DevOps Project

Problem Statement:

Create an end-to-end CI/CD pipeline in AWS platform using Jenkins as the orchestration tool, GitHub as scm, maven as the build tool, deploy in a docker instance and create a docker image, store the docker image in ECR, Kubernetes deployment using ECR image. Build sample java web app using maven.

Approach:

Requirements:

- ✓ CI/CD pipeline System
- ✓ Git - local version control system.
- ✓ GitHub - As Distributed version control system.
- ✓ Jenkins - Continuous Integration and Orchestration tool.
- ✓ Maven - As a Build Tool.
- ✓ ECR – AWS Container Registry
- ✓ docker -Containerization
- ✓ Kubernetes - As Container Orchestration Tool

Step-1:

- ➢ Install Jenkins, Maven, Git.
- ➢ Configure Jenkins global tools and install necessary plugins.
- ➢ Setup Tomcat server.
- ➢ Create a pipeline job in Jenkins.
- ➢ Create Jenkins file on pipeline job.
- ➢ Test deployment by accessing Tomcat URL.

Step-2:

- Create a Docker Host and Install Docker on it.
- Create a repo in ECR.
- Update Jenkins file to Integrate with Docker Host.
- Use declarative pipeline to build and push image to ECR.

Step-3:

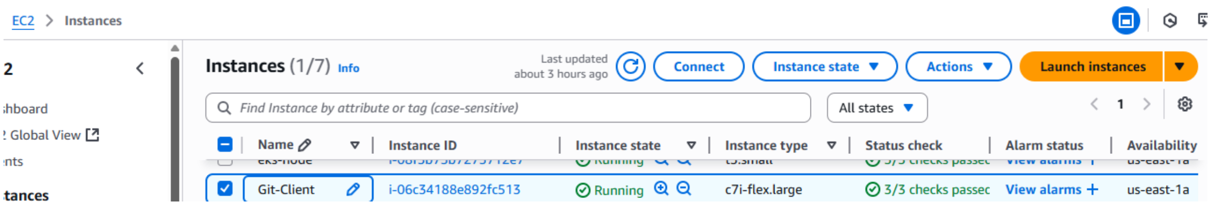
- Setup EKS Host.
- Install AWSCLI, Kubectl, Eksctl.
- Configure AWS.
- Create cluster.
- Create deployment.yaml, service.yaml.
- Update Jenkins file to Integrate with EKS Host.
- Use declarative pipeline to deploy.

Step-4:

- Deploy artifacts on the Kubernetes
- Write codes in the artifacts of docker and Kubernetes which we want to run.
- Now build the code in Jenkins.
- Check in Kubernetes the pods are getting created or not.
- Now copy the service IP and paste it in the browser and check the output.

Solution:

An EC2 instance is created for git operation

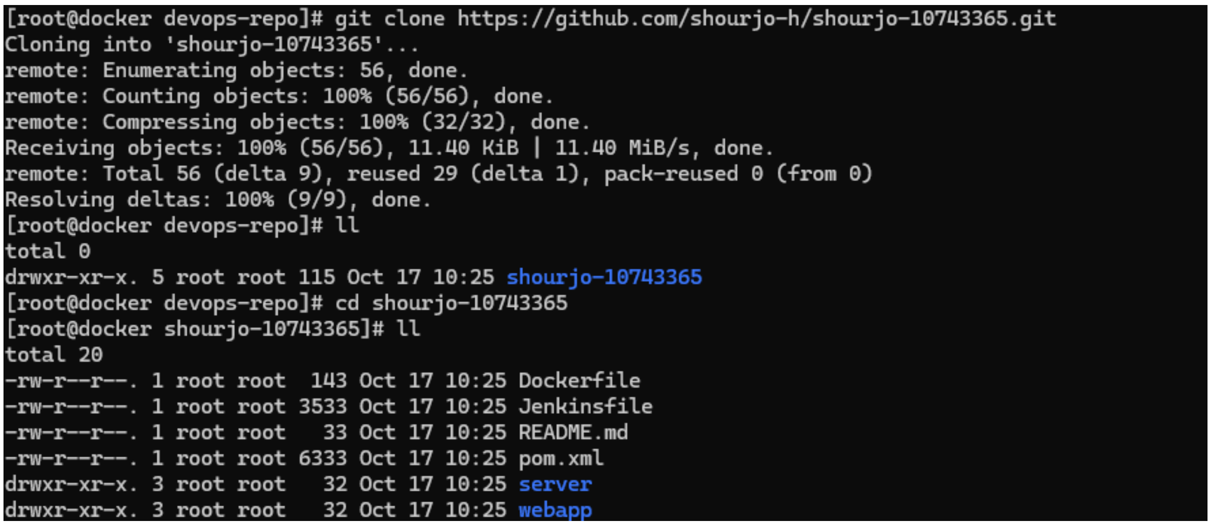

EC2 > Instances

Instances (1/7) Info

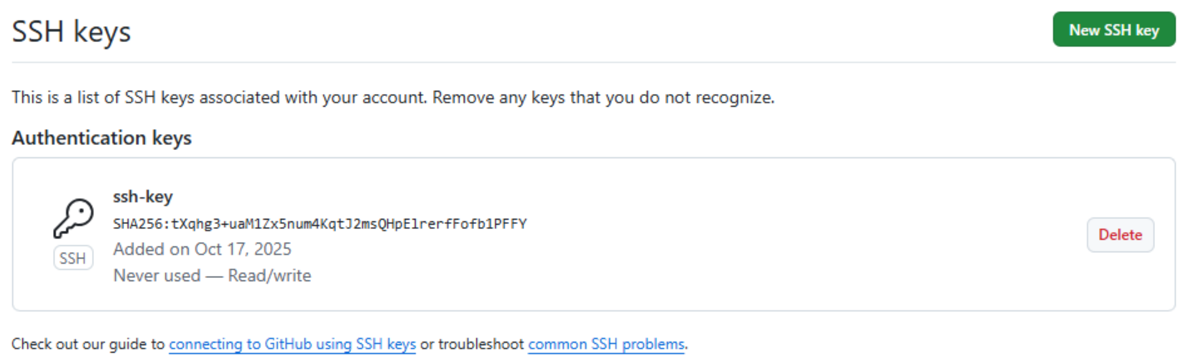
Last updated about 3 hours ago

	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability
✓	Git-Client	i-06c34188e892fc513	Running	c7i-flex.large	3/3 checks passec	View alarms +	us-east-1a

