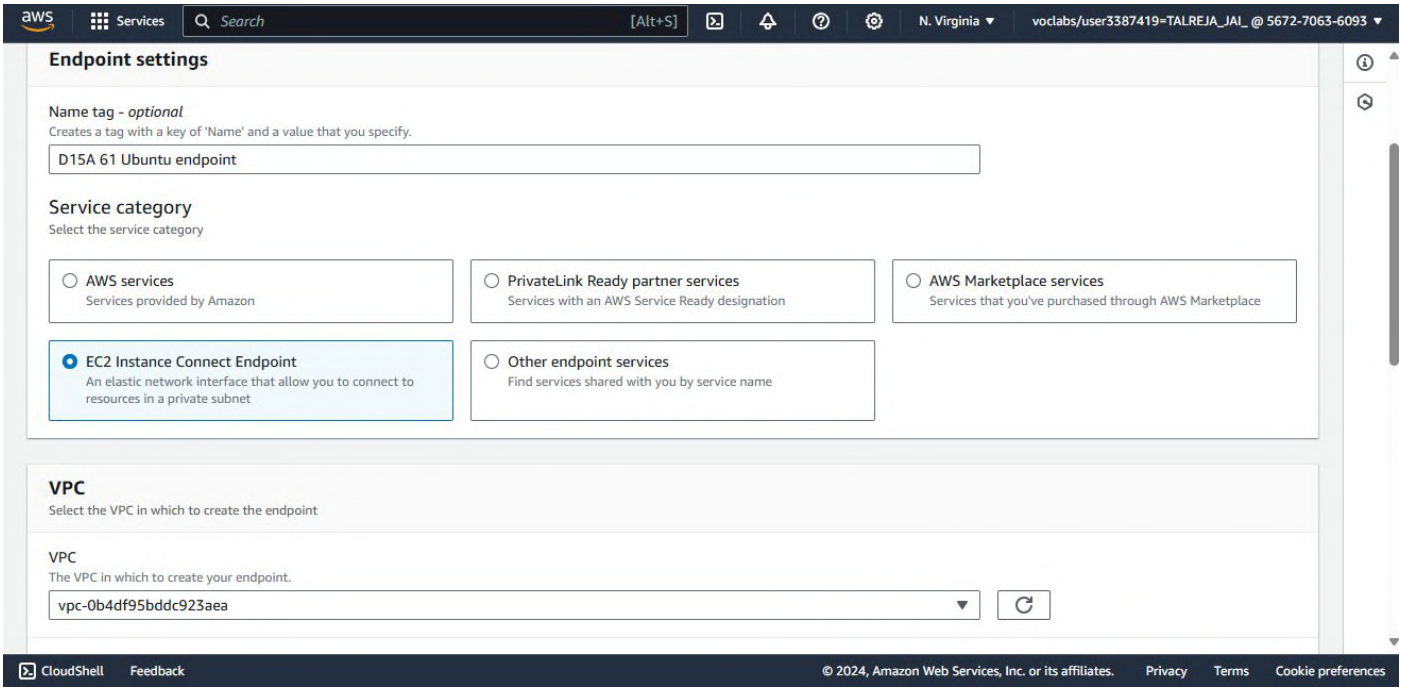
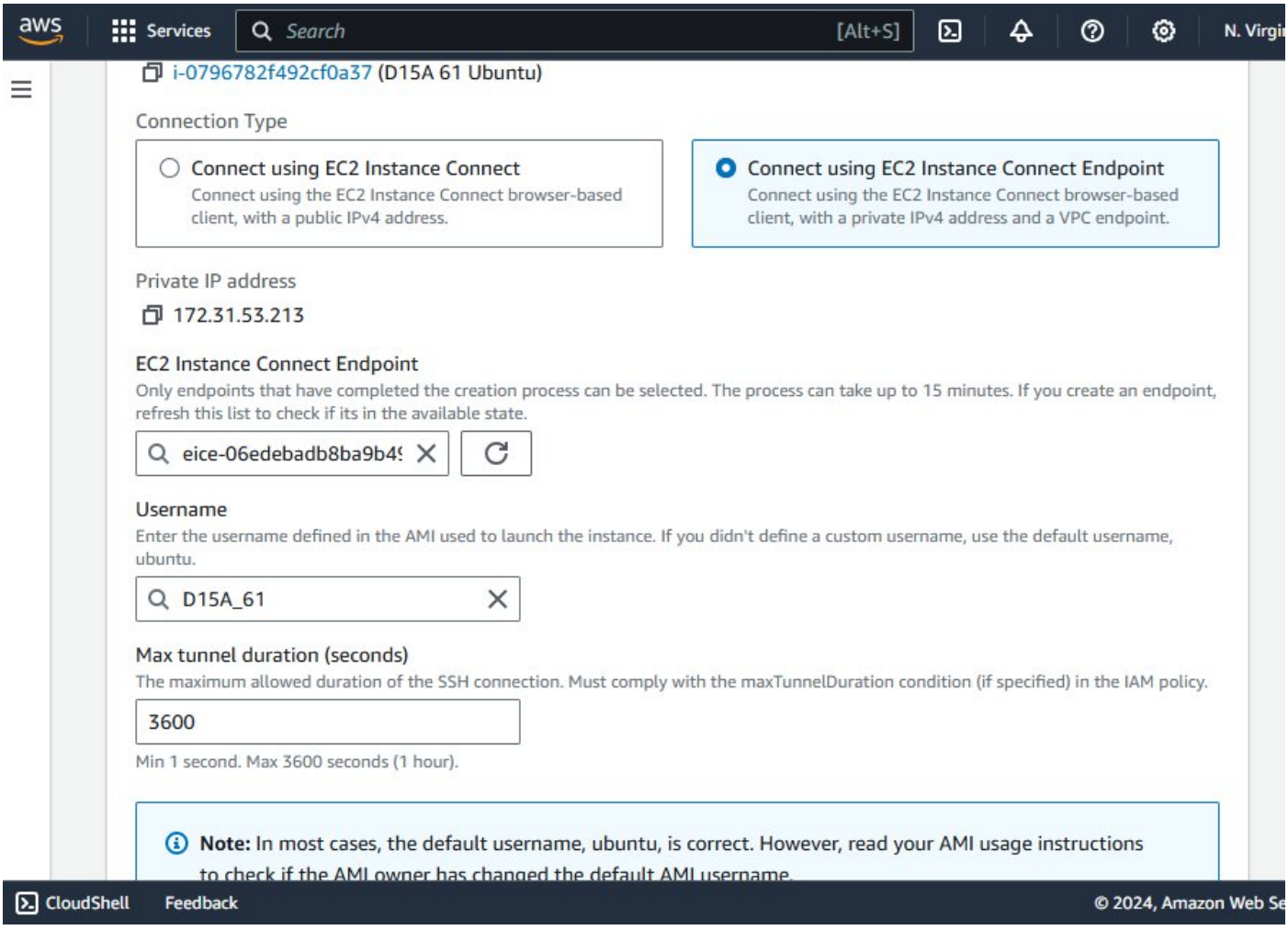
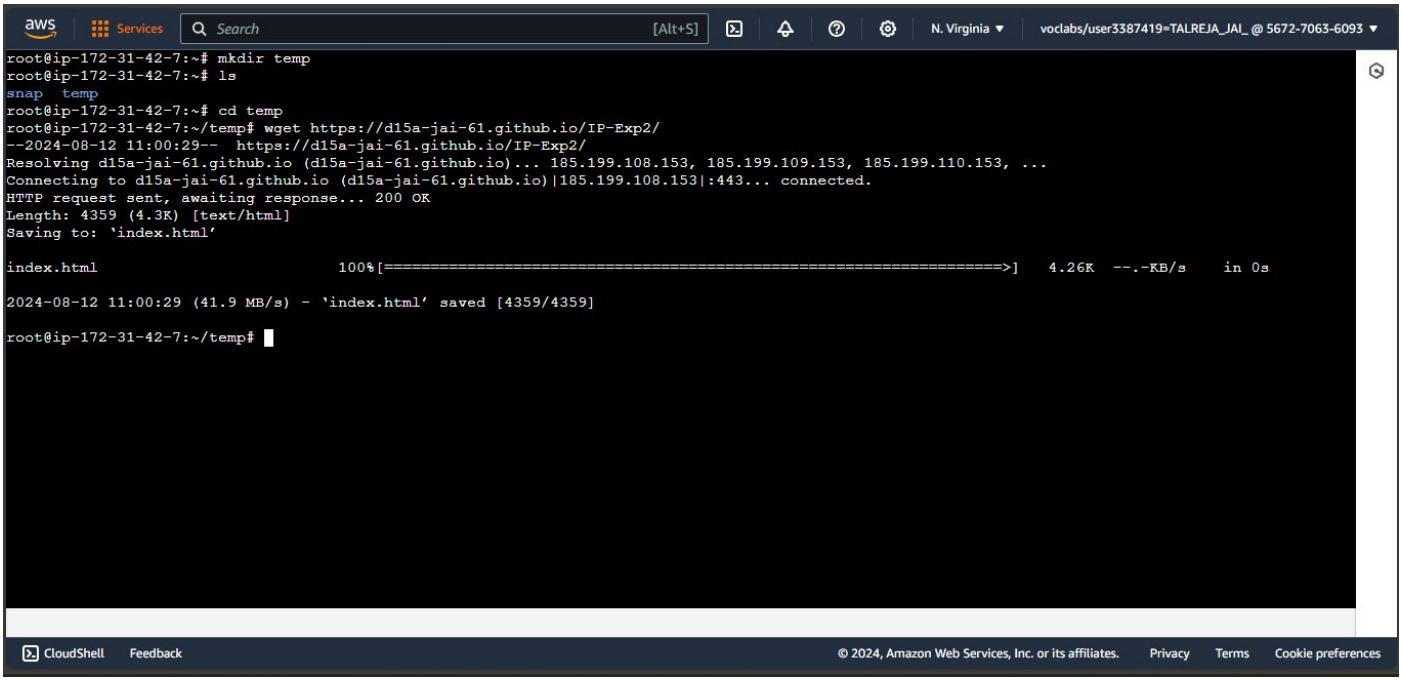
Jai Talreja 61 D15A

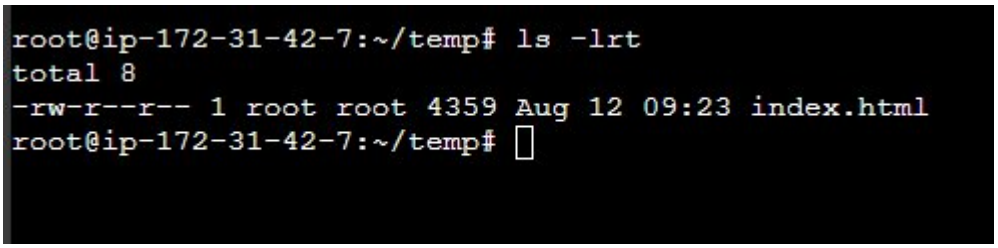
Screenshots of steps for creating and launching EC2 instance :-

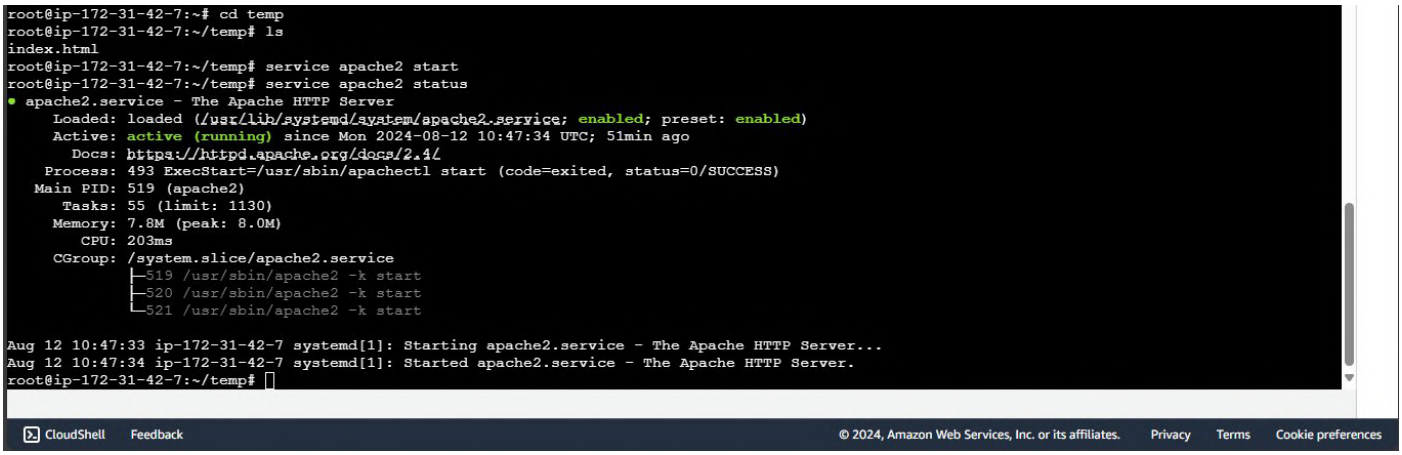




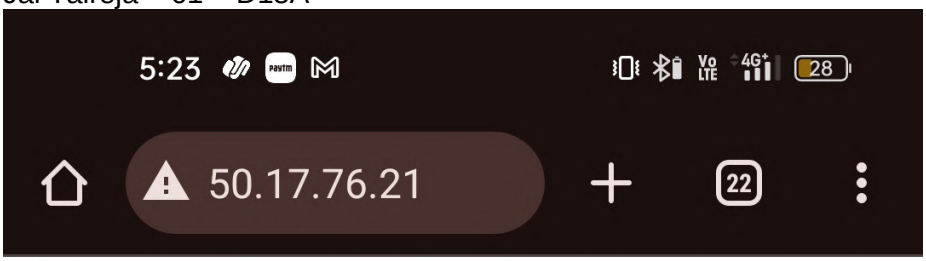
Jai Talreja 61 D15A







Jai Talreja 61 D15A



Petco

Your pet deserves the beat treatment in the world !

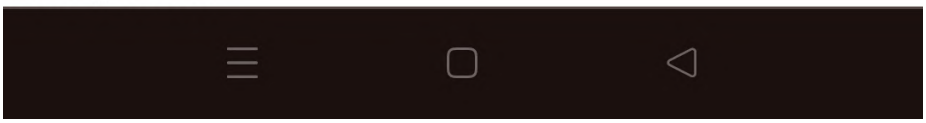
About us

Petco is a category-defining health and wellness company focused on improving the lives of pets, pet parents and our own Petco partners. Since our founding in 1965, we've been trailblazing new standards in pet care, delivering comprehensive wellness solutions through our products and services, and creating communities that deepen the pet-pet parent bond.

We employ more than 29,000 partners nationwide and operate more than 1,500 Petco locations across the U.S., Mexico and Puerto Rico — including a growing network of more than 200 in-store veterinary hospitals — and offer a complete online resource for pet health and wellness at petco.com and on the Petco app.

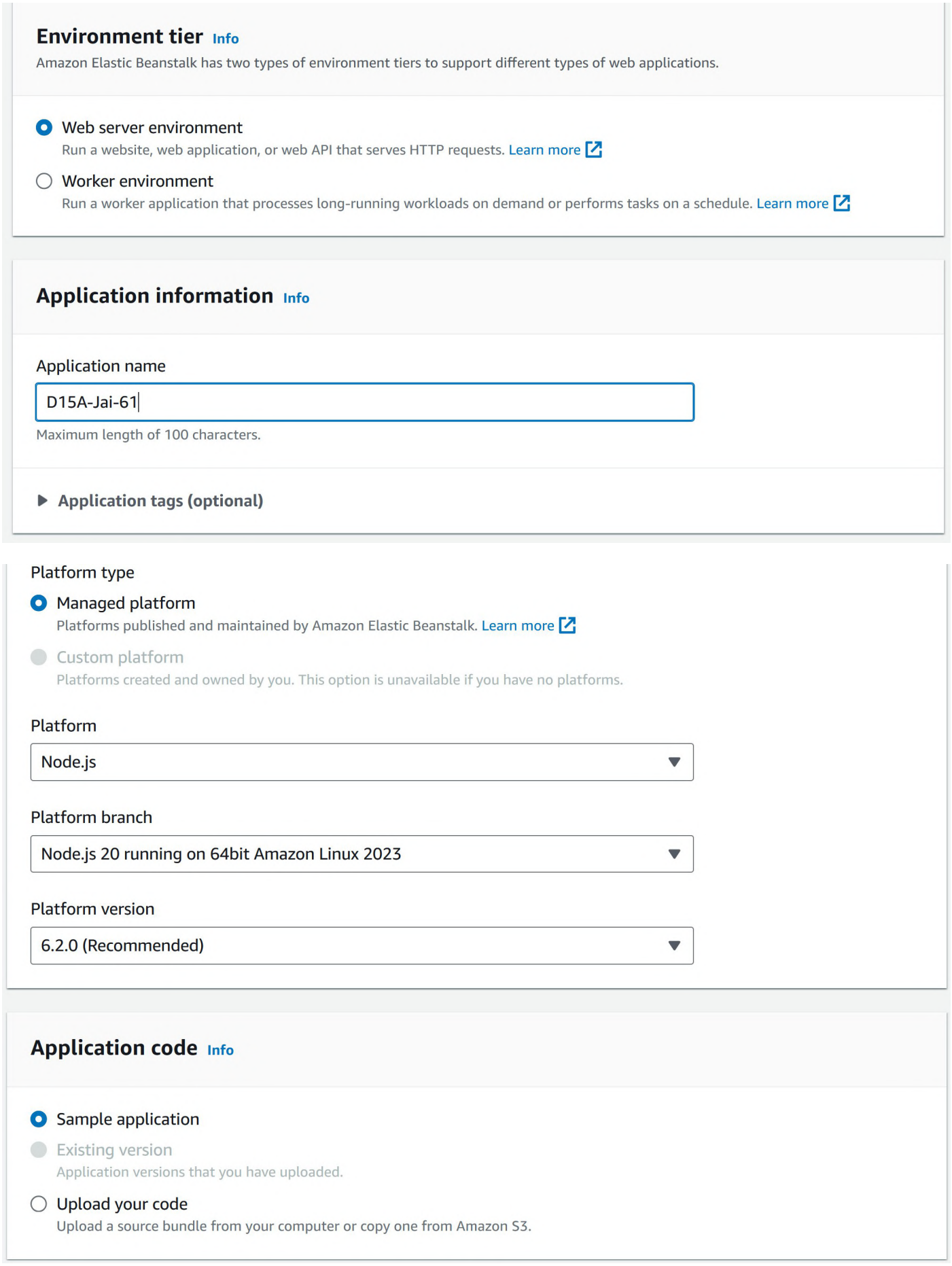
In tandem with Petco Love, an independent nonprofit organization, we have helped find homes for more than 7 million animals through in-store adoption events.

Working at Petco



Jai Talreja 61 D15A

Screenshots for setting up and deploying Elastic Beanstalk application :-



☐	us-east-1f	subnet-03b751cf5…	172.31.64.0/20
☐	us-east-1a	subnet-05ff6de98…	172.31.16.0/20
☐	us-east-1c	subnet-089dbc2d1…	172.31.0.0/20
☐	us-east-1b	subnet-0d4591b4f…	172.31.32.0/20
☐	us-east-1d	subnet-0f341fff62…	172.31.80.0/20

Enable database

Restore a snapshot - *optional*

Restore an existing snapshot from a previously used database.

Snapshot

None

Database settings

Choose an engine and instance type for your environment's database.

Engine

Engine version

Tags

Apply up to 50 tags. You can use tags to group and filter your resources. A tag is a key-value pair. The key must be unique within the resource and is case-sensitive. Learn more

No tags associated with the resource.

Add new tag

You can add 50 more tags.

Cancel Skip to review Previous Next

Configure updates, monitoring, and logging - *optional* Info

Monitoring Info

Health reporting

Enhanced health reporting provides free real-time application and operating system monitoring of the instances and other resources in your environment. The **EnvironmentHealth** custom metric is provided free with enhanced health reporting. Additional charges apply for each custom metric. For more information, see Amazon CloudWatch Pricing

System

- (•) Basic
- ( ) Enhanced

Health event streaming to CloudWatch Logs

Configure Elastic Beanstalk to stream environment health events to CloudWatch Logs. You can set the retention up to a maximum of ten years and configure Elastic Beanstalk to delete the logs when you terminate your environment.

Log streaming

- [ ] Activated (standard CloudWatch charges apply.)

Retention

7

Lifecycle

Managed platform updates Info

Activate managed platform updates to apply platform updates automatically during a weekly maintenance window that you choose. Your application stays available during the update process.

Managed updates

- [ ] Activated

Weekly update window

Tuesday at 01 : 14 UTC

Update level

Minor and patch

Instance replacement

If enabled, an instance replacement will be scheduled if no other updates are available.

