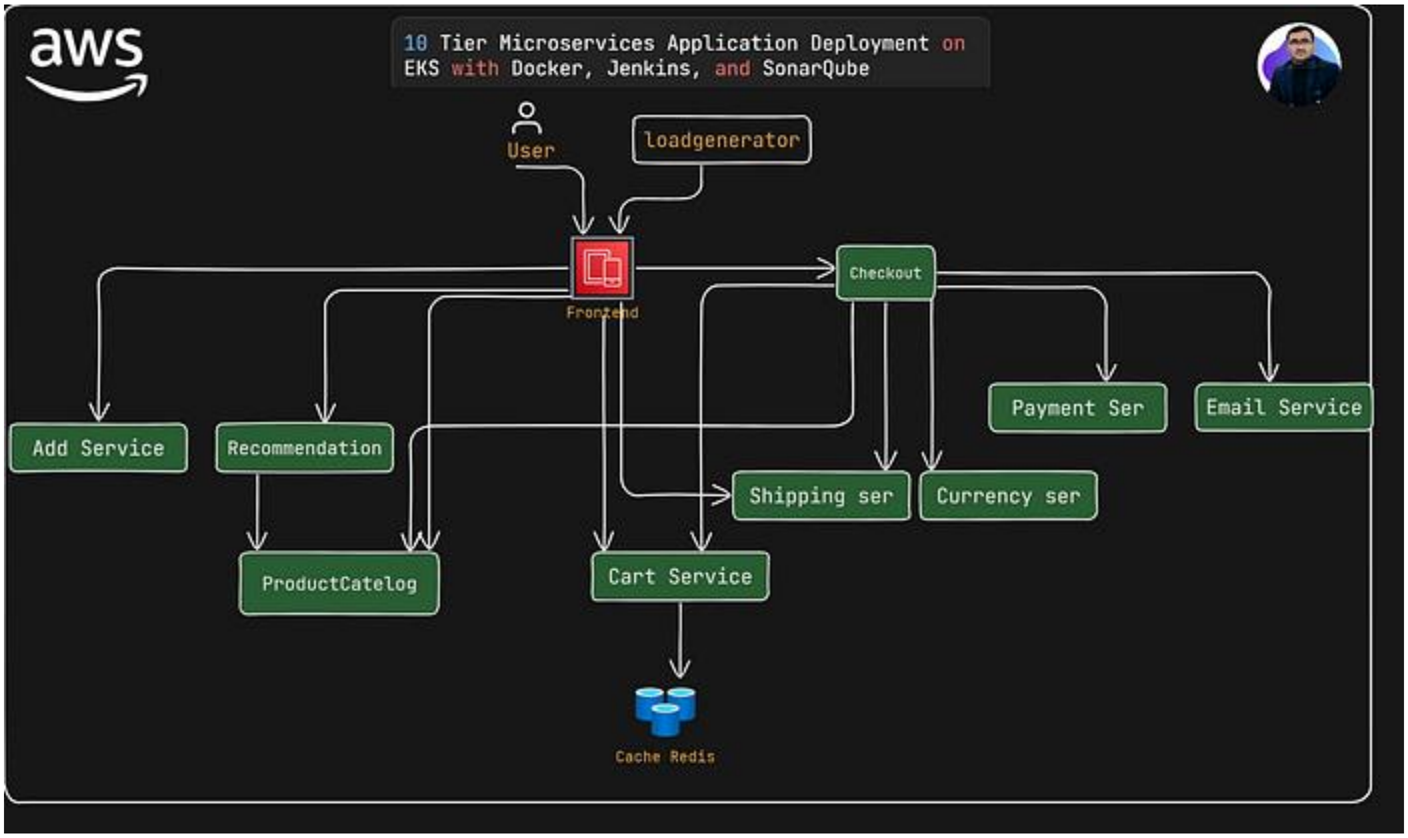
10-Tier Application Deployment on AWS EKS



Frontend: Golang
Exposes an HTTP server to serve the website. Does not require signup/login and generates session IDs for all users automatically.

[cartservice](): C#
Stores the items in the user's shopping cart in Redis and retrieves it.

productcatalogservice: Golang
Provides the list of products from a JSON file and ability to search products and get individual products.

currencyservice: Nodejs
Converts one money amount to another currency. Uses real values fetched from European Central Bank. It’s the highest QPS service.

paymentservice: Nodejs
Charges the given credit card info (mock) with the given amount and returns a transaction ID.

shippingservice: Golang
Gives shipping cost estimates based on the shopping cart. Ships items to the given address (mock)

emailservice: Python
Sends users an order confirmation email (mock).

checkoutservice: Golang
Retrieves user cart, prepares order and orchestrates the payment, shipping and the email notification.

recommendationservice: Python
Recommends other products based on what’s given in the cart.

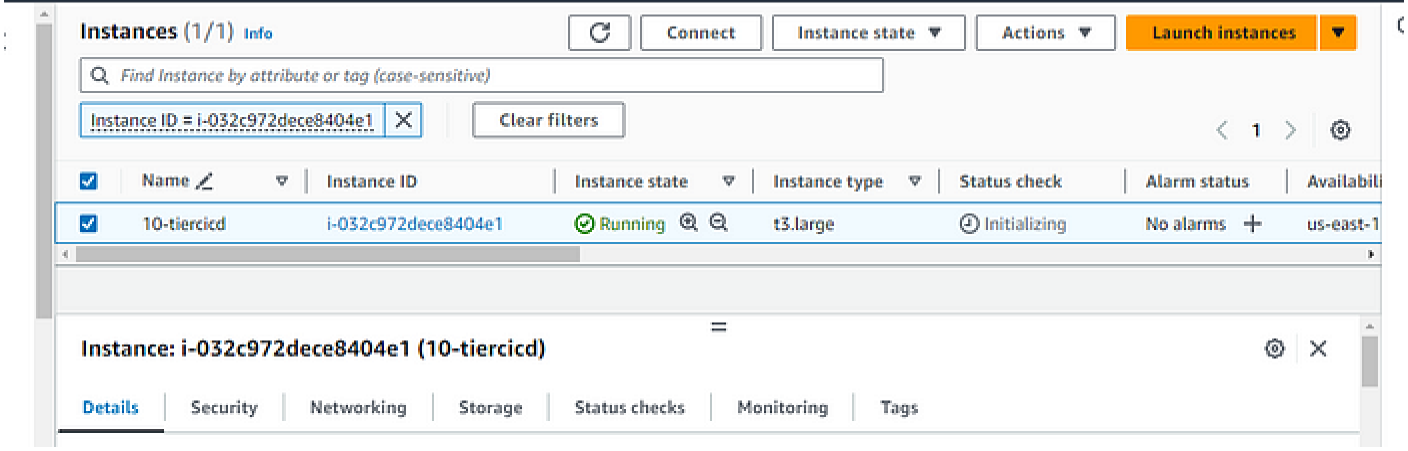
adservice: Java

Provides text ads based on given context words.

loadgenerator: Python/Locust

Continuously sends requests imitating realistic user shopping flows to the frontend.

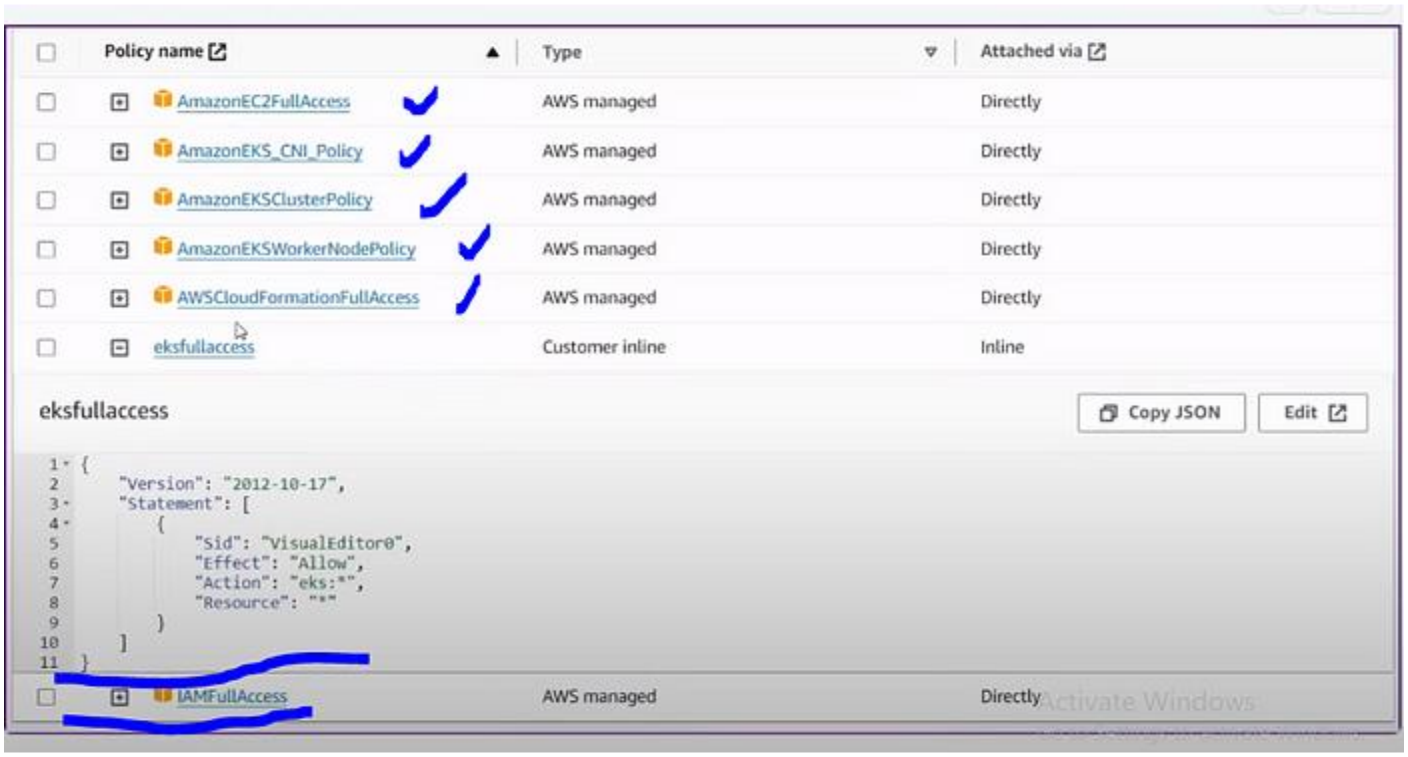
1. Create an AWS EC2 instance



Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availabili
10-tiercicd	i-032c972dece8404e1	Running	t3.large	Initializing	No alarms	us-east-1