Repository has been cloned


```
[root@docker devops-repo]# git clone https://github.com/shourjo-h/shourjo-10743365.git
Cloning into 'shourjo-10743365'...
remote: Enumerating objects: 56, done.
remote: Counting objects: 100% (56/56), done.
remote: Compressing objects: 100% (32/32), done.
Receiving objects: 100% (56/56), 11.40 KiB | 11.40 MiB/s, done.
remote: Total 56 (delta 9), reused 29 (delta 1), pack-reused 0 (from 0)
Resolving deltas: 100% (9/9), done.
[root@docker devops-repo]# ll
total 0
drwxr-xr-x. 5 root root 115 Oct 17 10:25 shourjo-10743365
[root@docker devops-repo]# cd shourjo-10743365
[root@docker shourjo-10743365]# ll
total 20
-rw-r--r--. 1 root root  143 Oct 17 10:25 Dockerfile
-rw-r--r--. 1 root root 3533 Oct 17 10:25 Jenkinsfile
-rw-r--r--. 1 root root   33 Oct 17 10:25 README.md
-rw-r--r--. 1 root root 6333 Oct 17 10:25 pom.xml
drwxr-xr-x. 3 root root   32 Oct 17 10:25 server
drwxr-xr-x. 3 root root   32 Oct 17 10:25 webapp
```

SSH key is generated and added in GitHub for connection with GitHub to push it to my GitHub.


SSH keys

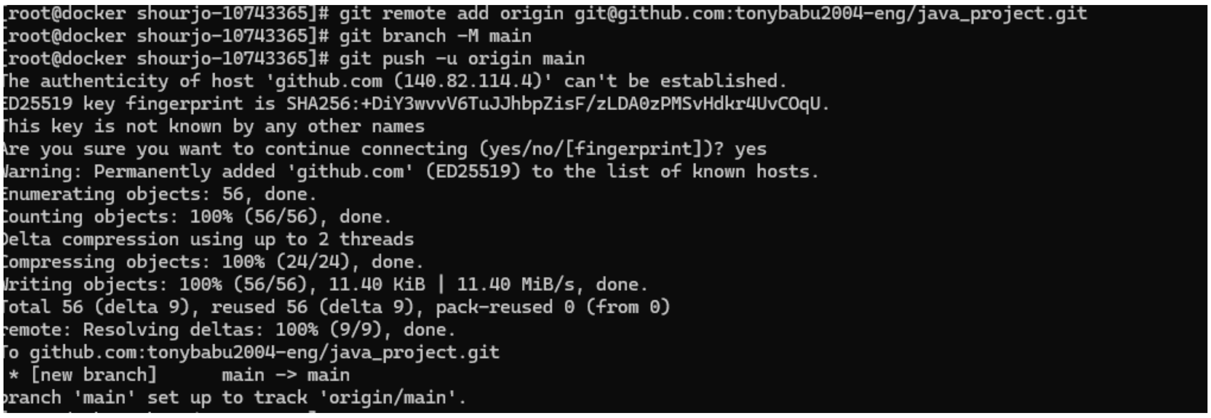
This is a list of SSH keys associated with your account. Remove any keys that you do not recognize.

Authentication keys

ssh-key
SHA256:tXqhg3+uaM1Zx5num4KqtJ2msQHpElrerfFofb1PFFY
Added on Oct 17, 2025
Never used — Read/write

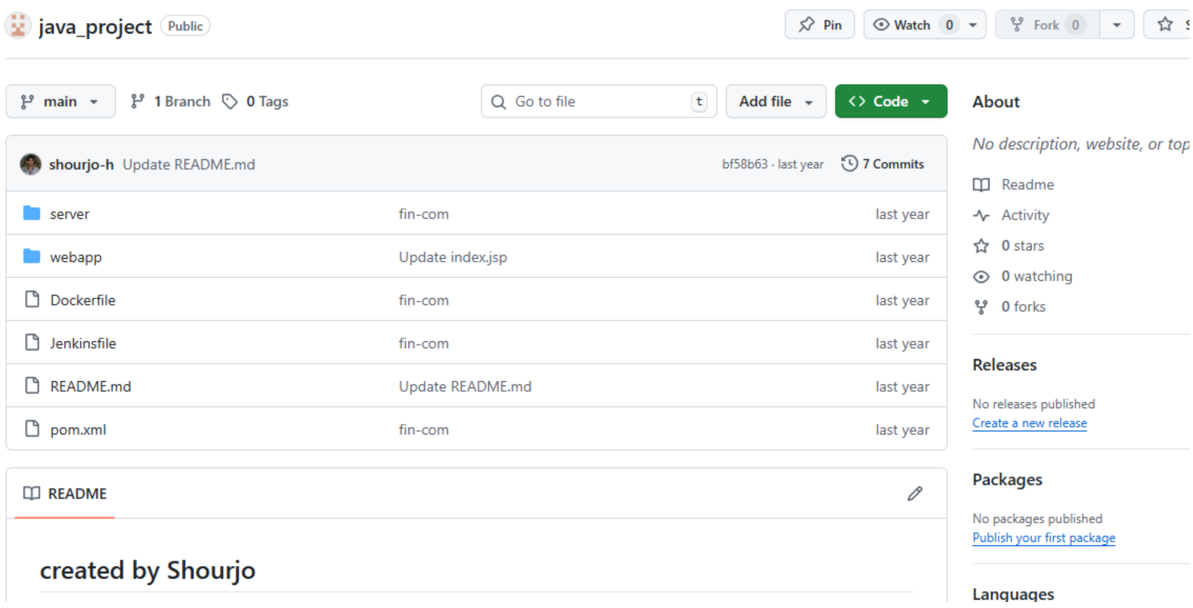
Check out our guide to connecting to GitHub using SSH keys or troubleshoot common SSH problems.

Created a repository name java_project and pushed it to the GitHub.



```
[root@docker shourjo-10743365]# git remote add origin git@github.com:tonybabu2004-eng/java_project.git
[root@docker shourjo-10743365]# git branch -M main
[root@docker shourjo-10743365]# git push -u origin main
he authenticity of host 'github.com (140.82.114.4)' can't be established.
D25519 key fingerprint is SHA256:+DiY3wvvV6TuJJhbpZisF/zLDA0zPMSvHdkr4UvCOqU.
his key is not known by any other names
re you sure you want to continue connecting (yes/no/[fingerprint])? yes
arning: Permanently added 'github.com' (ED25519) to the list of known hosts.
numerating objects: 56, done.
ounting objects: 100% (56/56), done.
elta compression using up to 2 threads
ompressing objects: 100% (24/24), done.
riting objects: 100% (56/56), 11.40 KiB | 11.40 MiB/s, done.
otal 56 (delta 9), reused 56 (delta 9), pack-reused 0 (from 0)
emote: Resolving deltas: 100% (9/9), done.
o github.com:tonybabu2004-eng/java_project.git
 * [new branch]      main -> main
ranch 'main' set up to track 'origin/main'.
```