- [ ] Activated

Ignore health check
false

Instance replacement
false

Platform software

Lifecycle
false

Log streaming
Deactivated

Proxy server
nginx

Logs retention
7

Rotate logs
Deactivated

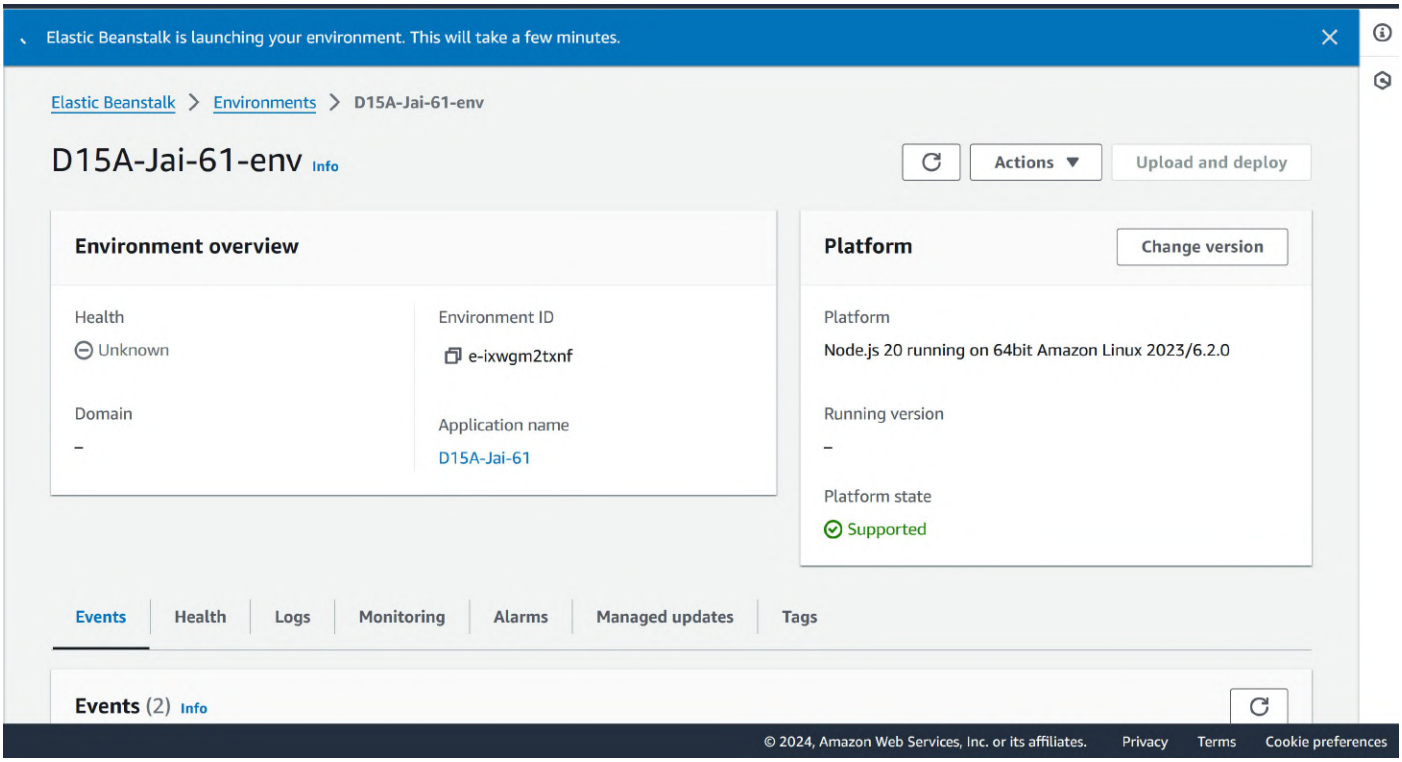
Update level
minor

X-Ray enabled
Deactivated

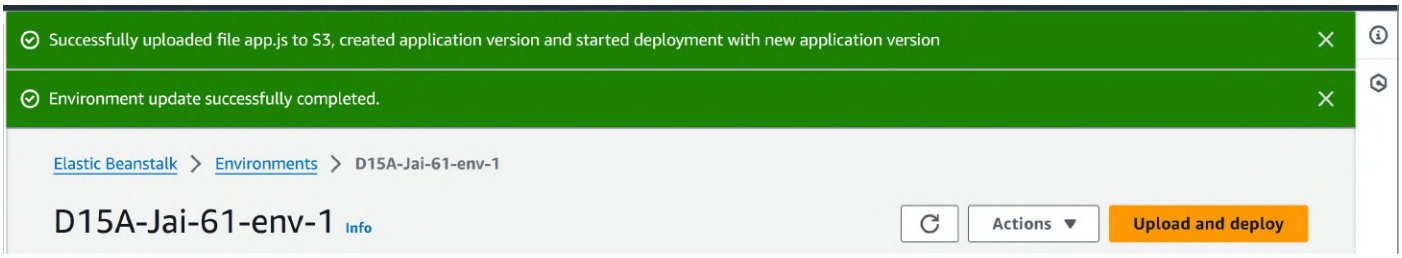
Environment properties

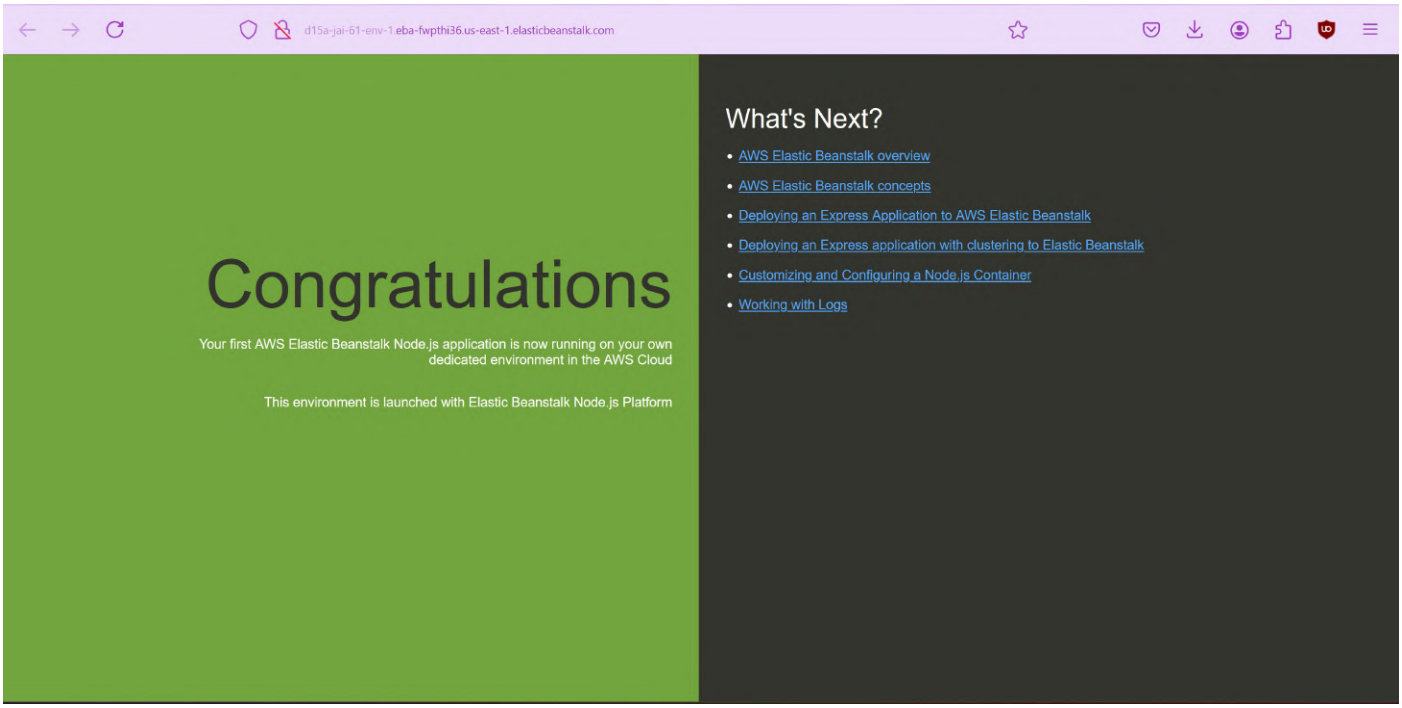
Key	Value
No environment properties	
There are no environment properties defined	

Cancel Previous Submit



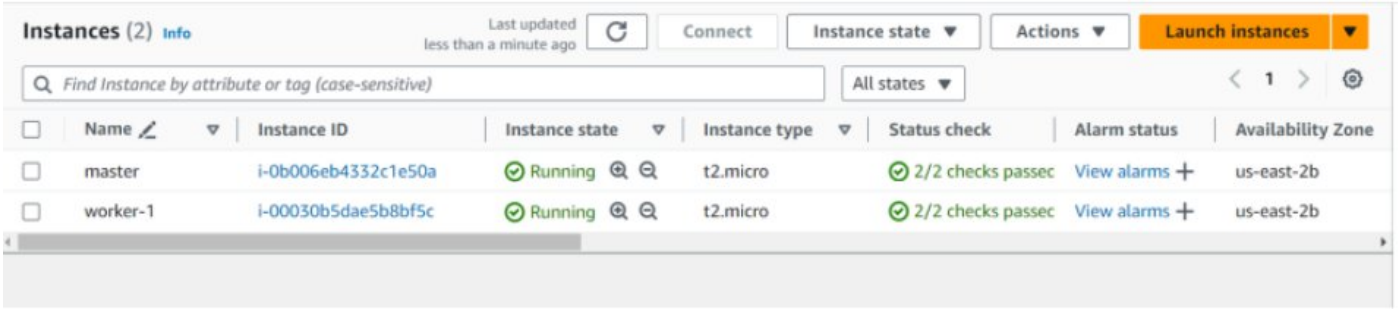
Jai Talreja 61 D15A

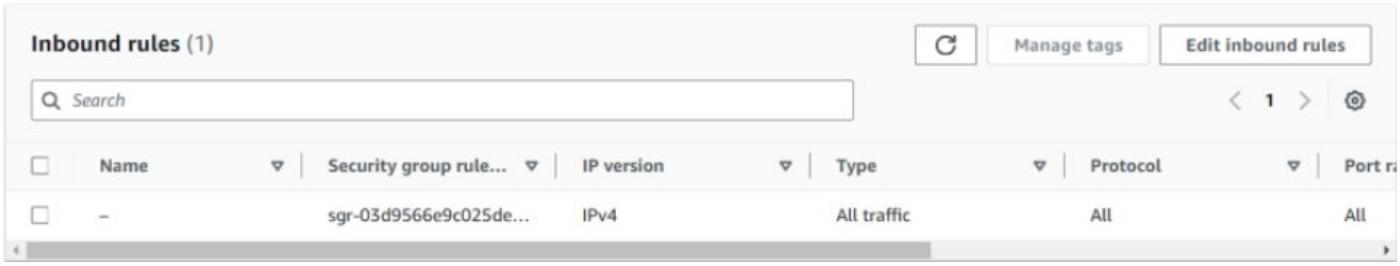




Jai Talreja 61 D15A

Screenshots for Experiment 3.






```
ubuntu@ip-172-31-18-146:~$ sudo hostnamectl set-hostname worker
```

```
ubuntu@master:~$ sudo apt-get update
Hit:1 http://us-east-2.ec2.archive.ubuntu.com/ubuntu noble InRelease
Get:2 http://us-east-2.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease [126 kB]
Get:3 http://us-east-2.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease [126 kB]
Get:4 http://security.ubuntu.com/ubuntu noble-security InRelease [126 kB]
Get:5 http://us-east-2.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 Packages [15.0 MB]
Get:6 http://us-east-2.ec2.archive.ubuntu.com/ubuntu noble/universe Translation-en [5982 kB]
```

```
ubuntu@master:~$ sudo apt-get install docker.io
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  bridge-utils containerd dns-root-data dnsmasq-base pigz runc ubuntu-fan
Suggested packages:
```

```
Processing triggers for dbus (1.14.10-4ubuntu4) ...
Processing triggers for man-db (2.12.0-4build2) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
```

Jai Talreja 61 D15A

```
ubuntu@master:~$  sudo systemctl enable docker
ubuntu@master:~$ sudo systemctl status docker
● docker.service - Docker Application Container Engine
     Loaded: loaded (/usr/lib/systemd/system/docker.service; enabled; preset: enabled)
     Active: active (running) since Sun 2024-09-15 15:23:25 UTC; 5min ago
TriggeredBy: ● docker.socket
       Docs: https://docs.docker.com
   Main PID: 2653 (dockerd)
      Tasks: 8
     Memory: 31.8M (peak: 33.4M)
        CPU: 299ms
     CGroup: /system.slice/docker.service
             └─2653 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd.sock
```

```
ubuntu@worker:~$ sudo apt-get update
Hit:1 http://us-east-2.ec2.archive.ubuntu.com/ubuntu noble InRelease
Hit:2 http://us-east-2.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:3 http://us-east-2.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease
Hit:4 http://security.ubuntu.com/ubuntu noble-security InRelease
Reading package lists... Done
```

```
ubuntu@master:~$ sudo apt-get install -y apt-transport-https ca-certificates curl gpg
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
ca-certificates is already the newest version (20240203).
ca-certificates set to manually installed.
gpg is already the newest version (2.4.4-2ubuntu17).
gpg set to manually installed.
The following NEW packages will be installed:
  apt-transport-https
The following packages will be upgraded:
  curl libcurl3t64-gnutls libcurl4t64
3 upgraded, 1 newly installed, 0 to remove and 130 not upgraded.
Need to get 904 kB of archives.
After this operation, 38.9 kB of additional disk space will be used.
```

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

```
ubuntu@master:~$ sudo apt-get install -y kubelet kubeadm kubectl
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  conntrack cri-tools kubernetes-cni
The following NEW packages will be installed:
  conntrack cri-tools kubeadm kubectl kubelet kubernetes-cni
0 upgraded, 6 newly installed, 0 to remove and 130 not upgraded.
Need to get 87.4 MB of archives.
After this operation, 314 MB of additional disk space will be used.
Get:1 http://us-east-2.ec2.archive.ubuntu.com/ubuntu noble/main amd64 conntrack amd64 1:1.4.8
Get:2 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.31/deb
Get:3 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.31/deb
Get:4 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.31/deb
Get:5 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.31/deb
Get:6 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.31/deb
Fetched 87.4 MB in 1s (70.9 MB/s)
Selecting previously unselected package conntrack.
(Reading database ... 68112 files and directories currently installed.)
```

```
ubuntu@master:~$ sudo apt-mark hold kubelet kubeadm kubectl
kubelet set on hold.
kubeadm set on hold.
kubectl set on hold.
ubuntu@master:~$
```

```
ubuntu@master:~$ sudo kubeadm init --pod-network-cidr=10.244.0.0/16 --ignore-preflight-errors=all
[init] Using Kubernetes version: v1.31.0
[preflight] Running pre-flight checks
        [WARNING NumCPU]: the number of available CPUs 1 is less than the required 2
        [WARNING Mem]: the system RAM (957 MB) is less than the minimum 1700 MB
        [WARNING FileExisting-socat]: socat not found in system path
```

```
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 172.31.25.196:6443 --token wb5xks.0x5kyf3rpycusd60 \
        --discovery-token-ca-cert-hash sha256:4fa35dc1d8d64ac6f31ebe712f3ec1d5c8a38ba8a8fb316579c08b671eabbd06
```

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

```
ubuntu@master:~$ kubectl apply -f https://github.com/flannel-io/flannel/releases/latest/download/kube-flannel.yml
namespace/kube-flannel created
serviceaccount/flannel created
clusterrole.rbac.authorization.k8s.io/flannel created
clusterrolebinding.rbac.authorization.k8s.io/flannel created
configmap/kube-flannel-cfg created
daemonset.apps/kube-flannel-ds created
```

```
ubuntu@worker:~$ sudo kubeadm join 172.31.25.196:6443 --token wb5xks.0x5kyf3rpycusd60 --discovery-token-ca-cert-hash sha256:4fa35dc1d8d64
ac6f31ebe712f3ec1d5c8a38ba8a8fb316579c08b671eabbd06 --ignore-preflight-errors=all
[preflight] Running pre-flight checks
        [WARNING FileAvailable--etc-kubernetes-kubelet.conf]: /etc/kubernetes/kubelet.conf already exists
        [WARNING FileExisting-socat]: socat not found in system path
        [WARNING Port-10250]: Port 10250 is in use
        [WARNING FileAvailable--etc-kubernetes-pki-ca.crt]: /etc/kubernetes/pki/ca.crt already exists
[preflight] Reading configuration from the cluster...
[preflight] FYI: You can look at this config file with 'kubectl -n kube-system get cm kubeadm-config -o yaml'
[kubelet-start] Writing kubelet configuration to file "/var/lib/kubelet/config.yaml"
[kubelet-start] Writing kubelet environment file with flags to file "/var/lib/kubelet/kubeadm-flags.env"
[kubelet-start] Starting the kubelet
[kubelet-check] Waiting for a healthy kubelet at http://127.0.0.1:10248/healthz. This can take up to 4m0s
[kubelet-check] The kubelet is healthy after 503.838994ms
[kubelet-start] Waiting for the kubelet to perform the TLS Bootstrap

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

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

```
ubuntu@master:~$ kubectl get pods --all-namespaces
NAMESPACE      NAME                              READY   STATUS             RESTARTS         AGE
kube-flannel   kube-flannel-ds-8z8zq             1/1     Running            1 (6m12s ago)    21m
kube-flannel   kube-flannel-ds-src45             1/1     Running            2 (5m16s ago)    16m
kube-system    coredns-7c65d6cfc9-4lmp4          1/1     Running            1 (6m12s ago)    30m
kube-system    coredns-7c65d6cfc9-fnrrt          1/1     Running            1 (6m12s ago)    30m
kube-system    etcd-master                       1/1     Running            1 (6m12s ago)    30m
kube-system    kube-apiserver-master             1/1     Running            1 (6m12s ago)    30m
kube-system    kube-controller-manager-master    1/1     Running            1 (6m12s ago)    30m
kube-system    kube-proxy-hw6zm                  0/1     CrashLoopBackOff   9 (29s ago)      16m
kube-system    kube-proxy-x6lmb                  1/1     Running            11 (2m8s ago)    30m
kube-system    kube-scheduler-master             1/1     Running            1 (6m12s ago)    30m
ubuntu@master:~$ kubectl delete node worker
node "worker" deleted
ubuntu@master:~$ kubectl get nodes
NAME     STATUS   ROLES           AGE   VERSION
master   Ready    control-plane   35m   v1.31.1
worker   Ready    <none>          87s   v1.31.1
```

Jai Talreja 61 D15A

Screenshots for experiment 4

```
ubuntu@master:~$ kubectl create deployment nginx --image=nginx
deployment.apps/nginx created
ubuntu@master:~$
```

```
ubuntu@master:~$ sudo nano deploy.yaml
```

```
ubuntu@master:~$ cat deploy.yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx-deployment
  labels:
    app: nginx
spec:
  replicas: 3
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
      - name: nginx
        image: nginx:1.14.2
        ports:
        - containerPort: 80
ubuntu@master:~$
```

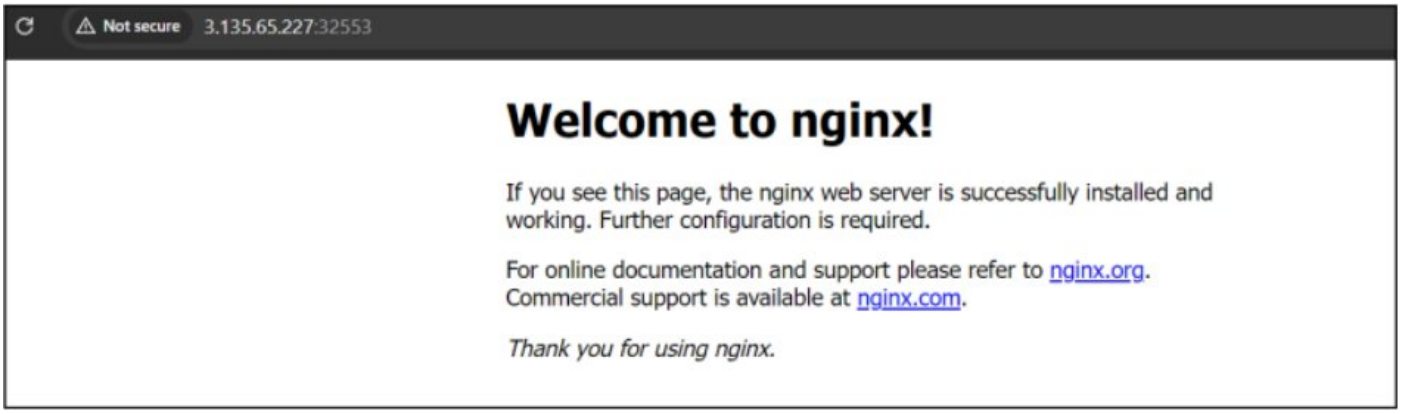
```
ubuntu@master:~$ kubectl create -f deploy.yaml
deployment.apps/nginx-deployment created
```

Jai Talreja 61 D15A

```
ubuntu@master:~$ kubectl get deploy
NAME               READY   UP-TO-DATE   AVAILABLE   AGE
nginx              1/1     1            1           40m
nginx-deployment   3/3     3            3           23m
ubuntu@master:~$
```

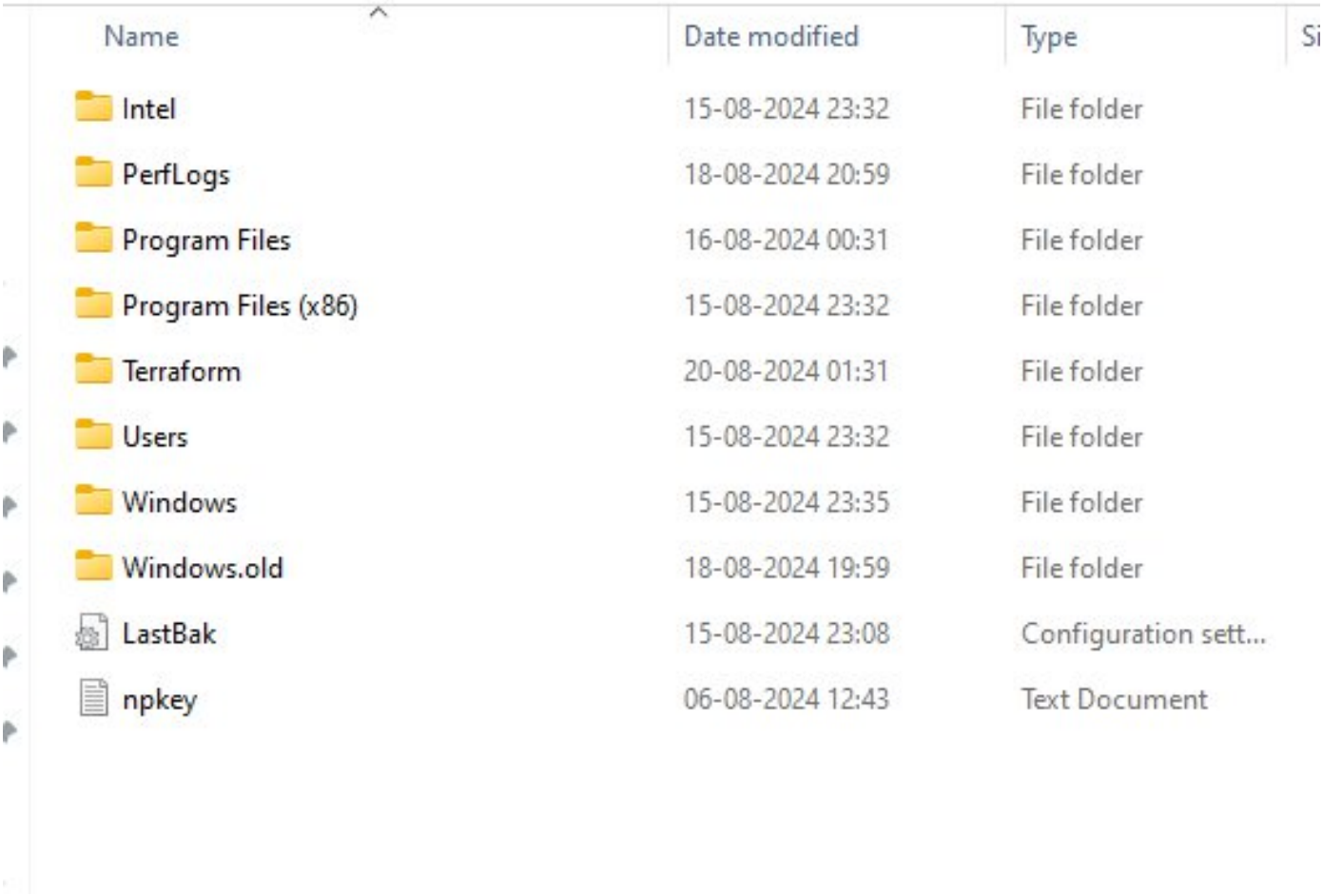
```
ubuntu@master:~$ kubectl expose deployment.apps/nginx-deployment \
> --type="LoadBalancer"
service/nginx-deployment exposed
ubuntu@master:~$
```

