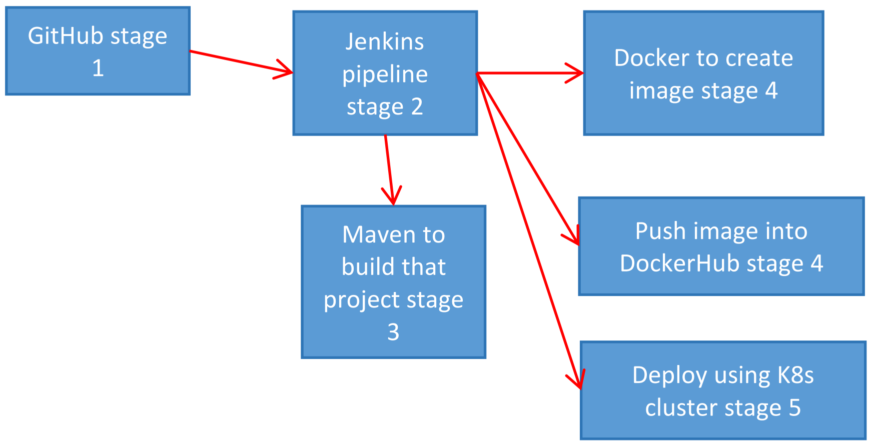
Jenkins Pipeline 2

CICD --> Jenkins Pipeline

Final project --> Terraform to automate React + SpringBoot app

1. GitHub
2. Maven in Jenkins pipeline stage
3. Docker image we push into Docker registry (stage)
4. Deploy application using Kubernetes cluster
5. Setup pipeline using Jenkins


GitHub stage 1
Jenkins pipeline stage 2
Docker to create image stage 4
Maven to build that project stage 3
Push image into DockerHub stage 4
Deploy using K8s cluster stage 5

GitHub repo --> Git clone (Stage 1) --> Build Jar/War (Stage 2) --> Docker build the image (Stage 3) --> Docker hub and push image (Stage 4) --> K8s to deploy application (Stage 5) . We will have Manifest.yml file then we will access application from browser

Final CICD pipelines
Jenkins pipeline 1 --> Git (Project clone) + Maven (Project build) + Docker (Docker image creation) + Docker Hub (Pushing image into Docker registry) + Kubernetes (Deploy application)
Until Pushing image into Docker registry it is CI and Kubernetes part is CD

==
1 ==> Create EKS Host VM is Created in AWS
===

a) Launch a Linux machine (Ubuntu VM) using AWS EC2 (t2.micro)
b) Connect to this machine and install kubectl

install kubectl

```
curl -LO "https://dl.k8s.io/release/$(curl -L -s
https://dl.k8s.io/release/stable.txt)/bin/linux/amd64/kubectl"
# Make executable and move to /usr/local/bin
chmod +x kubectl
sudo mv kubectl /usr/local/bin/
# Verify installation
kubectl version --client --output=yaml
```

install AWS CLI

```
# Install unzip (adjust for your package manager)
# For Debian/Ubuntu
```

```
$ sudo apt update && sudo apt install -y unzip

# For RHEL/CentOS
# sudo yum install -y unzip
# Download and install AWS CLI v2

$ curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o "awscliv2.zip"
 unzip awscliv2.zip
 sudo ./aws/install

# Clean up

$ rm -rf awscliv2.zip aws

# Verify installation
$ aws --version
```

Install eksctl

```
# Download and extract the latest eksctl

curl --silent --location
"https://github.com/weaveworks/eksctl/releases/latest/download/eksctl_$(uname -s)_amd64.tar.gz"
| tar xz -C /tmp

# Verify installation
eksctl version
```

===
2 ==> Create IAM Role and attach to EKS Management HOST
===

a)Create a new Role using IAM Service (Select Use case EC2)

b)Add below permission

AdministratorAccess

AmazonEC2FullAccess

AmazonVPCFullAccess

AWSCloudFormationFullAccess

IAMFullAccess

c)Enter Role Name(telusko_eks_role)

d) Attach created role to EKS management host vm --> Actions-> Security-> Modify IAM role --> add created IAM role

===
3 ==> Create EKS Cluster using eksctl
===

```
eksctl create cluster --name telusko-cluster --region ap-south-1 --node-type t2.medium  --zones ap-south-1a,ap-south-1b
```

kubeconfig as "/home/ubuntu/.kube/config"

EKS cluster "telusko-cluster" in "ap-south-1" region is ready

```
$ cat /home/ubuntu/.kube/config

$ kubectl get nodes

NAME                                            STATUS   ROLES    AGE   VERSION
ip-192-168-27-229.ap-south-1.compute.internal   Ready    <none>   12m   v1.32.3-eks-473151a
ip-192-168-39-106.ap-south-1.compute.internal   Ready    <none>   12m   v1.32.3-eks-473151a
```

Also check on AWS Console that cluster and also two new instance worker nodes would be created

===
4 ==> Jenkins Server Setup in Linux VM
===

1. Create Linux VM on AWS Cloud - Ubuntu (preferred to use min t2.medium as instance type)
   Get connected to Linux VM using ssh gitbash or terminal or any medium

2. install Java

   1.sudo apt update -> update the package manager
   2.sudo apt install openjdk-21-jdk -> install java
   java -version -> To check java is installed or not

```
sudo apt update
sudo apt install openjdk-17-jdk
```

3. Install Jenkins

```
# Create keyring directory if it doesn't exist
sudo mkdir -p /etc/apt/keyrings

# Download and add the Jenkins GPG key
sudo wget -O /etc/apt/keyrings/jenkins-keyring.asc https://pkg.jenkins.io/debian-stable/jenkins.io-2023.key

# Add Jenkins repo to your sources list
echo "deb [signed-by=/etc/apt/keyrings/jenkins-keyring.asc] https://pkg.jenkins.io/debian-stable binary/" \
| sudo tee /etc/apt/sources.list.d/jenkins.list > /dev/null

sudo apt update
sudo apt install -y jenkins
```

4. Start and verify Jenkins

```
sudo systemctl enable jenkins
sudo systemctl start jenkins
```

Verify Jenkins
```
sudo systemctl status Jenkins
```

5. Open Jenkins server in browser (also make sure edit inbond rules and add 8080 in security group)

http://public-ip:8080/

6: Copy Jenkins admin password

/var/lib/jenkins/secrets/initialAdminPassword

```
$ sudo cat /var/lib/jenkins/secrets/initialAdminPassword
```

==
===> Attach created role to Jenkins Server host vm --> Actions-> Security-> Modify IAM role --> add created iam role
===

===
5 ==> Configure Maven as Global Tool in Jenkins
===
Manage Jenkins --> Tools --> Maven Installation --> Add maven

==
6 ==> Docker Setup in Jenkins
==
--> Execute the commands

```
sudo apt update
sudo apt install -y ca-certificates curl gnupg lsb-release

echo \
"deb [arch=$(dpkg --print-architecture) \
signed-by=/etc/apt/keyrings/docker.gpg] \
https://download.docker.com/linux/ubuntu \
$(lsb_release -cs) stable" | \
sudo tee /etc/apt/sources.list.d/docker.list > /dev/null

echo \
"deb [arch=$(dpkg --print-architecture) \
signed-by=/etc/apt/keyrings/docker.gpg] \
https://download.docker.com/linux/ubuntu \
$(lsb_release -cs) stable" | \
sudo tee /etc/apt/sources.list.d/docker.list > /dev/null

sudo apt update
sudo apt install -y docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin

sudo systemctl enable docker
sudo systemctl start docker
```

Add jenkins into docker group

```
$ sudo usermod -aG docker jenkins

$ sudo systemctl restart jenkins

$ sudo docker -v ---> Verify Docker installation

$ sudo docker version
```

==
7 ==> Install AWS CLI and Kubectl in Jenkins Server
==
install aws cli

```
sudo apt install unzip
curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o "awscliv2.zip"
unzip awscliv2.zip
sudo ./aws/install
aws --version
```

----
install kubectl

```
curl -LO "https://dl.k8s.io/release/$(curl -L -s
https://dl.k8s.io/release/stable.txt)/bin/linux/amd64/kubectl"
# Make executable and move to /usr/local/bin
chmod +x kubectl
sudo mv kubectl /usr/local/bin/
# Verify installation
kubectl version --client
```

==
8 ==> Update EKS Cluster Config File in Jenkins Server
==
Execute the below command in EKS Management Machine and Copy config file data

```
$ cat .kube/config
```

Connect to Jenkins server execute the following command to add config file into Jenkins server

```
$ cd /var/lib/Jenkins

$ sudo mkdir .kube

$ sudo vi .kube/config
```
 (paste config file data copied from eks host machine)

Check eks nodes

```
$ kubectl get nodes

$ cd ~
$ ls -la
$ sudo vi .kube/config
```
(paste config file data copied from eks host machine)

```
$ kubectl get nodes
```

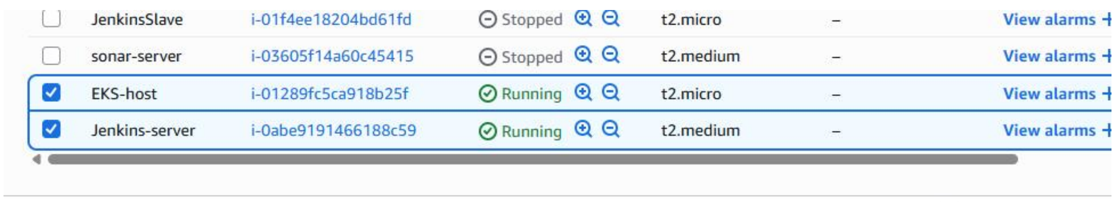
===
9 ==> Create Jenkins CICD Pipeline with all Stages
===
stage - 1 ==> Clone Git Repo
stage - 2 ==> Maven Build
stage - 3 ==> Create Docker Image
stage - 4 ==> Push Docker Image to Repository
stage - 5 ==> Deploy app in K8s eks Cluster

=====================================
10 ==> Install Blue Ocean Plugin
=====================================
Manage Jenkins --> plugins --> Available plugins --> Search Blue ocean --> install

You will see that in Jenkins dashboard and use the same to work with your pipeline

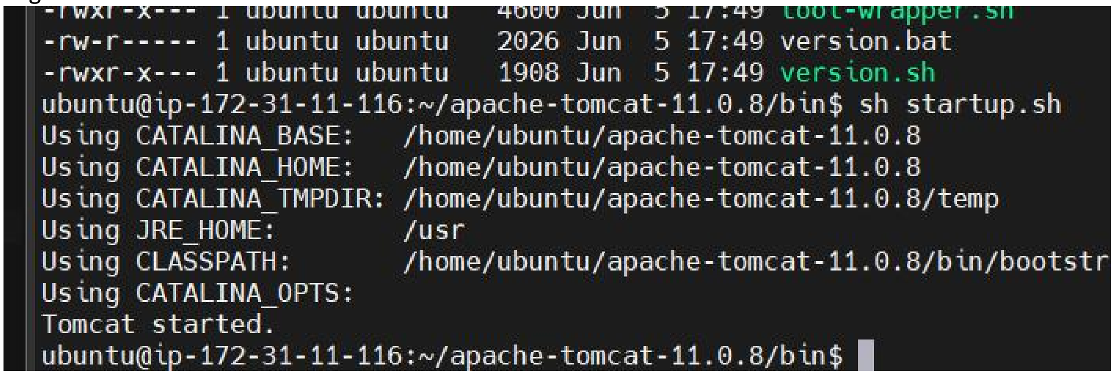
==

Jenkins Pipeline Step by Step



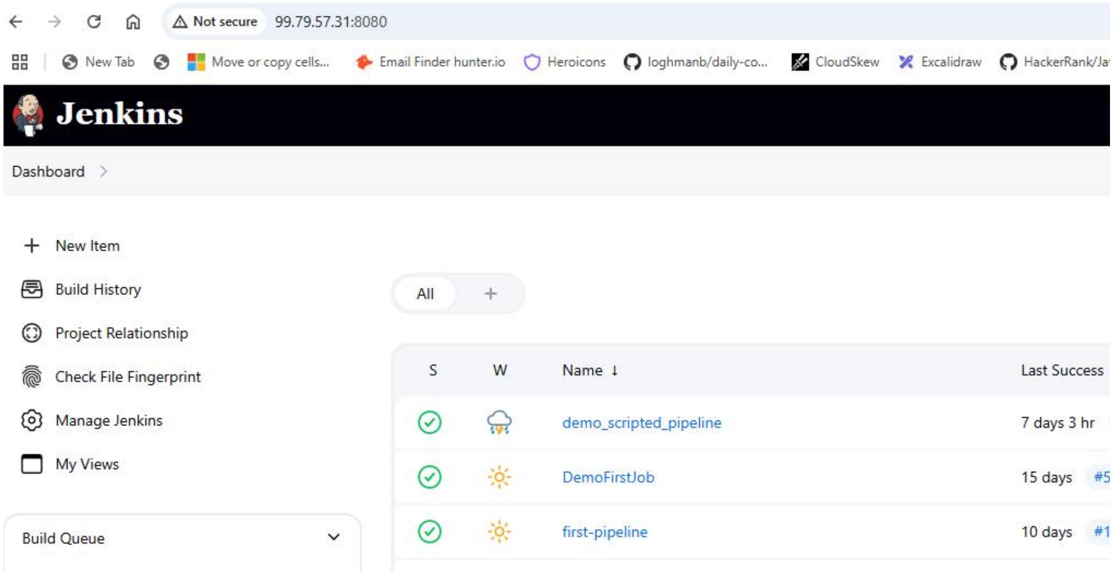
	Name	Instance ID	State	Type		
☐	JenkinsSlave	i-01f4ee18204bd61fd	Stopped	t2.micro	–	View alarms +
☐	sonar-server	i-03605f14a60c45415	Stopped	t2.medium	–	View alarms +
✓	EKS-host	i-01289fc5ca918b25f	Running	t2.micro	–	View alarms +
✓	Jenkins-server	i-0abe9191466188c59	Running	t2.medium	–	View alarms +

Log into Jenkins-server



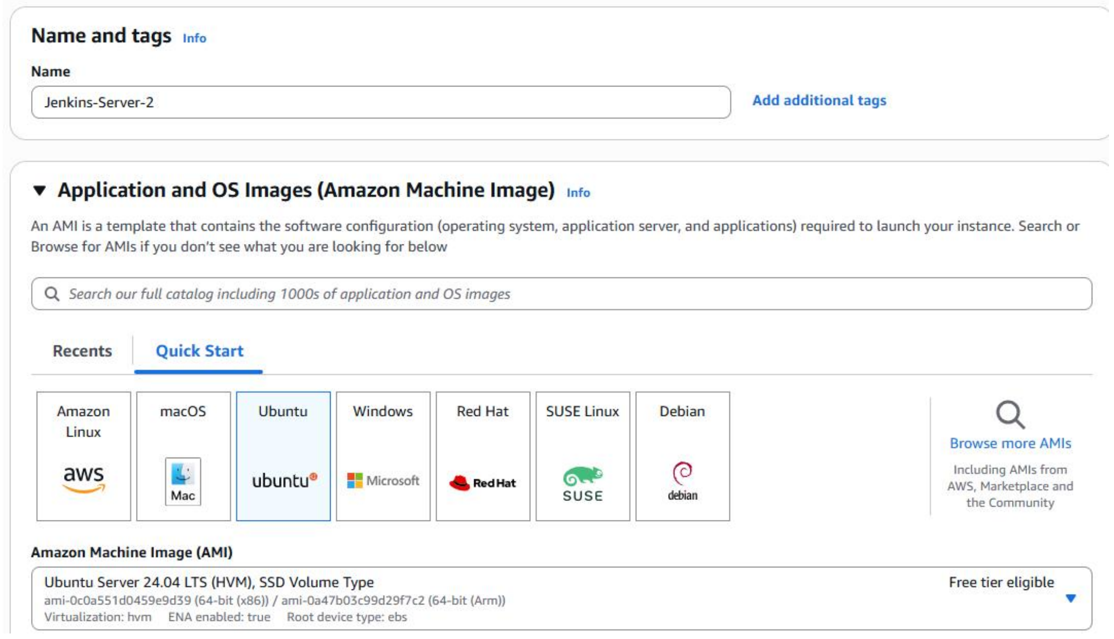
```
-rwxr-x--- 1 ubuntu ubuntu   4600 Jun  5 17:49 tool-wrapper.sh
-rw-r----- 1 ubuntu ubuntu   2026 Jun  5 17:49 version.bat
-rwxr-x--- 1 ubuntu ubuntu   1908 Jun  5 17:49 version.sh
ubuntu@ip-172-31-11-116:~/apache-tomcat-11.0.8/bin$ sh startup.sh
Using CATALINA_BASE:   /home/ubuntu/apache-tomcat-11.0.8
Using CATALINA_HOME:   /home/ubuntu/apache-tomcat-11.0.8
Using CATALINA_TMPDIR: /home/ubuntu/apache-tomcat-11.0.8/temp
Using JRE_HOME:        /usr
Using CLASSPATH:       /home/ubuntu/apache-tomcat-11.0.8/bin/bootstr
Using CATALINA_OPTS:
Tomcat started.
ubuntu@ip-172-31-11-116:~/apache-tomcat-11.0.8/bin$
```