Added in repository in GitHub



java_project Public

Pin | Watch 0 | Fork 0

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

shourjo-h Update README.md — bf58b63 · last year — 7 Commits

server	fin-com	last year
webapp	Update index.jsp	last year
Dockerfile	fin-com	last year
Jenkinsfile	fin-com	last year
README.md	Update README.md	last year
pom.xml	fin-com	last year

README

created by Shourjo

About

No description, website, or top

Readme
Activity
0 stars
0 watching
0 forks

Releases

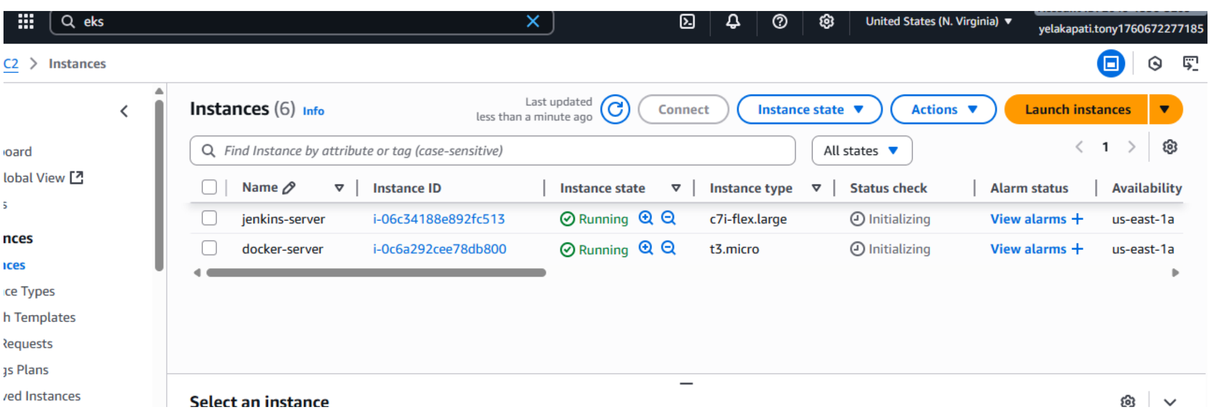
No releases published
Create a new release

Packages

No packages published
Publish your first package

Languages

Created Jenkins and Docker Instance



eks — United States (N. Virginia) — yelakapati.tony1760672277185

C2 > Instances

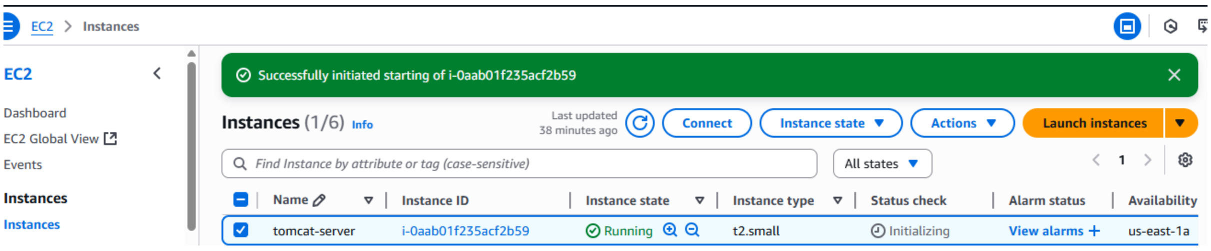
Instances (6) Info — Last updated less than a minute ago — Connect | Instance state | Actions | Launch instances

Find Instance by attribute or tag (case-sensitive) — All states

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability
jenkins-server	i-06c34188e892fc513	Running	c7i-flex.large	Initializing	View alarms +	us-east-1a
docker-server	i-0c6a292cee78db800	Running	t3.micro	Initializing	View alarms +	us-east-1a

Select an instance

Creating tomcat instance



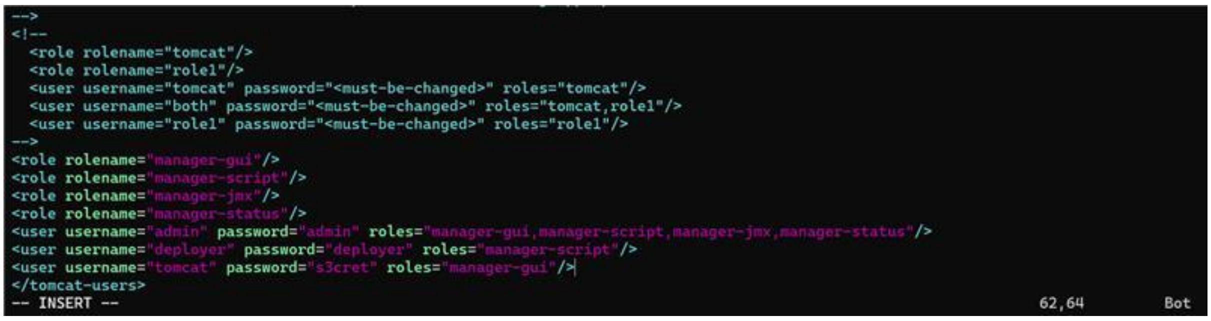
Successfully initiated starting of i-0aab01f235acf2b59

Instances (1/6) Info Last updated 38 minutes ago Connect Instance state Actions Launch instances

Find Instance by attribute or tag (case-sensitive) All states

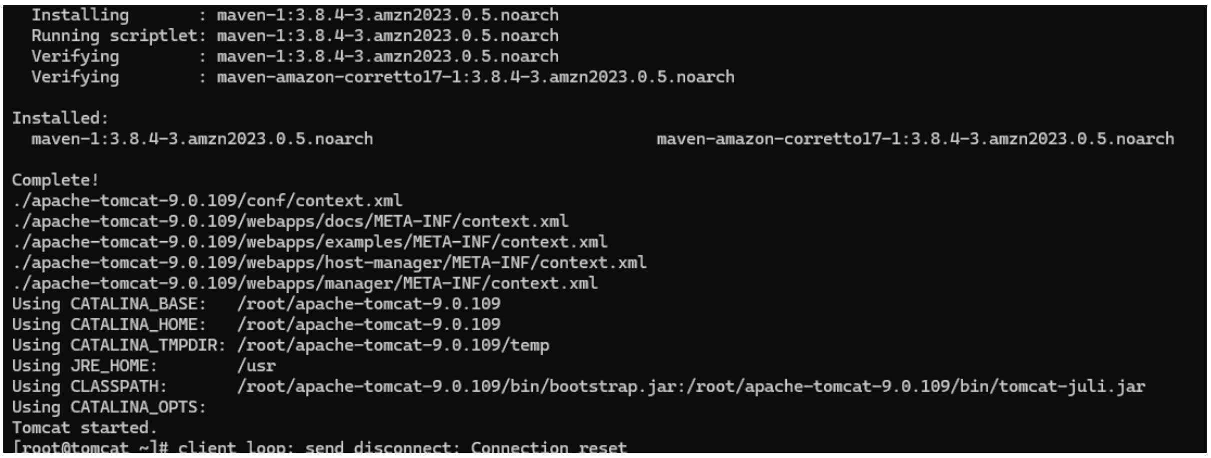
Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability
tomcat-server	i-0aab01f235acf2b59	Running	t2.small	Initializing	View alarms +	us-east-1a