Instance: i-032c972dece8404e1 (10-tiercicd)

2. Create a user and give the following permissions



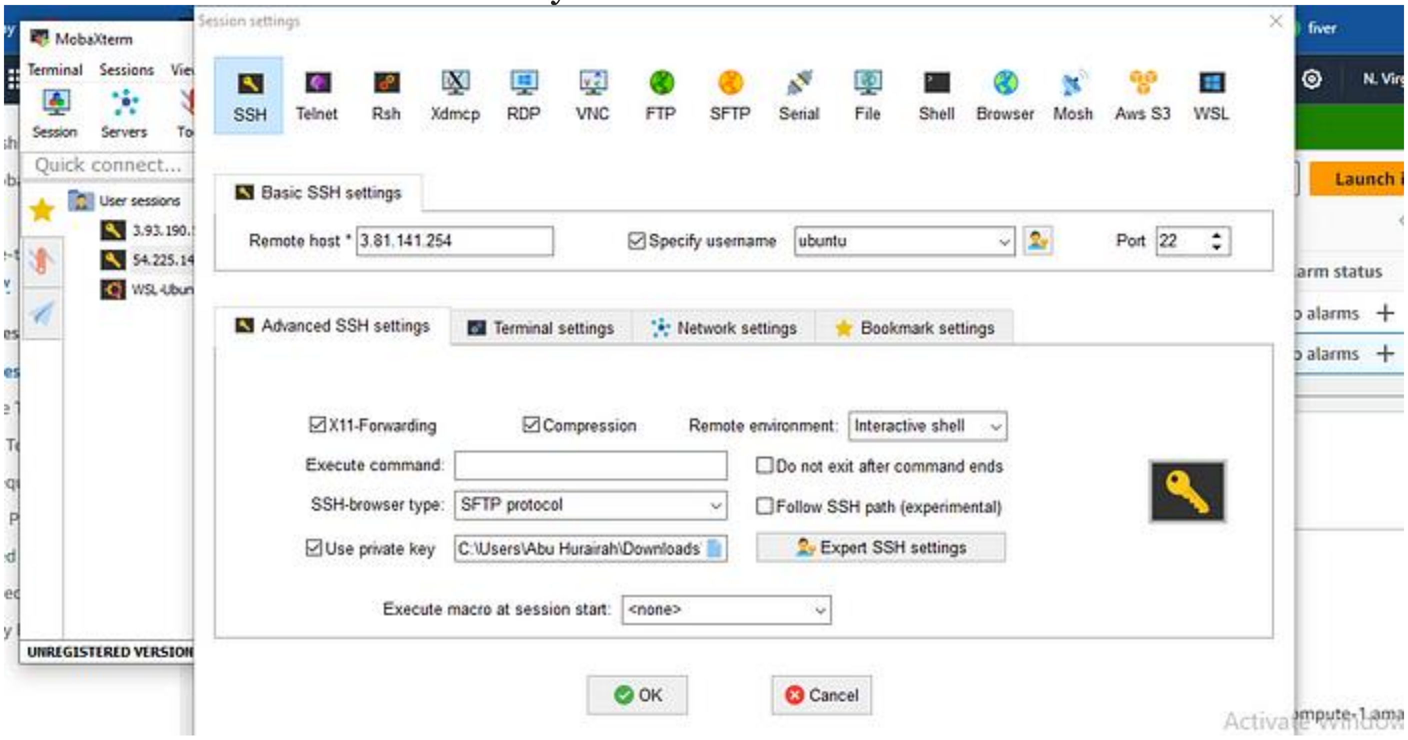
	Policy name	Type	Attached via
☐	AmazonEC2FullAccess ✓	AWS managed	Directly
☐	AmazonEKS_CNI_Policy ✓	AWS managed	Directly
☐	AmazonEKSClusterPolicy ✓	AWS managed	Directly
☐	AmazonEKSWorkerNodePolicy ✓	AWS managed	Directly
☐	AWSCloudFormationFullAccess ✓	AWS managed	Directly
☐	eksfullaccess	Customer inline	Inline

eksfullaccess

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor0",
            "Effect": "Allow",
            "Action": "eks:*",
            "Resource": "*"
        }
    ]
}
```

	Policy name	Type	Attached via
☐	IAMFullAccess	AWS managed	Directly

Connect with the EC2 instance you can use ssh or mobaXterm



Basic SSH settings

Remote host * 3.81.141.254 ☑ Specify username ubuntu Port 22

☑ X11-Forwarding ☑ Compression Remote environment: Interactive shell

Execute command:

☐ Do not exit after command ends

SSH-browser type: SFTP protocol

☐ Follow SSH path (experimental)

☑ Use private key C:\Users\Abu Hurairah\Downloads

Execute macro at session start: <none>

in the request, the host adds a public add of the instance
check to specify the user name and give the name of the instance
click on the advanced SSH setting
check to Use a private key and give the address of your key.

3. After connect install aws ctl on your server to give your credentials
https://docs.aws.amazon.com/cli/latest/userguide/getting-started-install.html

4. After connecting install Jenkins on your server
https://www.jenkins.io/doc/book/installing/linux/#debianubuntu

5. Now install kubctl on the linux
https://kubernetes.io/docs/tasks/tools/install-kubectl-linux/#install-kubectl-binary-with-curl-on-linux

6. Install eksctl
https://docs.aws.amazon.com/emr/latest/EMR-on-EKS-DevelopmentGuide/setting-up-eksctl.html

7. install docker and give permission

```
sudo apt-get install docker.io
sudo usermod -aG docker ubuntu
sudo newgrp docker
```

8. install sonarqube from docker image
docker run -d -p 9000:9000 sonarqube:lts-community

Now it is time to expose sonarqube and jenkins on ec2 instance
go to ec2 instance and edit its inbound rules

SonarQube is running on 9000
Jenkins is running on 8080

9. Install EKS

```
eksctl create cluster --name=my-eks2 \
        --region=ap-south-1 \
        --zones=ap-south-1a,ap-south-1b \
        --without-nodegroup

// after the above setup complete run this
eksctl utils associate-iam-oidc-provider \
 --region ap-south-1 \
 --cluster my-eks2 \
 --approve

eksctl create nodegroup --cluster=my-eks2 \
   --region=ap-south-1 \
   --name=node2 \
   --node-type=t3.medium \
   --nodes=3 \
   --nodes-min=2 \
   --nodes-max=3 \
   --node-volume-size=20 \
   --ssh-public-key=10-tier-key \
   --managed \
   --asg-access \
   --external-dns-access \
   --full-ecr-access \
   --appmesh-access \
   --alb-ingress-access
```

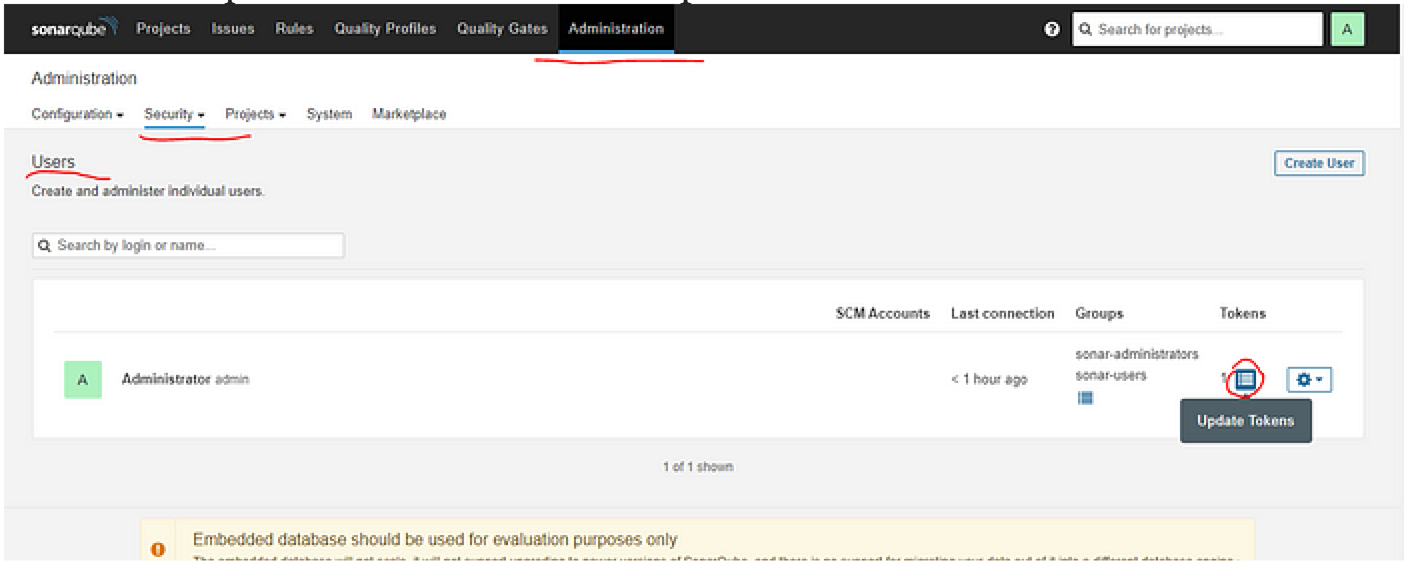
Install the following Plugins in Jenkins

go jenkins and and click on plugins.