Log into Jenkins



Not secure 99.79.57.31:8080

S	W	Name ↓	Last Success
✓		demo_scripted_pipeline	7 days 3 hr
✓		DemoFirstJob	15 days #5
✓		first-pipeline	10 days #1


Name and tags Info

Name

Jenkins-Server-2

Add additional tags

▼ Application and OS Images (Amazon Machine Image) Info

An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. Search or Browse for AMIs if you don't see what you are looking for below

Search our full catalog including 1000s of application and OS images

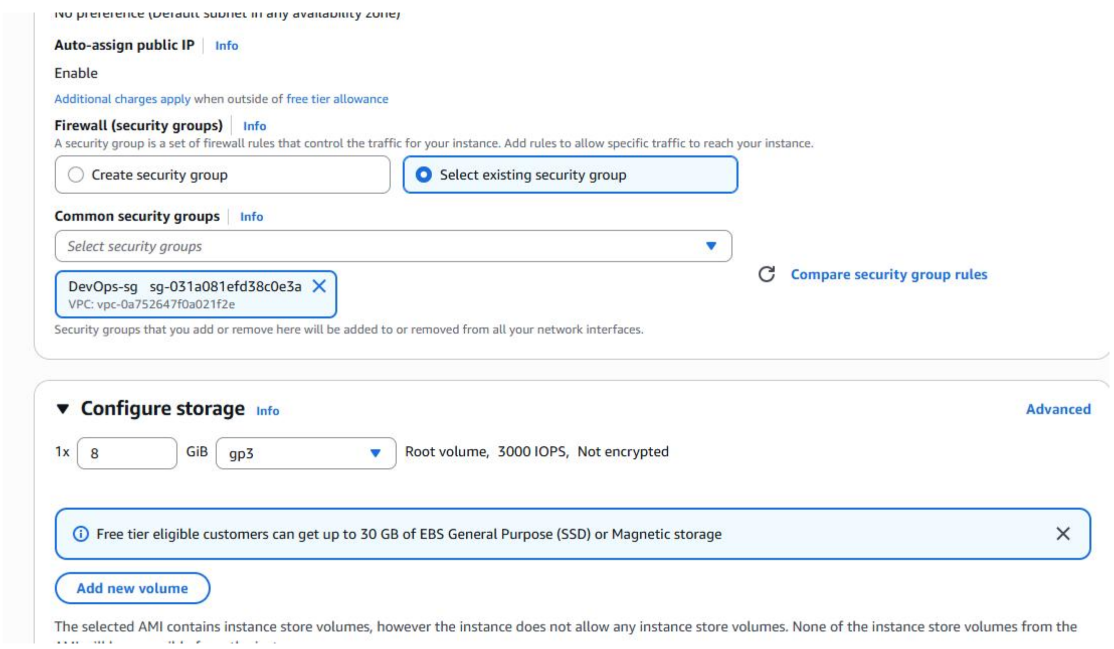
Recents | Quick Start

Amazon Linux | macOS | Ubuntu | Windows | Red Hat | SUSE Linux | Debian

Browse more AMIs
Including AMIs from AWS, Marketplace and the Community

Amazon Machine Image (AMI)

Ubuntu Server 24.04 LTS (HVM), SSD Volume Type — Free tier eligible
ami-0c0a551d0459e9d39 (64-bit (x86)) / ami-0a47b03c99d29f7c2 (64-bit (Arm))
Virtualization: hvm ENA enabled: true Root device type: ebs


No preference (Default subnet in any availability zone)

Auto-assign public IP Info

Enable

Additional charges apply when outside of free tier allowance

Firewall (security groups) Info

A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.

Create security group [✓ Select existing security group]

Common security groups Info

Select security groups

DevOps-sg sg-031a081efd38c0e3a
VPC: vpc-0a752647f0a021f2e

Compare security group rules

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

▼ Configure storage Info — Advanced

1x 8 GiB gp3 Root volume, 3000 IOPS, Not encrypted

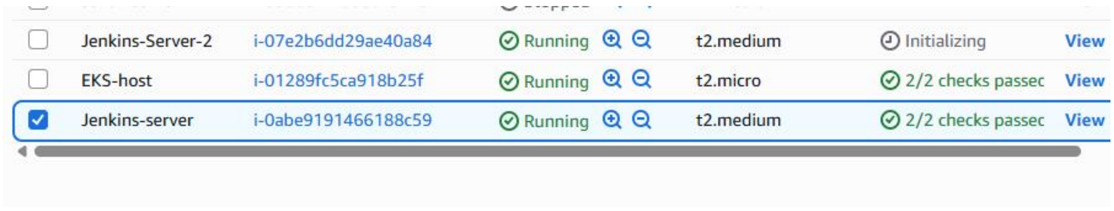
Free tier eligible customers can get up to 30 GB of EBS General Purpose (SSD) or Magnetic storage

Add new volume

The selected AMI contains instance store volumes, however the instance does not allow any instance store volumes. None of the instance store volumes from the

Launch Instance

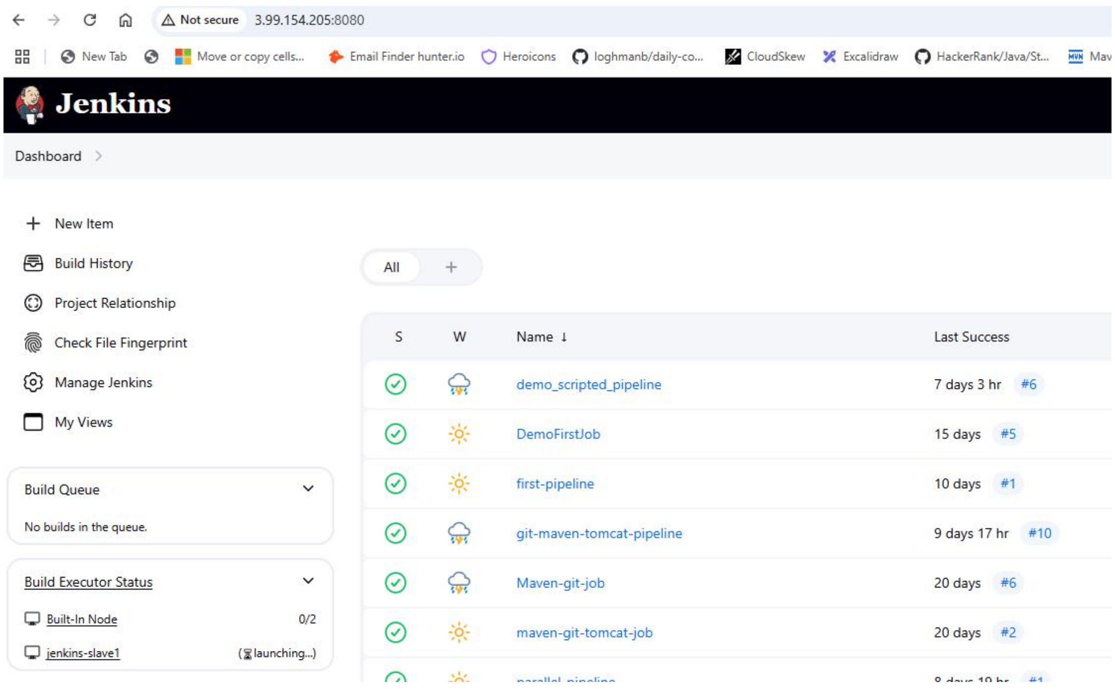
Lets use Jenkins-Server-2



☐	Jenkins-Server-2	i-07e2b6dd29ae40a84	Running	t2.medium	Initializing	View
☐	EKS-host	i-01289fc5ca918b25f	Running	t2.micro	2/2 checks passec	View
☑	Jenkins-server	i-0abe9191466188c59	Running	t2.medium	2/2 checks passec	View

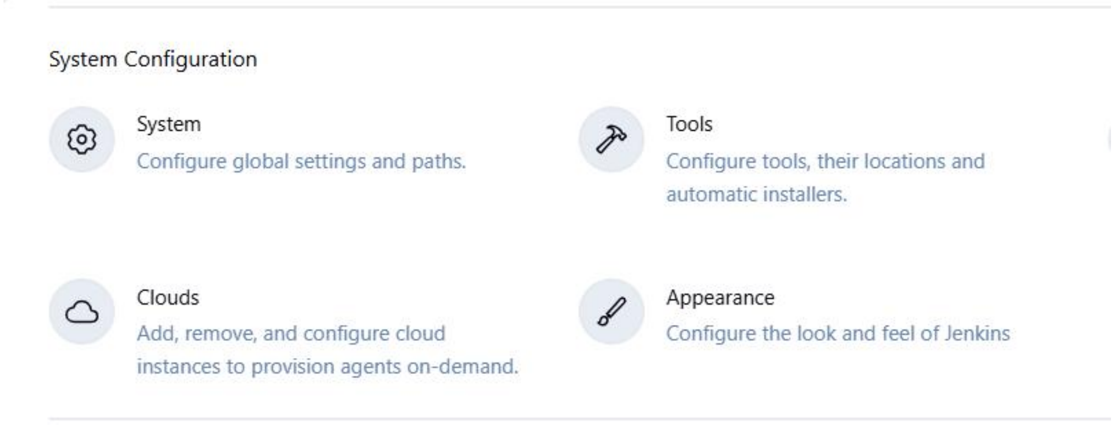
I am trying with Jenkins-Server original Jenkins server first

Logged back in

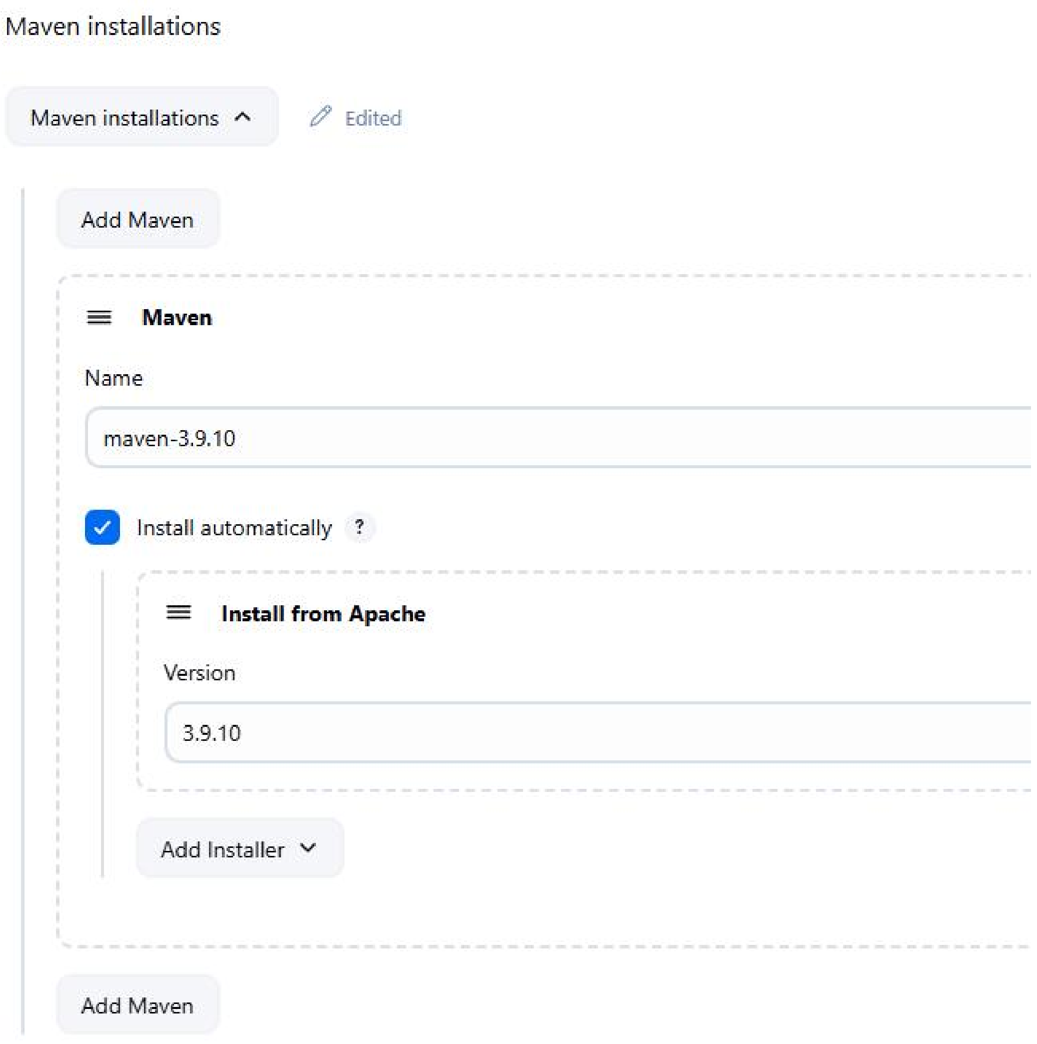


S	W	Name ↓	Last Success
✓		demo_scripted_pipeline	7 days 3 hr #6
✓		DemoFirstJob	15 days #5
✓		first-pipeline	10 days #1
✓		git-maven-tomcat-pipeline	9 days 17 hr #10
✓		Maven-git-job	20 days #6
✓		maven-git-tomcat-job	20 days #2

Go to Tools


System Configuration
System
Configure global settings and paths.
Tools
Configure tools, their locations and automatic installers.
Clouds
Add, remove, and configure cloud instances to provision agents on-demand.
Appearance
Configure the look and feel of Jenkins