Attaching roles in tomcat-users.xml in tomcat instance



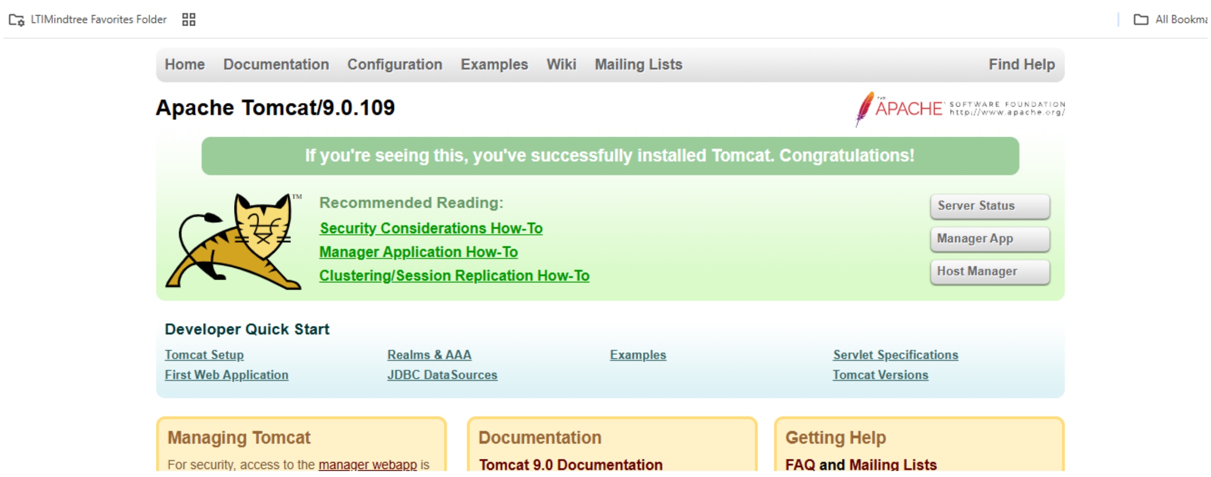
```
-->
<!--
  <role rolename="tomcat"/>
  <role rolename="role1"/>
  <user username="tomcat" password="<must-be-changed>" roles="tomcat"/>
  <user username="both" password="<must-be-changed>" roles="tomcat,role1"/>
  <user username="role1" password="<must-be-changed>" roles="role1"/>
-->
<role rolename="manager-gui"/>
<role rolename="manager-script"/>
<role rolename="manager-jmx"/>
<role rolename="manager-status"/>
<user username="admin" password="admin" roles="manager-gui,manager-script,manager-jmx,manager-status"/>
<user username="deployer" password="deployer" roles="manager-script"/>
<user username="tomcat" password="s3cret" roles="manager-gui"/>
</tomcat-users>
-- INSERT --                                                                      62,64          Bot
```

starting tomcat server



```
  Installing        : maven-1:3.8.4-3.amzn2023.0.5.noarch
  Running scriptlet: maven-1:3.8.4-3.amzn2023.0.5.noarch
  Verifying         : maven-1:3.8.4-3.amzn2023.0.5.noarch
  Verifying         : maven-amazon-corretto17-1:3.8.4-3.amzn2023.0.5.noarch

Installed:
  maven-1:3.8.4-3.amzn2023.0.5.noarch                            maven-amazon-corretto17-1:3.8.4-3.amzn2023.0.5.noarch

Complete!
./apache-tomcat-9.0.109/conf/context.xml
./apache-tomcat-9.0.109/webapps/docs/META-INF/context.xml
./apache-tomcat-9.0.109/webapps/examples/META-INF/context.xml
./apache-tomcat-9.0.109/webapps/host-manager/META-INF/context.xml
./apache-tomcat-9.0.109/webapps/manager/META-INF/context.xml
Using CATALINA_BASE:   /root/apache-tomcat-9.0.109
Using CATALINA_HOME:   /root/apache-tomcat-9.0.109
Using CATALINA_TMPDIR: /root/apache-tomcat-9.0.109/temp
Using JRE_HOME:        /usr
Using CLASSPATH:       /root/apache-tomcat-9.0.109/bin/bootstrap.jar:/root/apache-tomcat-9.0.109/bin/tomcat-juli.jar
Using CATALINA_OPTS:
Tomcat started.
[root@tomcat ~]# client_loop: send disconnect: Connection reset
```

Paste tomcat public ip in browser



Apache Tomcat/9.0.109

If you're seeing this, you've successfully installed Tomcat. Congratulations!

Recommended Reading:
Security Considerations How-To
Manager Application How-To
Clustering/Session Replication How-To