```
sonarqube scanner
sonarqube
docker
docker pipeline
docker common
cloud base docker build and publish
kubernetes
kubernetes cli
```

Now we need to configure Sonarqube with Jenkins

Go to Sonarqube and follow the below pictures



Administration

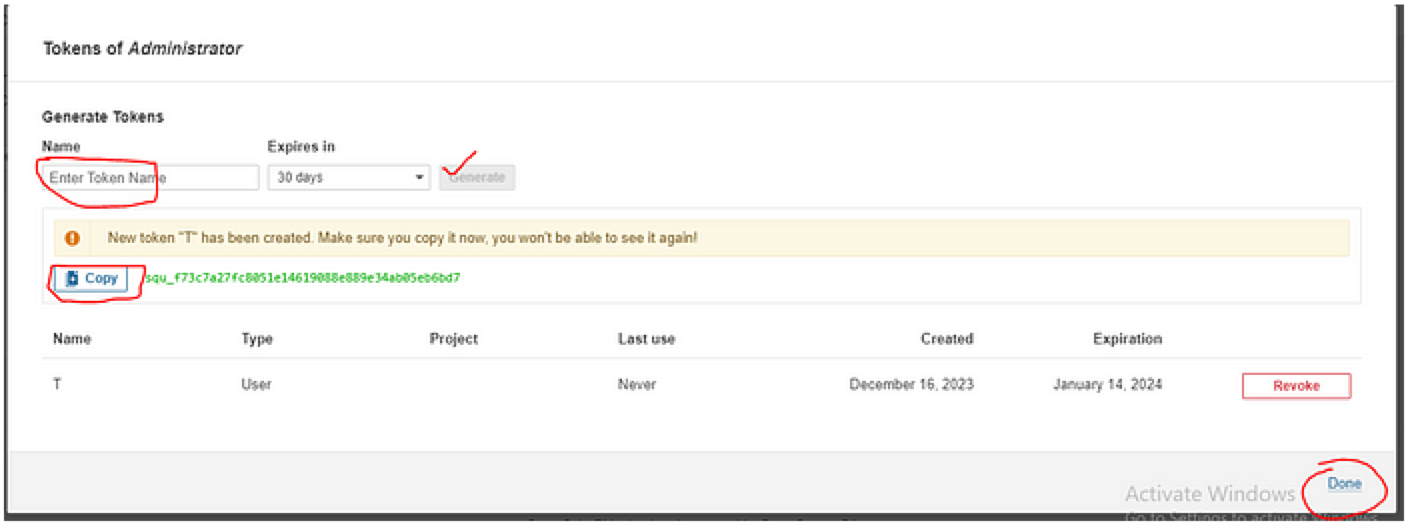
Users

Create and administer individual users.

	SCM Accounts	Last connection	Groups	Tokens
A Administrator admin		< 1 hour ago	sonar-administrators sonar-users	

1 of 1 shown

Embedded database should be used for evaluation purposes only


Tokens of *Administrator*

Generate Tokens

New token "T" has been created. Make sure you copy it now, you won't be able to see it again!

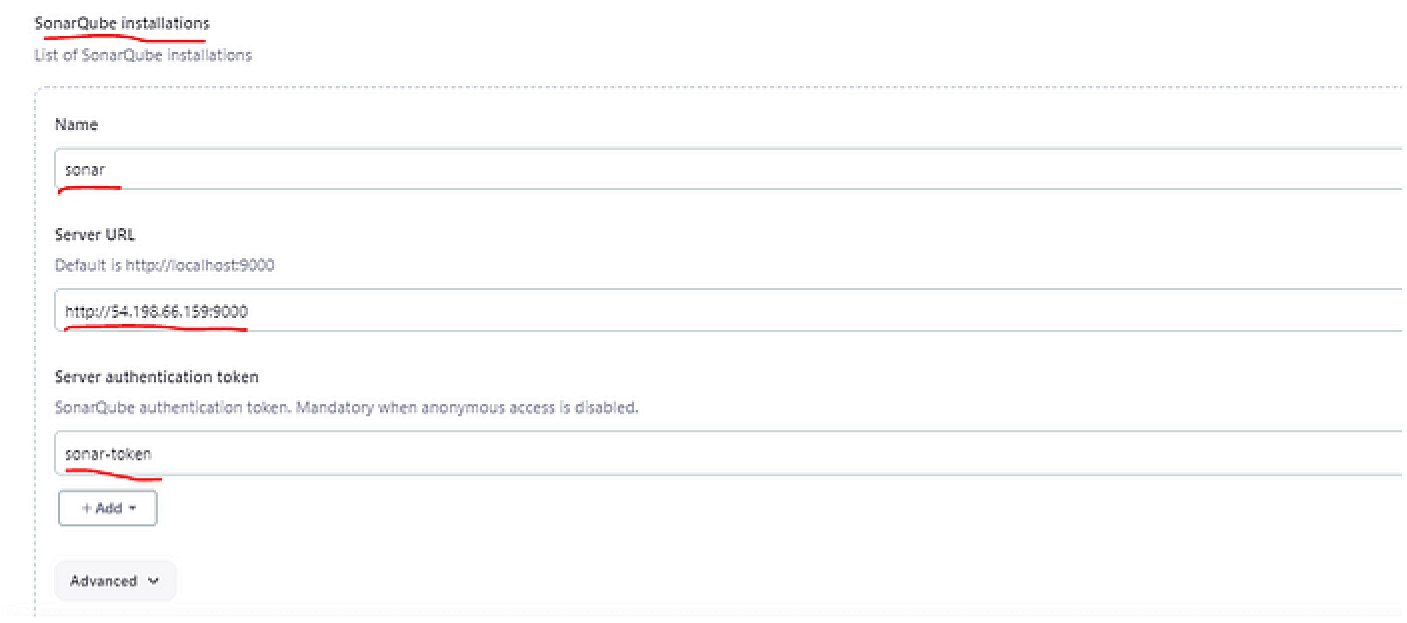
```
squ_f73c7a27fc8051e14619088e889e34ab05eb6bd7
```

Name	Type	Project	Last use	Created	Expiration	
T	User		Never	December 16, 2023	January 14, 2024	Revoke

Now come to Jenkins
click on manage Jenkins
click on Create credentials
click on global
the token from sonarqube is paste here
then add credentials

10. to connect Sonarqub sever we go manage Jenkins and click systems

scroll down to go Sonarqube installation


SonarQube installations

List of SonarQube installations

Name

sonar

Server URL

Default is http://localhost:9000

http://54.198.66.159:9000

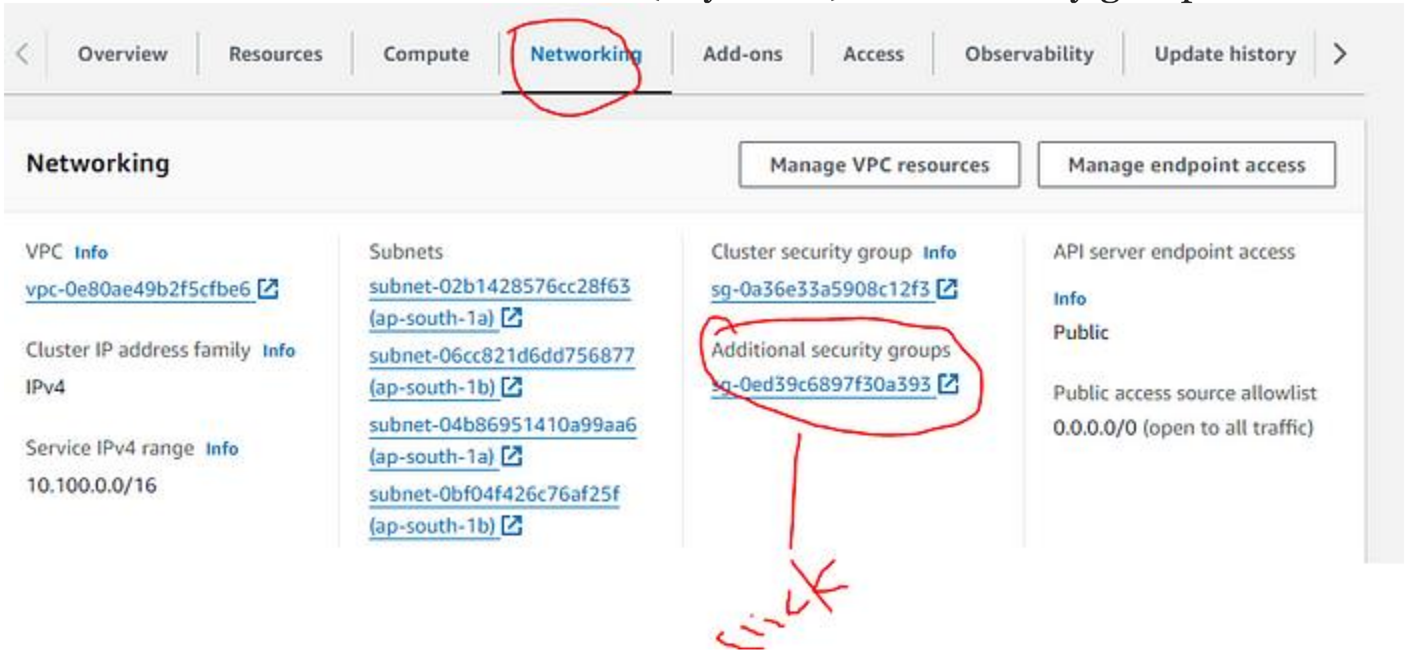
Server authentication token

SonarQube authentication token. Mandatory when anonymous access is disabled.

sonar-token

now click on apply

11. Go eks on AWS and add all traffic(anywhere) to its security group.



Networking

VPC Info

vpc-0e80ae49b2f5cfbe6

Cluster IP address family Info

IPv4

Service IPv4 range Info

10.100.0.0/16

Subnets

subnet-02b1428576cc28f63 (ap-south-1a)

subnet-06cc821d6dd756877 (ap-south-1b)

subnet-04b86951410a99aa6 (ap-south-1a)

subnet-0bf04f426c76af25f (ap-south-1b)

Cluster security group Info

sg-0a36e33a5908c12f3

Additional security groups

sg-0ed39c6897f30a393

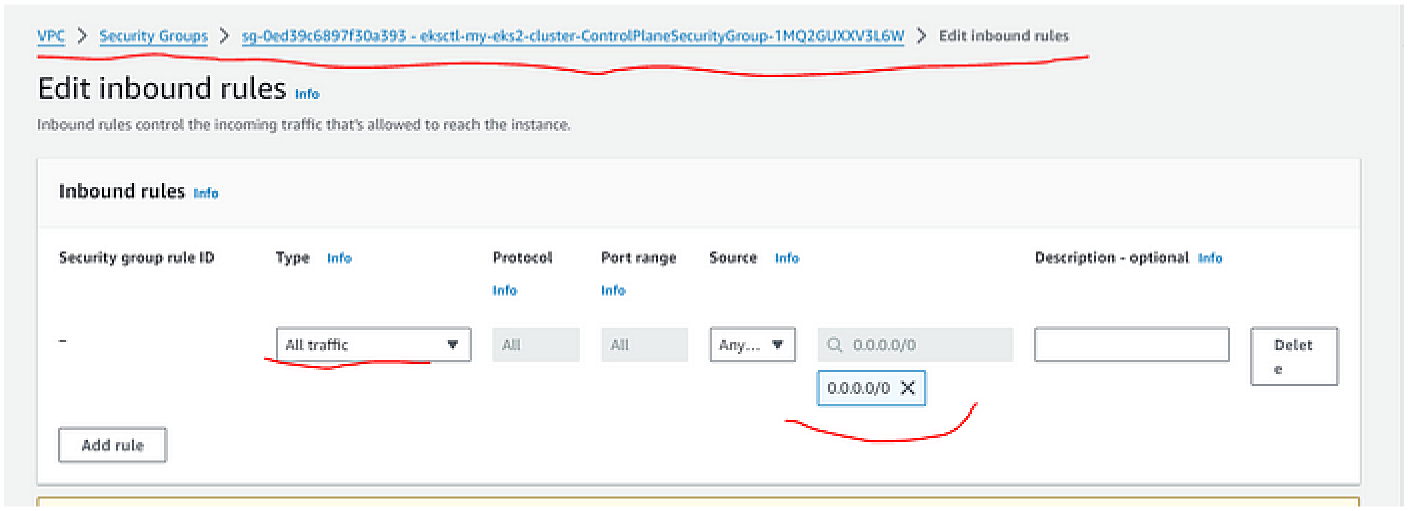
API server endpoint access

Info

Public

Public access source allowlist

0.0.0.0/0 (open to all traffic)



VPC > Security Groups > sg-0ed39c6897f30a393 - eksctl-my-eks2-cluster-ControlPlaneSecurityGroup-1MQ2GUXXV3L6W > Edit inbound rules

Edit inbound rules Info

Inbound rules control the incoming traffic that's allowed to reach the instance.

Inbound rules Info

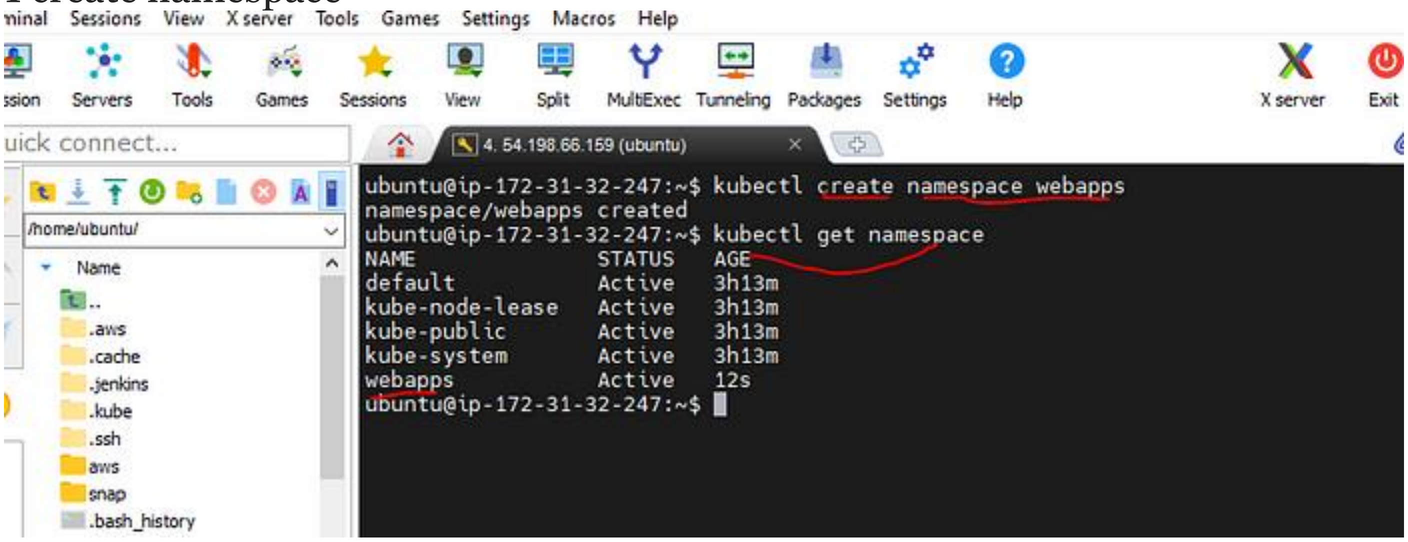
Security group rule ID	Type Info	Protocol Info	Port range Info	Source Info		Description - optional Info	
-	All traffic	All	All	Any...	0.0.0.0/0 0.0.0.0/0		Delete

Add rule

12. Create a Service Account and role and Assign that role create a secret service account, and generate a token

Creating Service Account

1 create namespace



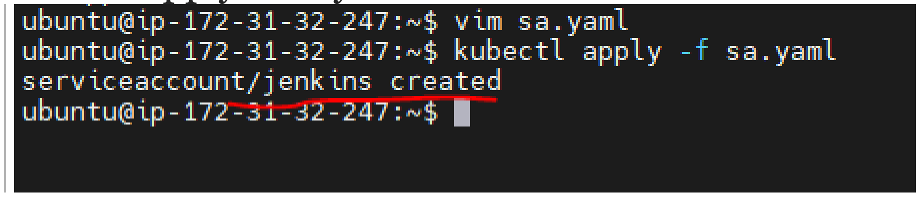
```
ubuntu@ip-172-31-32-247:~$ kubectl create namespace webapps
namespace/webapps created
ubuntu@ip-172-31-32-247:~$ kubectl get namespace
NAME              STATUS   AGE
default           Active   3h13m
kube-node-lease   Active   3h13m
kube-public       Active   3h13m
kube-system       Active   3h13m
webapps           Active   12s
ubuntu@ip-172-31-32-247:~$
```

2. Create sa.yml file and add the follow code