```
ubuntu@master:~$ kubectl get svc
NAME               TYPE           CLUSTER-IP      EXTERNAL-IP   PORT(S)        AGE
kubernetes         ClusterIP      10.96.0.1       <none>        443/TCP        140m
nginx-deployment   LoadBalancer   10.96.232.159   <pending>     80:32553/TCP   80s
```



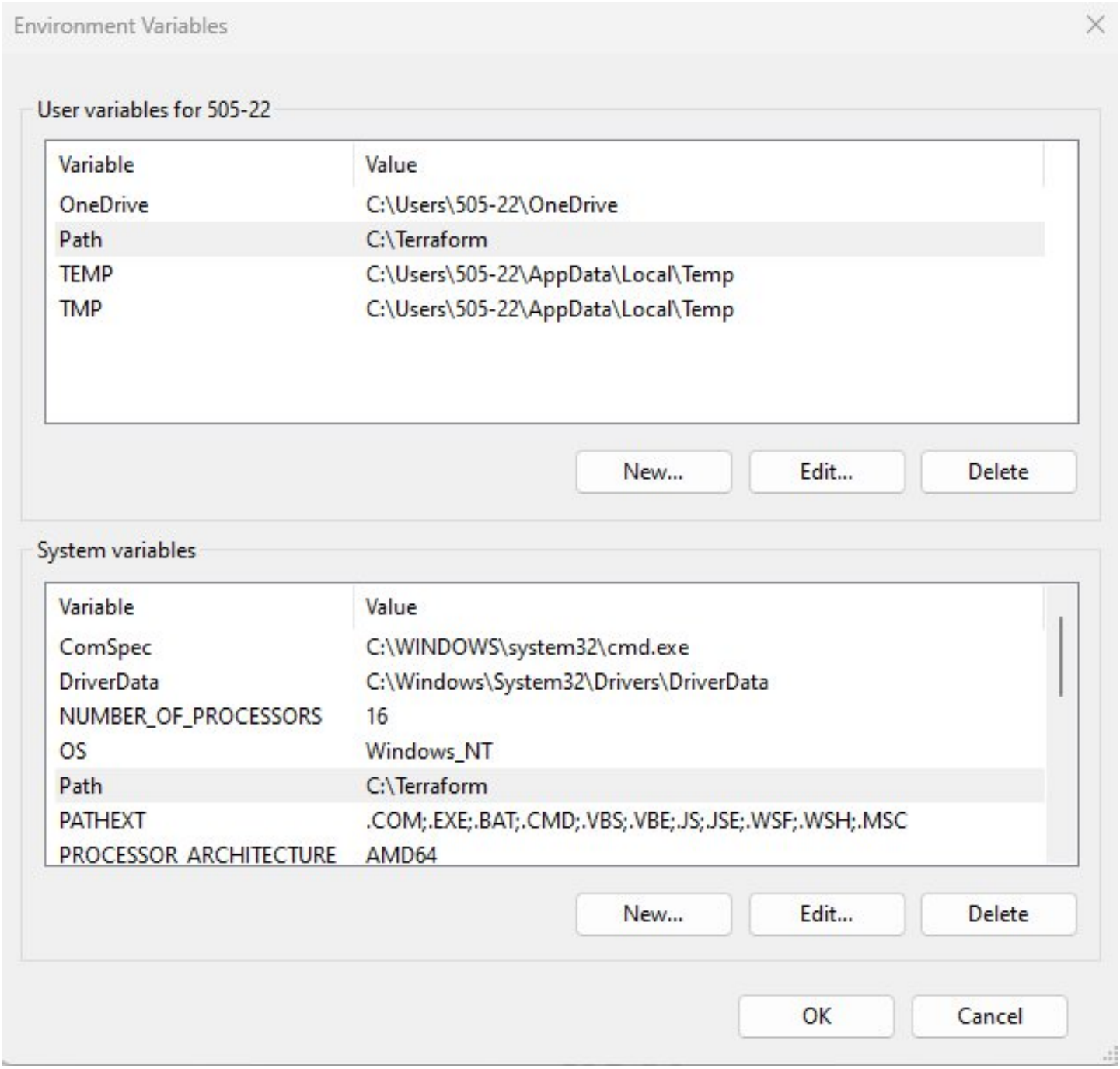
Jai Talreja 61 D15A

Screenshots for installing, configuring and running Terraform using PowerShell :-

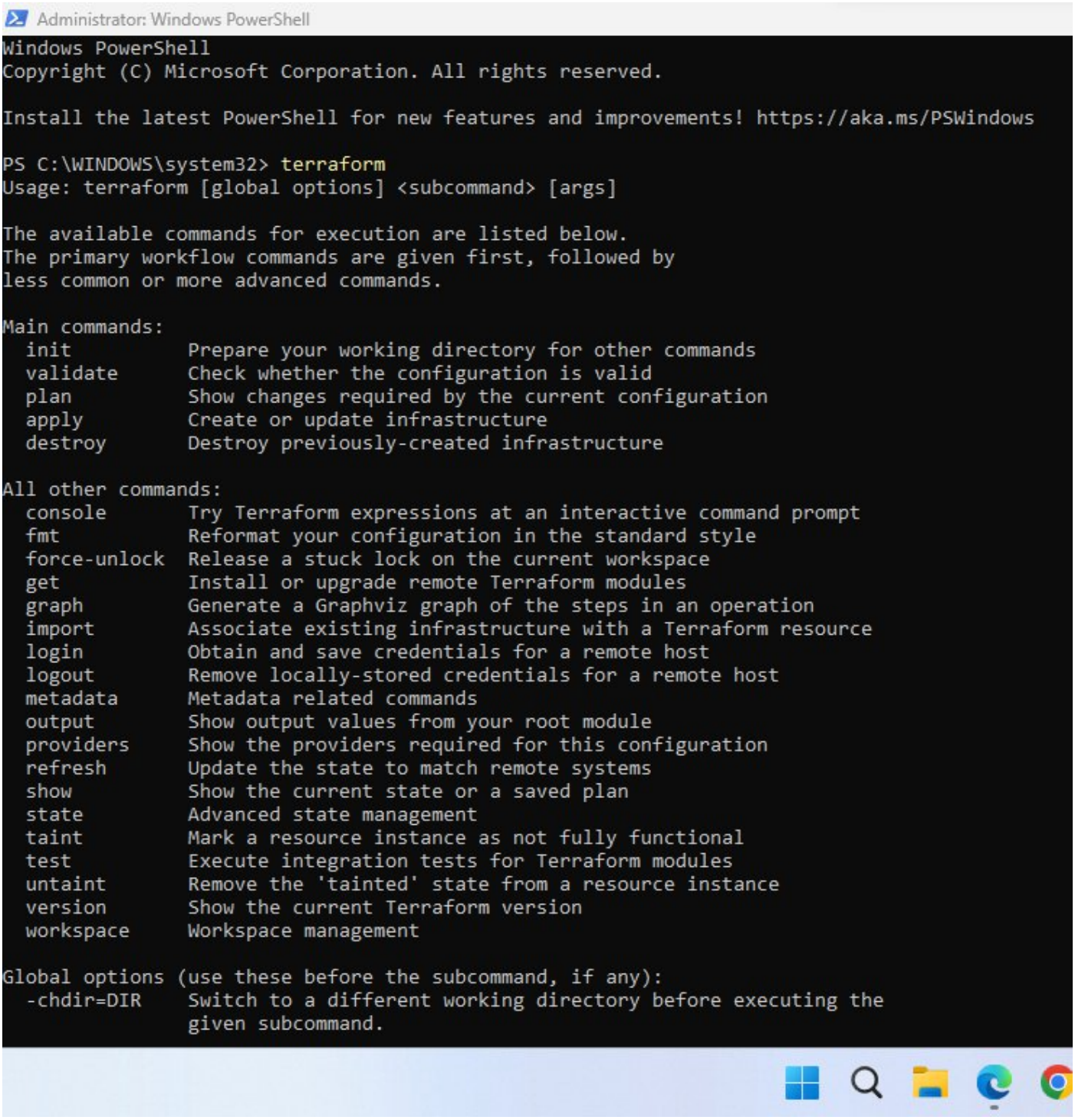


Name	Date modified	Type	Si
Intel	15-08-2024 23:32	File folder	
PerfLogs	18-08-2024 20:59	File folder	
Program Files	16-08-2024 00:31	File folder	
Program Files (x86)	15-08-2024 23:32	File folder	
Terraform	20-08-2024 01:31	File folder	
Users	15-08-2024 23:32	File folder	
Windows	15-08-2024 23:35	File folder	
Windows.old	18-08-2024 19:59	File folder	
LastBak	15-08-2024 23:08	Configuration sett...	
npkey	06-08-2024 12:43	Text Document	

Jai Talreja 61 D15A



Jai Talreja 61 D15A



```
jai@fedora:~/Documents/terraform_scripts/docker$ sudo systemctl start docker
[sudo] password for jai:
jai@fedora:~/Documents/terraform_scripts/docker$ terraform plan

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

Terraform will perform the following actions:

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

```
jai@fedora:~/Documents/terraform_scripts/docker$ terraform apply
docker_image.ubuntu: Refreshing state... [id=sha256:b1e9cef3f2977f8bdd19eb9ae04f

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

Terraform will perform the following actions:

  # docker_container.foo will be created
  + resource "docker_container" "foo" {
      + attach           = false
      + bridge           = (known after apply)
      + command          = (known after apply)
      + container_logs   = (known after apply)
      + entrypoint       = (known after apply)
      + env              = (known after apply)
      + exit_code        = (known after apply)
      + gateway          = (known after apply)
      + hostname         = (known after apply)
      + id               = (known after apply)
      + image            = "sha256:b1e9cef3f2977f8bdd19eb9ae04f83b315f80fe4f5c56
      + init             = (known after apply)
      + ip_address       = (known after apply)
      + ip_prefix_length = (known after apply)
      + ipc_mode         = (known after apply)
      + log_driver       = (known after apply)
      + logs             = false
      + must_run         = true
      + name             = "foo"
      + network_data     = (known after apply)
      + read_only        = false
      + remove_volumes   = true
      + restart          = "no"
      + rm               = false
      + runtime          = (known after apply)
      + security_opts    = (known after apply)
      + shm_size         = (known after apply)
      + start            = true
      + stdin_open       = true
      + stop_signal      = (known after apply)
      + stop_timeout     = (known after apply)
      + tty              = true

      + healthcheck (known after apply)
```

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

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

  Enter a value: yes

docker_container.foo: Creating...
docker_container.foo: Creation complete after 0s [id=dfe9d545df9f1bf34ba0dcb6aa106e9c52785ea49732a65708f46b1066af2fe8]

Apply complete! Resources: 1 added, 0 changed, 0 destroyed.
jai@fedora:~/Documents/terraform_scripts/docker$
```

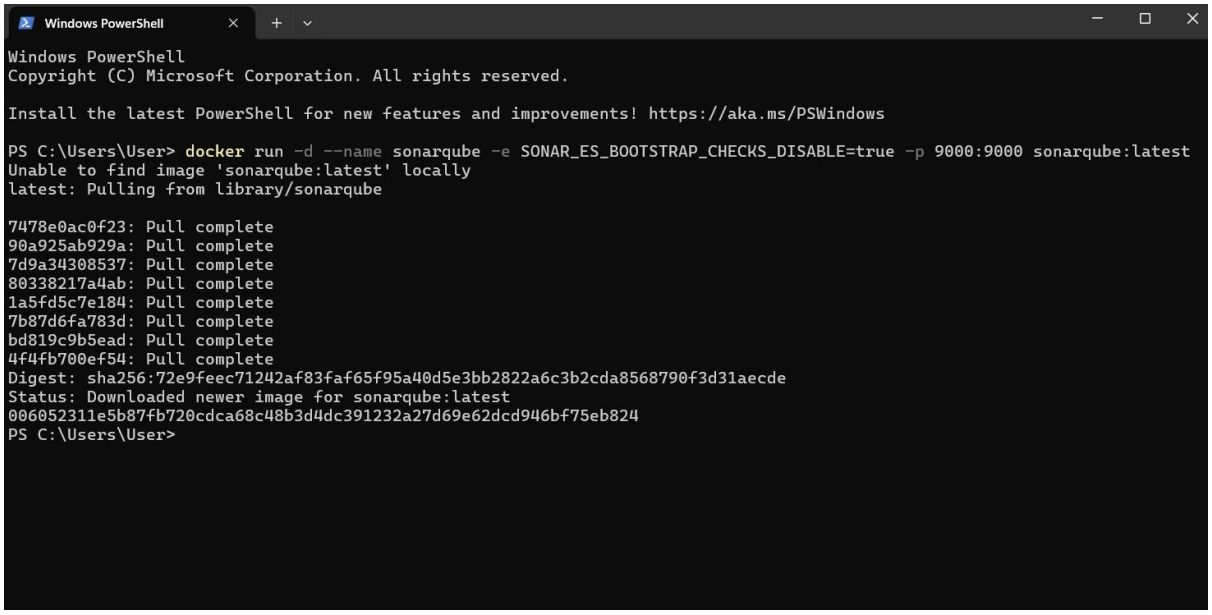
```
jai@fedora:~/Documents/terraform_scripts/docker$ terraform destroy
docker_image.ubuntu: Refreshing state... [id=sha256:b1e9cef3f2977f8bdd19eb9ae04f83b315f80fe4f5c5651fedf41482c12432f7ubuntu:latest]
docker_container.foo: Refreshing state... [id=dfe9d545df9f1bf34ba0dcb6aa106e9c52785ea49732a65708f46b1066af2fe8]

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

Terraform will perform the following actions:

  # docker_container.foo will be destroyed
  - resource "docker_container" "foo" {
      - attach            = false -> null
      - command           = [
          - "/bin/bash",
        ] -> null
      - cpu_shares        = 0 -> null
      - dns               = [] -> null
      - dns_opts          = [] -> null
      - dns_search        = [] -> null
      - entrypoint        = [] -> null
      - env               = [] -> null
      - gateway           = "172.17.0.1" -> null
      - group_add         = [] -> null
      - hostname          = "dfe9d545df9f" -> null
      - id                = "dfe9d545df9f1bf34ba0dcb6aa106e9c52785ea49732a65708f46b1066af2fe8" -> null
      - image             = "sha256:b1e9cef3f2977f8bdd19eb9ae04f83b315f80fe4f5c5651fedf41482c12432f7" -> null
      - init              = false -> null
      - ip_address        = "172.17.0.2" -> null
      - ip_prefix_length  = 16 -> null
      - ipc_mode          = "private" -> null
      - links             = [] -> null
      - log_driver        = "json-file" -> null
      - log_opts          = {} -> null
      - logs              = false -> null
      - max_retry_count   = 0 -> null
      - memory            = 0 -> null
      - memory_swap       = 0 -> null
      - must_run          = true -> null
      - name              = "foo" -> null
      - network_data      = [
          - {
              - gateway                   = "172.17.0.1"
              - global_ipv6_prefix_length = 0
              - ip_address                = "172.17.0.2"
              - ip_prefix_length          = 16
```

Jai Talreja 61 D15A

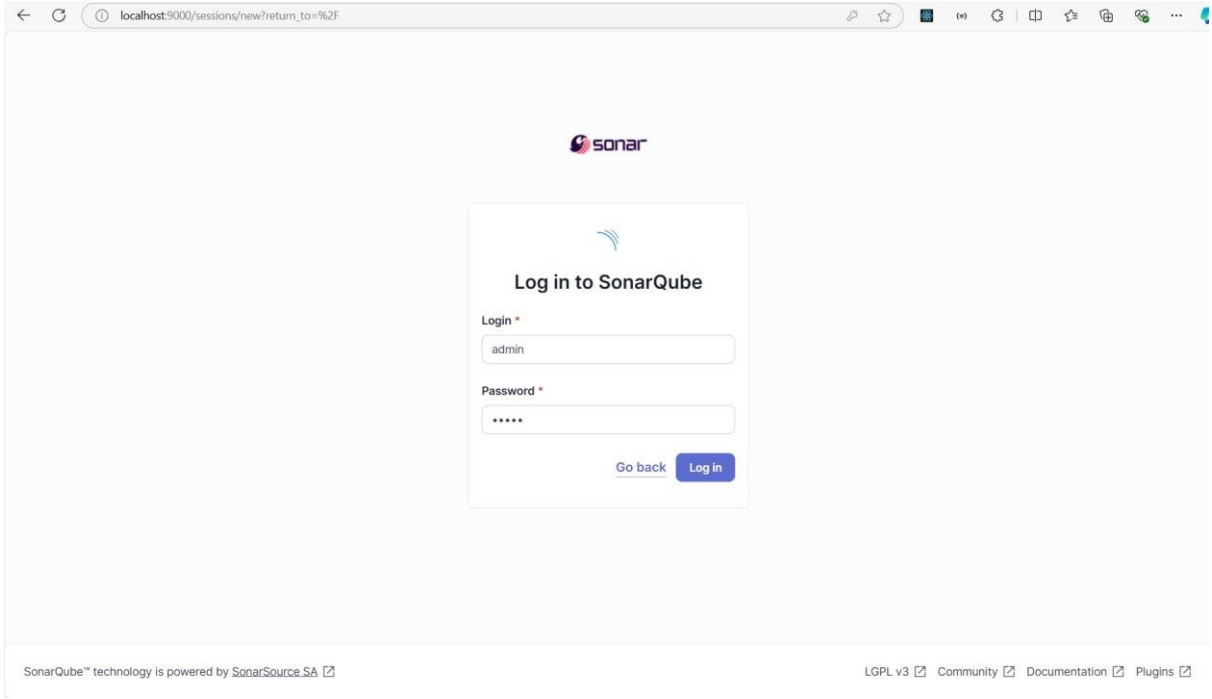
```
  # docker_image.ubuntu will be destroyed
  - resource "docker_image" "ubuntu" {
      - id          = "sha256:b1e9cef3f2977f8bdd19eb9ae04f83b315f80fe4f5c5651fedf41482c12432f7ubuntu:latest" -> null
      - image_id    = "sha256:b1e9cef3f2977f8bdd19eb9ae04f83b315f80fe4f5c5651fedf41482c12432f7" -> null
      - latest      = "sha256:b1e9cef3f2977f8bdd19eb9ae04f83b315f80fe4f5c5651fedf41482c12432f7" -> null
      - name        = "ubuntu:latest" -> null
      - repo_digest = "ubuntu@sha256:dfc10878be8d8fc9c61cbff33166cb1d1fe44391539243703c72766894fa834a" -> null
    }

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

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

  Enter a value: yes

docker_container.foo: Destroying... [id=dfe9d545df9f1bf34ba0dcb6aa106e9c52785ea49732a65708f46b1066af2fe8]
docker_container.foo: Destruction complete after 0s
docker_image.ubuntu: Destroying... [id=sha256:b1e9cef3f2977f8bdd19eb9ae04f83b315f80fe4f5c5651fedf41482c12432f7ubuntu:latest]
docker_image.ubuntu: Destruction complete after 0s