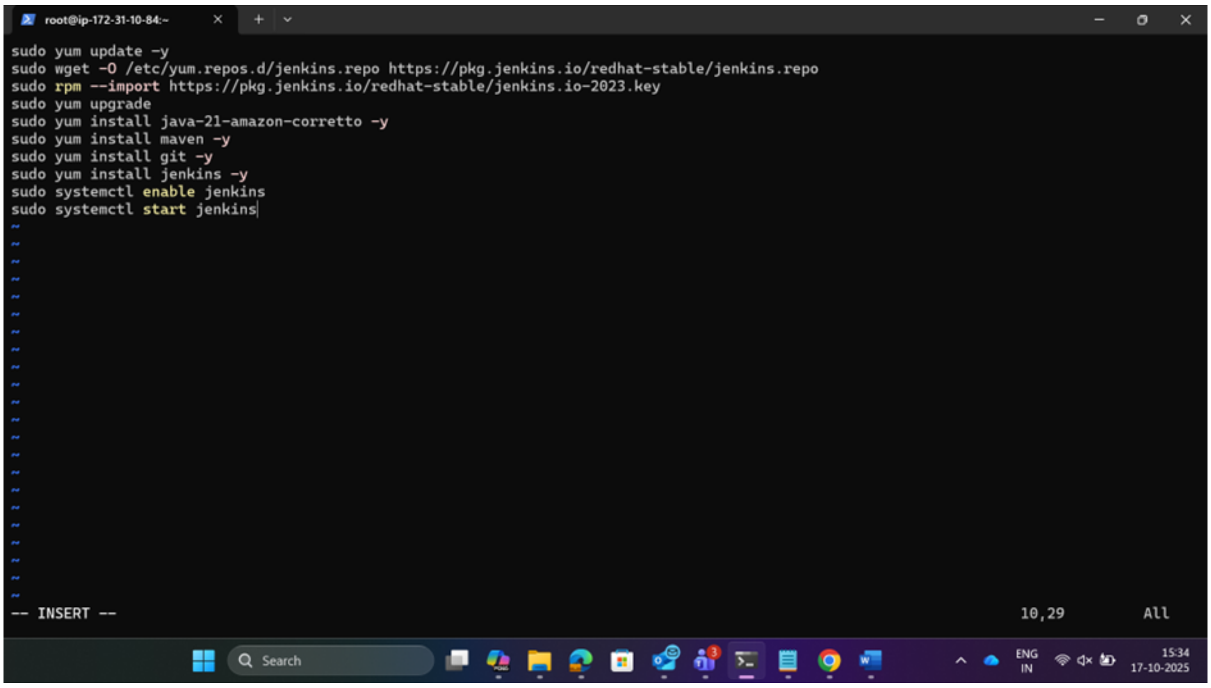
Developer Quick Start
Tomcat Setup
First Web Application
Realms & AAA
JDBC DataSources
Examples
Servlet Specifications
Tomcat Versions

Managing Tomcat
For security, access to the manager webapp is

Documentation
Tomcat 9.0 Documentation

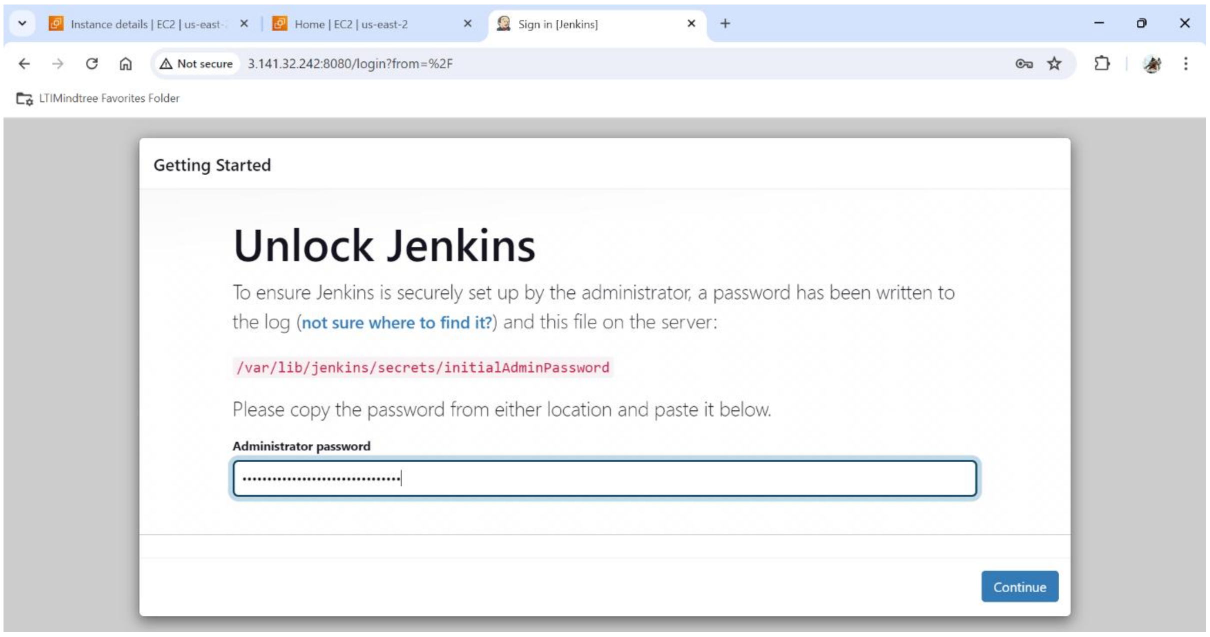
Getting Help
FAQ and Mailing Lists

Shell script for installing Jenkins



```
sudo yum update -y
sudo wget -O /etc/yum.repos.d/jenkins.repo https://pkg.jenkins.io/redhat-stable/jenkins.repo
sudo rpm --import https://pkg.jenkins.io/redhat-stable/jenkins.io-2023.key
sudo yum upgrade
sudo yum install java-21-amazon-corretto -y
sudo yum install maven -y
sudo yum install git -y
sudo yum install jenkins -y
sudo systemctl enable jenkins
sudo systemctl start jenkins
~
-- INSERT --                                                                          10,29         All
```

Java, Git, Maven, Jenkins is installed in Jenkins instance and started Jenkins


Instance details | EC2 | us-east-
Home | EC2 | us-east-2
Sign in [Jenkins]
Not secure
3.141.32.242:8080/login?from=%2F
LTIMindtree Favorites Folder

Getting Started

Unlock Jenkins

To ensure Jenkins is securely set up by the administrator, a password has been written to the log (not sure where to find it?) and this file on the server:

`/var/lib/jenkins/secrets/initialAdminPassword`

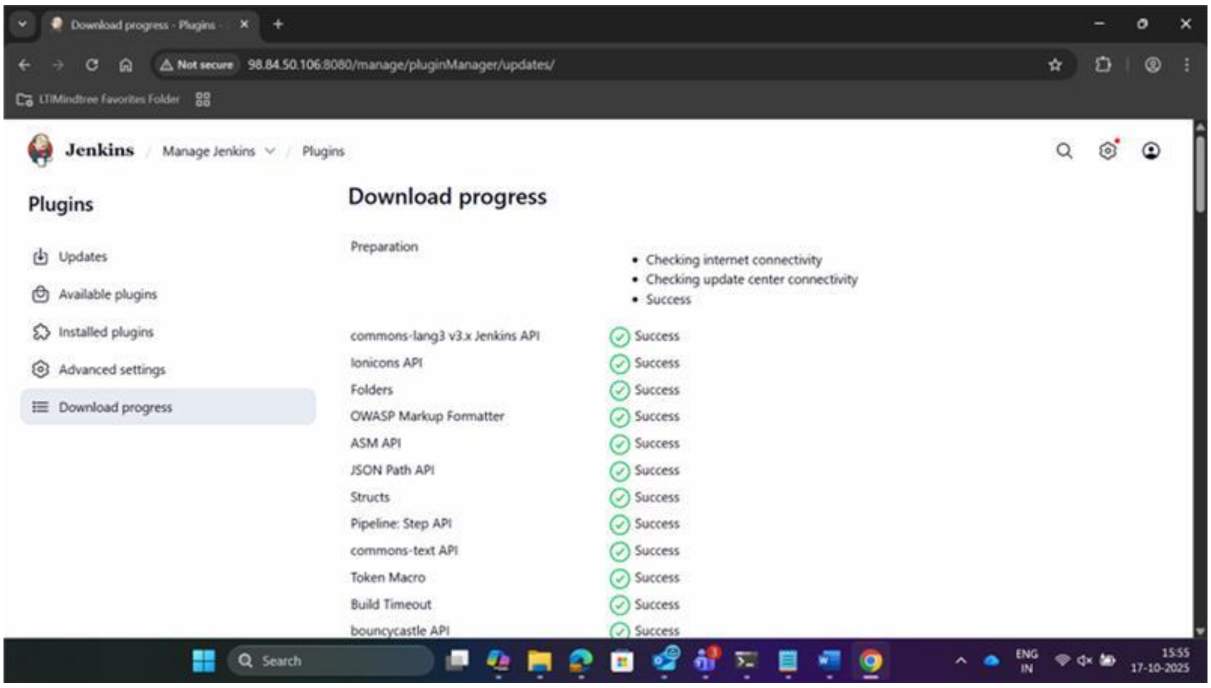
Please copy the password from either location and paste it below.