Maven is already there


Maven installations

Maven installations
Edited

Add Maven

Maven

Name

maven-3.9.10

✓ Install automatically

Install from Apache

Version

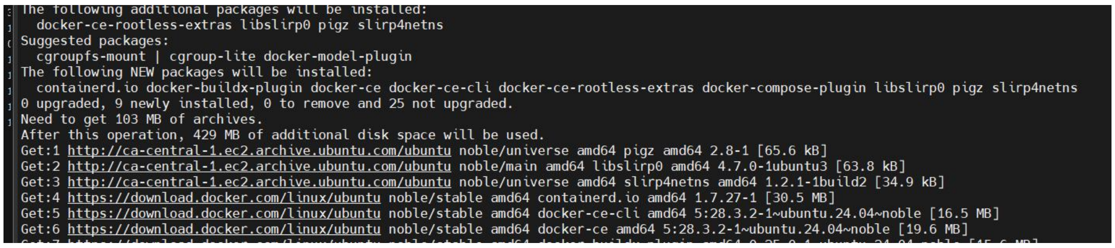
3.9.10

Add Installer

Add Maven

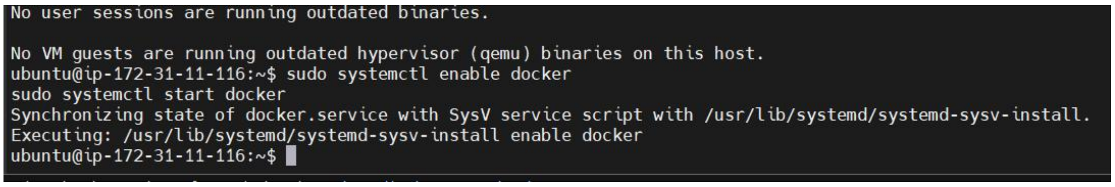
Docker Setup in Jenkins

```
# 1. Update existing packages
sudo apt update
sudo apt upgrade -y

# 2. Install dependencies
sudo apt install -y \
  ca-certificates \
  curl \
  gnupg \
  lsb-release

# 3. Add Docker's official GPG key
sudo mkdir -p /etc/apt/keyrings
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | \
 sudo gpg --dearmor -o /etc/apt/keyrings/docker.gpg

# 4. Set up the Docker repo
echo \
 "deb [arch=$(dpkg --print-architecture) \
 signed-by=/etc/apt/keyrings/docker.gpg] \
 https://download.docker.com/linux/ubuntu \
 $(lsb_release -cs) stable" | \
 sudo tee /etc/apt/sources.list.d/docker.list > /dev/null

# 5. Update package index with Docker repo
sudo apt update

# 6. Install Docker packages
sudo apt install -y \
 docker-ce \
 docker-ce-cli \
 containerd.io \
 docker-buildx-plugin \
 docker-compose-plugin
```



```
The following additional packages will be installed:
  docker-ce-rootless-extras libslirp0 pigz slirp4netns
Suggested packages:
  cgroupfs-mount | cgroup-lite docker-model-plugin
The following NEW packages will be installed:
  containerd.io docker-buildx-plugin docker-ce docker-ce-cli docker-ce-rootless-extras docker-compose-plugin libslirp0 pigz slirp4netns
0 upgraded, 9 newly installed, 0 to remove and 25 not upgraded.
Need to get 103 MB of archives.
After this operation, 429 MB of additional disk space will be used.
Get:1 http://ca-central-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 pigz amd64 2.8-1 [65.6 kB]
Get:2 http://ca-central-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 libslirp0 amd64 4.7.0-1ubuntu3 [63.8 kB]
Get:3 http://ca-central-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 slirp4netns amd64 1.2.1-1build2 [34.9 kB]
Get:4 https://download.docker.com/linux/ubuntu noble/stable amd64 containerd.io amd64 1.7.27-1 [30.5 MB]
Get:5 https://download.docker.com/linux/ubuntu noble/stable amd64 docker-ce-cli amd64 5:28.3.2-1~ubuntu.24.04~noble [16.5 MB]
Get:6 https://download.docker.com/linux/ubuntu noble/stable amd64 docker-ce amd64 5:28.3.2-1~ubuntu.24.04~noble [19.6 MB]
```

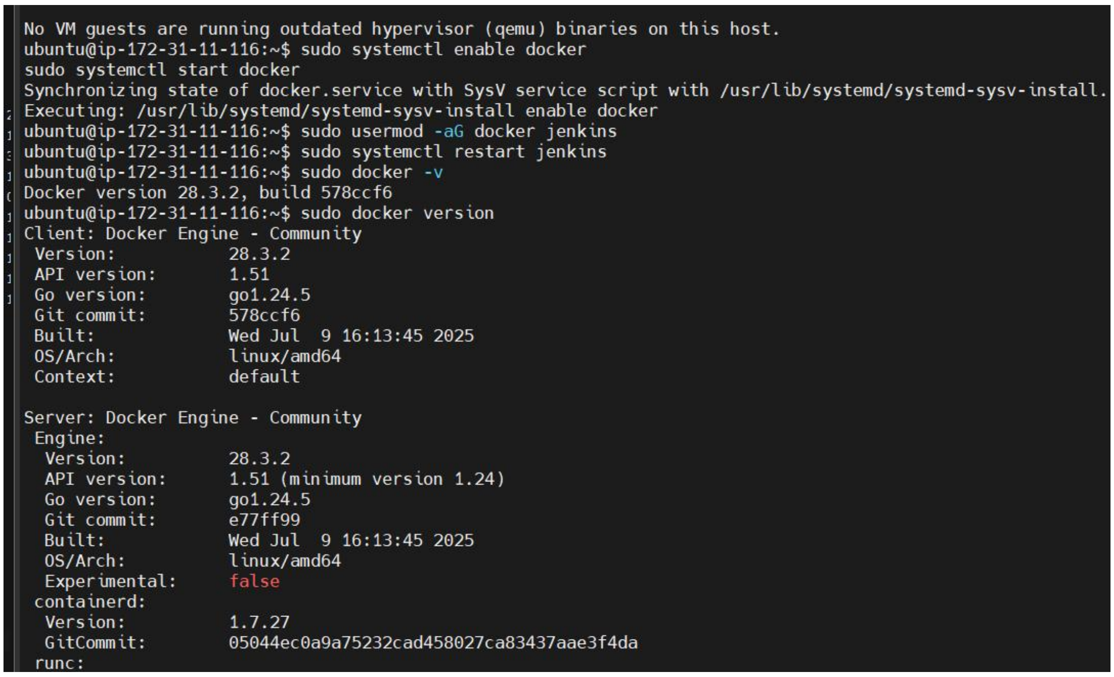
```
sudo systemctl enable docker
sudo systemctl start docker
```



```
No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
ubuntu@ip-172-31-11-116:~$ sudo systemctl enable docker
sudo systemctl start docker
Synchronizing state of docker.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable docker
ubuntu@ip-172-31-11-116:~$
```

Add jenkins into docker group

```
$ sudo usermod -aG docker jenkins

$ sudo systemctl restart jenkins

$ sudo docker -v ---> Verify Docker installation

$ sudo docker version
```



```
No VM guests are running outdated hypervisor (qemu) binaries on this host.
ubuntu@ip-172-31-11-116:~$ sudo systemctl enable docker
sudo systemctl start docker
Synchronizing state of docker.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable docker
ubuntu@ip-172-31-11-116:~$ sudo usermod -aG docker jenkins
ubuntu@ip-172-31-11-116:~$ sudo systemctl restart jenkins
ubuntu@ip-172-31-11-116:~$ sudo docker -v
Docker version 28.3.2, build 578ccf6
ubuntu@ip-172-31-11-116:~$ sudo docker version
Client: Docker Engine - Community
 Version:           28.3.2
 API version:       1.51
 Go version:        go1.24.5
 Git commit:        578ccf6
 Built:             Wed Jul  9 16:13:45 2025
 OS/Arch:           linux/amd64
 Context:           default

Server: Docker Engine - Community
 Engine:
  Version:          28.3.2
  API version:      1.51 (minimum version 1.24)
  Go version:       go1.24.5
  Git commit:       e77ff99
  Built:            Wed Jul  9 16:13:45 2025
  OS/Arch:          linux/amd64
  Experimental:     false
 containerd:
  Version:          1.7.27
  GitCommit:        05044ec0a9a75232cad458027ca83437aae3f4da
 runc:
```

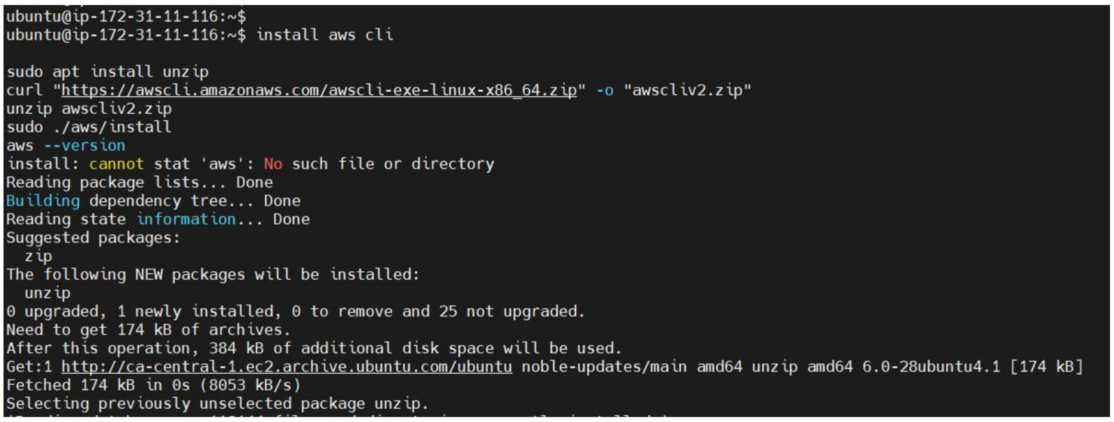
Install AWS CLI and Kubectl in Jenkins Server

install aws cli

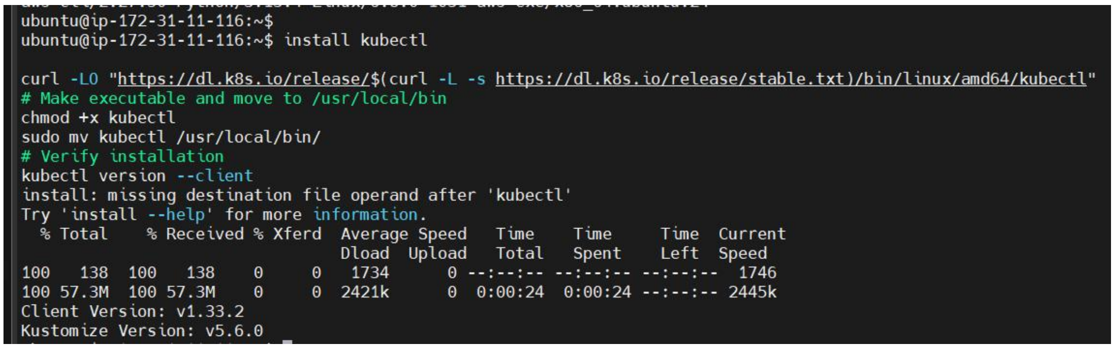
sudo apt install unzip
curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o "awscliv2.zip"
unzip awscliv2.zip
sudo ./aws/install
aws --version

----
install kubectl

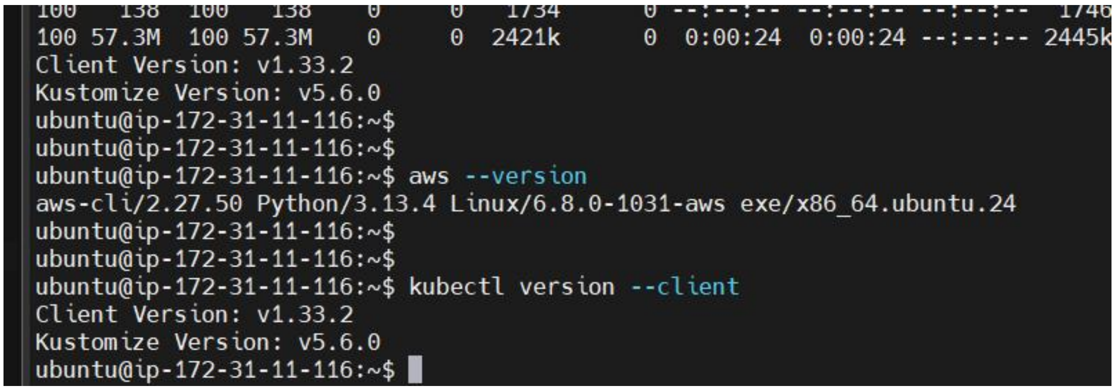
curl -LO "https://dl.k8s.io/release/$(curl -L -s
https://dl.k8s.io/release/stable.txt)/bin/linux/amd64/kubectl"
Make executable and move to /usr/local/bin
chmod +x kubectl
sudo mv kubectl /usr/local/bin/
# Verify installation
kubectl version --client



```
ubuntu@ip-172-31-11-116:~$
ubuntu@ip-172-31-11-116:~$ install aws cli

sudo apt install unzip
curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o "awscliv2.zip"
unzip awscliv2.zip
sudo ./aws/install
aws --version
install: cannot stat 'aws': No such file or directory
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Suggested packages:
  zip
The following NEW packages will be installed:
  unzip
0 upgraded, 1 newly installed, 0 to remove and 25 not upgraded.
Need to get 174 kB of archives.
After this operation, 384 kB of additional disk space will be used.
Get:1 http://ca-central-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 unzip amd64 6.0-28ubuntu4.1 [174 kB]
Fetched 174 kB in 0s (8053 kB/s)
Selecting previously unselected package unzip.
```

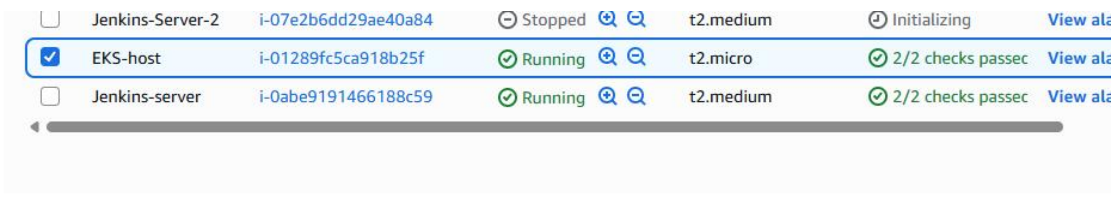


```
ubuntu@ip-172-31-11-116:~$
ubuntu@ip-172-31-11-116:~$ install kubectl

curl -LO "https://dl.k8s.io/release/$(curl -L -s https://dl.k8s.io/release/stable.txt)/bin/linux/amd64/kubectl"
# Make executable and move to /usr/local/bin
chmod +x kubectl
sudo mv kubectl /usr/local/bin/
# Verify installation
kubectl version --client
install: missing destination file operand after 'kubectl'
Try 'install --help' for more information.
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100   138  100   138    0     0   1734      0 --:--:-- --:--:-- --:--:--  1746
100 57.3M  100 57.3M    0     0  2421k      0  0:00:24  0:00:24 --:--:-- 2445k
Client Version: v1.33.2
Kustomize Version: v5.6.0
```