Destroy complete! Resources: 2 destroyed.
jai@fedora:~/Documents/terraform_scripts/docker$
```

Jai talreja 61 D15A

Screenshots for Experiment 7.

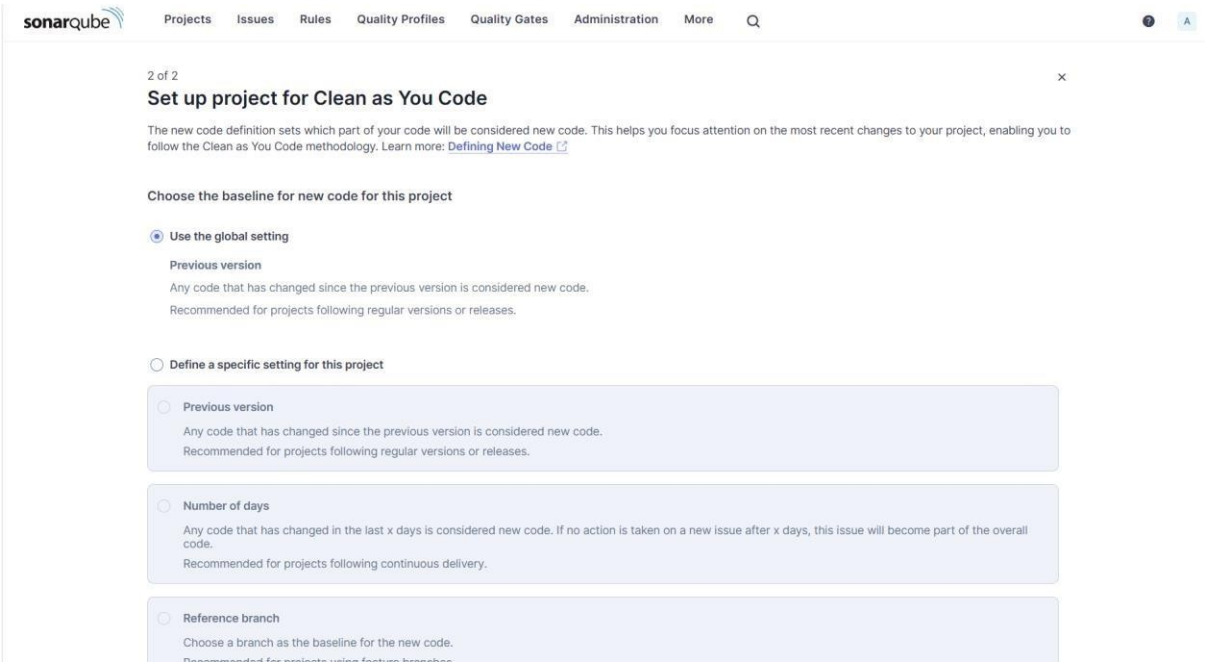


Logging in with admin and password.

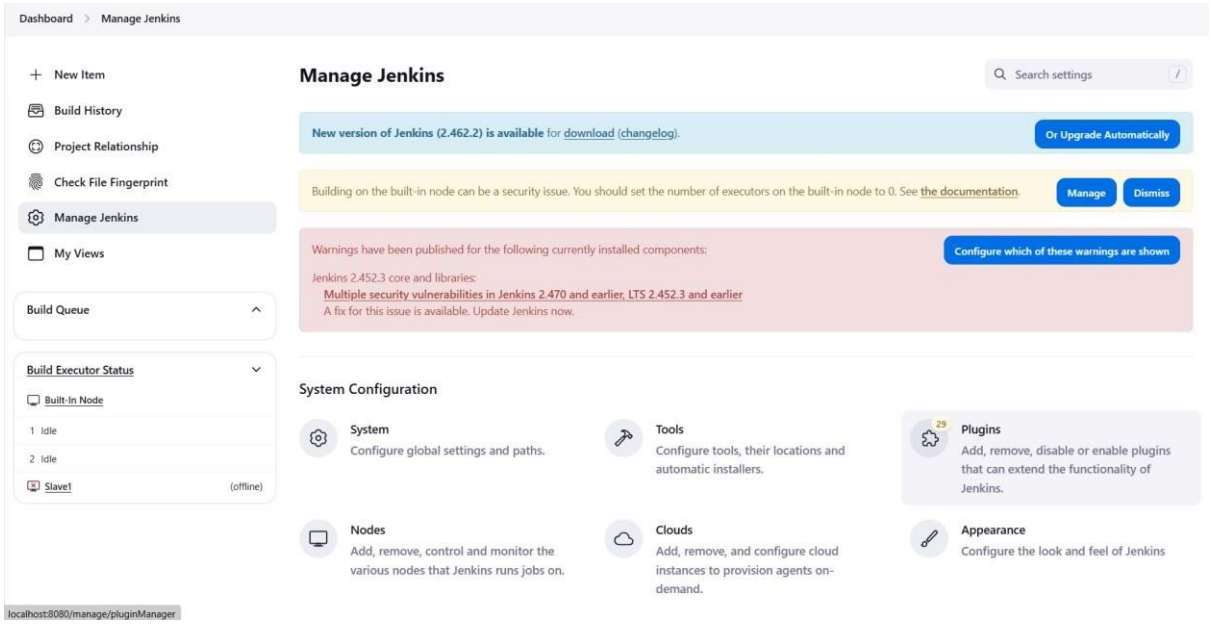


Create a freestyle project and enter its name and key as “sonarqube-test”.

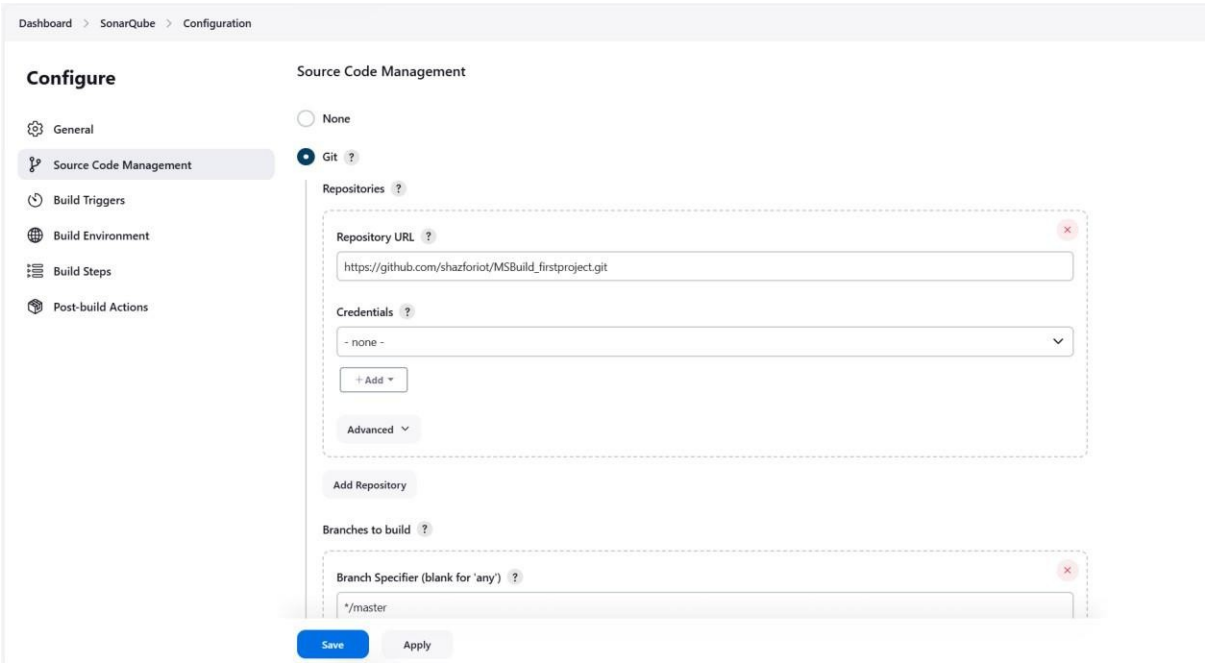
Jai talreja 61 D15A

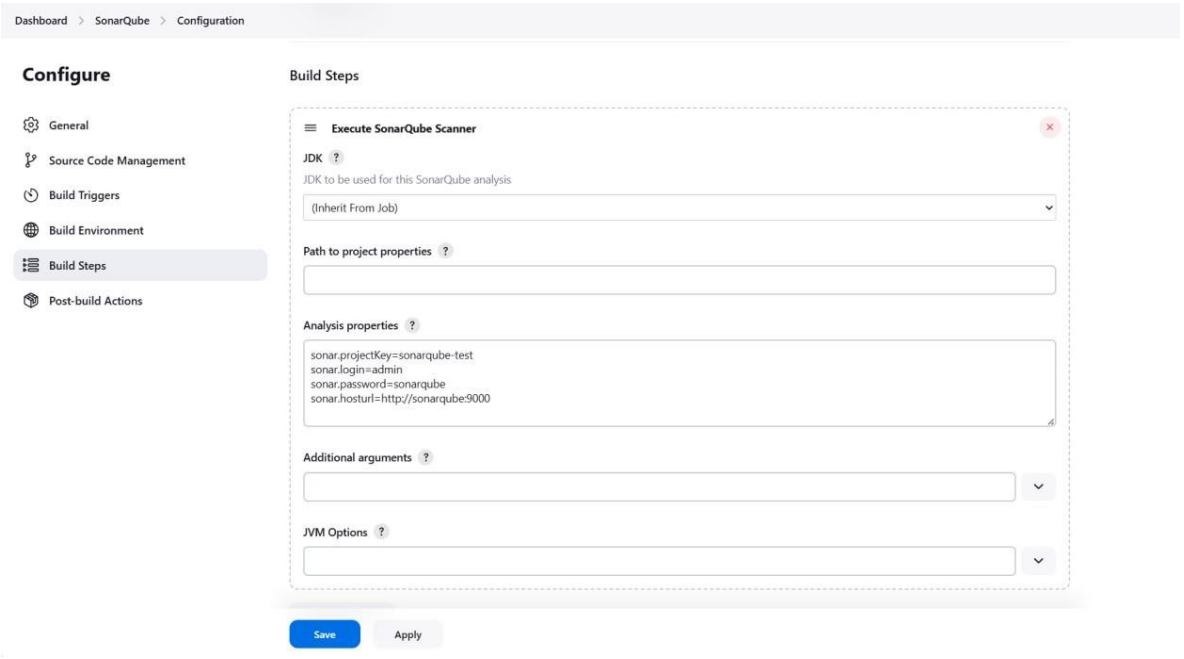


Installing and using the “SonarQube scanner for Jenkins” plugin.

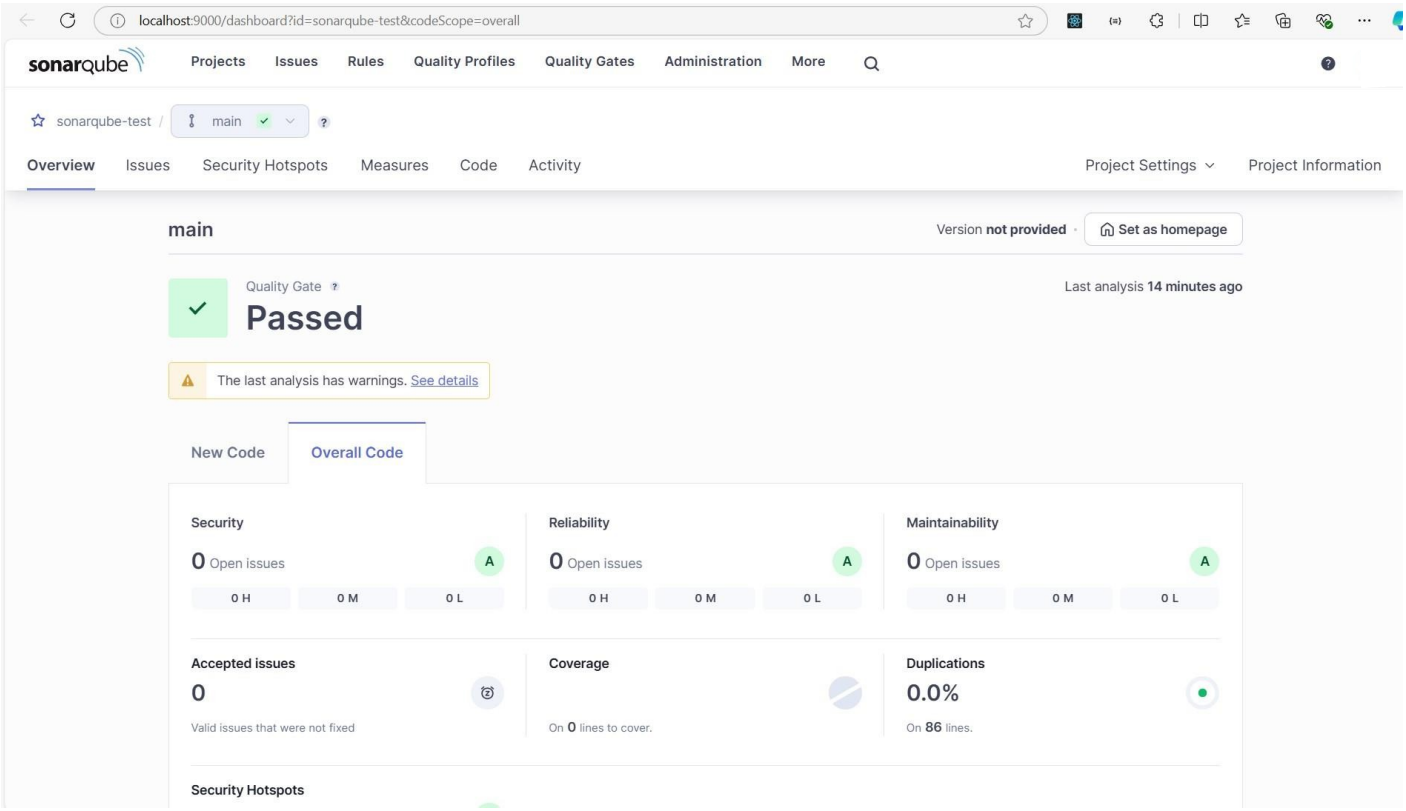


Jai talreja 61 D15A





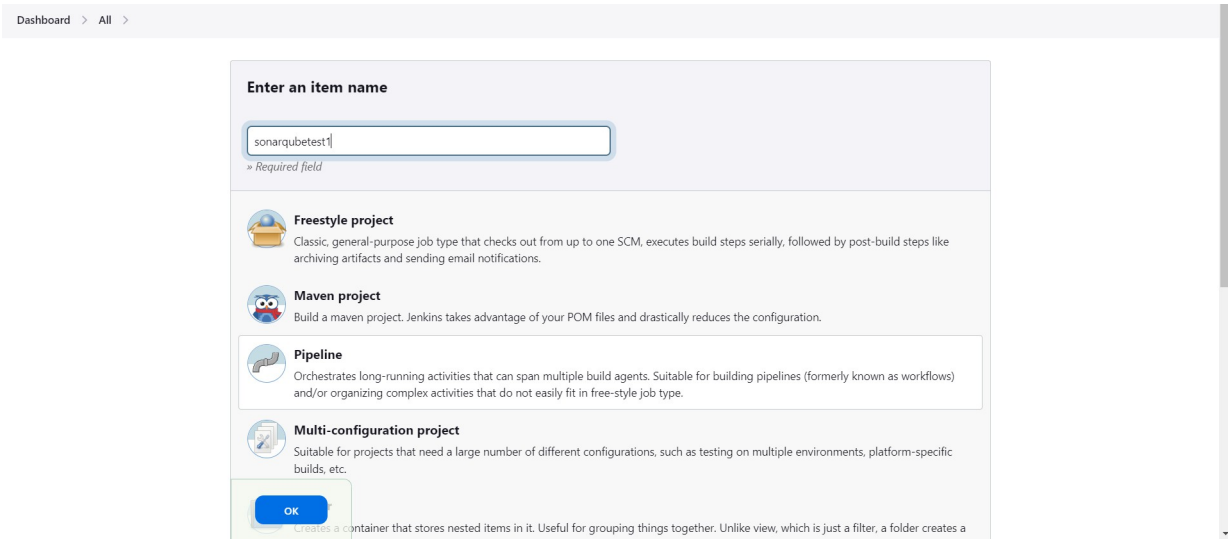
Jai talreja 61 D15A



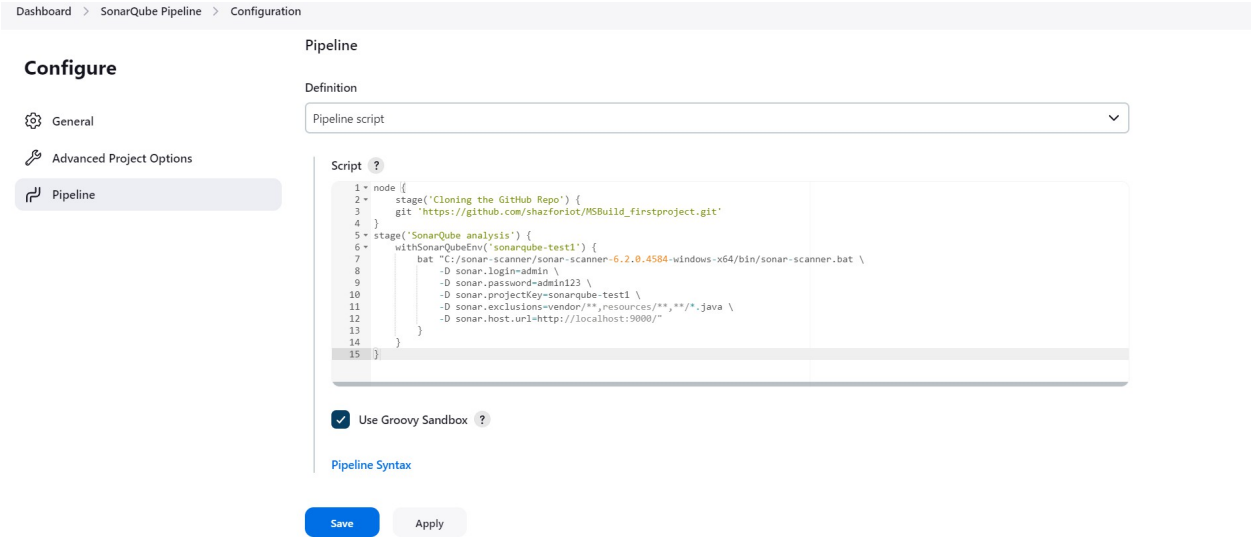
Jai Talreja 61 D15A

Screenshots for Experiment 8.

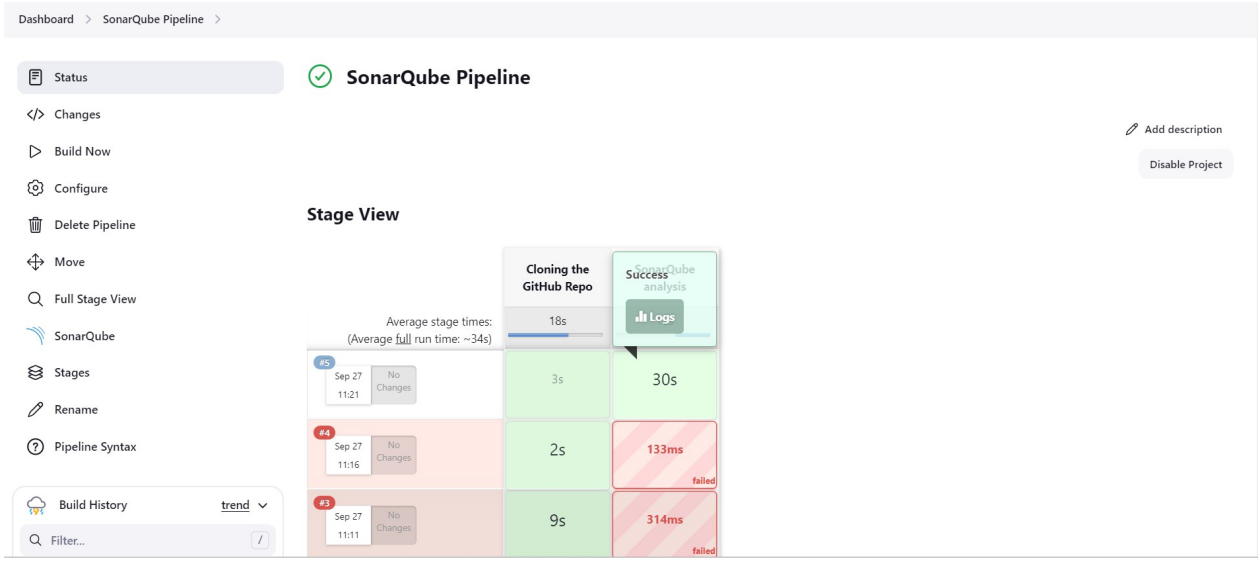
```
erShell does not load commands from the current location by default. If you trust this command, instead type: ".\sonar-s
canner.bat". See "get-help about_Command_Precedence" for more details.
PS C:\sonar-scanner\sonar-scanner-6.2.0.4584-windows-x64\bin> .\sonar-scanner.bat
11:02:02.120 INFO  Scanner configuration file: C:\sonar-scanner\sonar-scanner-6.2.0.4584-windows-x64\bin\..\conf\sonar-s
canner.properties
11:02:02.124 INFO  Project root configuration file: NONE
11:02:02.140 INFO  SonarScanner CLI 6.2.0.4584
11:02:02.142 INFO  Java 17.0.12 Eclipse Adoptium (64-bit)
11:02:02.142 INFO  Windows 11 10.0 amd64
11:02:02.160 INFO  User cache: C:\Users\navan\.sonar\cache
11:02:02.644 INFO  JRE provisioning: os[windows], arch[amd64]
11:02:06.241 INFO  EXECUTION FAILURE
11:02:06.243 INFO  Total time: 4.126s
11:02:06.244 ERROR Error during SonarScanner CLI execution
java.lang.IllegalStateException: Error status returned by url [https://api.sonarcloud.io/analysis/jres?os=windows&arch=a
md64]: 401
        at org.sonarsource.scanner.lib.internal.http.ServerConnection.callUrl(ServerConnection.java:182)
        at org.sonarsource.scanner.lib.internal.http.ServerConnection.callApi(ServerConnection.java:145)
        at org.sonarsource.scanner.lib.internal.http.ServerConnection.callRestApi(ServerConnection.java:123)
        at org.sonarsource.scanner.lib.internal.JavaRunnerFactory.getJreMetadata(JavaRunnerFactory.java:159)
        at org.sonarsource.scanner.lib.internal.JavaRunnerFactory.getJreFromServer(JavaRunnerFactory.java:138)
        at org.sonarsource.scanner.lib.internal.JavaRunnerFactory.createRunner(JavaRunnerFactory.java:85)
        at org.sonarsource.scanner.lib.internal.ScannerEngineLauncherFactory.createLauncher(ScannerEngineLauncherFactory
.java:53)
        at org.sonarsource.scanner.lib.ScannerEngineBootstrapper.bootstrap(ScannerEngineBootstrapper.java:118)
        at org.sonarsource.scanner.cli.Main.analyze(Main.java:75)
        at org.sonarsource.scanner.cli.Main.main(Main.java:63)
11:02:06.246 ERROR
11:02:06.246 ERROR Re-run SonarScanner CLI using the -X switch to enable full debug logging.
PS C:\sonar-scanner\sonar-scanner-6.2.0.4584-windows-x64\bin>
```



Jai Talreja 61 D15A



Jai Talreja 61 D15A

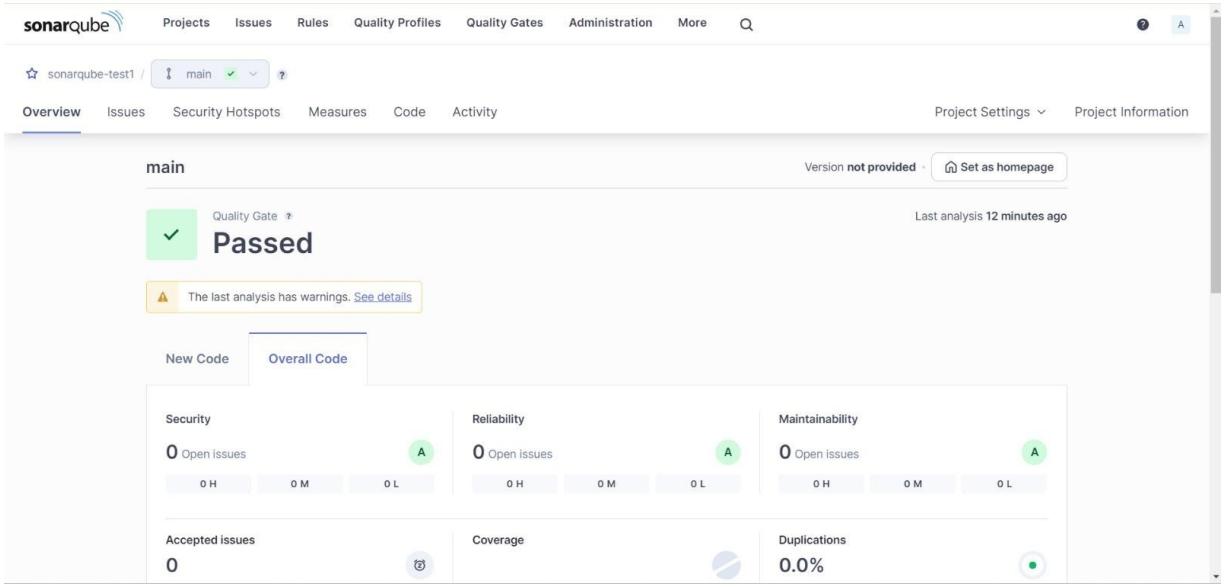


Console Output