Administrator password

································

Continue

Installing plugins

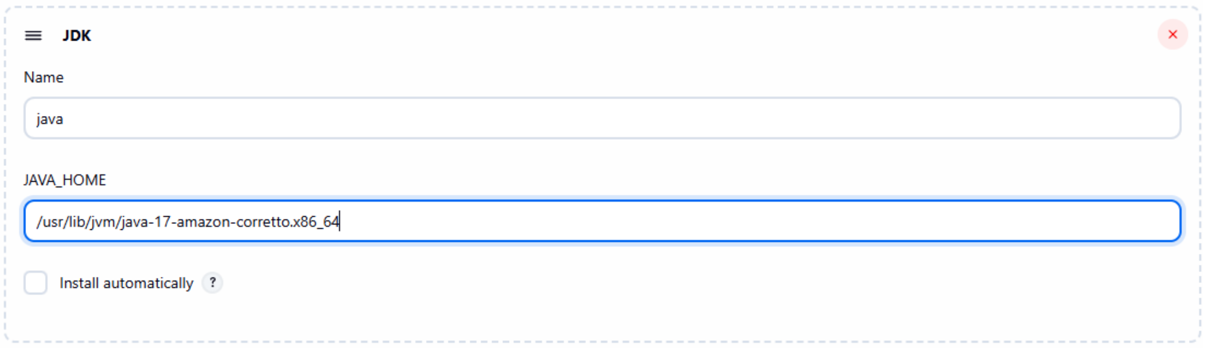


Download progress

Preparation
- Checking internet connectivity
- Checking update center connectivity
- Success

commons-lang3 v3.x Jenkins API	Success
Ionicons API	Success
Folders	Success
OWASP Markup Formatter	Success
ASM API	Success
JSON Path API	Success
Structs	Success
Pipeline: Step API	Success
commons-text API	Success
Token Macro	Success
Build Timeout	Success
bouncycastle API	Success

Configuring Java path in tools in Jenkins



Name

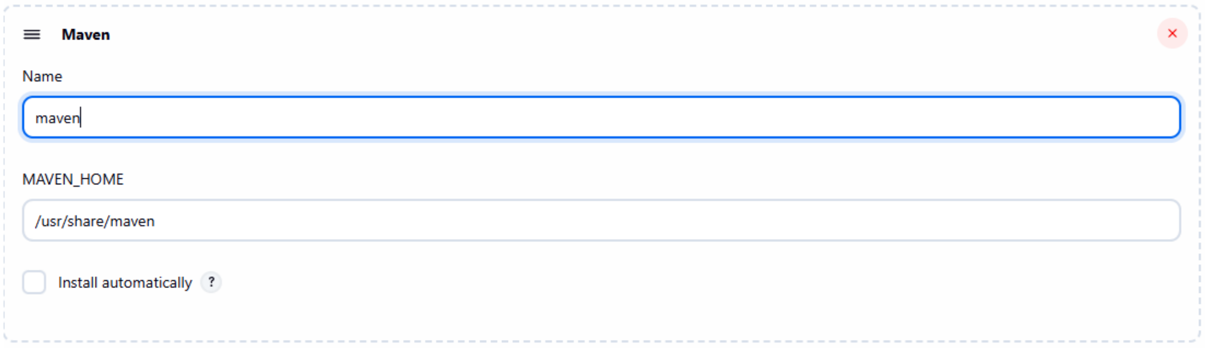
java

JAVA_HOME

/usr/lib/jvm/java-17-amazon-corretto.x86_64

Install automatically ?

Configuring Maven path in tools in Jenkins



Name

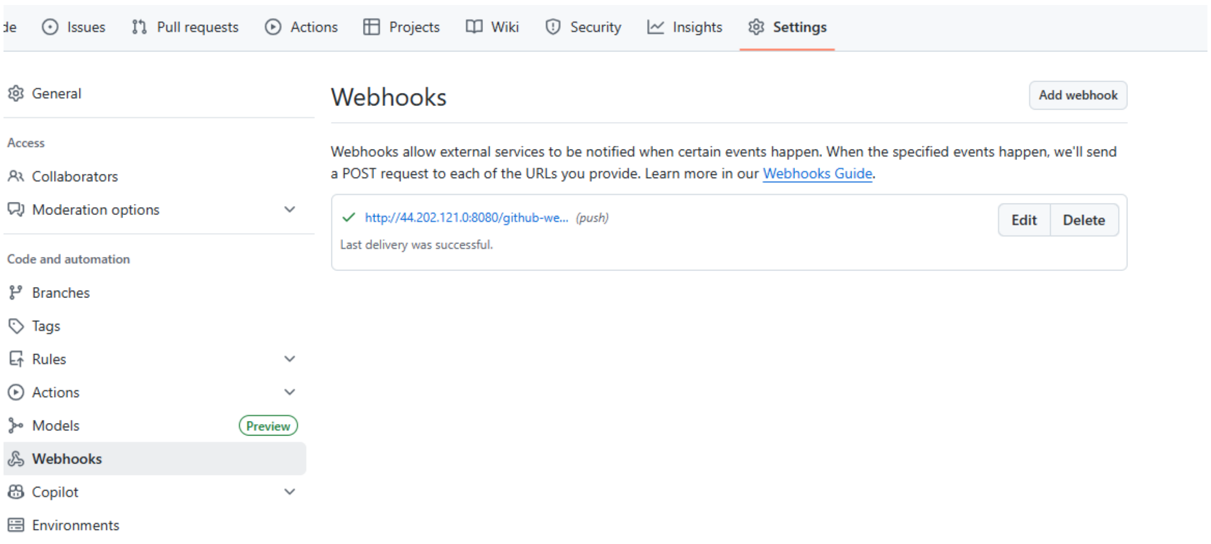
maven

MAVEN_HOME

/usr/share/maven

Install automatically ?

