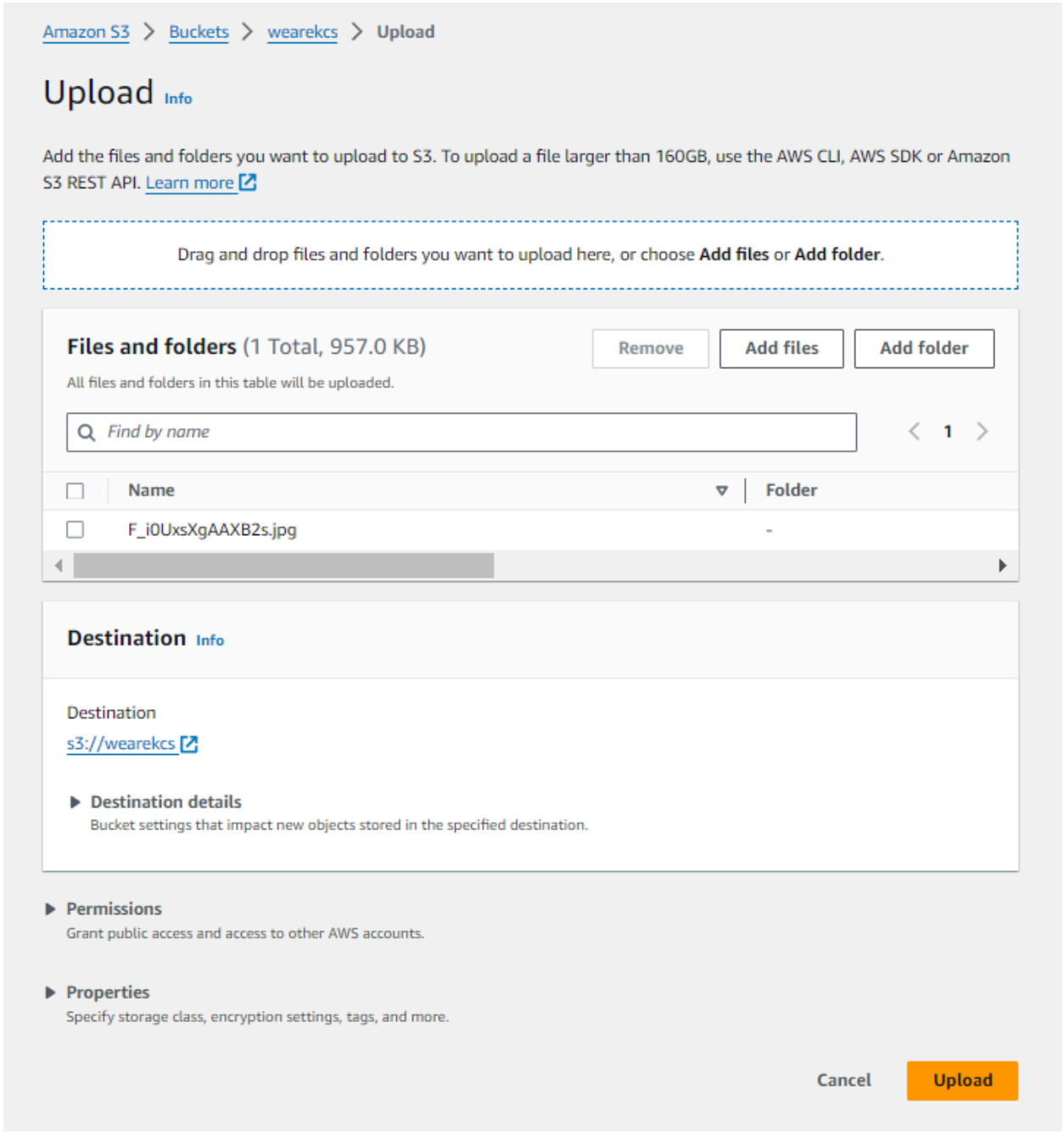
Step 8: Now Write code that logs a message like “An Image has been added” when triggered. Save the file and click on deploy.



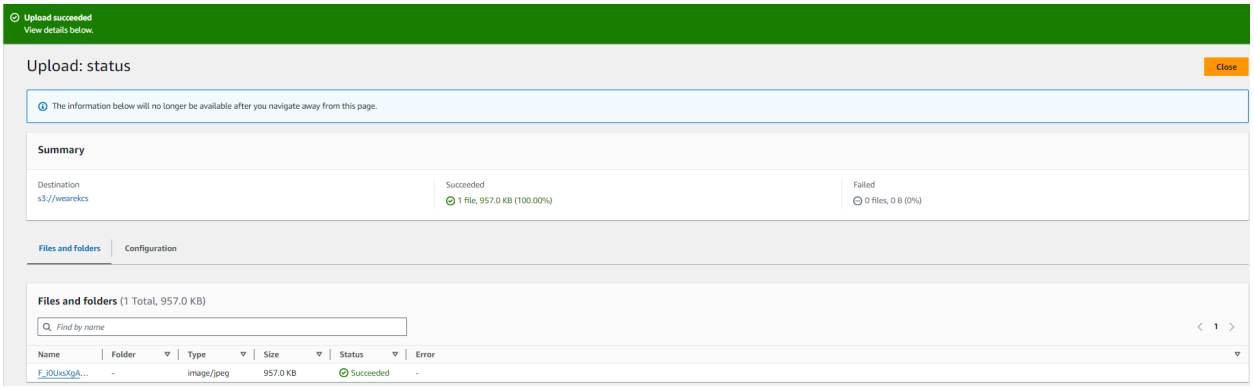


Step 9: Now upload any image to the bucket.

Name: Kshitij Hundre Div:D15C Roll No:18



Name: Kshitij Hundre Div:D15C Roll No:18



Step 10: Now to click on test in lambda to check whether it is giving log when image is added to S3

Step 11: Now Lets see the log on Cloud watch.To see it go to monitor section and then click on view cloudwatch logs.