```
Started by user jai navani
[Pipeline] Start of Pipeline
[Pipeline] node
Running on Jenkins in C:\ProgramData\Jenkins\.jenkins\workspace\SonarQube Pipeline
[Pipeline] {
[Pipeline] stage
[Pipeline] { (Cloning the GitHub Repo)
[Pipeline] git
The recommended git tool is: NONE
No credentials specified
 > git.exe rev-parse --resolve-git-dir C:\ProgramData\Jenkins\.jenkins\workspace\SonarQube Pipeline\.git # timeout=10
Fetching changes from the remote Git repository
 > git.exe config remote.origin.url https://github.com/shazforiot/MSBuild_firstproject.git # timeout=10
Fetching upstream changes from https://github.com/shazforiot/MSBuild_firstproject.git
 > git.exe --version # timeout=10
 > git --version # 'git version 2.45.2.windows.1'
 > git.exe fetch --tags --force --progress -- https://github.com/shazforiot/MSBuild_firstproject.git +refs/heads/*:refs/remotes/origin/* # timeout=10
 > git.exe rev-parse "refs/remotes/origin/master^{commit}" # timeout=10
Checking out Revision f2bc042c04c6e72427c380bcaee6d6fee7b49adf (refs/remotes/origin/master)
 > git.exe config core.sparsecheckout # timeout=10
 > git.exe checkout -f f2bc042c04c6e72427c380bcaee6d6fee7b49adf # timeout=10
 > git.exe branch -a -v --no-abbrev # timeout=10
 > git.exe branch -D master # timeout=10
```

Jai Talreja 61 D15A

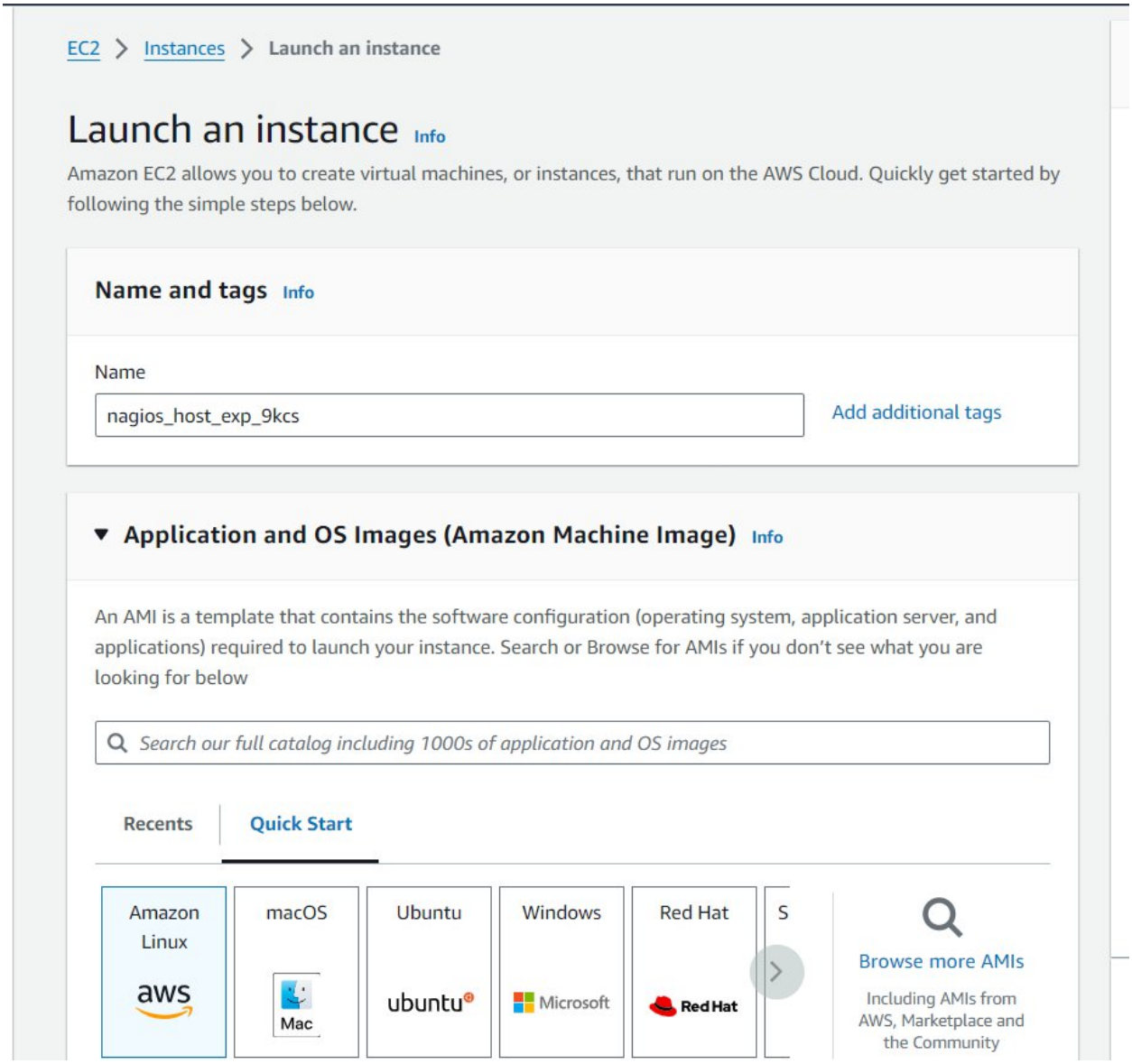
```
11:22:18.236 INFO  Sensor C# File Caching Sensor [csharp]
11:22:18.237 WARN  Incremental PR analysis: Could not determine common base path, cache will not be computed. Consider setting 'sonar.projectBaseDir'
property.
11:22:18.237 INFO  Sensor C# File Caching Sensor [csharp] (done) | time=1ms
11:22:18.237 INFO  Sensor Zero Coverage Sensor
11:22:18.251 INFO  Sensor Zero Coverage Sensor (done) | time=14ms
11:22:18.256 INFO  SCM Publisher SCM provider for this project is: git
11:22:18.257 INFO  SCM Publisher 4 source files to be analyzed
11:22:18.789 INFO  SCM Publisher 4/4 source files have been analyzed (done) | time=531ms
11:22:18.793 INFO  CPD Executor Calculating CPD for 0 files
11:22:18.795 INFO  CPD Executor CPD calculation finished (done) | time=0ms
11:22:18.810 INFO  SCM revision ID 'f2bc042c04c6e72427c380bcaee6d6fee7b49adf'
11:22:19.074 INFO  Analysis report generated in 134ms, dir size=201.0 kB
11:22:19.137 INFO  Analysis report compressed in 45ms, zip size=22.5 kB
11:22:19.351 INFO  Analysis report uploaded in 212ms
11:22:19.353 INFO  ANALYSIS SUCCESSFUL, you can find the results at: http://localhost:9000/dashboard?id=sonarqube-test1
11:22:19.354 INFO  Note that you will be able to access the updated dashboard once the server has processed the submitted analysis report
11:22:19.354 INFO  More about the report processing at http://localhost:9000/api/ce/task?id=971ae2f2-4e0f-49a7-88c4-ad4a6cceddf8
11:22:19.366 INFO  Analysis total time: 24.819 s
11:22:19.368 INFO  SonarScanner Engine completed successfully
11:22:19.454 INFO  EXECUTION SUCCESS
11:22:19.455 INFO  Total time: 29.696s
[Pipeline] }
[Pipeline] // withSonarQubeEnv
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```



Jai Talreja 61 D15A

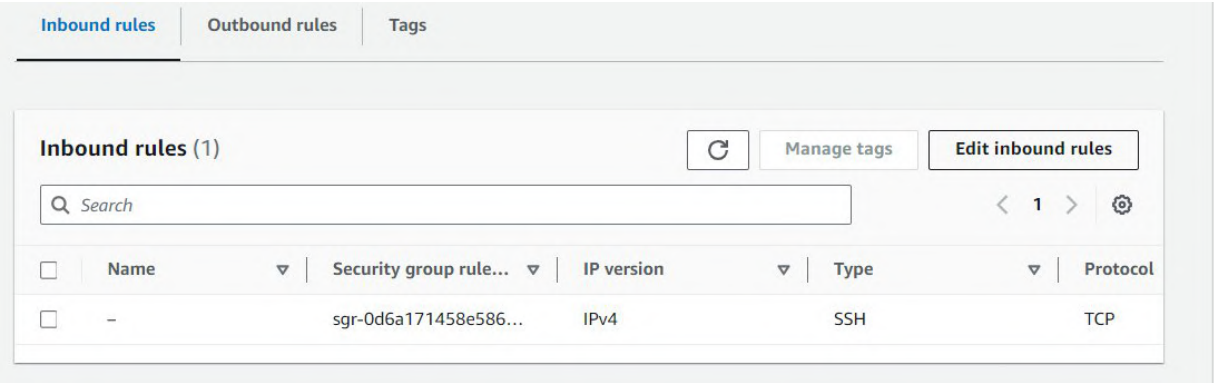
Screenshots for Experiment 9

Launch an ec2 instance
Give name use the default OS

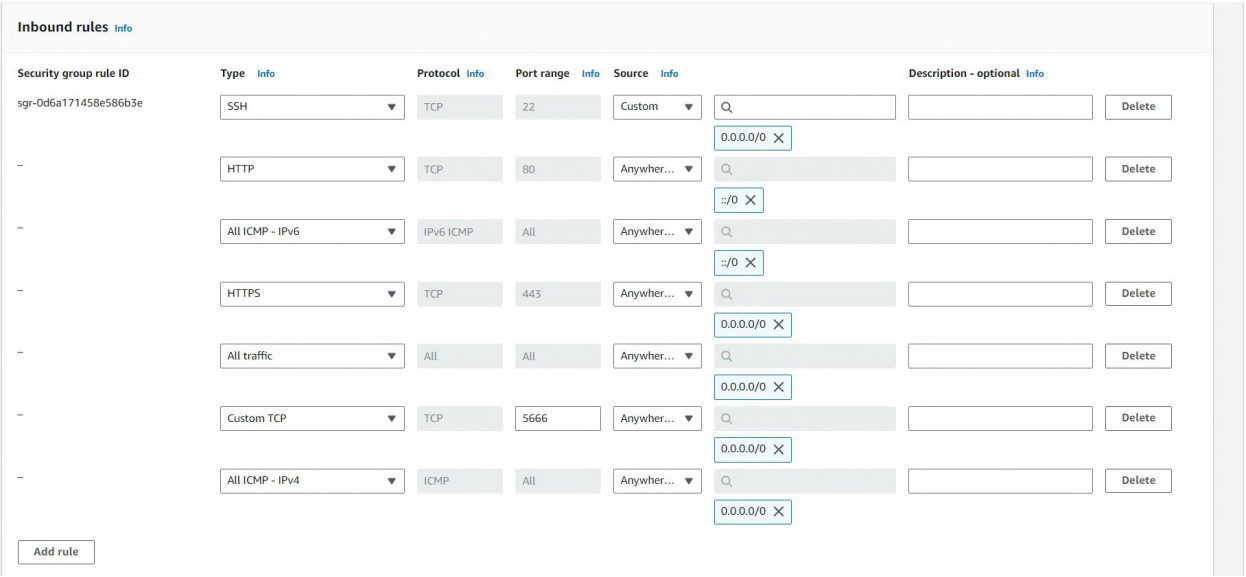


Jai Talreja 61 D15A

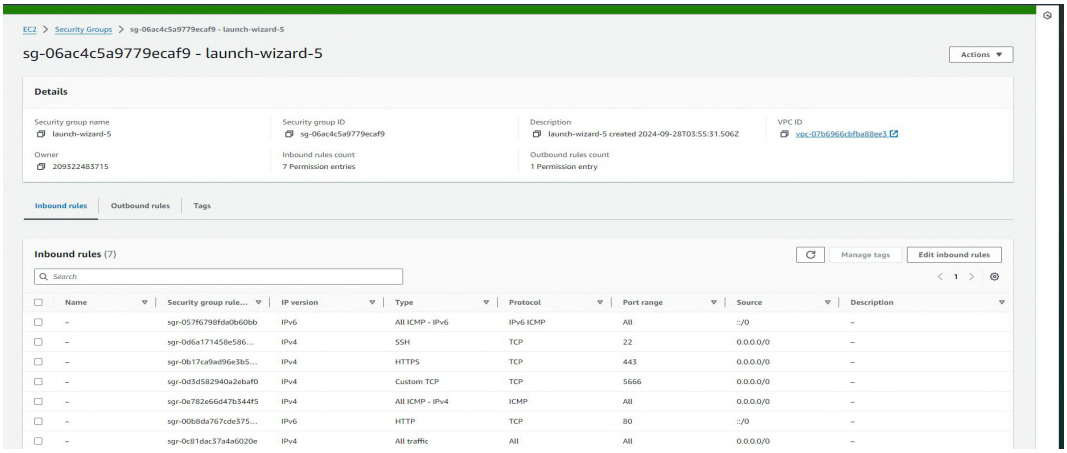
now click on edit inbound rules



now do the following configurations:
by clicking “add rules”



then click on save rules.



Jai Talreja 61 D15A

mkdir ~/downloads
cd ~/downloads

```
[ec2-user@ip-172-31-80-137 ~]$ mkdir ~/downloads
cd ~/downloads
[ec2-user@ip-172-31-80-137 downloads]$
```

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

```
cd ~/downloads
[ec2-user@ip-172-31-80-137 downloads]$ wget https://assets.nagios.com/downloads/nagioscore/releases/nagios-4.5.5.tar.gz
--2024-09-28 06:27:51--  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  5.30MB/s    in 0.4s

2024-09-28 06:27:52 (5.30 MB/s) - 'nagios-4.5.5.tar.gz' saved [2065473/2065473]

[ec2-user@ip-172-31-80-137 downloads]$
```

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

```
[ec2-user@ip-172-31-80-137 downloads]$ wget https://nagios-plugins.org/download/nagios-plugins-2.4.11.tar.gz
--2024-09-28 06:28:14--  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  5.90MB/s    in 0.4s

2024-09-28 06:28:15 (5.90 MB/s) - 'nagios-plugins-2.4.11.tar.gz' saved [2753049/2753049]

[ec2-user@ip-172-31-80-137 downloads]$
```

tar zxvf nagios-4.5.5.tar.gz

```
[ec2-user@ip-172-31-80-137 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
```

Jai Talreja 61 D15A

```
# 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 (define
    alias                   Nagios Admin            ; Full name of user
    email                   2022.shubham.jha@ves.ac.in ; <<***** CHANGE THIS TO YOUR EMAIL ADDRESS ******
}



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

^G Help       ^O Write Out  ^W Where Is   ^K Cut         ^T Execute    ^C Location   M-U Undo      M-A Set M
```

press Ctrl+O and then enter.
then press Ctrl +X

```
chmod g+s /usr/local/nagios/var/rw

*** External command directory configured ***

[ec2-user@ip-172-31-80-137 nagios-4.5.5]$ sudo nano /usr/local/nagios/etc/objects/contacts.cfg
[ec2-user@ip-172-31-80-137 nagios-4.5.5]$
```

sudo make install-webconf

```
[ec2-user@ip-172-31-80-137 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 ***

[ec2-user@ip-172-31-80-137 nagios-4.5.5]$
```

Adding password for nagios admin

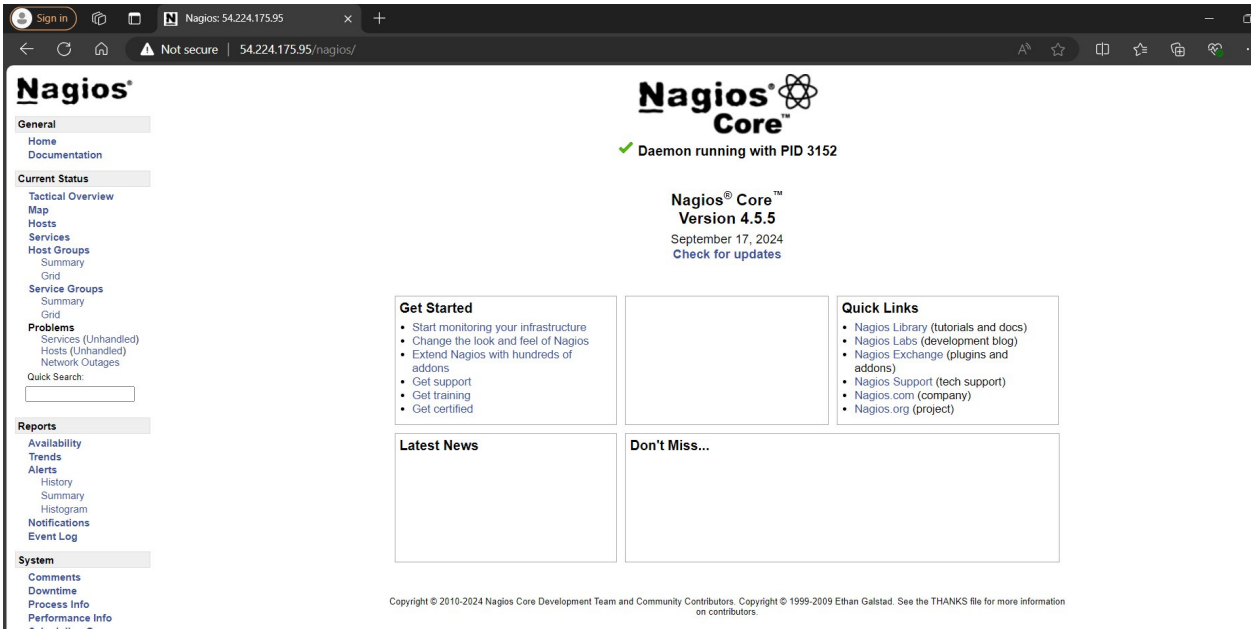
sudo htpasswd -c /usr/local/nagios/etc/htpasswd.users nagiosadmin

```
[ec2-user@ip-172-31-80-137 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-172-31-80-137 nagios-4.5.5]$
```

sudo service httpd restart

```
[ec2-user@ip-172-31-80-137 nagios-4.5.5]$ sudo service httpd restart
Redirecting to /bin/systemctl restart httpd.service
[ec2-user@ip-172-31-80-137 nagios-4.5.5]$
```

Jai Talreja 61 D15A

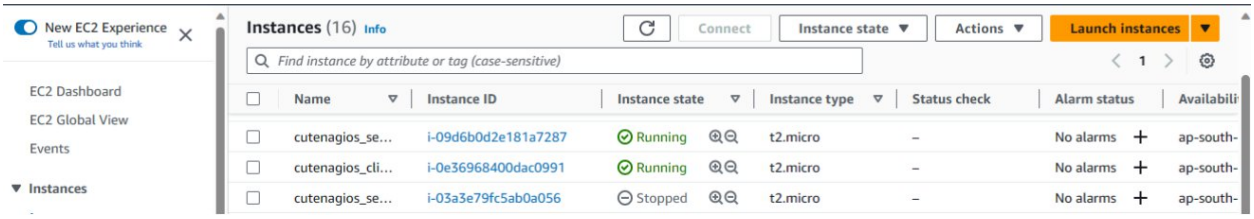


Jai Talreja 61 D15A

Screenshots for Experiment 10

```
● nagios.service - Nagios Core 4.4.14
     Loaded: loaded (/lib/systemd/system/nagios.service; enabled; vendor preset: enabled)
     Active: active (running) since Sat 2023-09-30 08:54:01 UTC; 20s ago
       Docs: https://www.nagios.org/documentation
    Process: 55285 ExecStartPre=/usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg (code=exited, status=0/SUCCESS)
    Process: 55286 ExecStart=/usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg (code=exited, status=0/SUCCESS)
   Main PID: 55287 (nagios)
      Tasks: 6 (limit: 1141)
     Memory: 5.3M
        CPU: 252ms
     CGroup: /system.slice/nagios.service
             ├─55287 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg
             ├─55288 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─55289 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─55290 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─55291 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             └─55292 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg

Sep 30 08:54:01 ip-172-31-44-151 nagios[55287]: qh: Socket '/usr/local/nagios/var/rw/nagios.qh' successfully initialized
lines 1-19
```

Monitoring a Linux machine, create an Ubuntu 20.04 server EC2 Instance in AWS.