Adding webhook to my GitHub repository to establish connection with Jenkins



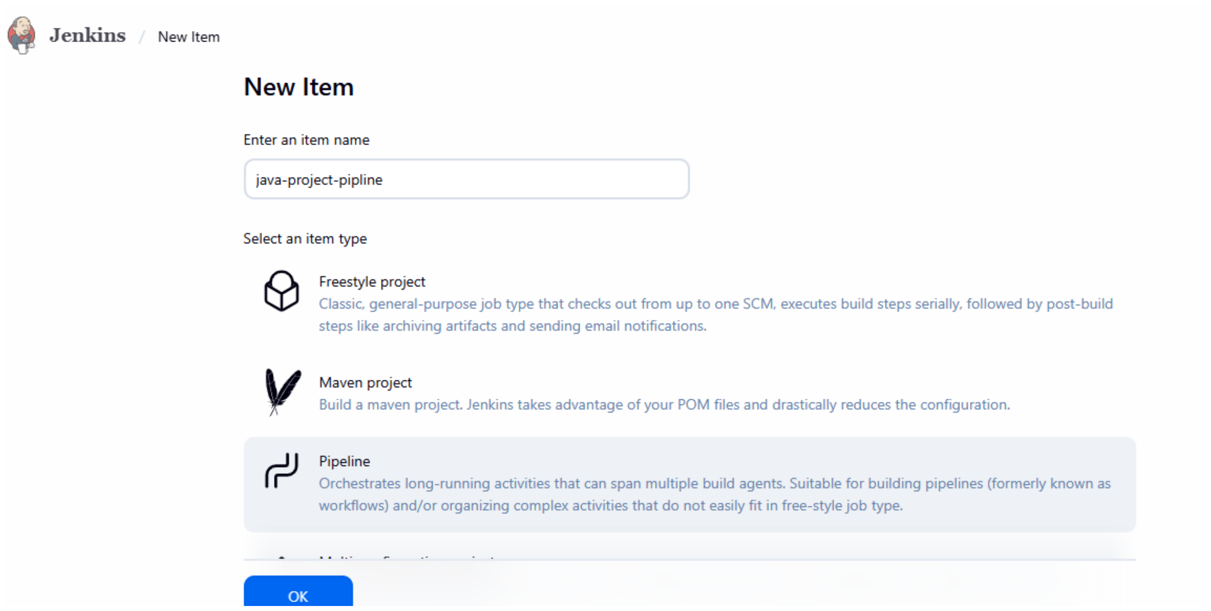
Webhooks

Webhooks allow external services to be notified when certain events happen. When the specified events happen, we'll send a POST request to each of the URLs you provide. Learn more in our Webhooks Guide.

✓ http://44.202.121.0:8080/github-we... *(push)*

Last delivery was successful.

Creating a new pipeline Job for maven project



New Item

Enter an item name

java-project-pipline

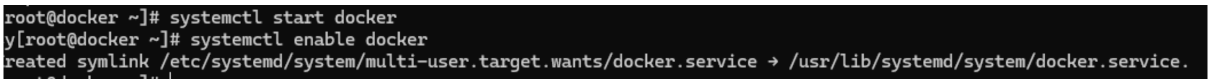
Select an item type

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.

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

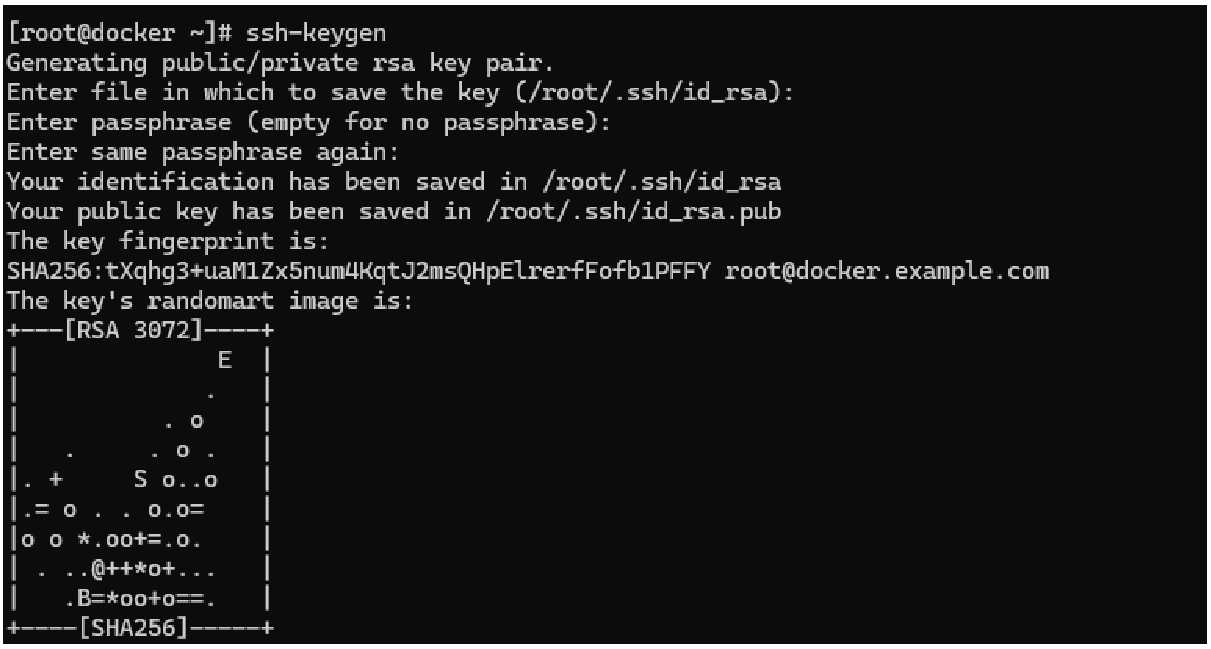
Pipeline
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.