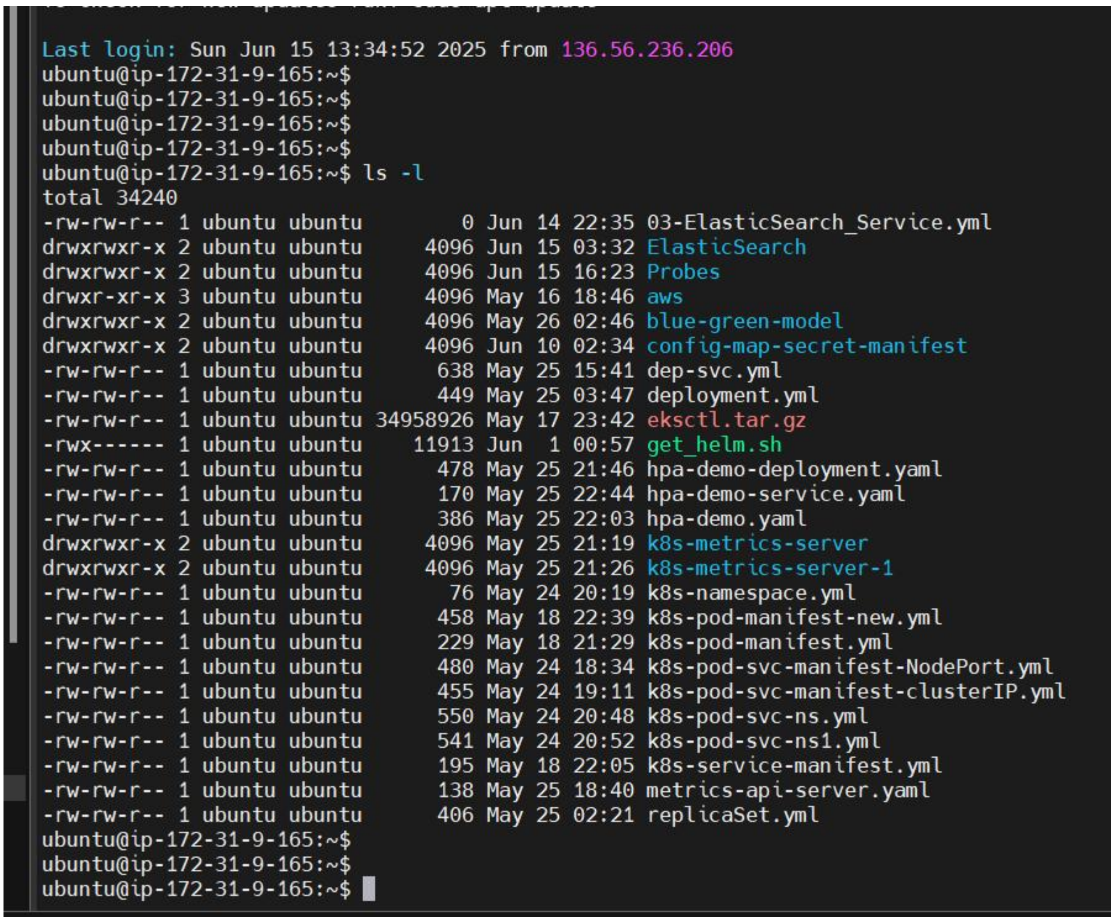


```
100   138  100   138    0     0   1734      0 --:--:-- --:--:-- --:--:--  1746
100 57.3M  100 57.3M    0     0  2421k      0  0:00:24  0:00:24 --:--:-- 2445k
Client Version: v1.33.2
Kustomize Version: v5.6.0
ubuntu@ip-172-31-11-116:~$
ubuntu@ip-172-31-11-116:~$
ubuntu@ip-172-31-11-116:~$ aws --version
aws-cli/2.27.50 Python/3.13.4 Linux/6.8.0-1031-aws exe/x86_64.ubuntu.24
ubuntu@ip-172-31-11-116:~$
ubuntu@ip-172-31-11-116:~$
ubuntu@ip-172-31-11-116:~$ kubectl version --client
Client Version: v1.33.2
Kustomize Version: v5.6.0
ubuntu@ip-172-31-11-116:~$
```

Start EKS-host machine



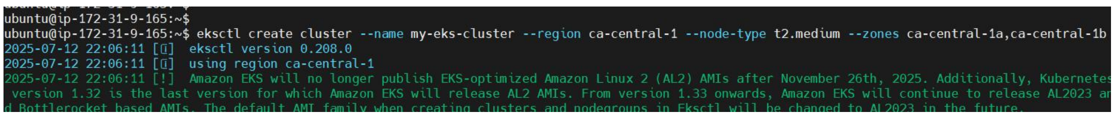
☐	Jenkins-Server-2	i-07e2b6dd29ae40a84	Stopped	t2.medium	Initializing	View ala
☑	EKS-host	i-01289fc5ca918b25f	Running	t2.micro	2/2 checks passec	View ala
☐	Jenkins-server	i-0abe9191466188c59	Running	t2.medium	2/2 checks passec	View ala



```
Last login: Sun Jun 15 13:34:52 2025 from 136.56.236.206
ubuntu@ip-172-31-9-165:~$
ubuntu@ip-172-31-9-165:~$
ubuntu@ip-172-31-9-165:~$
ubuntu@ip-172-31-9-165:~$
ubuntu@ip-172-31-9-165:~$ ls -l
total 34240
-rw-rw-r-- 1 ubuntu ubuntu        0 Jun 14 22:35 03-ElasticSearch_Service.yml
drwxrwxr-x 2 ubuntu ubuntu     4096 Jun 15 03:32 ElasticSearch
drwxrwxr-x 2 ubuntu ubuntu     4096 Jun 15 16:23 Probes
drwxr-xr-x 3 ubuntu ubuntu     4096 May 16 18:46 aws
drwxrwxr-x 2 ubuntu ubuntu     4096 May 26 02:46 blue-green-model
drwxrwxr-x 2 ubuntu ubuntu     4096 Jun 10 02:34 config-map-secret-manifest
-rw-rw-r-- 1 ubuntu ubuntu      638 May 25 15:41 dep-svc.yml
-rw-rw-r-- 1 ubuntu ubuntu      449 May 25 03:47 deployment.yml
-rw-rw-r-- 1 ubuntu ubuntu 34958926 May 17 23:42 eksctl.tar.gz
-rwx------ 1 ubuntu ubuntu    11913 Jun  1 00:57 get_helm.sh
-rw-rw-r-- 1 ubuntu ubuntu      478 May 25 21:46 hpa-demo-deployment.yaml
-rw-rw-r-- 1 ubuntu ubuntu      170 May 25 22:44 hpa-demo-service.yaml
-rw-rw-r-- 1 ubuntu ubuntu      386 May 25 22:03 hpa-demo.yaml
drwxrwxr-x 2 ubuntu ubuntu     4096 May 25 21:19 k8s-metrics-server
drwxrwxr-x 2 ubuntu ubuntu     4096 May 25 21:26 k8s-metrics-server-1
-rw-rw-r-- 1 ubuntu ubuntu       76 May 24 20:19 k8s-namespace.yml
-rw-rw-r-- 1 ubuntu ubuntu      458 May 18 22:39 k8s-pod-manifest-new.yml
-rw-rw-r-- 1 ubuntu ubuntu      229 May 18 21:29 k8s-pod-manifest.yml
-rw-rw-r-- 1 ubuntu ubuntu      480 May 24 18:34 k8s-pod-svc-manifest-NodePort.yml
-rw-rw-r-- 1 ubuntu ubuntu      455 May 24 19:11 k8s-pod-svc-manifest-clusterIP.yml
-rw-rw-r-- 1 ubuntu ubuntu      550 May 24 20:48 k8s-pod-svc-ns.yml
-rw-rw-r-- 1 ubuntu ubuntu      541 May 24 20:52 k8s-pod-svc-ns1.yml
-rw-rw-r-- 1 ubuntu ubuntu      195 May 18 22:05 k8s-service-manifest.yml
-rw-rw-r-- 1 ubuntu ubuntu      138 May 25 18:40 metrics-api-server.yaml
-rw-rw-r-- 1 ubuntu ubuntu      406 May 25 02:21 replicaSet.yml
ubuntu@ip-172-31-9-165:~$
ubuntu@ip-172-31-9-165:~$
ubuntu@ip-172-31-9-165:~$
```

Create EKS Cluster using eksctl

Create K8s cluster in **EKS-host VM**
eksctl create cluster --name my-eks-cluster --region ca-central-1 --node-type t2.medium --zones ca-central-1a,ca-central-1b



```
ubuntu@ip-172-31-9-165:~$
ubuntu@ip-172-31-9-165:~$ eksctl create cluster --name my-eks-cluster --region ca-central-1 --node-type t2.medium --zones ca-central-1a,ca-central-1b
2025-07-12 22:06:11 [ℹ]  eksctl version 0.208.0
2025-07-12 22:06:11 [ℹ]  using region ca-central-1
2025-07-12 22:06:11 [!]  Amazon EKS will no longer publish EKS-optimized Amazon Linux 2 (AL2) AMIs after November 26th, 2025. Additionally, Kubernetes
 version 1.32 is the last version for which Amazon EKS will release AL2 AMIs. From version 1.33 onwards, Amazon EKS will continue to release AL2023 an
d Bottlerocket based AMIs. The default AMI family when creating clusters and nodegroups in Eksctl will be changed to AL2023 in the future.
```

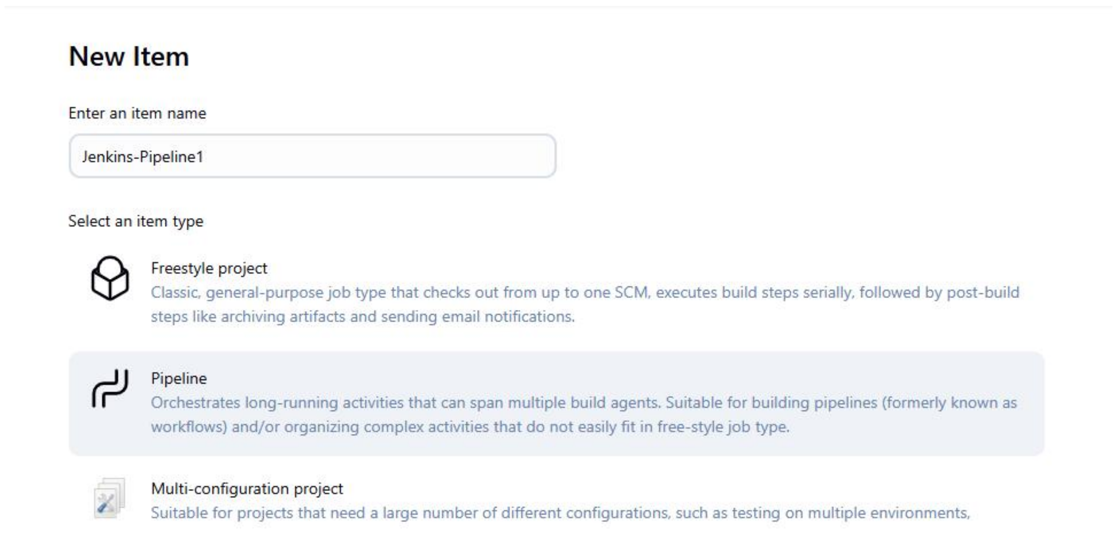
kubeconfig as "/home/ubuntu/.kube/config"

EKS cluster "telusko-cluster" in "ap-south-1" region is ready

$ cat /home/ubuntu/.kube/config

$ kubectl get nodes

New Item on Jenkins Pipeline


New Item

Enter an item name

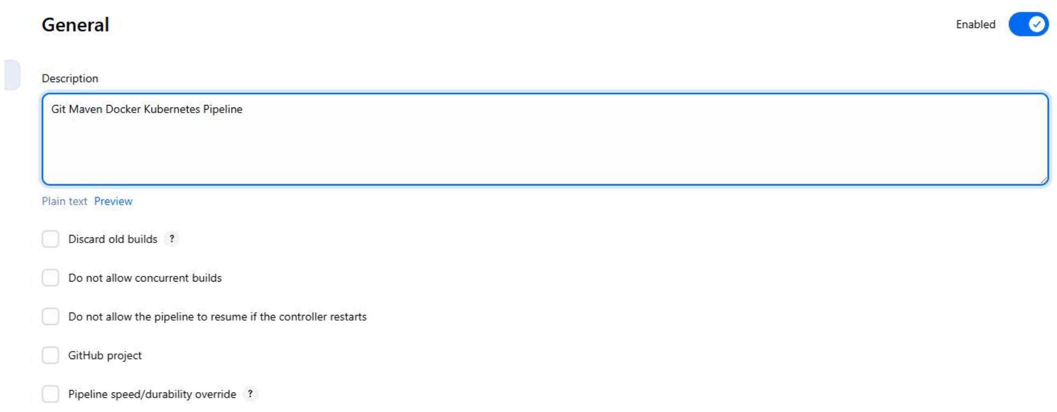
Jenkins-Pipeline1

Select an item type

Freestyle project
Classic, general-purpose job type that checks out from up to one SCM, executes build steps serially, followed by post-build steps like archiving artifacts and sending email notifications.

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.


General

Enabled

Description

Git Maven Docker Kubernetes Pipeline

Plain text Preview

Discard old builds ?

Do not allow concurrent builds

Do not allow the pipeline to resume if the controller restarts

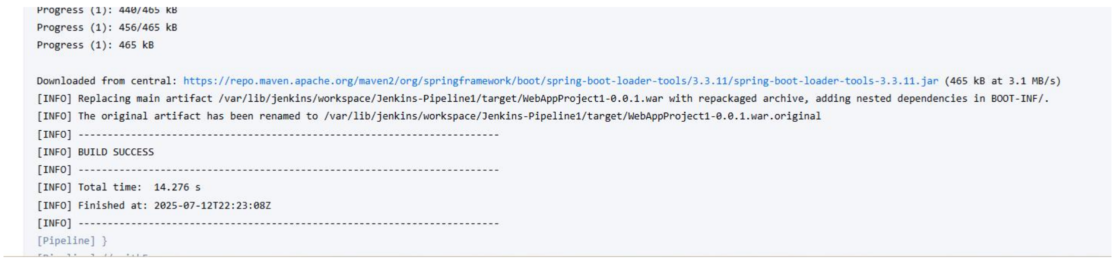
GitHub project

Pipeline speed/durability override ?

```
@Library('demo_shared_lib')_
pipeline {
  agent any

  tools {
    maven "maven-3.9.10"
  }

  stages {
    stage('git clone') {
      steps {
        git branch: 'main', url: 'https://github.com/Haider7214/SpringWebApp.git'
      }
    }
    stage('demo message') {
      steps {
        telusko()
      }
    }
    stage('maven build') {
      steps {
        mavenBuild()
      }
    }
    stage('parallel stage') {
      parallel {
            stage('code-review'){
                steps {
                    echo 'code review'
                    }
            }

            stage('nexus-upload'){
                steps {
                    echo 'nexus upload'
                    }
            }
        }

    }
    stage('app deployed') {
        steps {
            echo 'Deploying App with Tomcat'
        }
    }
  }
}
```