```
apiVersion: v1
kind: ServiceAccount
metadata:
  name: jenkins
  namespace: webapps
```

run the file

kubectl apply -f sa.yaml

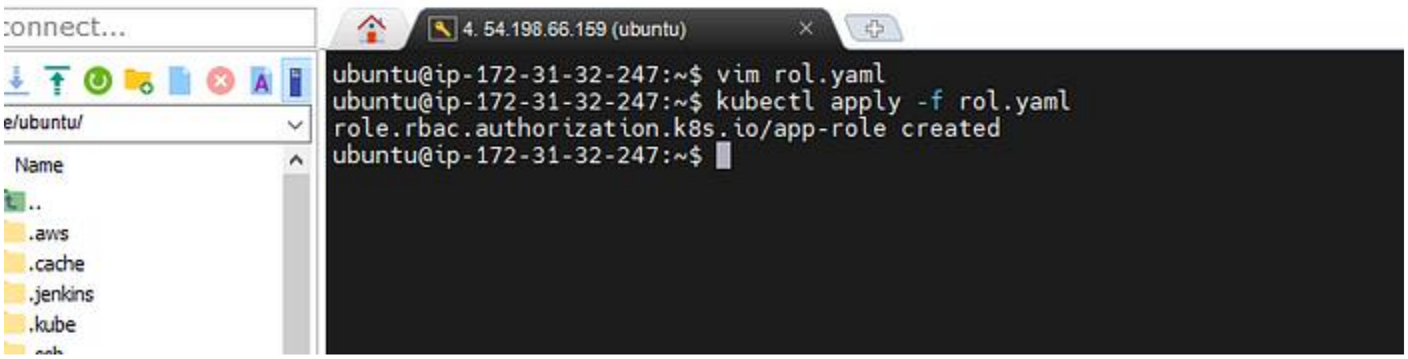


```
ubuntu@ip-172-31-32-247:~$ vim sa.yaml
ubuntu@ip-172-31-32-247:~$ kubectl apply -f sa.yaml
serviceaccount/jenkins created
ubuntu@ip-172-31-32-247:~$
```

3. Now we need to create role

```
- -
apiVersion: rbac.authorization.k8s.io/v1
kind: Role
metadata:
 name: app-role
 namespace: webapps
rules:
 - apiGroups:
 - ""
 - apps
 - autoscaling
 - batch
 - extensions
 - policy
 - rbac.authorization.k8s.io
 resources:
 - pods
 - configmaps
 - deployments
 - daemonsets
 - componentstatuses
 - events
 - endpoints
 - horizontalpodautoscalers
 - ingress
 - jobs
```