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

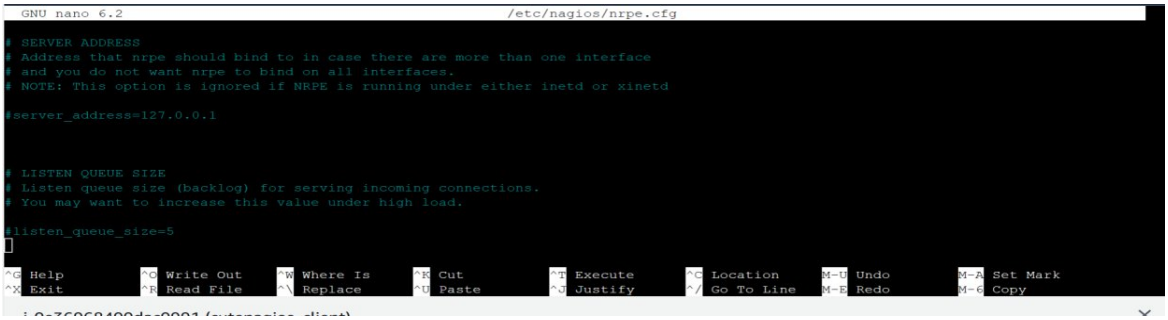
```
*** System restart required ***
Last login: Sat Sep 30 08:31:30 2023 from 13.233.177.3
ubuntu@ip-172-31-44-151:~$ sudo apt install gcc -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
gcc is already the newest version (4:11.2.0-1ubuntu1).
gcc set to manually installed.
0 upgraded, 0 newly installed, 0 to remove and 2 not upgraded.
ubuntu@ip-172-31-44-151:~$
```

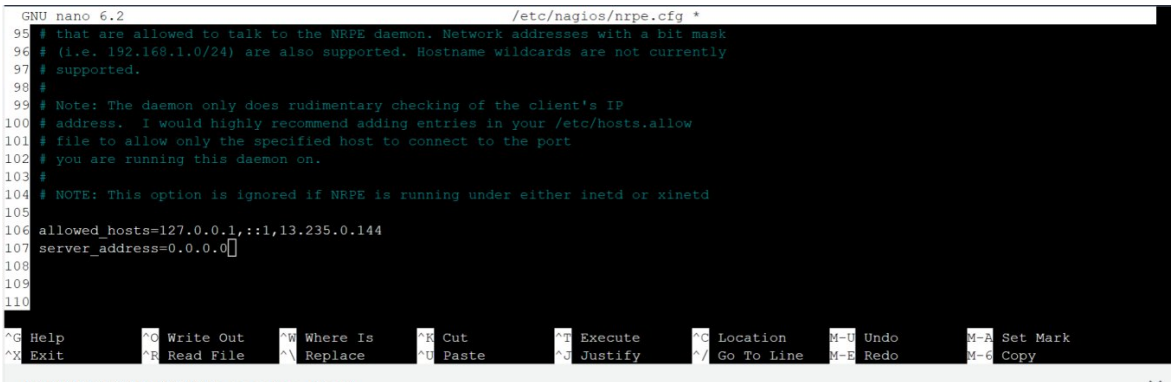
```
root@ip-172-31-44-151:/home/ubuntu# sudo apt install nagios-nrpe-server nagios-plugins
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Note, selecting 'monitoring-plugins' instead of 'nagios-plugins'
monitoring-plugins is already the newest version (2.3.1-1ubuntu4).
nagios-nrpe-server is already the newest version (4.0.3-1ubuntu2).
0 upgraded, 0 newly installed, 0 to remove and 2 not upgraded.
```

```
Get:4 http://security.ubuntu.com/ubuntu jammy-security InRelease [110 kB]
Fetched 229 kB in 1s (290 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
2 packages can be upgraded. Run 'apt list --upgradable' to see them.
root@ip-172-31-44-151:/home/ubuntu# sudo apt install gcc -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
gcc is already the newest version (4:11.2.0-1ubuntu1).
0 upgraded, 0 newly installed, 0 to remove and 2 not upgraded.
root@ip-172-31-44-151:/home/ubuntu# sudo apt install -y nagios-nrpe-server nagios-plugins
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Note, selecting 'monitoring-plugins' instead of 'nagios-plugins'
monitoring-plugins is already the newest version (2.3.1-1ubuntu4).
nagios-nrpe-server is already the newest version (4.0.3-1ubuntu2).
0 upgraded, 0 newly installed, 0 to remove and 2 not upgraded.
```

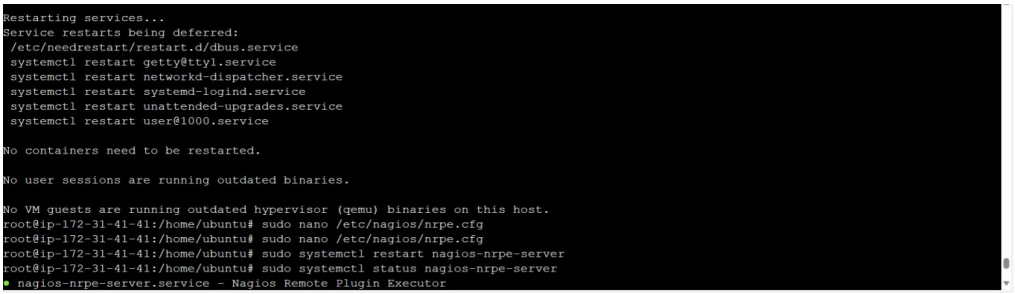
Jai Talreja 61 D15A

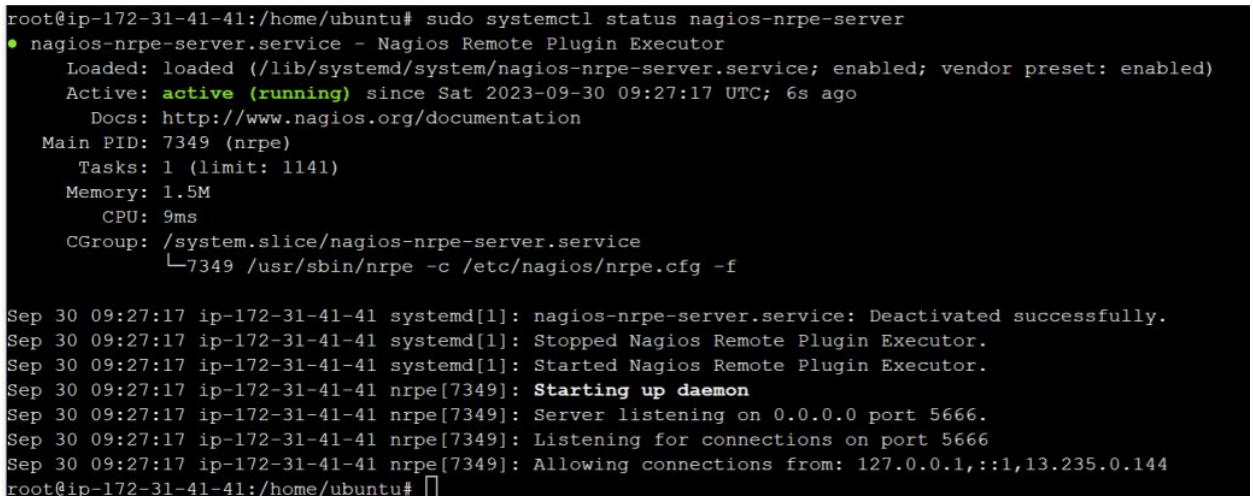
sudo nano /etc/nagios/nrpe.cfg





sudo systemctl restart nagios-nrpe-server





Jai Talreja 61 D15A

ps -ef | grep nagios

```
root@ip-172-31-44-151:/home/ubuntu# ps -ef | grep nagios
nagios     55287       1  0 08:54 ?        00:00:00 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg
nagios     55288   55287  0 08:54 ?        00:00:00 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
nagios     55289   55287  0 08:54 ?        00:00:00 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
nagios     55290   55287  0 08:54 ?        00:00:00 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
nagios     55291   55287  0 08:54 ?        00:00:00 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
nagios     55292   55287  0 08:54 ?        00:00:00 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg
nagios     56327       1  0 08:58 ?        00:00:00 /usr/sbin/nrpe -c /etc/nagios/nrpe.cfg -f
root       60903   60158  0 09:32 pts/1    00:00:00 grep --color=auto nagios
root@ip-172-31-44-151:/home/ubuntu# sudo su
root@ip-172-31-44-151:/home/ubuntu# mkdir /usr/local/nagios/etc/objects/monitorhosts
root@ip-172-31-44-151:/home/ubuntu# mkdir /usr/local/nagios/etc/objects/monitorhosts/linuxhosts
```

1.sudo su 2.mkdir /usr/local/nagios/etc/objects/monitorhosts 3.mkdir /usr/local/nagios/etc/objects/monitorhosts/linuxhosts Copy the sample localhost.cfg file to linuxhost folder 4.cp /usr/local/nagios/etc/objects/localhost.cfg /usr/local/nagios/etc/objects/monitorhosts/linuxhosts/linuxserver.cfg

```
root@ip-172-31-44-151:/home/ubuntu# cp /usr/local/nagios/etc/objects/localhost.cfg /usr/local/nagios/etc/objects/monitorhosts/linuxhos
ts/linuxserver.cfg
root@ip-172-31-44-151:/home/ubuntu# nano /usr/local/nagios/etc/objects/monitorhosts/linuxhosts/linuxserver.cfg
```

```
  GNU nano 6.2                  /usr/local/nagios/etc/objects/monitorhosts/linuxhosts/linuxserver.cfg
##############################################################################

# Define a host for the local machine

define host {

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


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

i-03a3e79fc5ab0a056 (cutenagios_server)

```
  GNU nano 6.2                  /usr/local/nagios/etc/objects/monitorhosts/linuxhosts/linuxserver.cfg *
##############################################################################
#
# HOST GROUP DEFINITION
#
##############################################################################

# Define an optional hostgroup for Linux machines

define hostgroup {

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


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

Jai Talreja 61 D15A

```
  GNU nano 6.2                                  /usr/local/nagios/etc/nagios.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
Save modified buffer?
 Y Yes
 N No            ^C Cancel
```

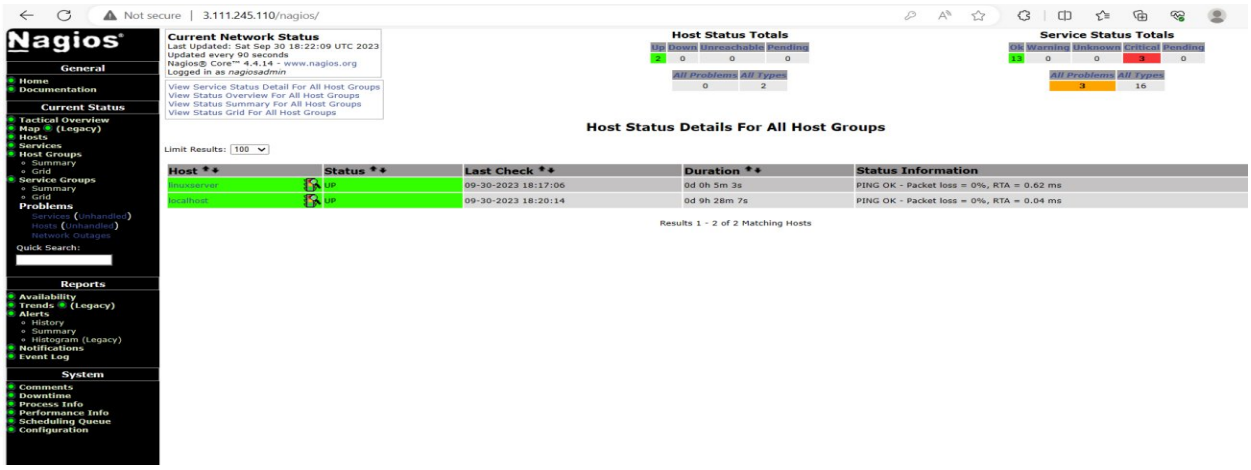
```
root@ip-172-31-44-151:/home/ubuntu# nano /usr/local/nagios/etc/objects/monitorhosts/linuxhosts/linuxserver.cfg
root@ip-172-31-44-151:/home/ubuntu#   /usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg

Nagios Core 4.4.14
Copyright (c) 2009-present Nagios Core Development Team and Community Contributors
Copyright (c) 1999-2009 Ethan Galstad
Last Modified: 2023-08-01
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 8 services.
        Checked 1 hosts.
        Checked 1 host groups.
```

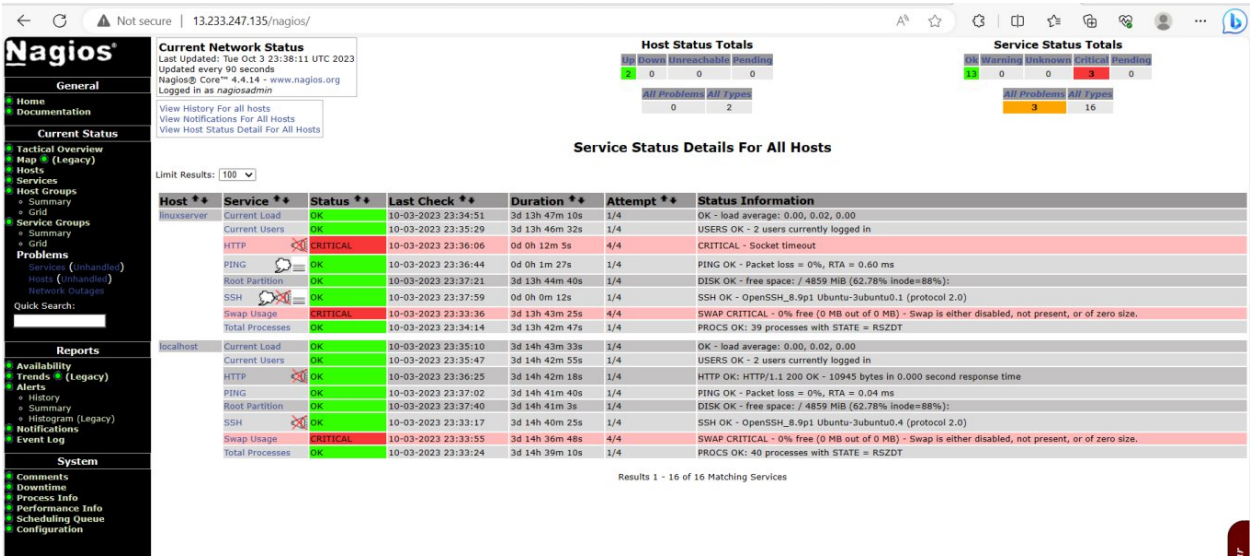
```
        Checked 1 contacts.
        Checked 1 contact groups.
        Checked 24 commands.
        Checked 5 time periods.
        Checked 0 host escalations.
        Checked 0 service escalations.
Checking for circular paths...
        Checked 1 hosts
        Checked 0 service dependencies
        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
root@ip-172-31-44-151:/home/ubuntu# nano /usr/local/nagios/etc/nagios.cfg
```

Sudo systemctl status nagios

```
● nagios.service - Nagios Core 4.4.14
     Loaded: loaded (/lib/systemd/system/nagios.service; enabled; vendor preset: enabled)
     Active: active (running) since Sat 2023-09-30 08:54:01 UTC; 20s ago
       Docs: https://www.nagios.org/documentation
    Process: 55285 ExecStartPre=/usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg (code=exited, status=0/SUCCESS)
    Process: 55286 ExecStart=/usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg (code=exited, status=0/SUCCESS)
   Main PID: 55287 (nagios)
      Tasks: 6 (limit: 1141)
     Memory: 5.3M
        CPU: 252ms
     CGroup: /system.slice/nagios.service
             ├─55287 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg
             ├─55288 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─55289 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─55290 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─55291 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             └─55292 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg

Sep 30 08:54:01 ip-172-31-44-151 nagios[55287]: qh: Socket '/usr/local/nagios/var/rw/nagios.qh' successfully initialized
lines 1-19
```