Apply and Save

Build Now
Build was successful

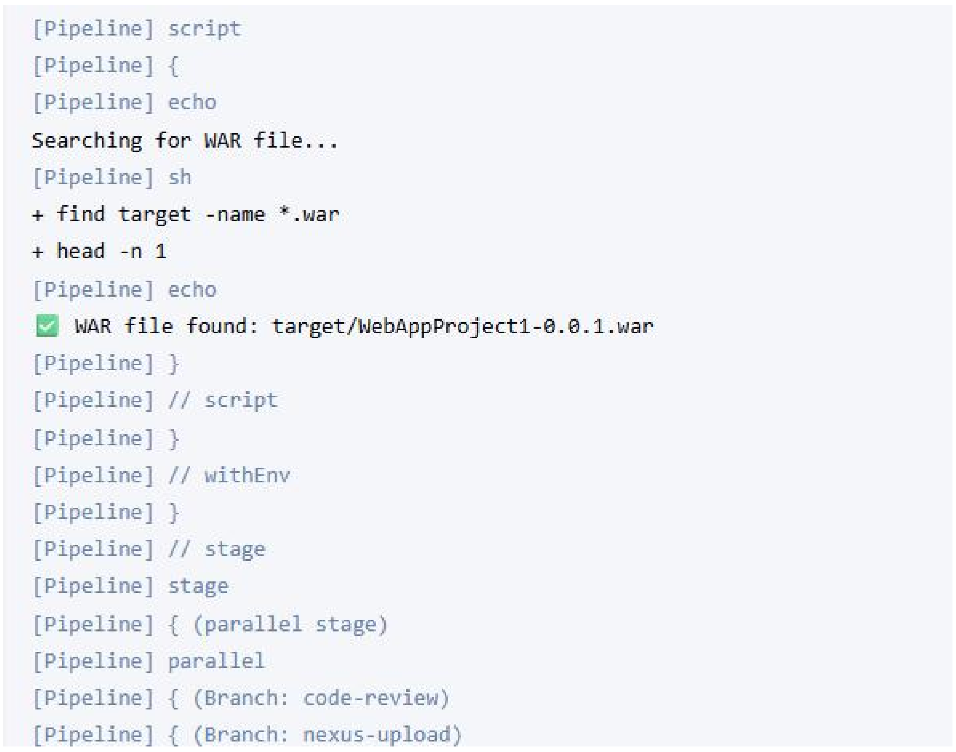


```
Progress (1): 440/465 kB
Progress (1): 456/465 kB
Progress (1): 465 kB

Downloaded from central: https://repo.maven.apache.org/maven2/org/springframework/boot/spring-boot-loader-tools/3.3.11/spring-boot-loader-tools-3.3.11.jar (465 kB at 3.1 MB/s)
[INFO] Replacing main artifact /var/lib/jenkins/workspace/Jenkins-Pipeline1/target/WebAppProject1-0.0.1.war with repackaged archive, adding nested dependencies in BOOT-INF/.
[INFO] The original artifact has been renamed to /var/lib/jenkins/workspace/Jenkins-Pipeline1/target/WebAppProject1-0.0.1.war.original
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  14.276 s
[INFO] Finished at: 2025-07-12T22:23:08Z
[INFO] ------------------------------------------------------------------------
[Pipeline] }
```

```
@Library('demo_shared_lib')_
pipeline {
    agent any

    tools {
        maven "maven-3.9.10"
    }

    stages {
        stage('git clone') {
            steps {
                git branch: 'main', url: 'https://github.com/Haider7214/SpringWebApp.git'
            }
        }
        stage('demo message') {
            steps {
                telusko()
            }
        }
        stage('maven build') {
            steps {
```

```
            mavenBuild()
         }
      }

      stage('Find WAR file') {
         steps {
                    script {
                               echo "Searching for WAR file..."
                               def warPath = sh(
                                  script: "find target -name '*.war' | head -n 1",
                                  returnStdout: true
                               ).trim()

                               if (warPath) {
                                  echo "✅ WAR file found: ${warPath}"
                               } else {
                                  error("❌ WAR file not found!")
                               }
                             }
                    }
            }
      stage('parallel stage') {
         parallel {
            stage('code-review'){
               steps {
                  echo 'code review'
                  }
            }

            stage('nexus-upload'){
               steps {
                  echo 'nexus upload'
                  }
            }
         }

      }
      stage('app deployed') {
         steps {
            echo 'Deploying App with Tomcat'
         }
      }
   }
}
```

Build was successful



```
[Pipeline] script
[Pipeline] {
[Pipeline] echo
Searching for WAR file...
[Pipeline] sh
+ find target -name *.war
+ head -n 1
[Pipeline] echo
✅ WAR file found: target/WebAppProject1-0.0.1.war
[Pipeline] }
[Pipeline] // script
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (parallel stage)
[Pipeline] parallel
[Pipeline] { (Branch: code-review)
[Pipeline] { (Branch: nexus-upload)
```

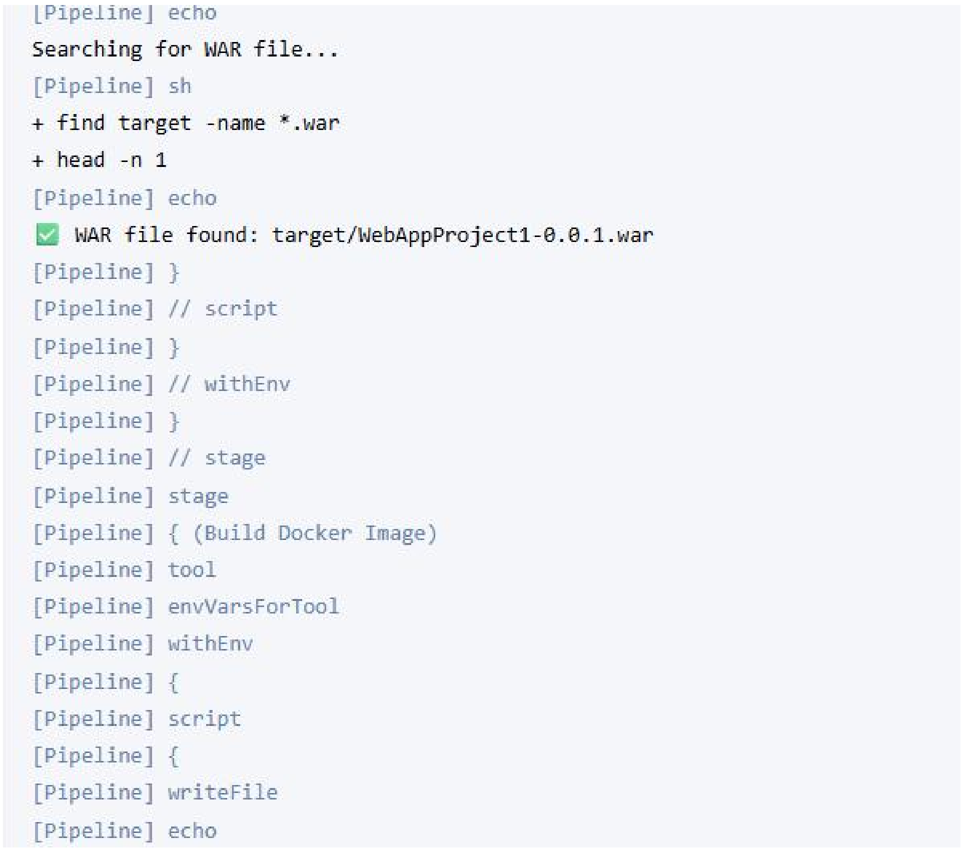
```
WAR file found: target/WebAppProject1-0.0.1.war
```

Create Dockerfile

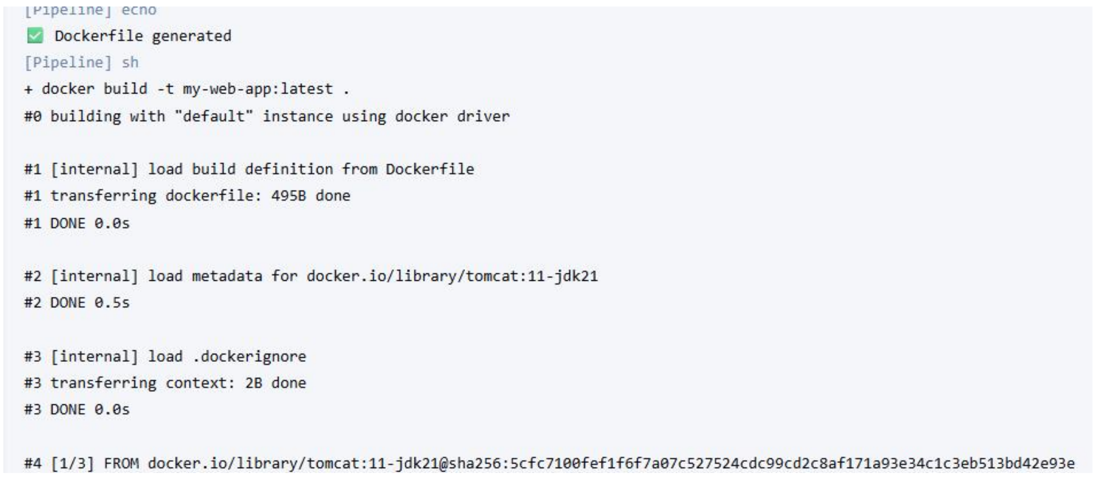
```
@Library('demo_shared_lib')_
pipeline {
  agent any

  environment {
        IMAGE_NAME = “my-web-app”
        DOCKER_TAG=”latest”
  }

  tools {
    maven "maven-3.9.10"
  }

  stages {
    stage('git clone') {
      steps {
        git branch: 'main', url: 'https://github.com/Haider7214/SpringWebApp.git'
      }
    }
    stage('demo message') {
      steps {
        telusko()
      }
    }
    stage('maven build') {
      steps {
        sh ‘mvn clean compile test package’
      }
    }
```

```
        stage('Find WAR file') {
          steps {
                        script {
                                        echo "Searching for WAR file..."
                                        def warPath = sh(
                                          script: "find target -name '*.war' | head -n 1",
                                          returnStdout: true
                                        ).trim()

                                        if (warPath) {
                                          echo "✅ WAR file found: ${warPath}"
                                        } else {
                                          error("❌ WAR file not found!")
                                        }
                                      }
                        }
            }

        stage('Build Docker Image') {
          steps {
            script {
              writeFile file: 'Dockerfile', text: '''
              # Use an official Tomcat base image
              FROM tomcat:11-jdk21
              LABEL maintainer="DemoDockerfile"

              # Remove default webapps
              RUN rm -rf /usr/local/tomcat/webapps/*

              # Copy WAR to Tomcat webapps
              COPY target/*.war /usr/local/tomcat/webapps/

              # Expose port
              EXPOSE 8080
'''
              echo "✅ Dockerfile generated"

              sh "docker build -t ${IMAGE_NAME}:${DOCKER_TAG} ."
            }
            }
        }
}

@Library('demo_shared_lib')_
pipeline {
agent any

environment {
            IMAGE_NAME = "my-web-app"
            DOCKER_TAG = "latest"
}

   tools {
      maven "maven-3.9.10"
   }
```

```
stages {
  stage('git clone') {
    steps {
      git branch: 'main', url: 'https://github.com/Haider7214/SpringWebApp.git'
    }
  }
  stage('demo message') {
    steps {
      telusko()
    }
  }
  stage('maven build') {
    steps {
      sh 'mvn clean compile test package'
    }
  }


  stage('Find WAR file') {
    steps {
                script {
        echo "Searching for WAR file..."
        def warPath = sh(
          script: "find target -name '*.war' | head -n 1",
          returnStdout: true
        ).trim()

        if (warPath) {
          echo "✅ WAR file found: ${warPath}"
        } else {
          error("❌ WAR file not found!")
        }
      }
    }
      }

  stage('Build Docker Image') {
    steps {
      script {
        writeFile file: 'Dockerfile', text: '''
        # Use an official Tomcat base image
        FROM tomcat:11-jdk21
        LABEL maintainer="DemoDockerfile"

        # Remove default webapps
        RUN rm -rf /usr/local/tomcat/webapps/*

        # Copy WAR to Tomcat webapps
        COPY target/*.war /usr/local/tomcat/webapps/

        # Expose port
        EXPOSE 8080
      '''
        echo "✅ Dockerfile generated"

            sh "docker build -t ${IMAGE_NAME}:${DOCKER_TAG} ."
            }
          }
        }
    }
}
```



```
[Pipeline] echo
Searching for WAR file...
[Pipeline] sh
+ find target -name *.war
+ head -n 1
[Pipeline] echo
✅ WAR file found: target/WebAppProject1-0.0.1.war
[Pipeline] }
[Pipeline] // script
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (Build Docker Image)
[Pipeline] tool
[Pipeline] envVarsForTool
[Pipeline] withEnv
[Pipeline] {
[Pipeline] script
[Pipeline] {
[Pipeline] writeFile
[Pipeline] echo
```