```
  - limitranges
  - namespaces
  - nodes
  - pods
  - persistentvolumes
  - persistentvolumeclaims
  - resourcequotas
  - replicasets
  - replicationcontrollers
  - serviceaccounts
  - services
  verbs:
  - get
  - list
  - watch
  - create
  - update
  - patch
  - delete
```



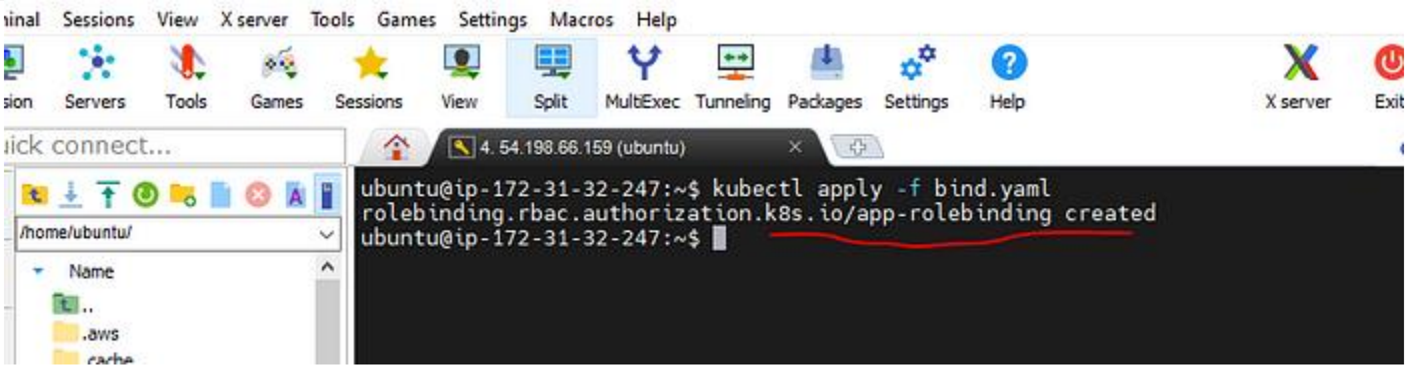
```
ubuntu@ip-172-31-32-247:~$ vim rol.yaml
ubuntu@ip-172-31-32-247:~$ kubectl apply -f rol.yaml
role.rbac.authorization.k8s.io/app-role created
ubuntu@ip-172-31-32-247:~$
```

4. now assigning the role to the service account

role binding

```
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  name: app-rolebinding
  namespace: webapps
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: Role
  name: app-role
subjects:
  - namespace: webapps
```

```
    kind: ServiceAccount
    name: jenkins
```

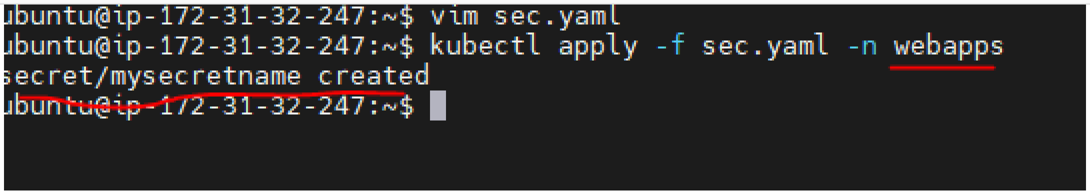


```
ubuntu@ip-172-31-32-247:~$ kubectl apply -f bind.yaml
rolebinding.rbac.authorization.k8s.io/app-rolebinding created
ubuntu@ip-172-31-32-247:~$
```

5. now creating a token for service account

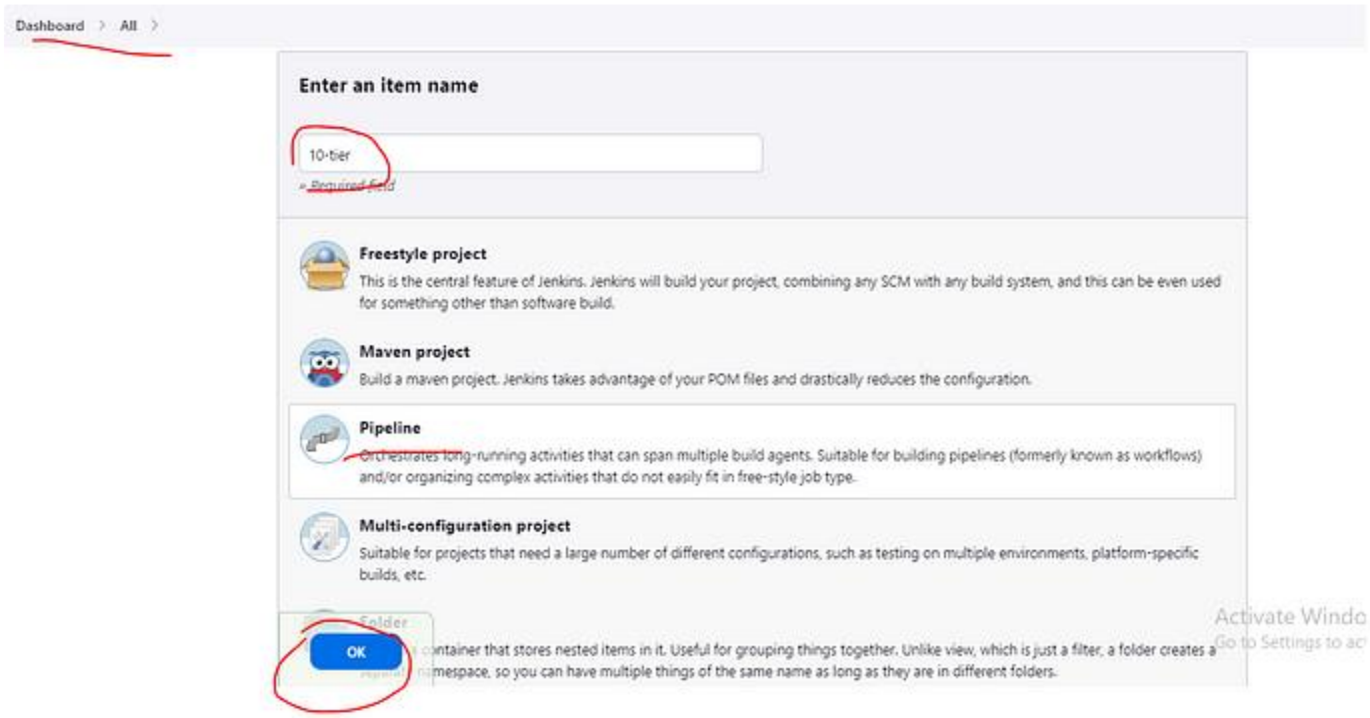
we have created the secret file with the following command

```
apiVersion: v1
kind: Secret
type: kubernetes.io/service-account-token
metadata:
  name: mysecretname
  annotations:
    kubernetes.io/service-account.name: jenkins
```