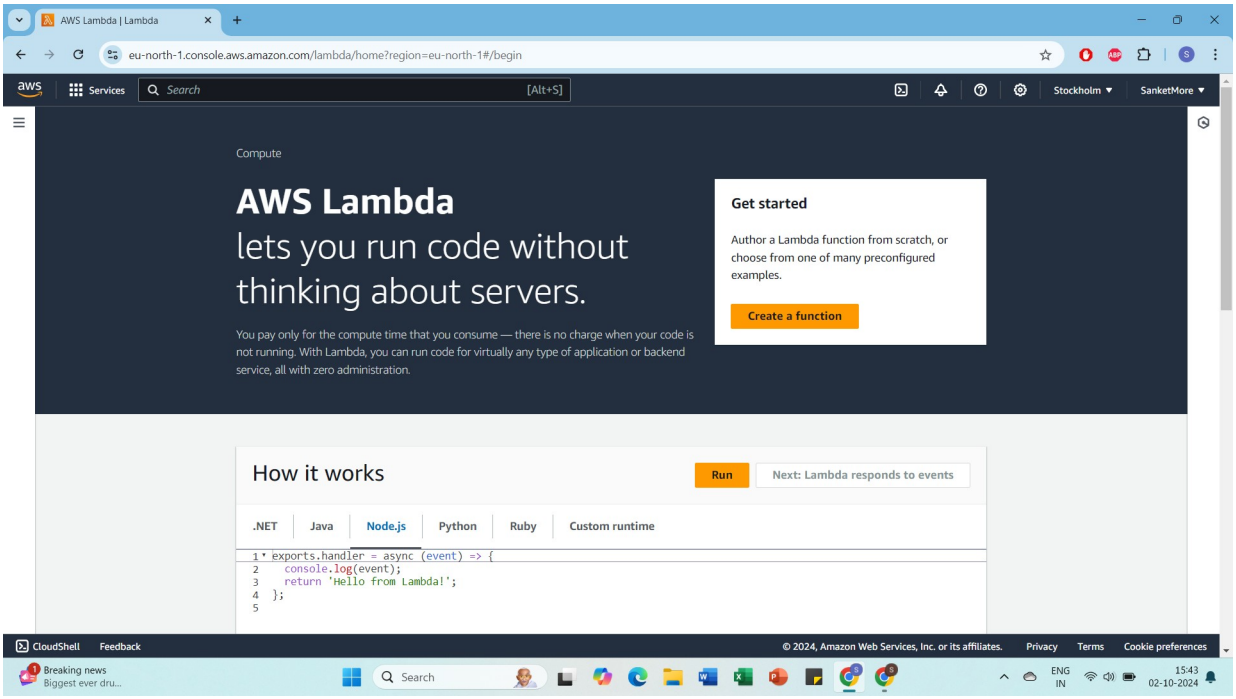
Jai Talreja 61 D15A

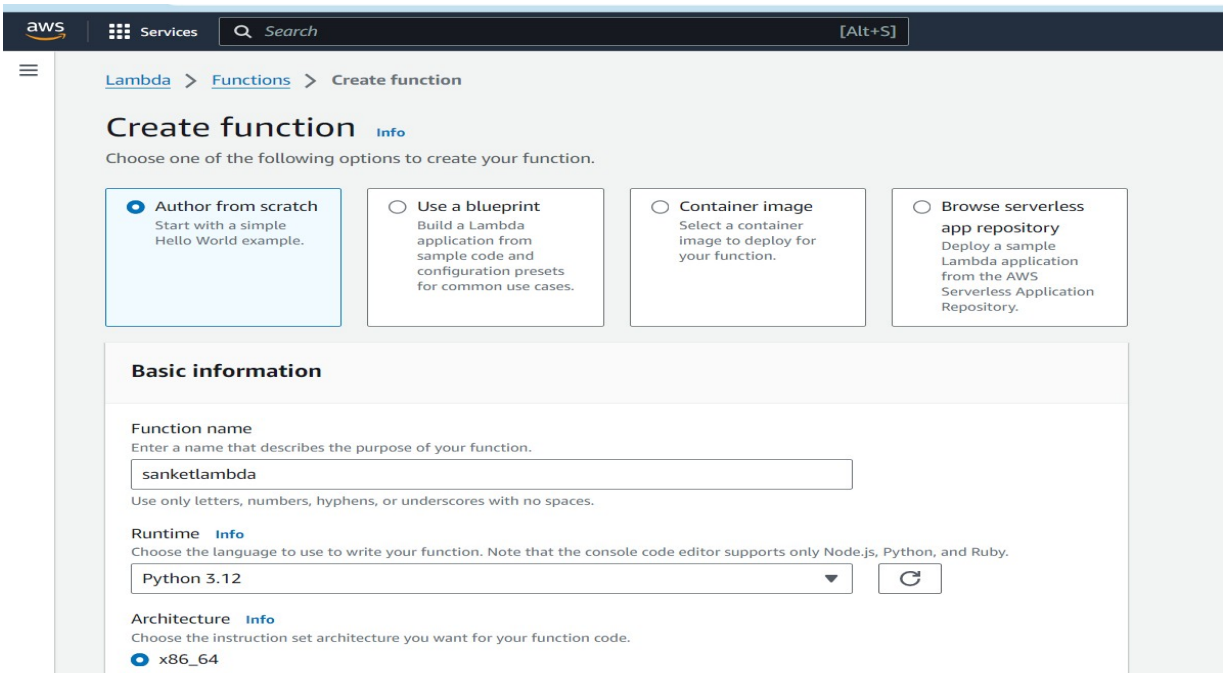


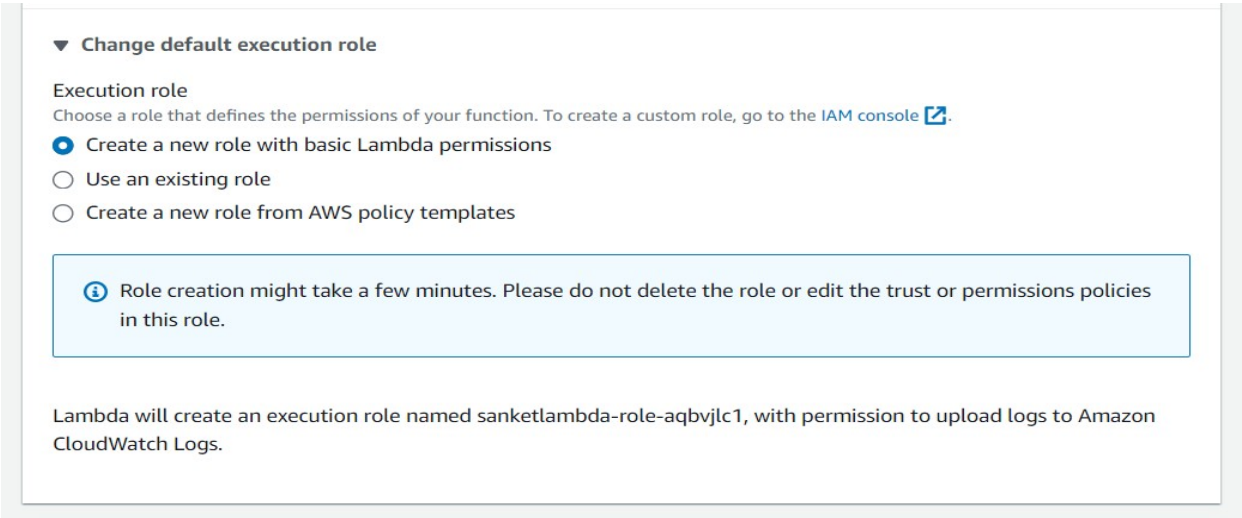


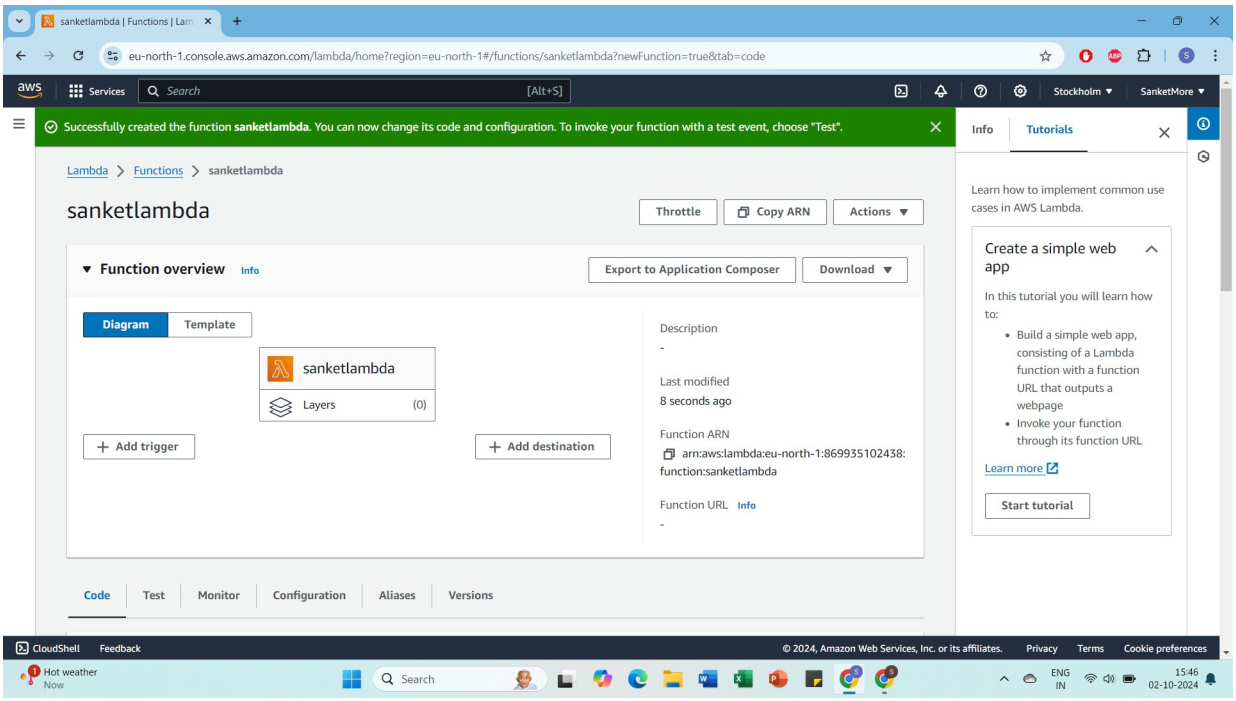
Jai Talreja 61 D15A

Screenshots for Experiment 11

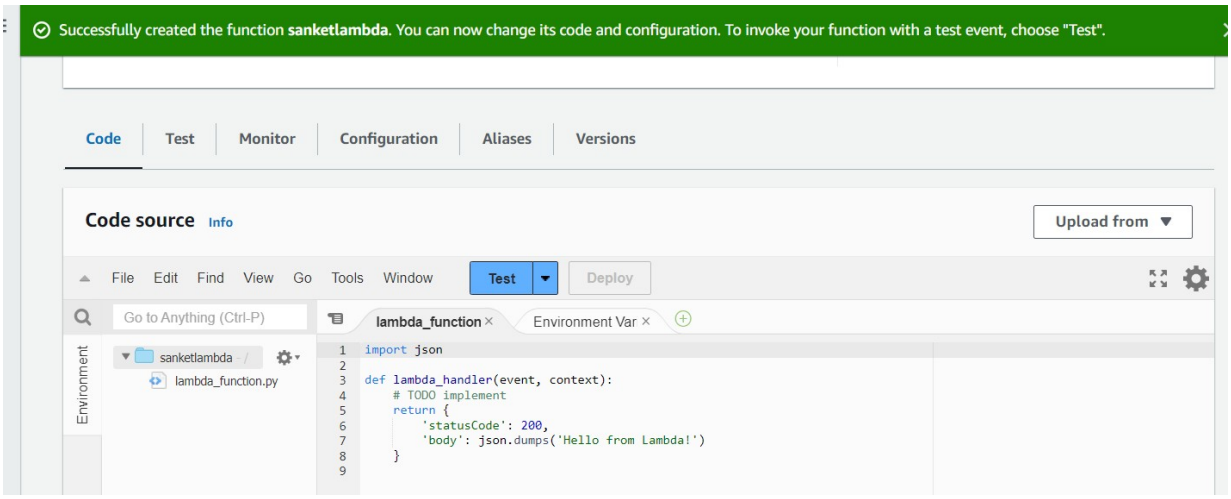


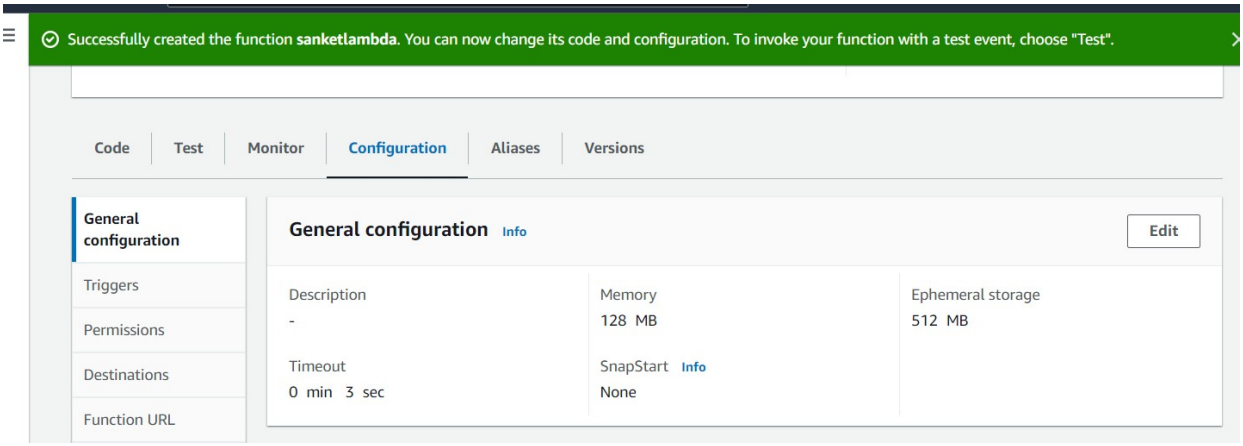


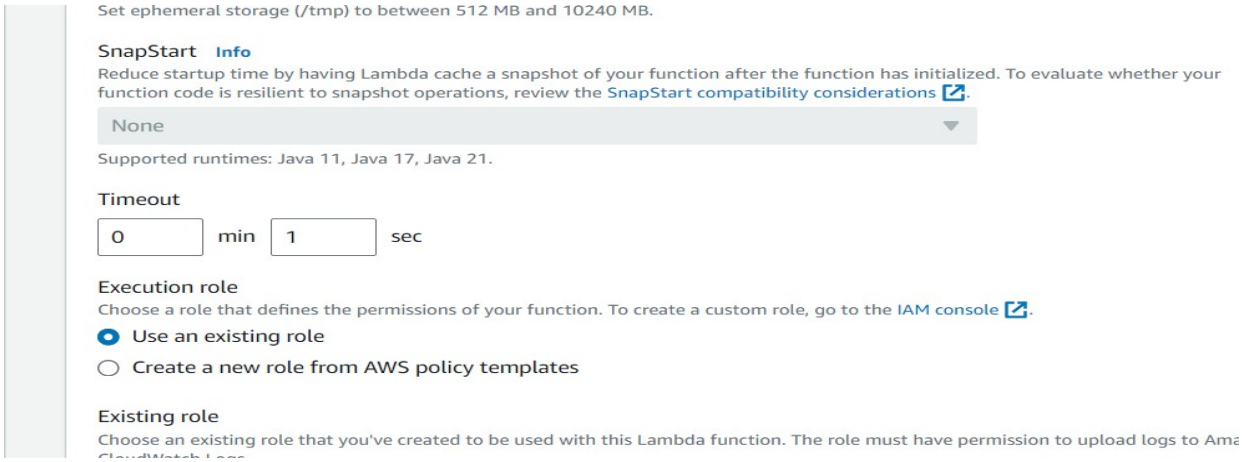

Change default execution role
Execution role
Choose a role that defines the permissions of your function. To create a custom role, go to the IAM console.
Create a new role with basic Lambda permissions
Use an existing role
Create a new role from AWS policy templates
Role creation might take a few minutes. Please do not delete the role or edit the trust or permissions policies in this role.
Lambda will create an execution role named sanketlambda-role-aqbvjlc1, with permission to upload logs to Amazon CloudWatch Logs.


sanketlambda | Functions | Lam
eu-north-1.console.aws.amazon.com/lambda/home?region=eu-north-1#/functions/sanketlambda?newFunction=true&tab=code
Services
Search
[Alt+S]
Stockholm
SanketMore
Successfully created the function sanketlambda. You can now change its code and configuration. To invoke your function with a test event, choose "Test".
Lambda > Functions > sanketlambda
sanketlambda
Throttle
Copy ARN
Actions
Function overview
Info
Export to Application Composer
Download
Diagram
Template
sanketlambda
Layers
(0)
+ Add trigger
+ Add destination
Description
-
Last modified
8 seconds ago
Function ARN
arn:aws:lambda:eu-north-1:869935102438:function:sanketlambda
Function URL
Info
-
Code
Test
Monitor
Configuration
Aliases
Versions
Info
Tutorials
Learn how to implement common use cases in AWS Lambda.
Create a simple web app
In this tutorial you will learn how to:
Build a simple web app, consisting of a Lambda function with a function URL that outputs a webpage
Invoke your function through its function URL
Learn more
Start tutorial
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
Hot weather
Now
Search
ENG
IN
15:46
02-10-2024