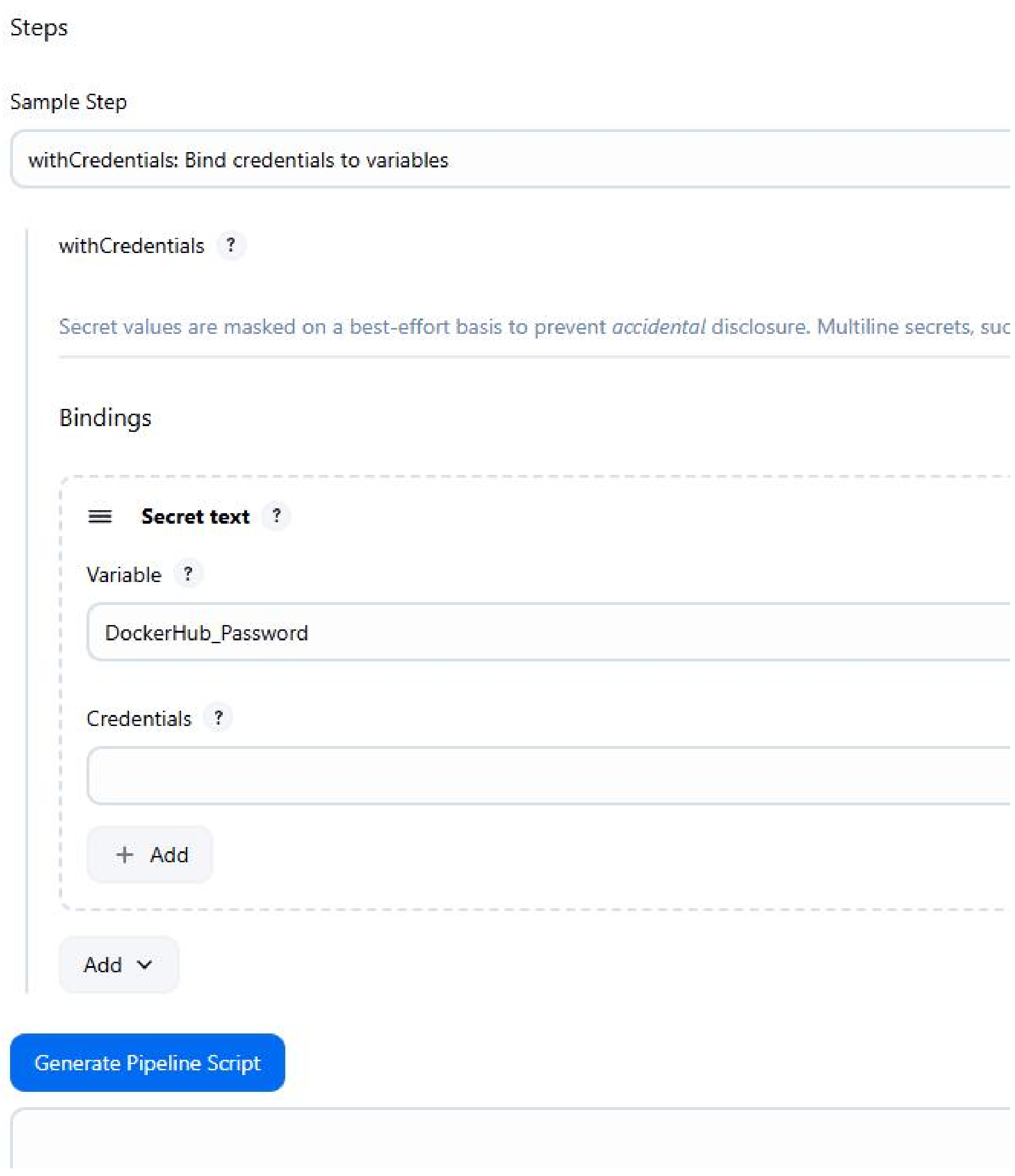


```
[Pipeline] echo
✅ Dockerfile generated
[Pipeline] sh
+ docker build -t my-web-app:latest .
#0 building with "default" instance using docker driver

#1 [internal] load build definition from Dockerfile
#1 transferring dockerfile: 495B done
#1 DONE 0.0s

#2 [internal] load metadata for docker.io/library/tomcat:11-jdk21
#2 DONE 0.5s

#3 [internal] load .dockerignore
#3 transferring context: 2B done
#3 DONE 0.0s

#4 [1/3] FROM docker.io/library/tomcat:11-jdk21@sha256:5cfc7100fef1f6f7a07c527524cdc99cd2c8af171a93e34c1c3eb513bd42e93e
```

```
#6 [2/3] RUN rm -rf /usr/local/tomcat/webapps/*
#6 DONE 1.4s

#7 [3/3] COPY target/*.war /usr/local/tomcat/webapps/
#7 DONE 0.4s

#8 exporting to image
#8 exporting layers
#8 exporting layers 0.2s done
#8 writing image sha256:3b91ee0eb2ba8fe61ea00c411337790e23250419f49ca8b8fd6a3a10e5ec5460 done
#8 naming to docker.io/library/my-web-app:latest done
#8 DONE 0.2s
[Pipeline] }
[Pipeline] // script
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```

Now to push this image we got to add credentials

Generate Pipeline Script


Steps

Sample Step

withCredentials: Bind credentials to variables

withCredentials ?

Secret values are masked on a best-effort basis to prevent *accidental* disclosure. Multiline secrets, suc

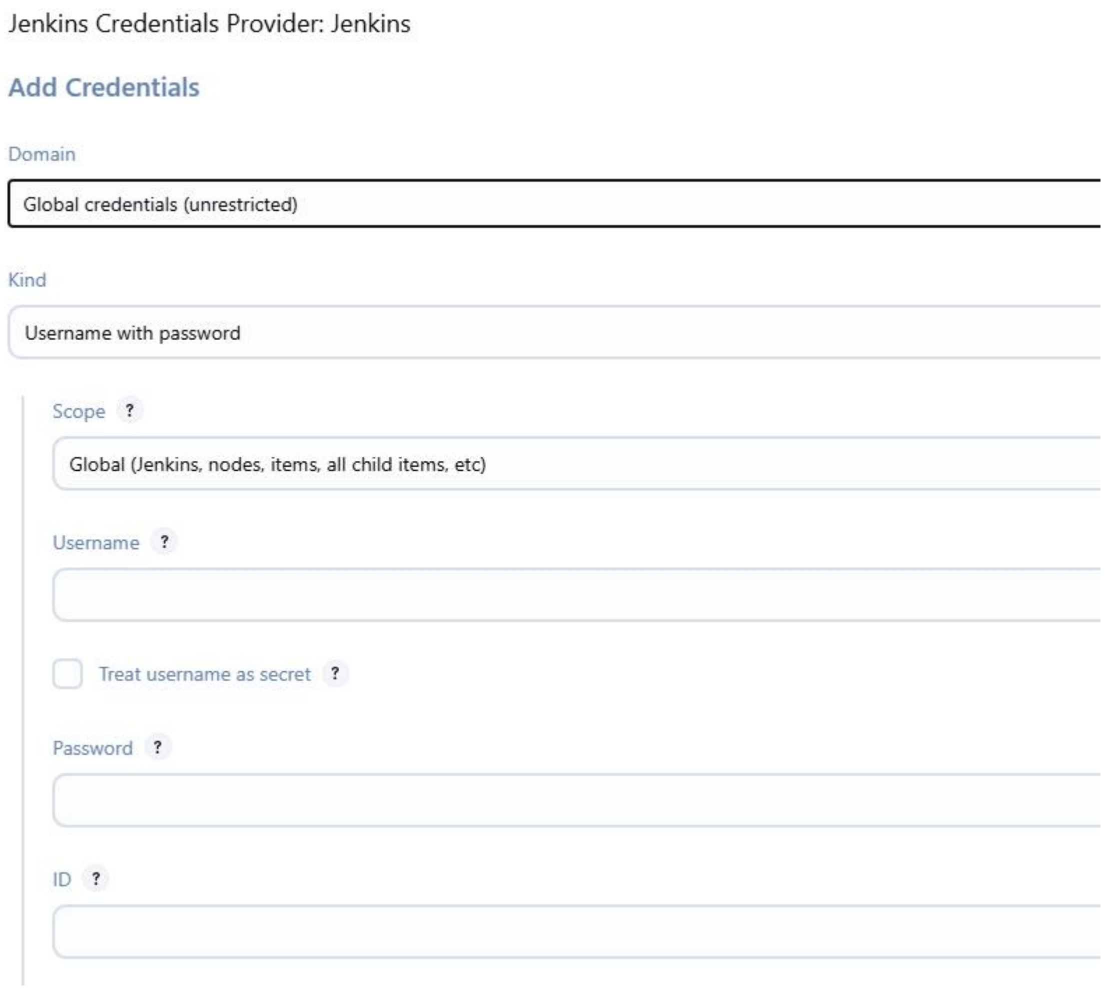
Bindings

Secret text ?

Variable ?

DockerHub_Password

Credentials ?



Jenkins Credentials Provider: Jenkins

Add Credentials

Domain

Global credentials (unrestricted)

Kind

Username with password

Scope ?

Global (Jenkins, nodes, items, all child items, etc)

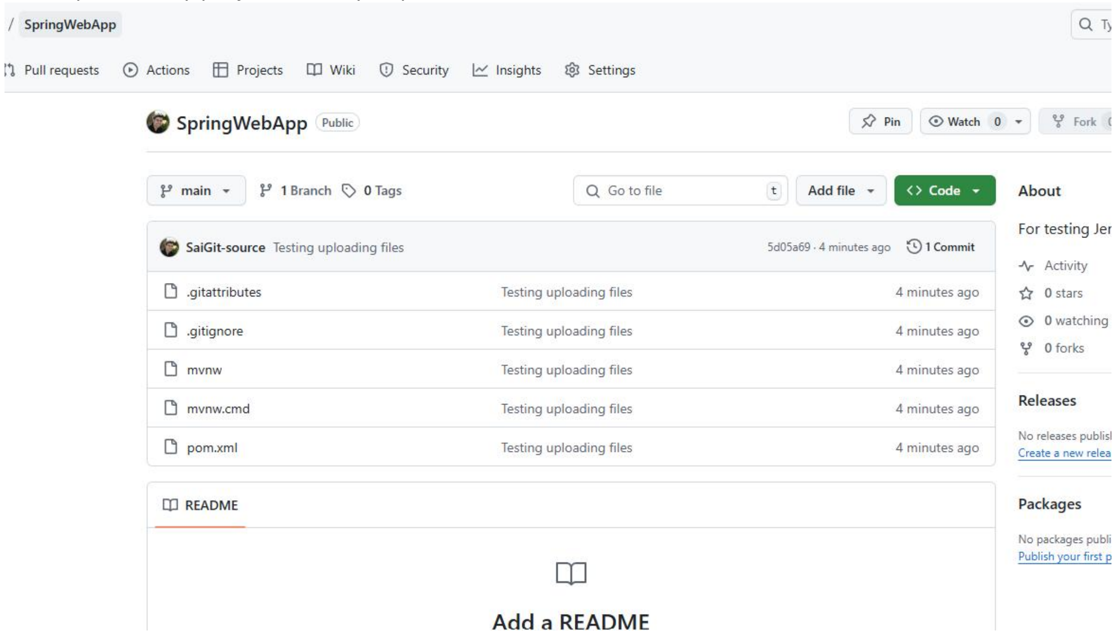
Username ?

Treat username as secret ?

Password ?

ID ?

I have pushed my project into my Repo



/ SpringWebApp

Pull requests · Actions · Projects · Wiki · Security · Insights · Settings

SpringWebApp Public

Pin · Watch 0 · Fork

main · 1 Branch · 0 Tags · Go to file · Add file · Code

SaiGit-source Testing uploading files 5d05a69 · 4 minutes ago 1 Commit

.gitattributes	Testing uploading files	4 minutes ago
.gitignore	Testing uploading files	4 minutes ago
mvnw	Testing uploading files	4 minutes ago
mvnw.cmd	Testing uploading files	4 minutes ago
pom.xml	Testing uploading files	4 minutes ago

README

Add a README

About

For testing Jer

Activity

0 stars

0 watching

0 forks

Releases

No releases publisl

Create a new relea

Packages

No packages publi

Publish your first p

Generate Pipeline Script

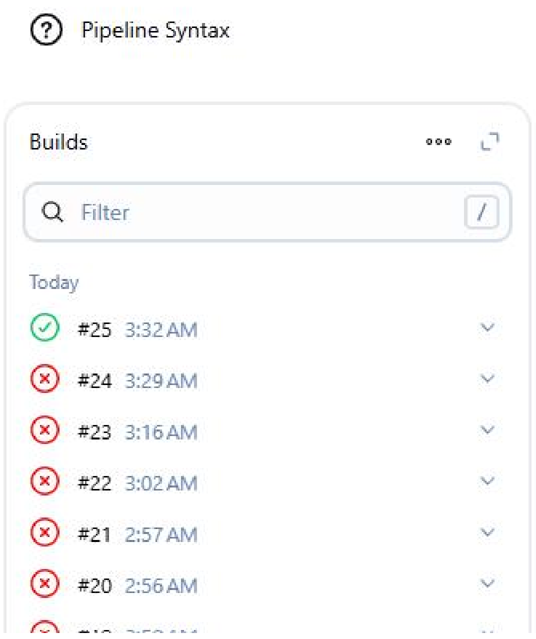
```
withCredentials([string(credentialsId: 'My-Docker-Git-pwd', variable: 'Docker_Hub_PWD')]) {
  // some block
}


pipeline {
agent any

environment {
        IMAGE_NAME = "my-web-app"
        DOCKER_TAG = "latest"
}

  tools {
    maven "maven-3.9.10"
  }

  stages {
    stage('git clone') {
      steps {
        git branch: 'main', url: 'https://github.com/SaiGit-source/SpringWebApp.git'
      }
    }
    stage('demo message') {
      steps {
        telusko()
      }
    }
    stage('maven build') {
      steps {
        sh 'mvn clean compile test package'
      }
    }

    stage('Find WAR file') {
      steps {
              script {
          echo "Searching for WAR file..."
          def warPath = sh(
            script: "find target -name '*.war' | head -n 1",
            returnStdout: true
          ).trim()

          if (warPath) {
            echo "✅ WAR file found: ${warPath}"
          } else {
            error("❌ WAR file not found!")
          }
        }
      }
        }

    stage('Build Docker Image') {
      steps {
        script {
          writeFile file: 'Dockerfile', text: '''
```

```
                # Use an official Tomcat base image
                FROM tomcat:11-jdk21
                LABEL maintainer="DemoDockerfile"

                # Remove default webapps
                RUN rm -rf /usr/local/tomcat/webapps/*

                # Copy WAR to Tomcat webapps
                COPY target/*.war /usr/local/tomcat/webapps/

                # Expose port
                EXPOSE 8080
            '''
                echo "✅ Dockerfile generated"

                    sh "docker build -t ${IMAGE_NAME}:${DOCKER_TAG} ."
            }
        }
    }

    stage('Docker push') {
        steps {
            withCredentials([string(credentialsId: 'My-Docker-Git-pwd', variable: 'Docker_Hub_PWD')]) {
                sh 'docker login -u SaiGit-Source -p ${Docker_Hub_PWD}'
                sh 'docker tag ${IMAGE_NAME}:${DOCKER_TAG} SaiGit-
Source/${IMAGE_NAME}:${DOCKER_TAG}'
                sh 'docker push SaiGit-Source/${IMAGE_NAME}:${DOCKER_TAG}'
            }
        }
    }
  }
}
```

```
pipeline {
agent any

environment {
        IMAGE_NAME = "my-web-app"
        DOCKER_TAG = "latest"
}

  tools {
    maven "maven-3.9.10"
  }

  stages {
    stage('git clone') {
      steps {
        git branch: 'main', url: 'https://github.com/SaiGit-source/SpringWebApp.git'
      }
    }
    stage('maven build') {
      steps {
        sh 'mvn clean package'
      }
    }
    stage('Build Docker Image') {
      steps {
        script {
          writeFile file: 'Dockerfile', text: '''
          # Use an official Tomcat base image
          FROM tomcat:latest
          LABEL maintainer="DemoDockerfile"