```
ubuntu@ip-172-31-32-247:~$ vim sec.yaml
ubuntu@ip-172-31-32-247:~$ kubectl apply -f sec.yaml -n webapps
secret/mysecretname created
ubuntu@ip-172-31-32-247:~$
```

Go to Jenkins and add a pipeline


Dashboard > All >

Enter an item name

10-tier

» Required field

Freestyle project
This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.

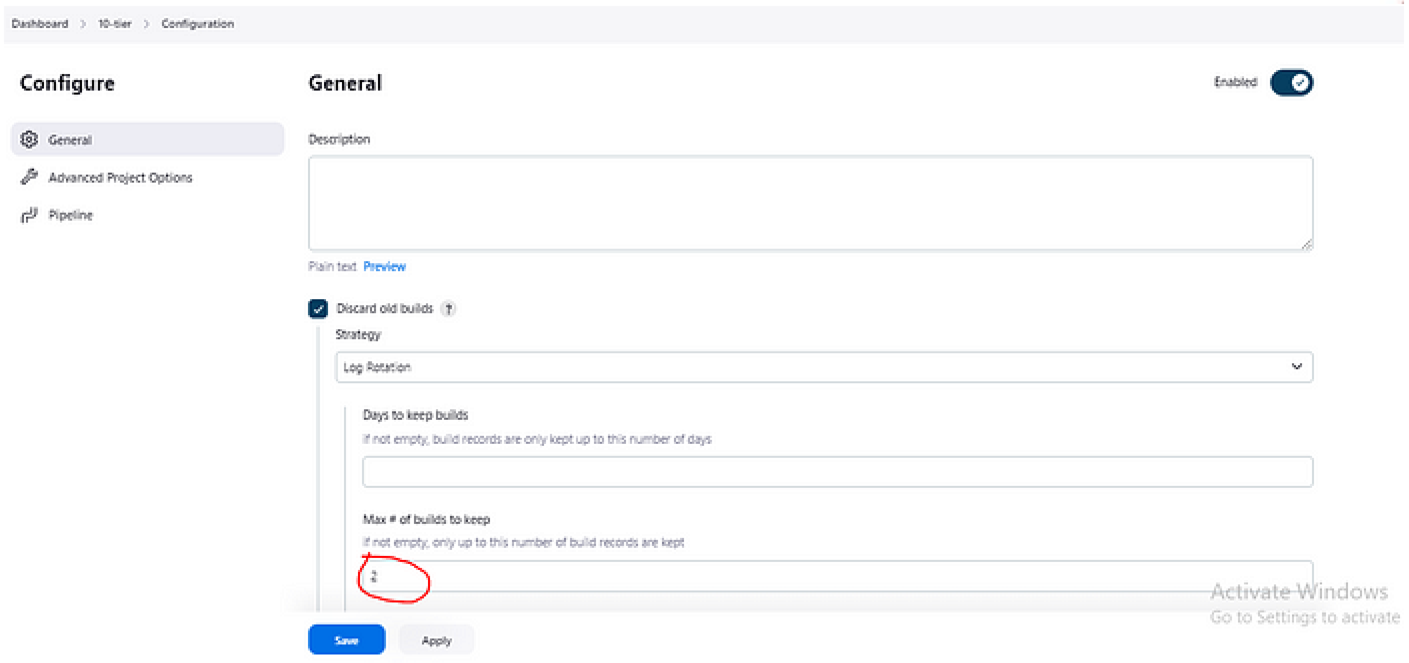
Maven project
Build a maven project. Jenkins takes advantage of your POM files and drastically reduces the configuration.

Pipeline
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.

Multi-configuration project
Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.

Folder
container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a namespace, so you can have multiple things of the same name as long as they are in different folders.

OK


Dashboard > 10-tier > Configuration

Configure

General
Advanced Project Options
Pipeline

General

Enabled

Description

Plain text Preview

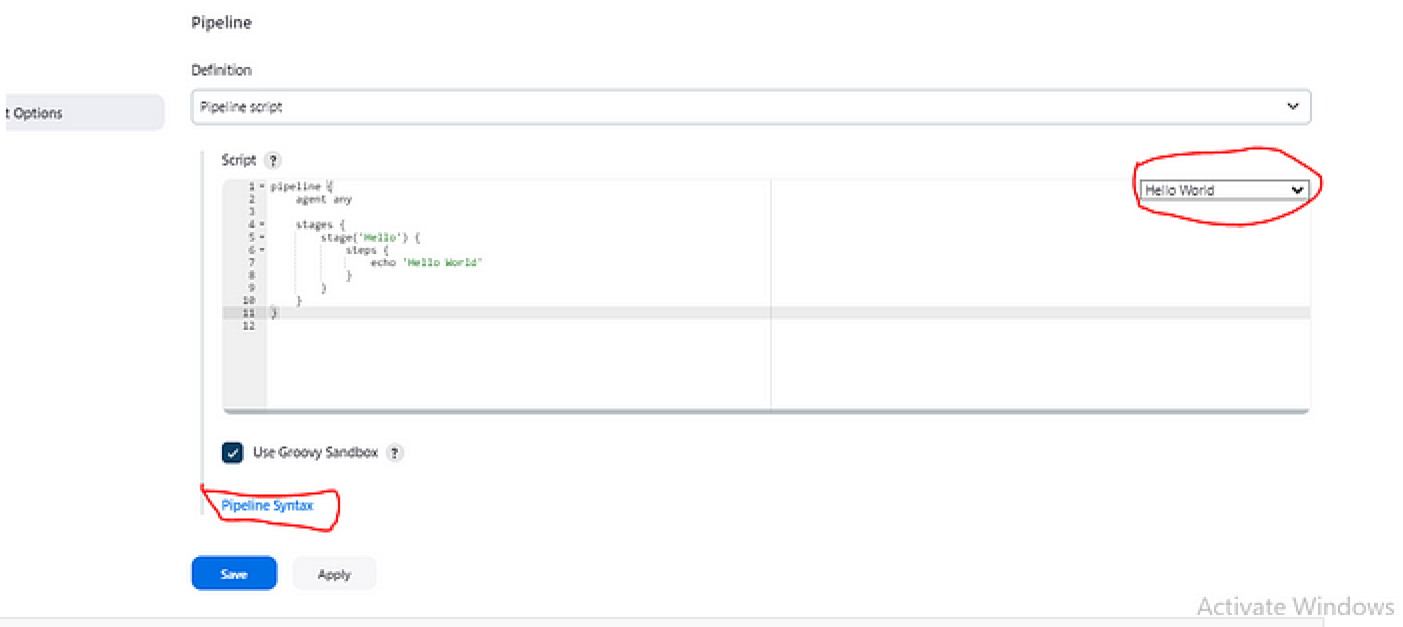
✓ Discard old builds ?

Strategy
Log Rotation

Days to keep builds
If not empty, build records are only kept up to this number of days

Max # of builds to keep
If not empty, only up to this number of build records are kept
2

Save Apply



```
pipeline {
    agent any

    stages {
        stage('Hello') {
            steps {
                echo 'Hello World'
            }
        }
    }
}
```

```
pipeline {
    agent any
    environment{
        SCANNER_HOME= tool 'sonar-scanner'
    }
    stages {
        stage('git checkout') {
            steps {
                git branch: 'latest', url:
'https://github.com/samsorrahman/10-Tier-MicroService-Appliction.git'
            }
        }
        stage('SonarQube')  {
            steps {
                withSonarQubeEnv('sonar') {
                    sh ''' $SCANNER_HOME/bin/sonar-scanner -
Dsonar.projectKey=10-Tier -Dsonar.ProjectName=10-Tier -Dsonar.java.binaries=.
'''
                }

            }


        }
        stage('adservice'){
            steps{
             script{
              withDockerRegistry(credentialsId: 'docker-cred', toolName:
'docker') {
                            dir('/var/lib/jenkins/workspace/10-
Tier/src/adservice') {
                            sh 'docker build -t samsorrahman/adservice:latest
.'
                            sh "docker push samsorrahman/adservice:latest"
                            sh "docker rmi samsorrahman/adservice:latest"
                        }
                    }
            }
            }
        }

        stage('cartservice'){
            steps{
             script{
              withDockerRegistry(credentialsId: 'docker-cred', toolName:
'docker') {
                        dir('/var/lib/jenkins/workspace/10-
Tier/src/cartservice/src/') {
                            sh 'docker build -t
samsorrahman/cartservice:latest .'
                            sh "docker push samsorrahman/cartservice:latest"
                            sh "docker rmi samsorrahman/cartservice:latest"
                        }
                    }
            }
            }
        }

        stage('checkoutservice'){
            steps{
             script{
              withDockerRegistry(credentialsId: 'docker-cred', toolName:
'docker') {
                        dir('/var/lib/jenkins/workspace/10-
Tier/src/checkoutservice/') {
                            sh 'docker build -t
samsorrahman/checkoutservice:latest .'
                            sh "docker push
samsorrahman/checkoutservice:latest"
                            sh "docker rmi
samsorrahman/checkoutservice:latest"
                        }
                    }
            }
            }
        }

        stage('currencyservice'){
            steps{
             script{
              withDockerRegistry(credentialsId: 'docker-cred', toolName:
'docker') {
                        dir('/var/lib/jenkins/workspace/10-
Tier/src/currencyservice/') {
```

```
                                sh 'docker build -t
    samsorrahman/currencyservice:latest .'
                                sh "docker push
    samsorrahman/currencyservice:latest"
                                sh "docker rmi
    samsorrahman/currencyservice:latest"
                            }
                        }
                }
                }
            }

            stage('emailservice'){
                steps{
                 script{
                  withDockerRegistry(credentialsId: 'docker-cred', toolName:
    'docker') {
                            dir('/var/lib/jenkins/workspace/10-
    Tier/src/emailservice/') {
                                sh 'docker build -t
    samsorrahman/emailservice:latest .'
                                sh "docker push samsorrahman/emailservice:latest"
                                sh "docker rmi samsorrahman/emailservice:latest"
                            }
                        }
                }
                }
            }

            stage('frontend'){
                steps{
                 script{
                  withDockerRegistry(credentialsId: 'docker-cred', toolName:
    'docker') {
                            dir('/var/lib/jenkins/workspace/10-
    Tier/src/frontend/') {
                                sh 'docker build -t samsorrahman/frontend:latest
    .'
                                sh "docker push samsorrahman/frontend:latest"
                                sh "docker rmi samsorrahman/frontend:latest"
                            }
                        }
                }
                }
            }

            stage('loadgenerator'){
                steps{
                 script{
                  withDockerRegistry(credentialsId: 'docker-cred', toolName:
    'docker') {
                            dir('/var/lib/jenkins/workspace/10-
    Tier/src/loadgenerator/') {
                                sh 'docker build -t
```

```
samsorrahman/loadgenerator:latest .'
                            sh "docker push
samsorrahman/loadgenerator:latest"
                            sh "docker rmi samsorrahman/loadgenerator:latest"
                        }
                    }
            }
            }
        }

        stage('paymentservice'){
            steps{
             script{
              withDockerRegistry(credentialsId: 'docker-cred', toolName:
'docker') {
                        dir('/var/lib/jenkins/workspace/10-
Tier/src/paymentservice/') {
                            sh 'docker build -t
samsorrahman/paymentservice:latest .'
                            sh "docker push
samsorrahman/paymentservice:latest"
                            sh "docker rmi
samsorrahman/paymentservice:latest"
                        }
                    }
            }
            }
        }

        stage('productcatalogservice'){
            steps{
             script{
              withDockerRegistry(credentialsId: 'docker-cred', toolName:
'docker') {
                        dir('/var/lib/jenkins/workspace/10-
Tier/src/productcatalogservice/') {
                            sh 'docker build -t
samsorrahman/productcatalogservice:latest .'
                            sh "docker push
samsorrahman/productcatalogservice:latest"
                            sh "docker rmi
samsorrahman/productcatalogservice:latest"
                        }
                    }
            }
            }
        }

        stage('recommendationservice'){
            steps{
             script{
              withDockerRegistry(credentialsId: 'docker-cred', toolName:
'docker') {
                        dir('/var/lib/jenkins/workspace/10-
Tier/src/recommendationservice/') {
                            sh 'docker build -t
samsorrahman/recommendationservice:latest .'
                            sh "docker push
samsorrahman/recommendationservice:latest"
                            sh "docker rmi
samsorrahman/recommendationservice:latest"
                        }
                    }
            }
            }
        }