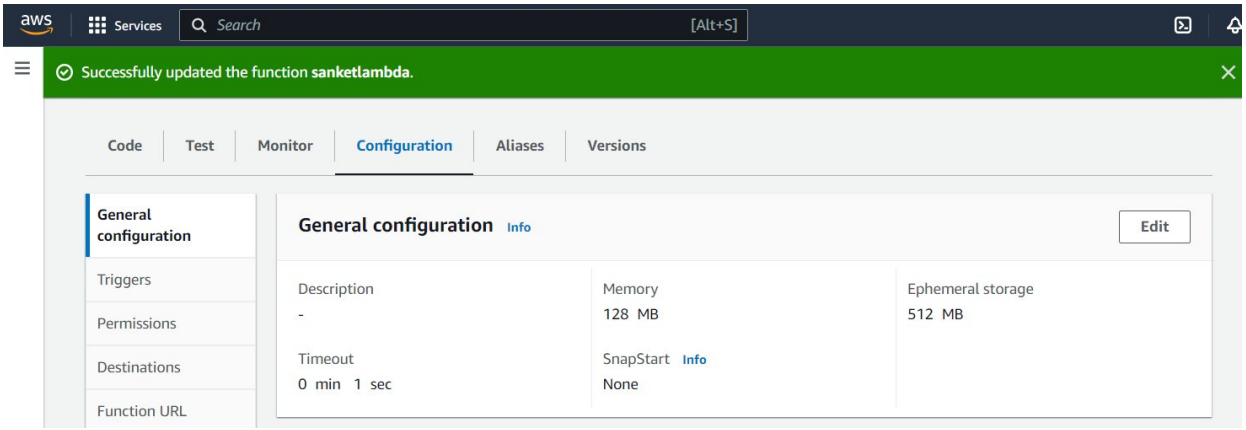
Jai Talreja 61 D15A

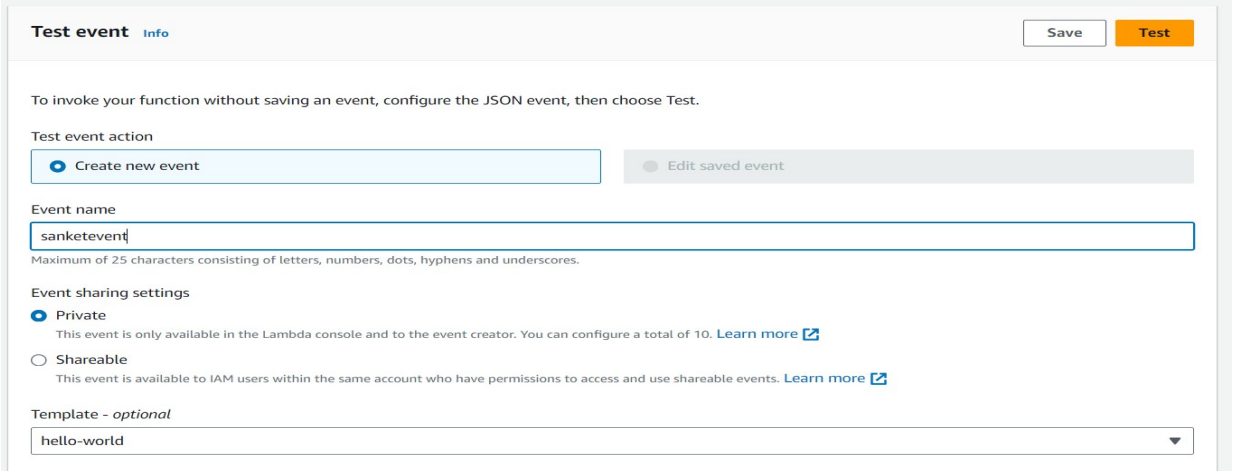




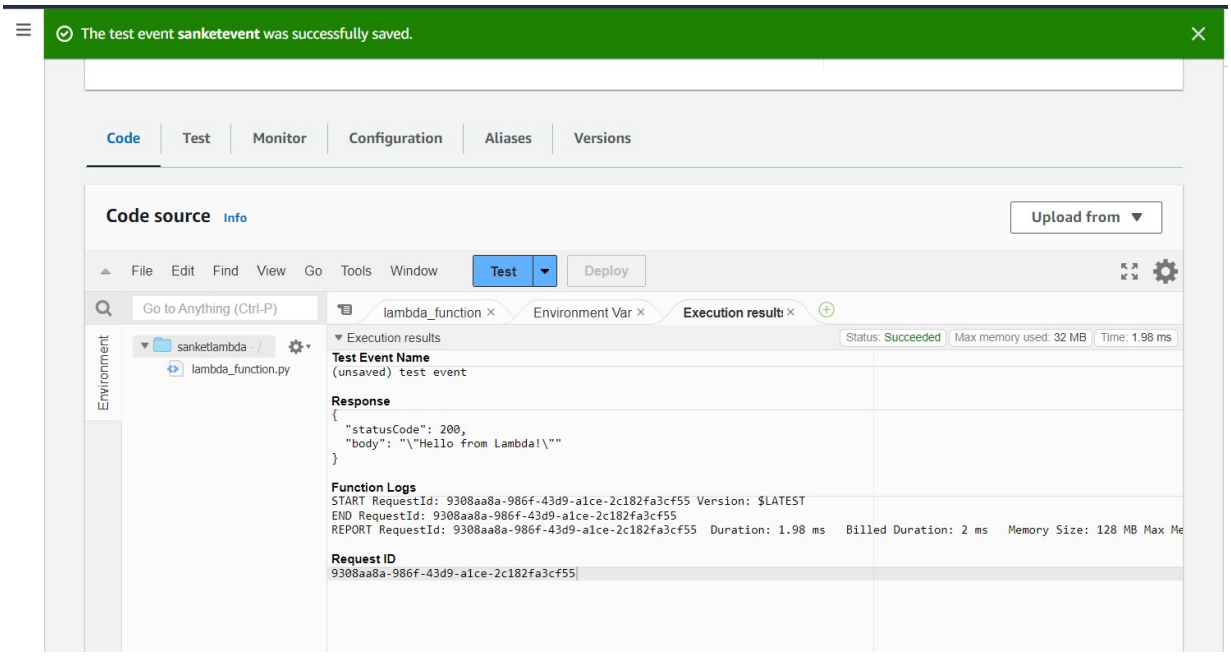


Jai Talreja 61 D15A





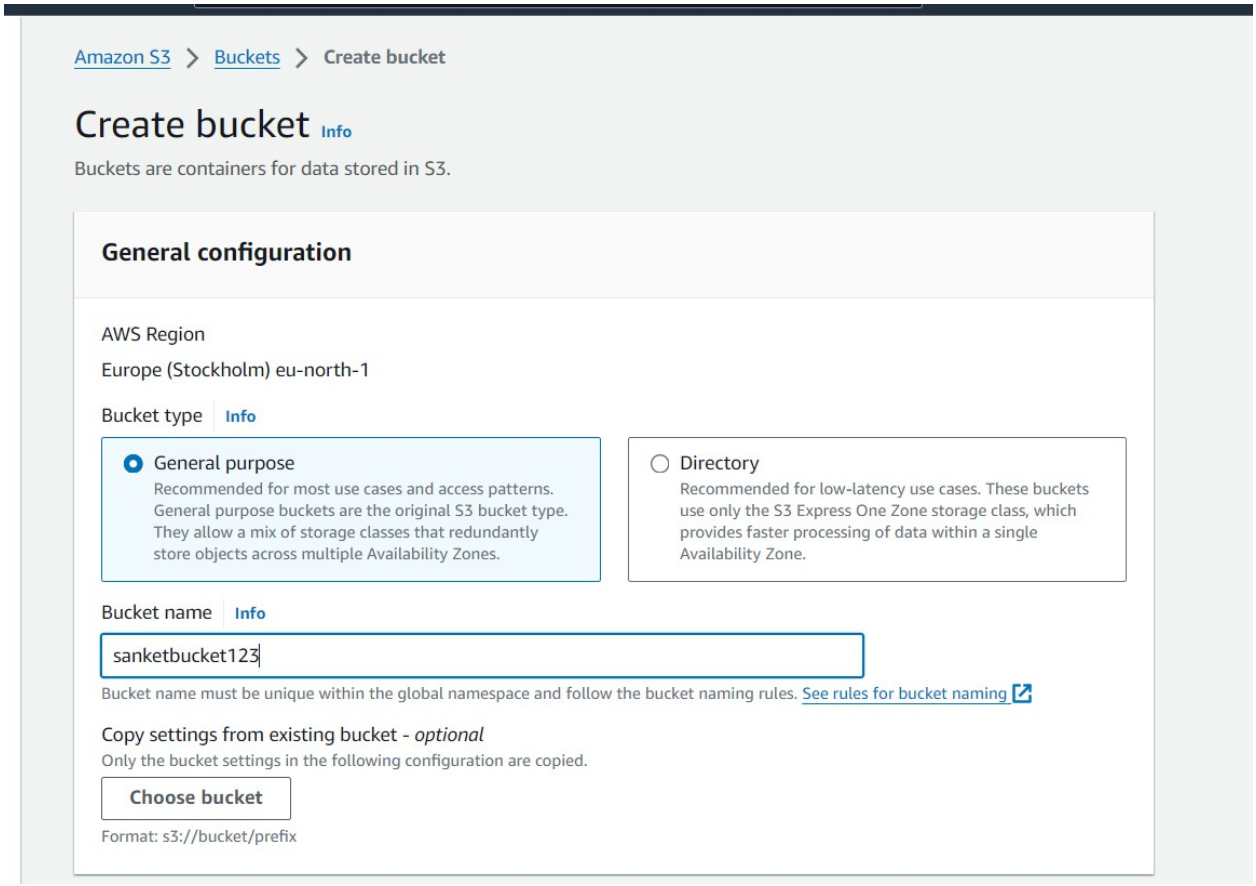
Jai Talreja 61 D15A

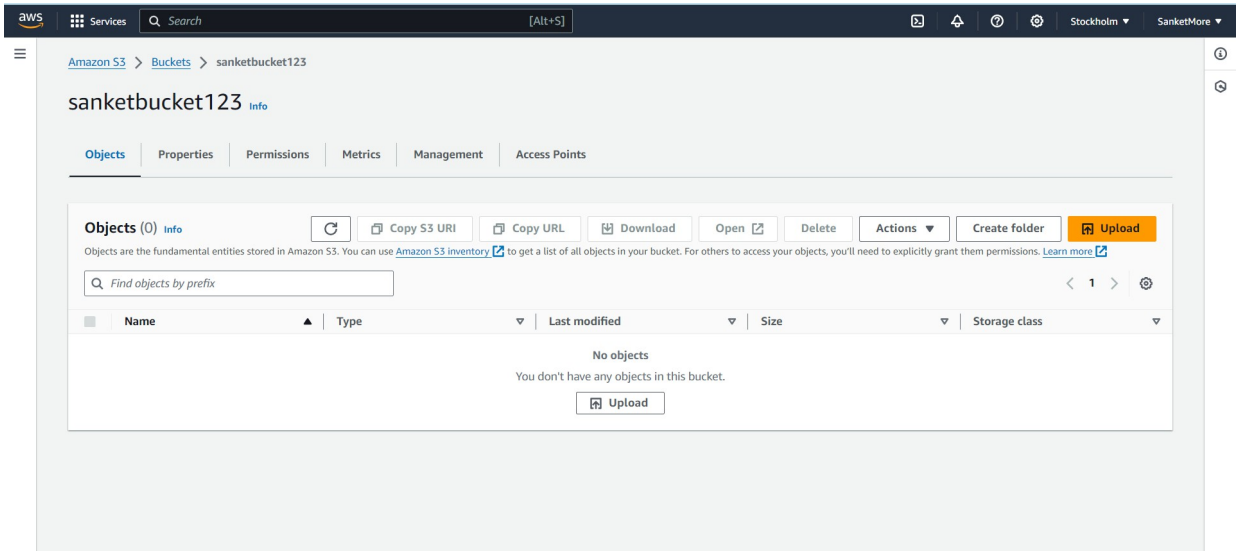


Jai Talreja 61 D15A

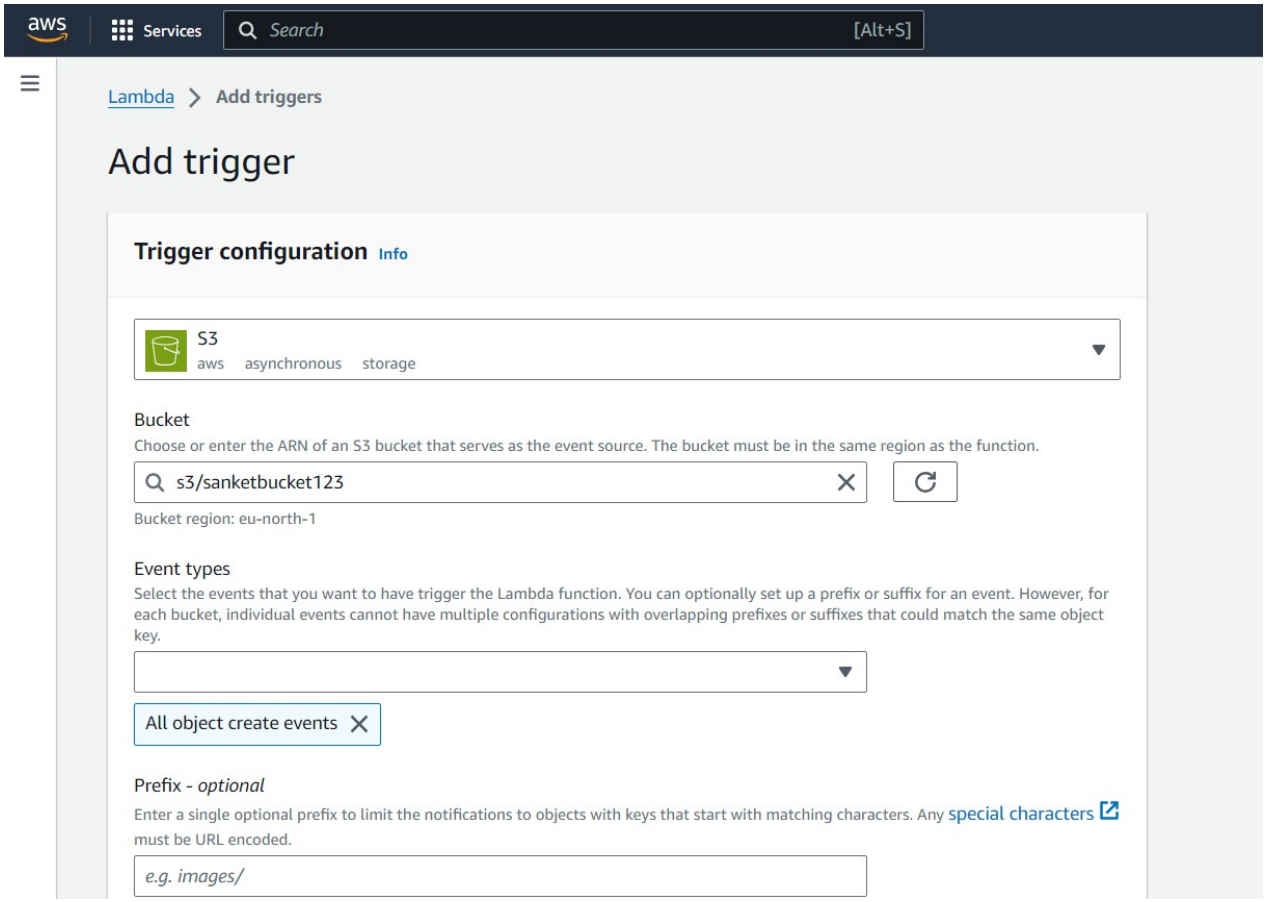
Screenshots for Experiment 12

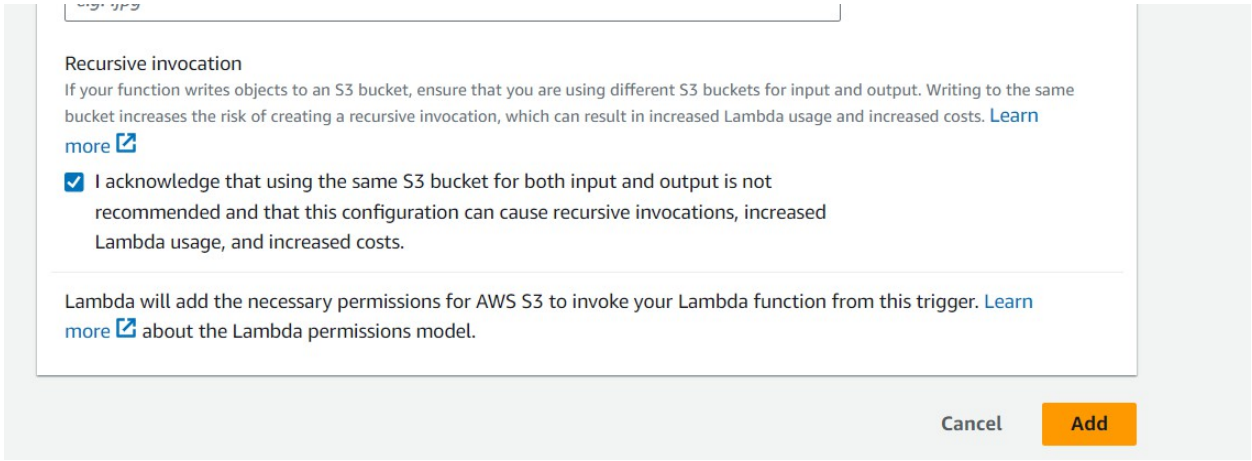
Create an S3 bucket of the same location as that of the Lambda function



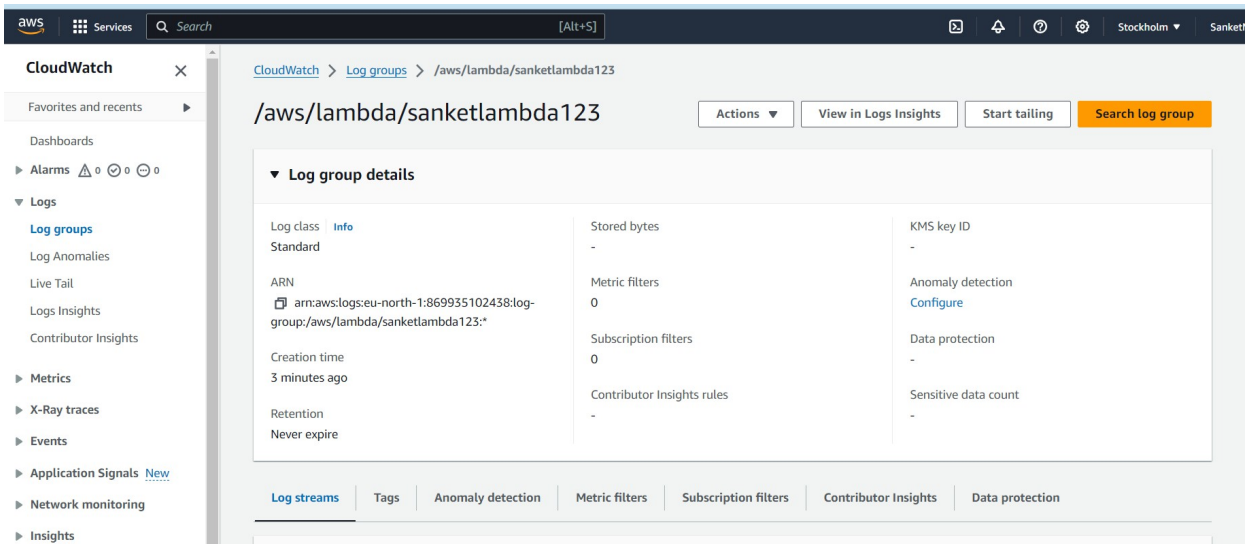


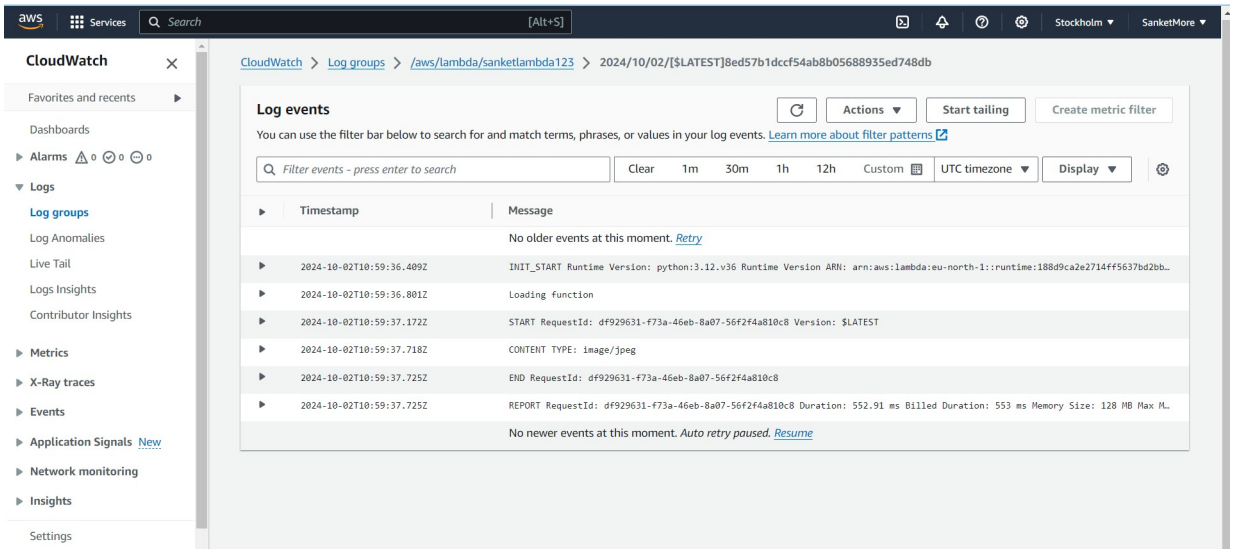
Jai Talreja 61 D15A





Jai Talreja 61 D15A





Sai Tabreja D151 61

04/05

AdvDevOps Assignment 1

1

Ans:1) Steps to host a video streaming service using S3 on AWS.

(1) Create an S3 bucket

- Log in to your AWS management console
- Navigate to the S3 service.
- Click on "Create Bucket" and follow the prompts to set up your bucket.
- Name your bucket
- Choose a region to host the bucket from.
- Disable ACLs (Access Control Lists).
- Block public access if you only want users to connect via Cloudfront.
- Disable Bucket versioning.
- Use default encryption.
- Click "Create Bucket".

(2) Upload your videos

- Click on the created bucket to open it.
- Click on "Upload" and choose the video files you want to upload.
- Click on "Upload" again.

(3) Configure bucket policy

- In the S3 bucket, under "Permissions" tab, update bucket policy to allow CloudFront access to your videos.

(4) Setup a CDN (Content Delivery Network) via CloudFront.

- Choose your S3 bucket as the origin.
- Click on "Redirect HTTP to HTTPS".
- Keep cache policy as 'Caching Optimized'
- Check "[redacted]" enable security protections" for Web App Firewall.

- Use all edge locations for best performance
- Leave the remaining settings at default values.
- Copy the policy and update your bucket.

(5) Access videos via CloudFront.

- Use your CloudFront distribution name and video key to access it.

(6) Secure your Content

- Use IAM roles and policies.
- Enable encryption.

(7) Monitor and Optemize

- Use CloudFront's analytics and monitoring tools.
- Optimize Caching settings in CloudFront settings.

Ans:2) BMW Group's Cloud Transformation with AWS
A Case Study

Overview:- The BMW Group, renowned for its premium automobiles and motorcycles, embarked on a digital transformation journey to enhance its operational efficiency and innovation capabilities.

By migrating its on-premises data lake to Amazon Web Services (AWS), BMW has been able to leverage cloud technology to meet global demands and drive Innovation.

Challenges:- BMW faced several challenges with its existing infrastructure, like,

Scalability issues to accommodate the increasing volume of data generated by BMW's global operations.

Fragmented data causing data silos, causing inability to fulfill needs of,

Real-Time access.

Solutions:- (1) Cloud Data Hub (CDH)

- BMW developed a centralized data lake using

Amazon S3 for scalable and secure storage and Amazon Redshift, which enables fast querying and analysis of large datasets.

(2) Machine Learning and Predictive Analysis

- Amazon SageMaker to train and deploy machine learning models.
- Amazon Kinesis to facilitate real-time data streaming and analytics.

(3) Blockchain Technology

- Amazon Elastic Kubernetes Service (EKS) to enhance to traceability to of automative parts.

(4) Multilingual Business Processes using Amazon Translate.

Outcomes :- The migration to AWS has yielded several significant benefits for BMW.

- Enhanced Scalability
- Improved Innovation
- Real-Time Insights
- Operational Efficiency

Services used :-

- S3
- RedShift
- SageMaker
- Kinesis
- EKS
- Translate

Conclusion :-

BMW's collaboration with AWS highlights the transformative power of cloud technology in driving scalability, innovation, and real-time data processing.

This partnership has enabled BMW to stay at the forefront of the automobile industry, delivering premium experiences to its customers, worldwide.

Hotstar's success with AWS
A Case Study

Overview:- Hotstar, now known as Disney+Hotstar is one of India's leading OTT streaming platforms.

It has leveraged Amazon Web Services (AWS) to handle massive traffic spikes, especially during live events like the Indian Premier League (IPL) cricket matches.

By using AWS, Hotstar has been able to scale its infrastructure to support millions of concurrent viewers.

Challenges:- Hotstar faced the following challenges with its existing infrastructure

Scalability to handle sudden traffic spikes,

Performance, to ensure low-latency and high-quality streaming.

Cost management by scaling infrastructure

Solutions :- To address its challenges, Hotstar emptemented the following services.

- Amazon EC2 and Elastic Load Balancing (ELB) for running applications.
- Amazon CloudFront for content delivery.
- Amazon kinesis for Real-Time analytics.
- EC2 ~~s~~ spot instances to optemize costs.

Outcomes :- The migration to AWS has provided Hotstar with several benefits.

- Scalability
- Performance
- Cost Efficiency
- Real-time insights

Conclusion :- Hotstar's collaboration with AWS showcases how leveraging cloud technology enabled the platform to deliver high-quality streaming experiences to millions of users.

Ans:3) Kubernetes is used because it automates the deployment, scaling, and management of containerized applications.

It ensures high availability, efficient resource utilization, and easy scaling, making it ideal for managing complex, distributed system.

Advantages :-

- Scalability
- High availability
- Portability
- Resource Efficiency
- Extensibility
- Automation

Disadvantages :-

- Complexity
- Resource Intensive
- Operational Overhead
- Security
- Cost

How Adidas leverages kubernetes :-

- Containerization and Agile Development.
- By using kubernetes, Adidas can deploy applications faster and more quickly.

- Improved Developer Experience

  With Kubernetes, Adidas no longer faces delays in getting development environments set up.

- Enhanced E-Commerce Performance

  Kubernetes has reduced site load times to half and allowed more frequent releases.

- Scalability and Reliability

  Adidas manages thousands of pods and hundreds of nodes, ensuring their systems can handle high traffic volumes, especially during peak shopping events.

- Continuous Delivery

  Rapid push updates help Adidas stay competitive in the fast-paced retail market.

Ans:- Nagios is an open-source monitoring and alerting system designed to track the health and performance of IT infrastructure, networks, and applications.

Key-Features :-

- Server monitoring
- Network monitoring
- Service availability
- Resource Utilizations
- Performance metrics and Trend Analysis
- Log file monitoring
- Alerting and Notification
- Application monitoring
- Security monitoring
- Environmental monitoring

Nagios is extensively used in E-Services to ensure the reliability and performance of various IT components.

It provides real-time monitoring, easy-to-understand graphics, and intuitive statistics, which helped in reducing troubleshooting time and ensuring high availability of services.

By using Nagios, E-services can,

- Centralize monitoring
- Improve response times
- Enhance reliability
- Simplify management

Jai Tabrej D15A 61

AdvDevOps Assignment 2 1

Q.1] Create a REST API with the Serverless Framework.

Ans) "Serverless Framework" refers to the user not needing to manage and maintain their own servers.

Steps :-

1) Install serverless framework CLI globally using 'node package manager' (npm).

2) Create node JS serverless project, inside a directory intended for it.

This service will house all your Lambda functions, configurations and cloud resources.

3) The project scaffold creates essential files like handler.js which contains code for lambda functions and serverless.yml.

4) Create a REST API resource that handles HTTP requests.

5.) Deploy the service and upload new resources to AWS and set up the infra

6.) Once deployed, test the REST API using tools like curl or Postman by making post requests to generated API!

7.) To store submitted data you integrate AWS DynamoDB as a database.

8.) Add functionalities like list all candidates by ID.

9.) You need to ensure that serverless framework is given right permissions to interact with AWS resources like DynamoDB.

10.) After deployment serverless framework provides service information like deployed endpoints, API key, log stre

Case Study for SonarQube

Creating ~~Quality~~ profile in SonarQube for testing project quality.

SonarQube is an Open Source platform used for Continuous inspection of quality.

SonarQube detects bugs and code smells along with vulnerabilities in projects across various programming languages.

Profile Creation:

Quality profiles in SonarQube are essential configurations that define rules applied during code analysis. Each project has a quality profile for every supported language with default being 'Sonar Way'.

Custom profiles can be created by copying or extending existing ones.
Permissions to manage quality profiles are restricted to users with administrator privileges. Permissions can also be exported from others instances via backup and restore.

2.) Sonarcloud to analyze GitHub code:

Sonarcloud is a Cloud-Based counter-p of SonarQube that integrates directly with GitHub, BitBucket, Azure and GetHub repositories.

To get started with SonarCloud via GitHub setup with each project corresponding to GitHub repositories. Automatic analyses happens directly in SonarCloud while CI based analyses integrates with your build process a the analyses is complete results c be viewed in both SonarCloud an GitHub including including security import issue.

3.) ~~Sonarlint~~ in Java IDE:

Sonarlint is an IDE that performs on-the-fly code analyses as you write code. It helps develops en development environments, such as IntelliJ Idea or Eclipse. This approa ensures immediate feedback on code quality. Promoting clean & maintainab code from the beginning.

Analyzing Python projects with SonarQube:

SonarQube supports Python test coverage reporting but it requires third party tool like coverage. For setup, you can use Tox and Coverage. Py to configure and run tests in your tox.ini. The build process can also be automated using GitHub Actions, which installs dependencies, runs tests and invokes SonarQube scan. Ensure report is in XML format.

Analyzing Node projects with Sonar Qube:

You can configure SonarQube to analyze Node JS projects by installing the appropriate plugin and using SonarScanner to scan the projects.
SonarQube will check the code against industry standard rules and best practices, flagging issues related to security vulnerabilities, bugs and performances optimization.

Q.3 At a large organizations, your cente operation team may get many repetitive infrastructure requests. You can create and use Terraform modules that codify the standards for deploying & managing services in your large Terraform Cloud can also integrate wit ticketing system like services.

Ans) Implementing a 'self-serve' infrastru model using Terraform can transform 3 how large organizations manage th infrastructure independantly.

Organization can enhance efficiency reduce bottlenecks, and ensure compliance with established needs.

→ Overwhelming number of repititive requests leads to delays in service deliveries.

→ Benefits:

:) Modularity and Reusability
:) Standardizations
:) Increased Efficiency
:) Integration with ticketing systems

Implementation Steps :

1) Impdentify Infrastructure components
- Begin by identifying which components of your infrastructure can be modularized.

2) Establish Governance and Best Practices to ensure clarity and maintainability.

3) Develop Terraform modules to include input variables for customizations and outputs for integration with other modules.

4) Testing and Validation of best modules for module management, utilize the Terraform registry.

5) Version Control for modules to track changes over-time. This helps manage dependencies effectively and minimize disruptions during updates.

6) Documentation for each module, usage examples, input/output descriptions and any dependencies.

7) Encourage Collaboration and promote consistency in deployments and facilitat knowledge within the organisation. Their approach not only streamlines pro but also enhances agility in respond to changing business needs.

Ultimately it leads to a more responsive IT environment that support innovation and growth within the organization.