          # Remove default webapps
          RUN rm -rf /usr/local/tomcat/webapps/*

          # Copy WAR to Tomcat webapps
          COPY target/*.war /usr/local/tomcat/webapps/ROOT.war

          # Expose port
          EXPOSE 8080
        '''
          echo "✅ Dockerfile generated"

              sh "docker build -t ${IMAGE_NAME}:${DOCKER_TAG} ."
        }
      }
    }

    stage('Docker push') {
      steps {
        withCredentials([string(credentialsId: 'Sai-Docker-Pwd', variable: 'Docker_Hub_PWD_New')])
{
            sh 'docker login -u saidocker567 -p ${Docker_Hub_PWD_New}'
            sh 'docker tag ${IMAGE_NAME}:${DOCKER_TAG}
saidocker567/${IMAGE_NAME}:${DOCKER_TAG}'
            sh 'docker push saidocker567/${IMAGE_NAME}:${DOCKER_TAG}'
        }
      }
    }
  }
}
```

Build succeeded after numerous attempts

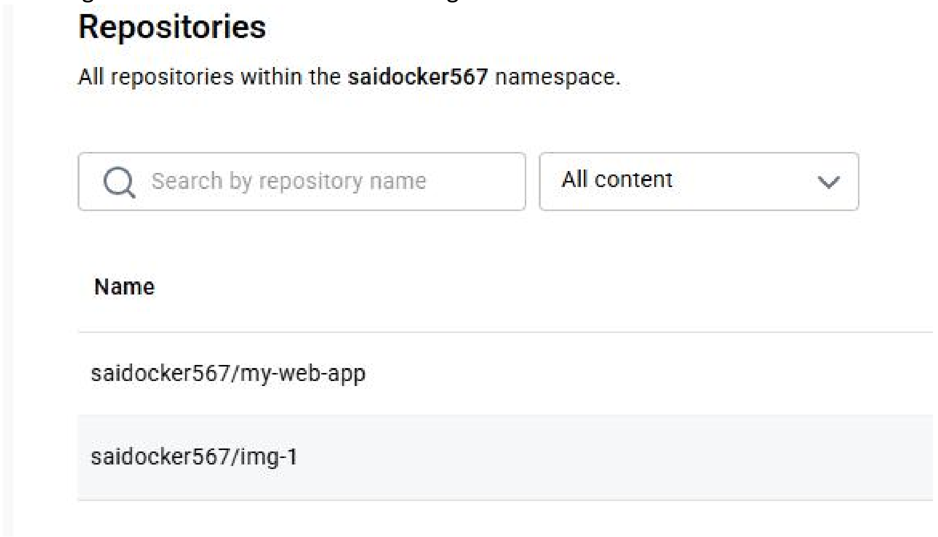
•



Today

✓	#25	3:32 AM
✗	#24	3:29 AM
✗	#23	3:16 AM
✗	#22	3:02 AM
✗	#21	2:57 AM
✗	#20	2:56 AM

If we log into Docker Hub we see the image

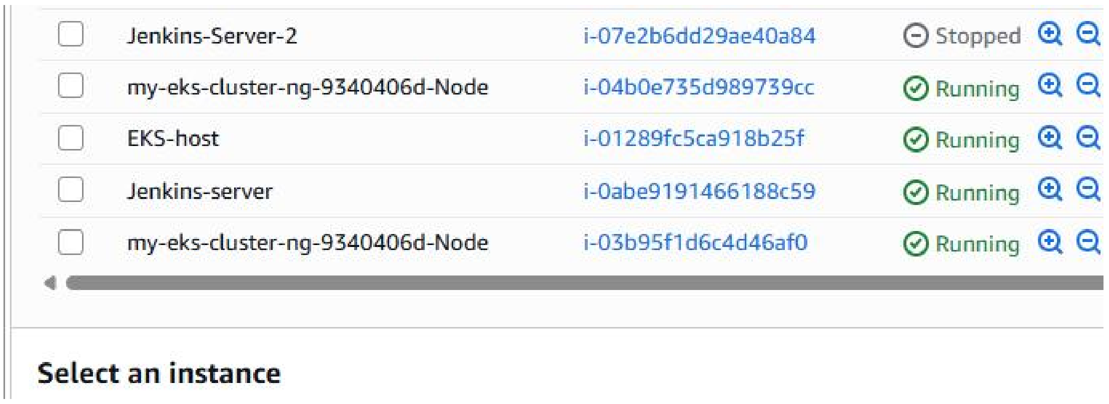


Repositories

All repositories within the **saidocker567** namespace.

Name
saidocker567/my-web-app
saidocker567/img-1

Go to EC2 and we can see EKS clusters created



☐	Jenkins-Server-2	i-07e2b6dd29ae40a84	Stopped
☐	my-eks-cluster-ng-9340406d-Node	i-04b0e735d989739cc	Running
☐	EKS-host	i-01289fc5ca918b25f	Running
☐	Jenkins-server	i-0abe9191466188c59	Running
☐	my-eks-cluster-ng-9340406d-Node	i-03b95f1d6c4d46af0	Running

Select an instance

1:45
Run on EKS host
kubectl delete all --all

eksctl delete cluster --name my-eks-cluster --region ca-central-1

Terminated Jenkins-Server-2

Starting EKS clusters in EKS-host VM
eksctl create cluster --name my-eks-cluster --region ca-central-1 --node-type t2.medium --zones ca-central-1a,ca-central-1b

8 ==> Update EKS Cluster Config File in Jenkins Server
Execute the below command in EKS Management Machine and Copy config file data

$ cat .kube/config

Connect to Jenkins server execute the following command to add config file into Jenkins server

$ cd /var/lib/jenkins

$ sudo mkdir .kube

$ sudo vi .kube/config
 (paste config file data copied from eks host machine)

Check eks nodes

$ kubectl get nodes

$ cd ~
$ ls -la
$ sudo vi .kube/config
(paste config file data copied from eks host machine)

$ kubectl get nodes

EKS-host VM
ubuntu@ip-172-31-9-165:~$ cat .kube/config
apiVersion: v1
clusters:
- cluster:
    certificate-authority-data:
LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0tCk1JSURCVENDQWUyZ0F3SUJBZ0lJRHFuUkFZVWR6M3d3R
FFZSktvWklodmNOQVFFTEJRQXdGVEVUTUJFR0ExVUUKQXhNS2EzVmlaWEp1WlhSbGN6QWVGdzB5Tl
RBM01UTXhNekl5TWpkYUZ3MHpOVEEzTVRFeE16STNNamRhTUJVeApFekFSQmdOVkJBTVRDbXQxW
W1WeWJtVjBaWE13Z2dFaU1BMEdDU3FHU0liM0RRRUJBUVVBQTRJQkR3QXdnZ0VLCkFvSUJBUUNXbj
ZoVEZtWm5wSmdheHllZnF3dVBsUEx2S1U1eUpHcUh6OXc5M3d5SlBvUEhPLzRXd1VZZSt2NTQKSFRTc
U53bUlQNHVyRUNNRDN0TE1OZktqYVZIVHBIbnR6YzNJU1I5cU1vZGkyTjEwMWp1T1M1dFptcnRlSFJIb
Apnd0Y1UnhzVTBPZys3VWdFNkpzVWlybUhTWXZwY0xUZS9iUEhrZ1BQMjlPNzNaVEszNytWbXRsQm1
KU0QybWUzCjVhMjJkZXRUQzd1aGFUTktBSzRUdWwvK0M3U3dJcFlxODJOeWplZm5OKyt5SlpiRmgvb3
ZlTEJtQ3ZwUGI5U24KeERhZ1JzRzBrb3BJVmZDTHg2cTFqdjRNK1pVYksweGlLMTRzeVoxTFZ3YS9RQUpy
TGp5cTJHQmwzVEpxaGt2NgpuSVFtQzRpbjlqL3l2VmhCMk0wYWRCdlp6ZUNYQWdNQkFBR2pXVEJYTU
E0R0ExVWREd0VCL3dRRUF3SUNwREFQQkJnTlZIUk1CQWY4RUJUQURBUUgvTUIwR0ExVWREZ1FXQkJ
UMjlyOGxab3laMVc2RXZVNy9CUnVWc1k5YlN6QVYKQmdOVkhSRUVEakFNZ2dwcmRXSmxjbTVsZEdW
ek1BMEdDU3FHU0liM0RRRUJDd1VBQTRJQkFRQllxNmQ2dEdxNgpXYXQ3R01rRVVlT0duRlZPWWtNV0
9mUVAxSkVDSW5kQTNRc3YrNlk4YUJxRVhRMUJVQnRCVDl3a2ZpNlBDT1BZCm9ydUJ2K2RrT1pLbWVG
UWV4WW0rbHd3ZXJ0TTc0UlRaZWdNOWtmTHg1azJXYnYyTmgxVHhyQkpsT1ZPUERHSW8KOXpycDFx
M2ptUHVQT3JqNWowVU0xN0V0SFNiL2RwbENrUVJSbW1xcWFnMlBrZlpXUjJMTzF5ek1qcWFIdGUzaw
pmTkw0bFJVLzZHMThnVG5aYmNpZGlXQlozZlhKZXhqRDc5YUtiUFRJRG1aYWhNUWNqbDN6ekRQQzNz

```
Nkl2emRmCjZjN0hqeVFLczRMTVEySXJ3OXBsUHJucXlrQXlIcS9sUmh0eVNwSjZoVUlyemwzNHpqYXpVR
XphT0lSb0lvL0cKU05BK2NwSE90VUMxCi0tLS0tRU5EIENFUlRJRklDQVRFLS0tLS0K
    server: https://A4534B05E0A24CC4222E7AEBE8F2A919.gr7.ca-central-1.eks.amazonaws.com
  name: my-eks-cluster.ca-central-1.eksctl.io
contexts:
- context:
    cluster: my-eks-cluster.ca-central-1.eksctl.io
    user: i-01289fc5ca918b25f@my-eks-cluster.ca-central-1.eksctl.io
  name: i-01289fc5ca918b25f@my-eks-cluster.ca-central-1.eksctl.io
current-context: i-01289fc5ca918b25f@my-eks-cluster.ca-central-1.eksctl.io
kind: Config
preferences: {}
users:
- name: i-01289fc5ca918b25f@my-eks-cluster.ca-central-1.eksctl.io
  user:
    exec:
      apiVersion: client.authentication.k8s.io/v1beta1
      args:
      - eks
      - get-token
      - --output
      - json
      - --cluster-name
      - my-eks-cluster
      - --region
      - ca-central-1
      command: aws
      env:
      - name: AWS_STS_REGIONAL_ENDPOINTS
        value: regional
      provideClusterInfo: false
```

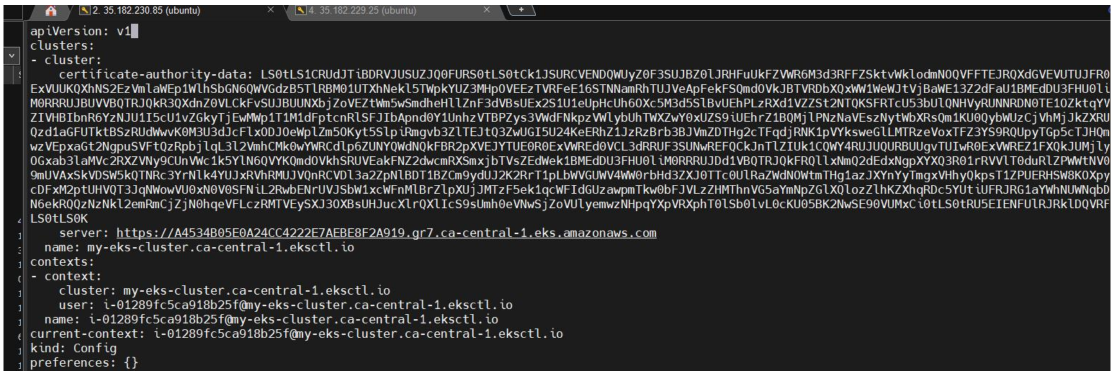
All on Jenkins server

```
ubuntu@ip-172-31-11-116:~/apache-tomcat-11.0.8$ kubectl version --client
Client Version: v1.33.2
Kustomize Version: v5.6.0

ubuntu@ip-172-31-11-116:~/apache-tomcat-11.0.8$ cd /var/lib/jenkins
ubuntu@ip-172-31-11-116:/var/lib/jenkins$
```

All in Jenkins server

```
ubuntu@ip-172-31-11-116:/var/lib/jenkins$
ubuntu@ip-172-31-11-116:/var/lib/jenkins$
ubuntu@ip-172-31-11-116:/var/lib/jenkins$
ubuntu@ip-172-31-11-116:/var/lib/jenkins$ sudo mkdir .kube
ubuntu@ip-172-31-11-116:/var/lib/jenkins$ cd  .kube/
ubuntu@ip-172-31-11-116:/var/lib/jenkins/.kube$ sudo vi config
```

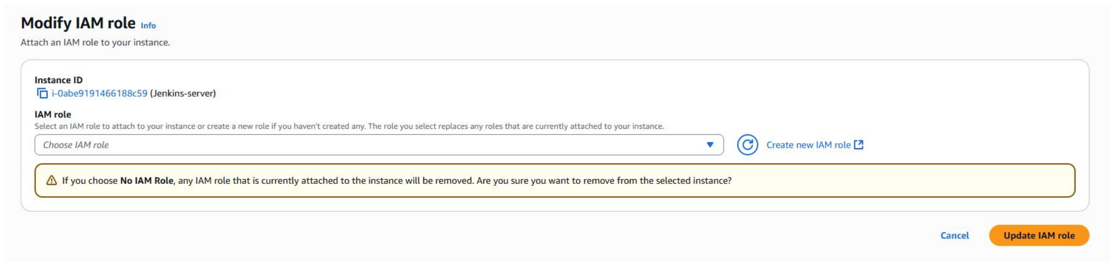
Copy paste contents from config file from EKS-Host



```
apiVersion: v1
clusters:
- cluster:
    certificate-authority-data: LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0tCk1JSURCVENDQWUyZ0F3SUJBZ0lJRHFuUkFZVWR6M3d3RFFZSktvWklodmNOQVFFTEJRQXdGVEVUTUJFR0
ExVUUKQXhNS2EzVmlaWEp1WlhSbGN6QWVGdzB5TlRBM01UTXhNekl5TWpkYUZ3MHpOVEEzTVRFeE16STNNamRhTUJVeApFekFSQmdOVkJBTVRDbXQxWW1WeWJtVjBaWE13Z2dFaU1BMEdDU3FHU0li
M0RRRUJBUVVBQTRJQkR3QXdnZ0VLCkFvSUJBUUNXbjZoVEZtWm5wSmdheHllZnF3dVBsUEx2S1U1eUpHcUh6OXc5M3d5SlBvUEhPLzRXd1VZZSt2NTQKSFRTcU53bUlQNHVyRUNNRDN0TE1OZktqYV
ZIVHBIbnR6YzNJU1I5cU1vZGkyTjEwMWp1T1M1dFptcnRlSFJIbApnd0Y1UnhzVTBPZys3VWdFNkpzVWlybUhTWXZwY0xUZS9iUEhrZ1BQMjlPNzNaVEszNytWbXRsQm1KU0QybWUzCjVhMjJkZXRU
Qzd1aGFUTktBSzRUdWwvK0M3U3dJcFlxODJOeWplZm5OKyt5SlpiRmgvb3ZlTEJtQ3ZwUGI5U24KeERhZ1JzRzBrb3BJVmZDTHg2cTFqdjRNK1pVYksweGlLMTRzeVoxTFZ3YS9RQUpyTGp5cTJHQm
wzVEpxaGt2NgpuSVFtQzRpbjlqL3l2VmhCMk0wYWRCdlp6ZUNYQWdNQkFBR2pXVEJYTUE0R0ExVWREd0VCL3dRRUF3SUNwREFQCkJnTlZIUk1CQWY4RUJUQURBUUgvTUIwR0ExVWREZ1FXQkJUMjly
OGxab3laMVc2RXZVNy9CUnVWc1k5YlN6QVYKQmd0VkhSRUVEakFNZ2dwcmRXSmxjbTVsZEdWek1BMEdDU3FHU0liM0RRRUJDd1VBQTRJQkFRQllxNmQ2dEdxNgpXYXQ3R01rRVVlT0duRlZPWWtNV0
9mUVAxSkVDSW5kQTNRc3YrNlk4YUJxRVhRMUJVQnRCVDl3a2ZpNlBDT1BZCm9ydUJ2K2RrT1pLbWVGUWV4WW0rbHd3ZXJ0TTc0UlRaZWdNOWtmTHg1azJXYnYyTmgxVHhyQkpsT1ZPUERHSW8KOXpy
cDFxM2ptUHVQT3JqNWowVU0xN0V0SFNiL2RwbENrUVJSbW1xcWFnMlBrZlpXUjJMTzF5ek1qcWFIdGUzawpmTkw0bFJVLzZHMThnVG5aYmNpZGlXQlozZlhKZXhqRDc5YUtiUFRJRG1aYWhNUWNqbD
N6ekRQQzNzNkl2emRmCjZjN0hqeVFLczRMTVEySXJ3OXBsUHJucXlrQXlIcS9sUmh0eVNwSjZoVUlyemwzNHpqYXpVRXphT0lSb0lvL0cKU05BK2NwSE90VUMxCi0tLS0tRU5EIENFUlRJRklDQVRF
LS0tLS0K
    server: https://A4534B05E0A24CC4222E7AEBE8F2A919.gr7.ca-central-1.eks.amazonaws.com
  name: my-eks-cluster.ca-central-1.eksctl.io
contexts:
- context:
    cluster: my-eks-cluster.ca-central-1.eksctl.io
    user: i-01289fc5ca918b25f@my-eks-cluster.ca-central-1.eksctl.io
  name: i-01289fc5ca918b25f@my-eks-cluster.ca-central-1.eksctl.io
current-context: i-01289fc5ca918b25f@my-eks-cluster.ca-central-1.eksctl.io
kind: Config
preferences: {}
```

Esc -> :wq

===> Attach created role to Jenkins Server host vm --> Actions-> Security-> Modify IAM role --> add created iam role


Modify IAM role Info

Attach an IAM role to your instance.

Instance ID
i-0abe9191466188c59 (Jenkins-server)

IAM role
Select an IAM role to attach to your instance or create a new role if you haven't created any. The role you select replaces any roles that are currently attached to your instance.

Choose IAM role

Create new IAM role

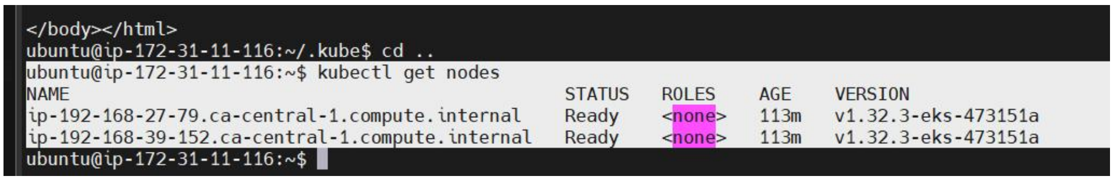
If you choose **No IAM Role**, any IAM role that is currently attached to the instance will be removed. Are you sure you want to remove from the selected instance?

eks-role is already there
Update IAM role

Copy paste the config content into this config file
ubuntu@ip-172-31-11-116:~/.kube$ sudo vi config
ubuntu@ip-172-31-11-116:~/.kube$ pwd
/home/ubuntu/.kube

Path is in home
All in Jenkins Server
ubuntu@ip-172-31-11-116:~$ kubectl get nodes

```
NAME                                          STATUS   ROLES    AGE    VERSION
ip-192-168-27-79.ca-central-1.compute.internal    Ready    <none>   113m   v1.32.3-eks-473151a
ip-192-168-39-152.ca-central-1.compute.internal   Ready    <none>   113m   v1.32.3-eks-473151a
```