        stage('shippingservice'){
            steps{
             script{
              withDockerRegistry(credentialsId: 'docker-cred', toolName:
'docker') {
                        dir('/var/lib/jenkins/workspace/10-
Tier/src/shippingservice/') {
                            sh 'docker build -t
samsorrahman/shippingservice:latest .'
                            sh "docker push
samsorrahman/shippingservice:latest"
                            sh "docker rmi
samsorrahman/shippingservice:latest"
                        }
                    }
            }
            }
        }

        stage('K8-Deploy'){
            steps{
                withKubeConfig(caCertificate: '', clusterName: 'my-eks2',
contextName: '', credentialsId: 'k8-token', namespace: 'webapps',
restrictKubeConfigAccess: false, serverUrl:
'https://EBCE08CF45C3AA5A574E126370E5D4FC.gr7.ap-south-1.eks.amazonaws.com')
{
                    sh 'kubectl apply -f deployment-service.yml'
                    sh 'kubectl get pods'
                    sh 'kubectl get svc'
                }
            }
        }
}
}
```

Change samsorrahman with your dockerhub account username and also chang the K8-Deploy key with your own key

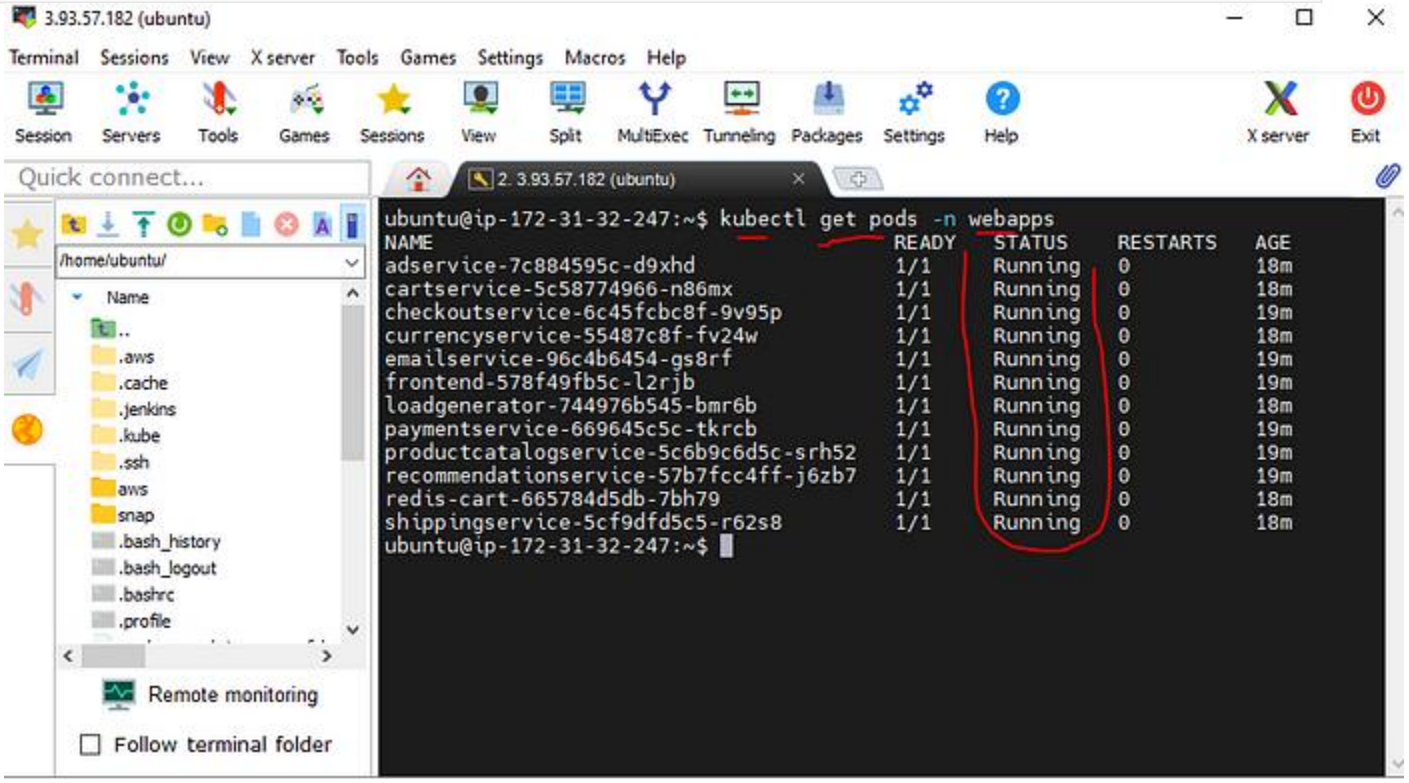
How to get the key run the following command on terminal

```
kubectl -n examplens describe secret mysecretname
```

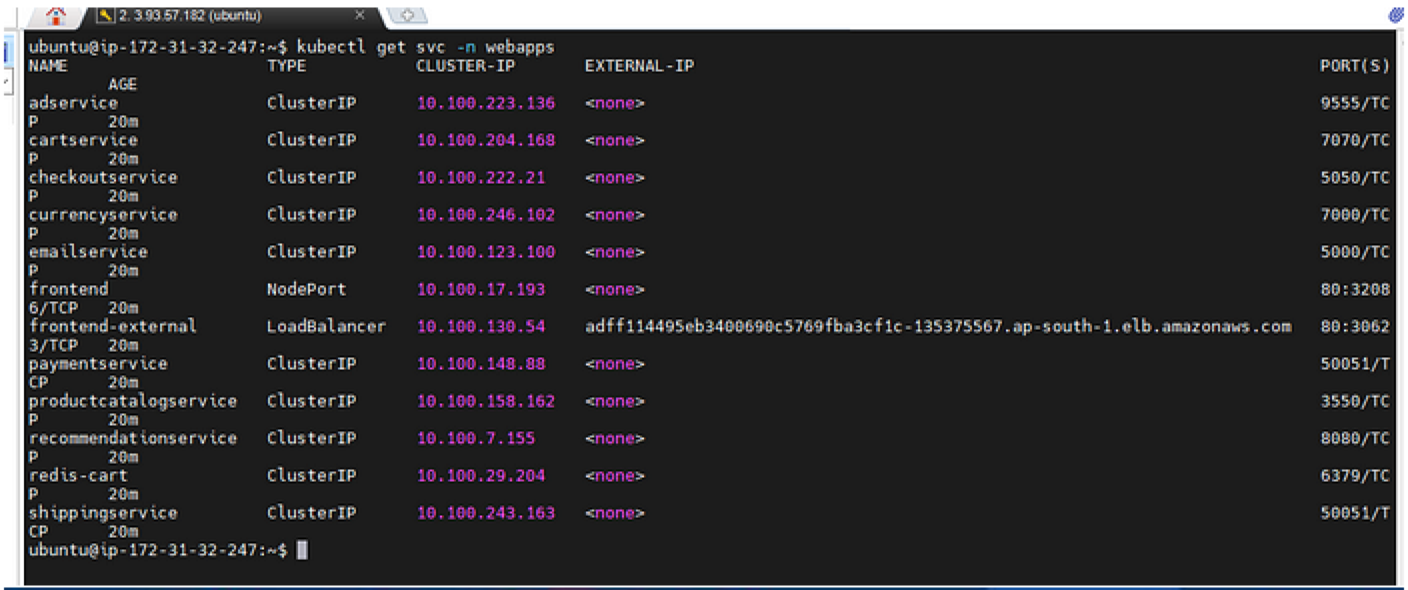
Now run the pipeline

after running
run the following command on terminal

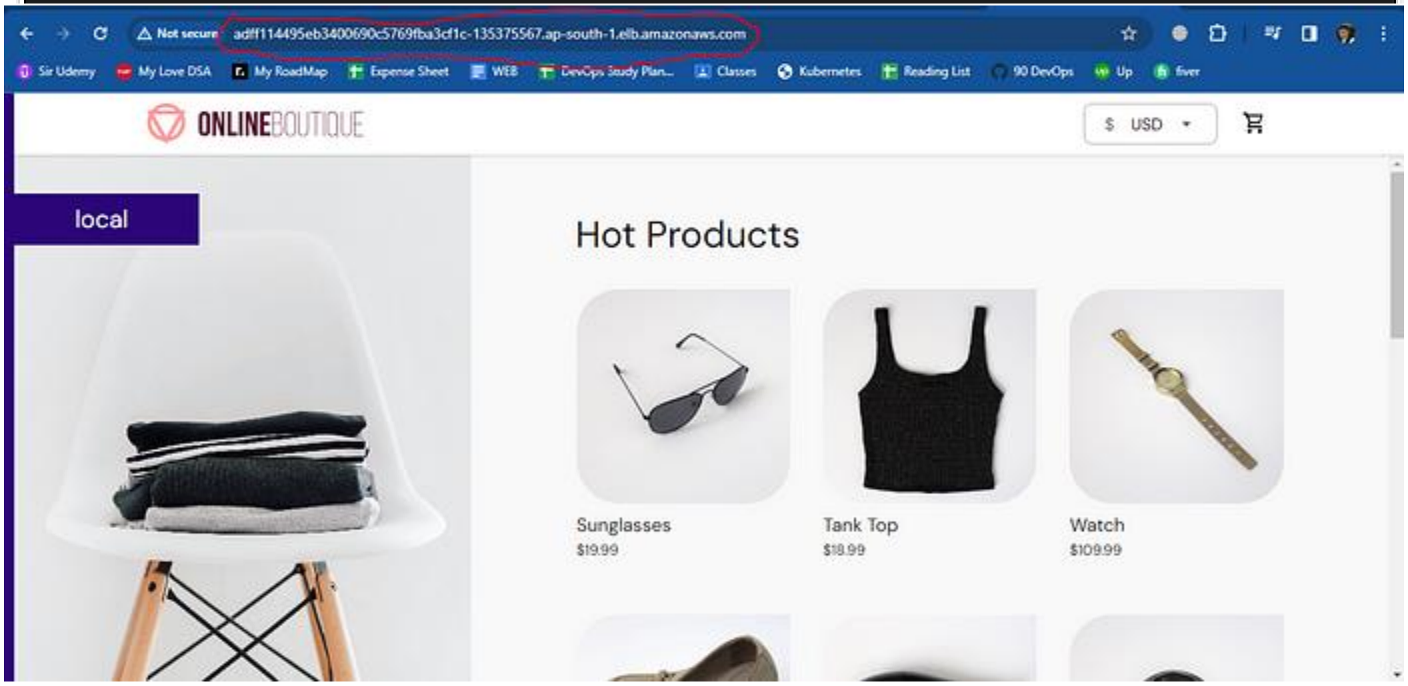
```
kubectl get pods -n webapps
```

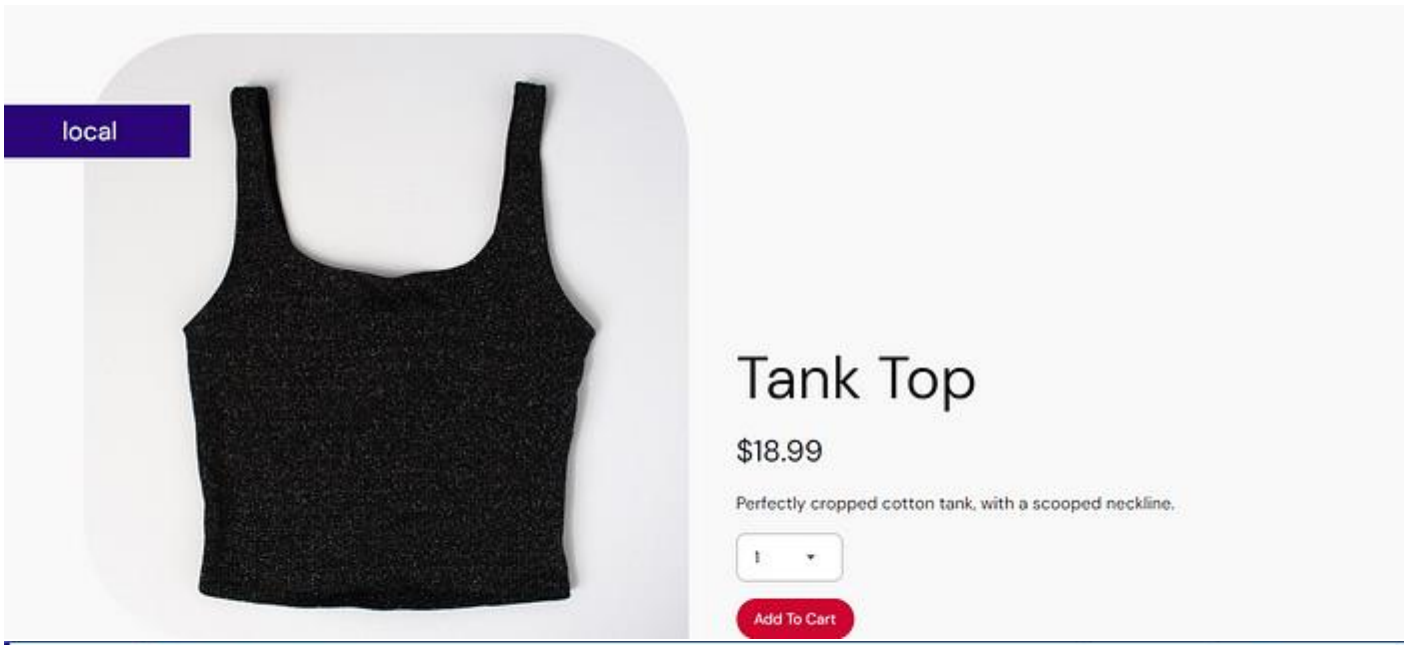