```
</body></html>
ubuntu@ip-172-31-11-116:~/.kube$ cd ..
ubuntu@ip-172-31-11-116:~$ kubectl get nodes
NAME                                              STATUS   ROLES    AGE    VERSION
ip-192-168-27-79.ca-central-1.compute.internal    Ready    <none>   113m   v1.32.3-eks-473151a
ip-192-168-39-152.ca-central-1.compute.internal   Ready    <none>   113m   v1.32.3-eks-473151a
ubuntu@ip-172-31-11-116:~$
```

1:56
For adding K8s into Pipeline script

```
pipeline {
agent any

environment {
        IMAGE_NAME = "my-web-app"
        DOCKER_TAG = "latest"
}
```

```
    tools {
        maven "maven-3.9.10"
    }

    stages {
        stage('git clone') {
            steps {
                git branch: 'main', url: 'https://github.com/SaiGit-source/SpringWebApp.git'
            }
        }
        stage('maven build') {
            steps {
                sh 'mvn clean package'
            }
        }
        stage('Build Docker Image') {
            steps {
                script {
                    writeFile file: 'Dockerfile', text: '''
                    # Use an official Tomcat base image
                    FROM tomcat:latest
                    LABEL maintainer="DemoDockerfile"

                    # Remove default webapps
                    RUN rm -rf /usr/local/tomcat/webapps/*

                    # Copy WAR to Tomcat webapps
                    COPY target/*.war /usr/local/tomcat/webapps/ROOT.war

                    # Expose port
                    EXPOSE 8080
                '''
                    echo "✅ Dockerfile generated"

                        sh "docker build -t ${IMAGE_NAME}:${DOCKER_TAG} ."
                }
            }
        }

        stage('Docker push') {
            steps {
                withCredentials([string(credentialsId: 'Sai-Docker-Pwd', variable: 'Docker_Hub_PWD_New')])
                {
                        sh 'docker login -u saidocker567 -p ${Docker_Hub_PWD_New}'
                        sh 'docker tag ${IMAGE_NAME}:${DOCKER_TAG}
saidocker567/${IMAGE_NAME}:${DOCKER_TAG}'
                        sh 'docker push saidocker567/${IMAGE_NAME}:${DOCKER_TAG}'
                }
            }
        }

        stage('Deploy to Kubernetes') {
            steps {
                script {
                    // Write Kubernetes Deployment YAML
                    writeFile file: 'deployment.yaml', text: '''
                apiVersion: apps/v1
                kind: Deployment
                metadata:
                  name: web-app-deployment
                spec:
                  replicas: 2
                  selector:
                    matchLabels:
                      app: web-app
                  template:
                    metadata:
                      labels:
                        app: web-app
                    spec:
                      containers:
                      - name: web-container
                        image: saidocker567/my-web-app:latest
                        ports:
                        - containerPort: 8080
                                '''

                                // Write Kubernetes Service YAML
                                writeFile file: 'service.yaml', text: '''
                apiVersion: v1
                kind: Service
                metadata:
                  name: web-app-service
                spec:
                  type: LoadBalancer
                  selector:
                    app: web-app
                  ports:
                    - protocol: TCP
                      port: 80
                      targetPort: 8080
                                '''

                // Apply Kubernetes manifests
                sh 'kubectl apply -f deployment.yaml'
                sh 'kubectl apply -f service.yaml'
            }
        }
    }

}
```

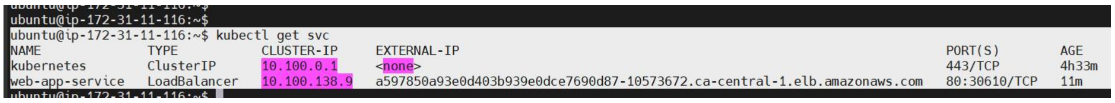
We have to write a K8s manifest file in the pipeline similar to Dockerfile

https://github.com/SaiGit-source/SpringWebApp

This part could be avoided if we already have a K8s manifest file in the Github repo since I don't have the manifest file I have to create one in the pipeline itself

```
[Pipeline] writeFile
[Pipeline] sh
+ kubectl apply -f deployment.yaml
deployment.apps/web-app-deployment created
[Pipeline] sh
+ kubectl apply -f service.yaml
service/web-app-service created
[Pipeline] }
[Pipeline] // script
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```

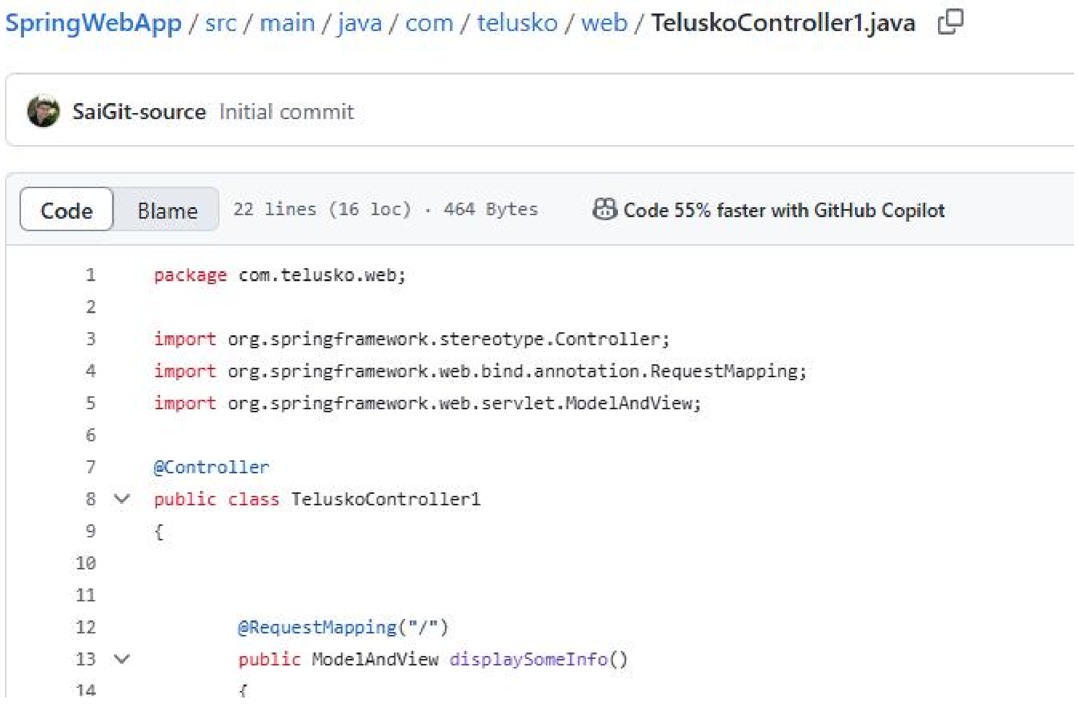
```
ubuntu@ip-172-31-11-116:~$ kubectl get svc
NAME           TYPE         CLUSTER-IP    EXTERNAL-IP                                       PORT(S)
AGE
kubernetes      ClusterIP     10.100.0.1     <none>                                          443/TCP
4h33m
web-app-service   LoadBalancer   10.100.138.9   a597850a93e0d403b939e0dce7690d87-10573672.ca-
central-1.elb.amazonaws.com   80:30610/TCP   11m
```



```
ubuntu@ip-172-31-11-116:~$
ubuntu@ip-172-31-11-116:~$ kubectl get svc
NAME              TYPE           CLUSTER-IP     EXTERNAL-IP                                                                PORT(S)        AGE
kubernetes        ClusterIP      10.100.0.1     <none>                                                                     443/TCP        4h33m
web-app-service   LoadBalancer   10.100.138.9   a597850a93e0d403b939e0dce7690d87-10573672.ca-central-1.elb.amazonaws.com   80:30610/TCP   11m
ubuntu@ip-172-31-11-116:~$
```

Copy this
a597850a93e0d403b939e0dce7690d87-10573672.ca-central-1.elb.amazonaws.com

In the application controller, we have only ‘/’ as endpoint



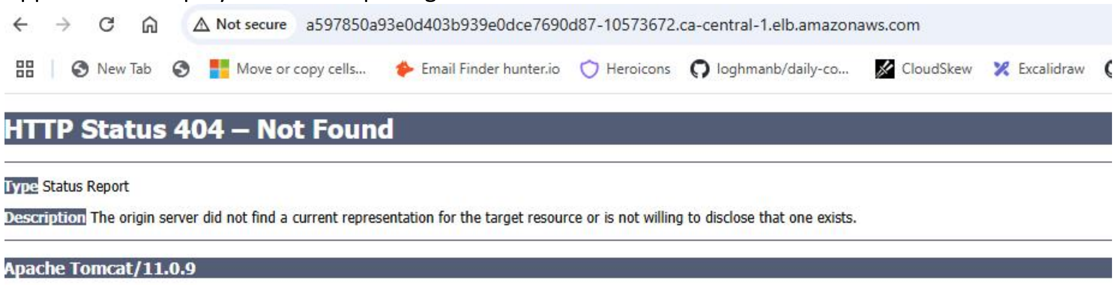
SpringWebApp / src / main / java / com / telusko / web / TeluskoController1.java

SaiGit-source Initial commit

Code
Blame
22 lines (16 loc) · 464 Bytes
Code 55% faster with GitHub Copilot

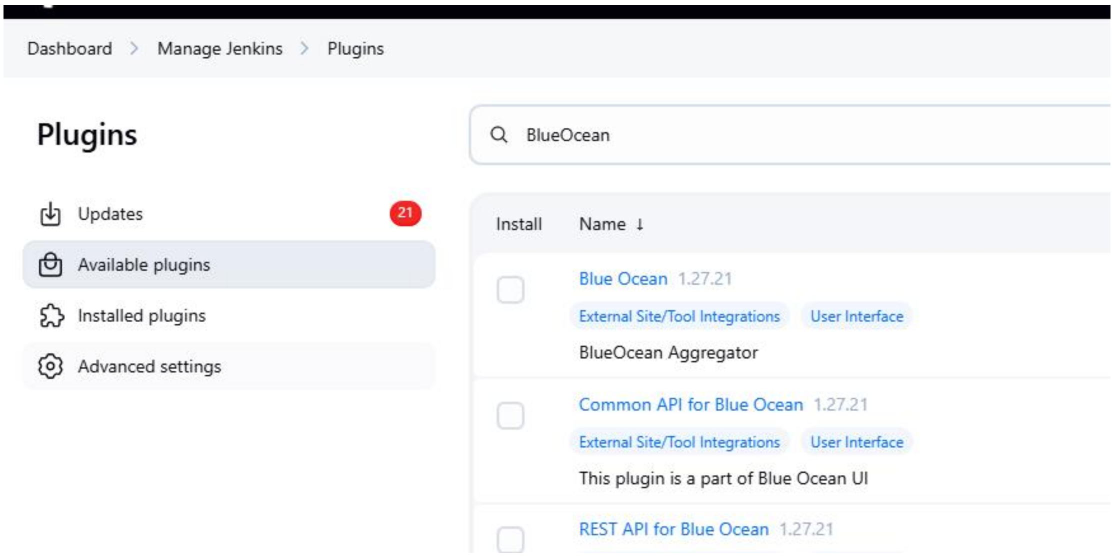
```java
package com.telusko.web;

import org.springframework.stereotype.Controller;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.servlet.ModelAndView;

@Controller
public class TeluskoController1
{


        @RequestMapping("/")
        public ModelAndView displaySomeInfo()
        {
```

Application is deployed but not opening



```
HTTP Status 404 – Not Found

Type Status Report

Description The origin server did not find a current representation for the target resource or is not willing to disclose that one exists.

Apache Tomcat/11.0.9
```

Look for BlueOcean, we can check the Logs from each stage and see what failed the App

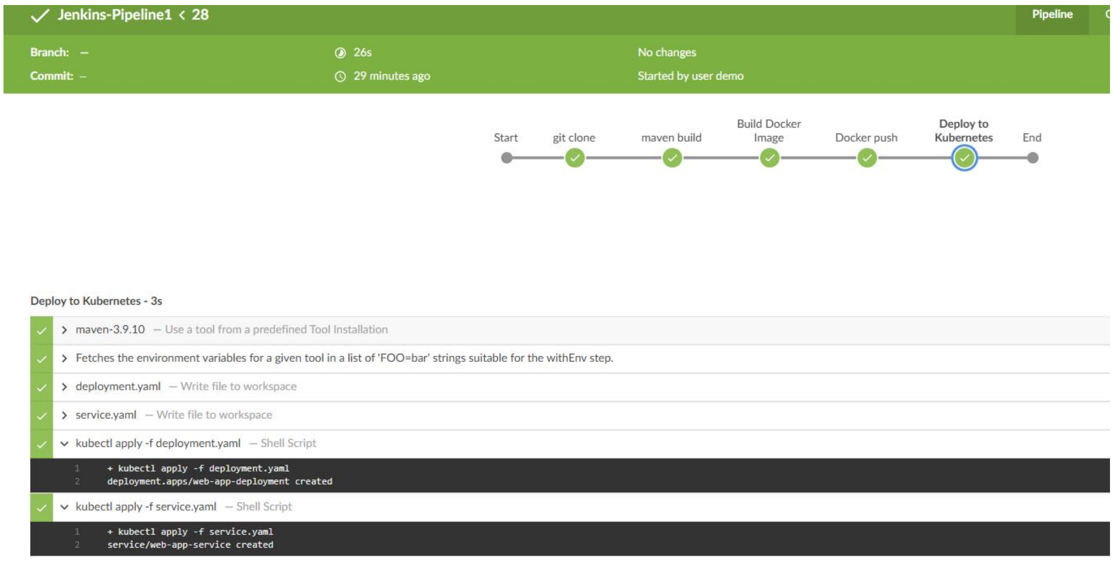


Blue Ocean 1.27.21
External Site/Tool Integrations User Interface
BlueOcean Aggregator

Common API for Blue Ocean 1.27.21
External Site/Tool Integrations User Interface
This plugin is a part of Blue Ocean UI

REST API for Blue Ocean 1.27.21

Click Install
Click on Dashboard
Open Blue Ocean



Branch: —
Commit: —
26s
29 minutes ago
No changes
Started by user demo

Deploy to Kubernetes - 3s

```
> maven-3.9.10 — Use a tool from a predefined Tool Installation
> Fetches the environment variables for a given tool in a list of 'FOO=bar' strings suitable for the withEnv step.
> deployment.yaml — Write file to workspace
> service.yaml — Write file to workspace
v kubectl apply -f deployment.yaml — Shell Script
    + kubectl apply -f deployment.yaml
    deployment.apps/web-app-deployment created
v kubectl apply -f service.yaml — Shell Script
    + kubectl apply -f service.yaml
    service/web-app-service created
```

Second way to deploy pipeline

```
pipeline {
  agent any
  environment {
    IMAGE_NAME = "my-web-app"
    DOCKER_TAG = "latest"
  }
  tools {
                maven "maven"
        }

  stages {
    stage('git clone') {
      steps {
        git branch: 'main', url: 'https://github.com/Haider7214/WebAppMaven.git'
      }
```

```
        }
        stage('maven build') {
            steps {
                sh 'mvn clean compile test package'
            }
        }
        stage('Build Docker Image') {
            steps {
                script {
                    sh "docker build -t ${IMAGE_NAME}:${DOCKER_TAG} ."

                }
            }
        }
        stage('Docker Push') {
            steps {
                withCredentials([string(credentialsId: 'Docker-pwd', variable: 'Docker_Hub_PWD')]) {
                  sh "docker login -u haidertelusko -p ${Docker_Hub_PWD}"
                  sh "docker tag ${IMAGE_NAME}:${DOCKER_TAG}
haidertelusko/${IMAGE_NAME}:${DOCKER_TAG}"
                  sh "docker push haidertelusko/${IMAGE_NAME}:${DOCKER_TAG}"
                }

            }
        }
         stage('k8s - deployment') {
            steps {
                sh 'kubectl apply -f k8s-deployment.yaml'

            }
        }
    }
}
```