```
ubuntu@ip-172-31-32-247:~$ kubectl get pods -n webapps
NAME                                      READY   STATUS    RESTARTS   AGE
adservice-7c884595c-d9xhd                 1/1     Running   0          18m
cartservice-5c58774966-n86mx              1/1     Running   0          18m
checkoutservice-6c45fcbc8f-9v95p          1/1     Running   0          19m
currencyservice-55487c8f-fv24w            1/1     Running   0          18m
emailservice-96c4b6454-gs8rf              1/1     Running   0          19m
frontend-578f49fb5c-l2rjb                 1/1     Running   0          19m
loadgenerator-744976b545-bmr6b            1/1     Running   0          18m
paymentservice-669645c5c-tkrcb            1/1     Running   0          19m
productcatalogservice-5c6b9c6d5c-srh52    1/1     Running   0          19m
recommendationservice-57b7fcc4ff-j6zb7    1/1     Running   0          19m
redis-cart-665784d5db-7bh79               1/1     Running   0          18m
shippingservice-5cf9dfd5c5-r62s8          1/1     Running   0          18m
ubuntu@ip-172-31-32-247:~$
```



```
ubuntu@ip-172-31-32-247:~$ kubectl get svc -n webapps
NAME                    TYPE           CLUSTER-IP       EXTERNAL-IP                                                                 PORT(S)
        AGE
adservice               ClusterIP      10.100.223.136   <none>                                                                      9555/TC
P       20m
cartservice             ClusterIP      10.100.204.168   <none>                                                                      7070/TC
P       20m
checkoutservice         ClusterIP      10.100.222.21    <none>                                                                      5050/TC
P       20m
currencyservice         ClusterIP      10.100.246.102   <none>                                                                      7000/TC
P       20m
emailservice            ClusterIP      10.100.123.100   <none>                                                                      5000/TC
P       20m
frontend                NodePort       10.100.17.193    <none>                                                                      80:3208
6/TCP   20m
frontend-external       LoadBalancer   10.100.130.54    adff114495eb3400690c5769fba3cf1c-135375567.ap-south-1.elb.amazonaws.com    80:3062
3/TCP   20m
paymentservice          ClusterIP      10.100.148.88    <none>                                                                      50051/T
CP      20m
productcatalogservice   ClusterIP      10.100.158.162   <none>                                                                      3550/TC
P       20m
recommendationservice   ClusterIP      10.100.7.155     <none>                                                                      8080/TC
P       20m
redis-cart              ClusterIP      10.100.29.204    <none>                                                                      6379/TC
P       20m
shippingservice         ClusterIP      10.100.243.163   <none>                                                                      50051/T
CP      20m
ubuntu@ip-172-31-32-247:~$
```

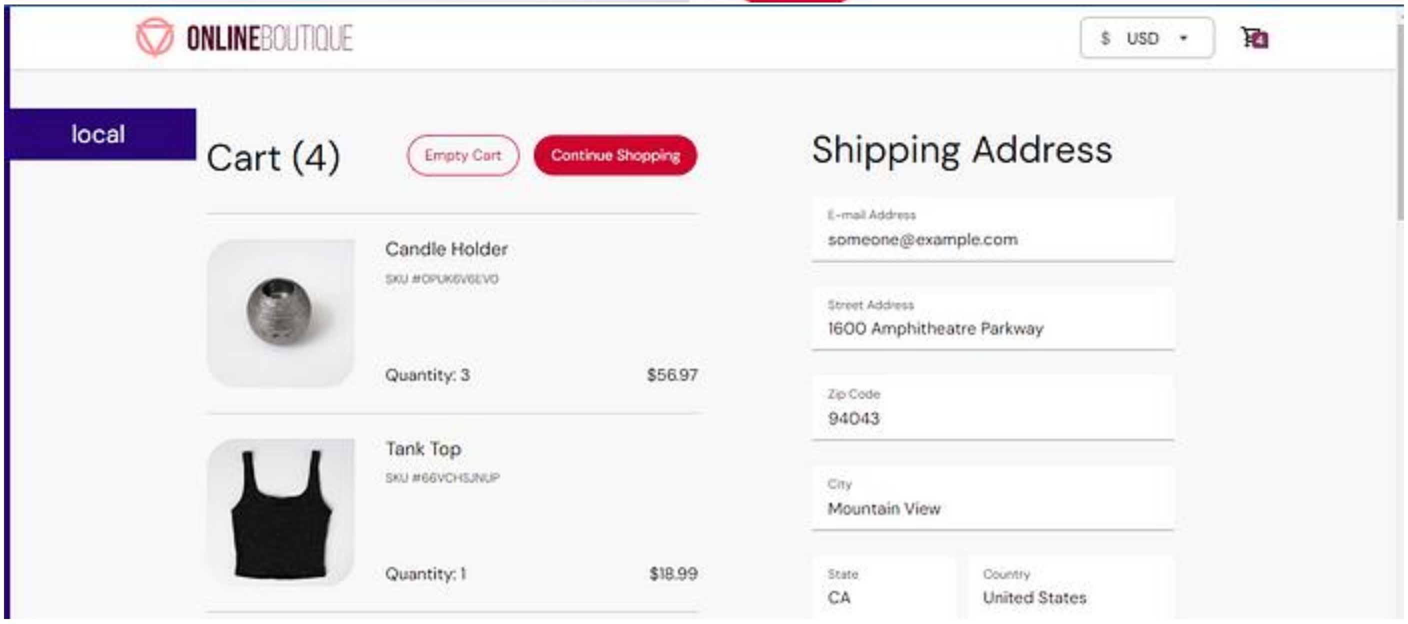




Tank Top

$18.99

Perfectly cropped cotton tank, with a scooped neckline.



Cart (4)

Candle Holder
SKU #OPUK6V6EV0
Quantity: 3 $56.97

Tank Top
SKU #66VCHSJNUP
Quantity: 1 $18.99

Shipping Address

E-mail Address: someone@example.com
Street Address: 1600 Amphitheatre Parkway
Zip Code: 94043
City: Mountain View
State: CA
Country: United States

Thank for Reading :)

My LinkedIn -> https://www.linkedin.com/in/samsor-rahman18/
My GitHub -> https://github.com/samsorrahman