Now in Docker instance install Docker and start Docker



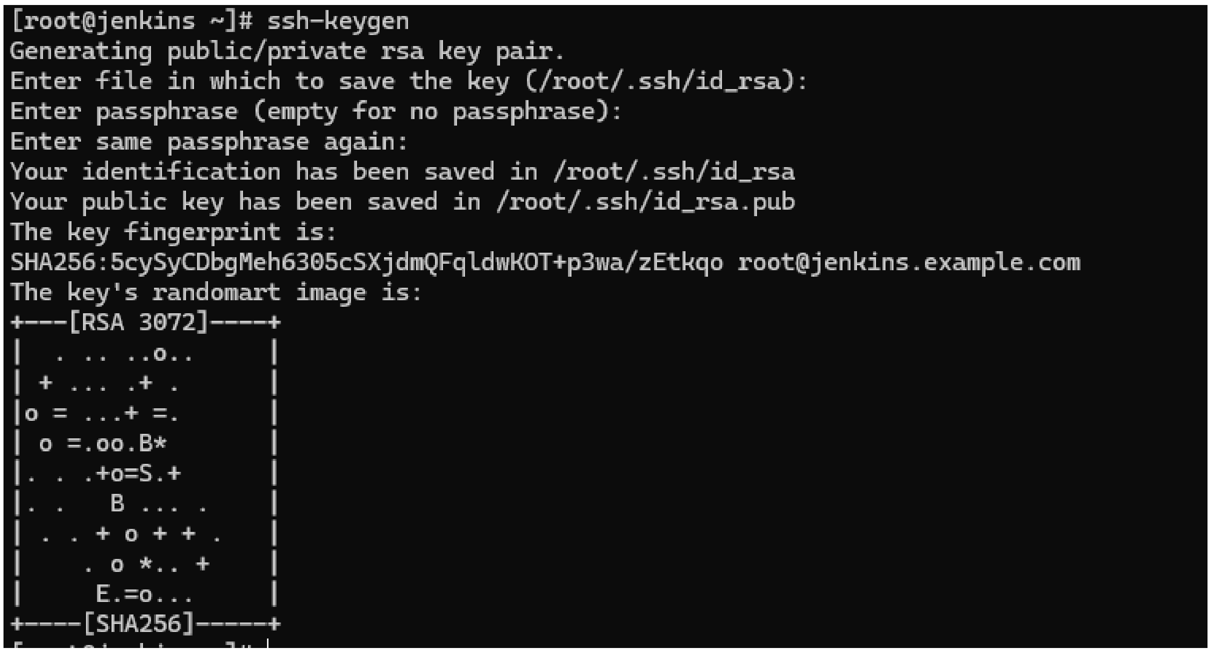
```
root@docker ~]# systemctl start docker
y[root@docker ~]# systemctl enable docker
reated symlink /etc/systemd/system/multi-user.target.wants/docker.service → /usr/lib/systemd/system/docker.service.
```

Generate SSH-KEY in docker



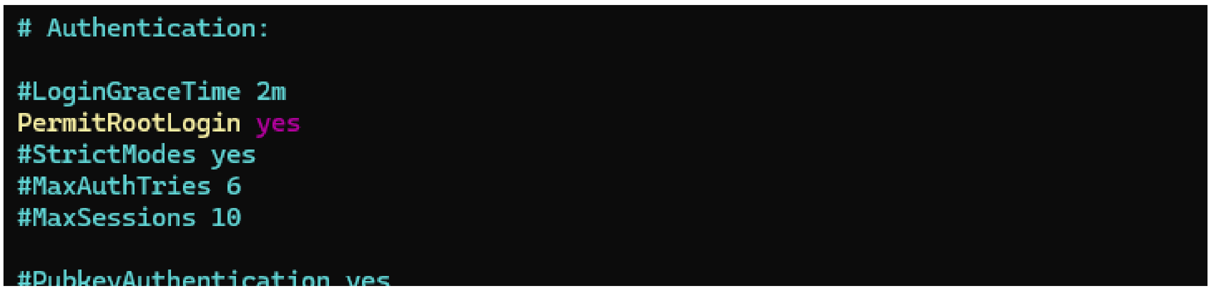
```
[root@docker ~]# ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/root/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /root/.ssh/id_rsa
Your public key has been saved in /root/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:tXqhg3+uaM1Zx5num4KqtJ2msQHpElrerfFofb1PFFY root@docker.example.com
The key's randomart image is:
+---[RSA 3072]----+
|              E  |
|             .   |
|          . o    |
|   .     . o .   |
|. +     S o..o   |
|.= o . . o.o=    |
|o o *.oo+=.o.    |
| .  ..@++*o+...  |
|    .B=*oo+o==.  |
+----[SHA256]-----+
```

Generate SSH-KEY in Jenkins

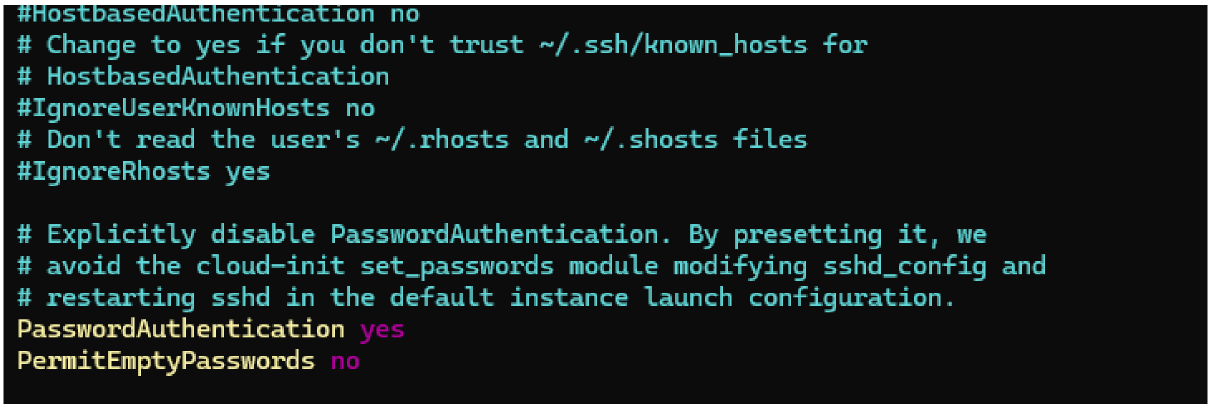


```
[root@jenkins ~]# ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/root/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /root/.ssh/id_rsa
Your public key has been saved in /root/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:5cySyCDbgMeh6305cSXjdmQFqldwKOT+p3wa/zEtkqo root@jenkins.example.com
The key's randomart image is:
+---[RSA 3072]----+
|  . .. ..o..     |
| + ... .+ .      |
|o = ...+ =.      |
| o =.oo.B*       |
|. .  .+o=S.+     |
|. .   B ... .    |
| . . + o + + .   |
|    . o *.. +    |
|     E.=o...     |
+----[SHA256]-----+
```

In both Jenkins and Docker go to /etc/ssh/sshd_config file and enable Password authentication and permit root login

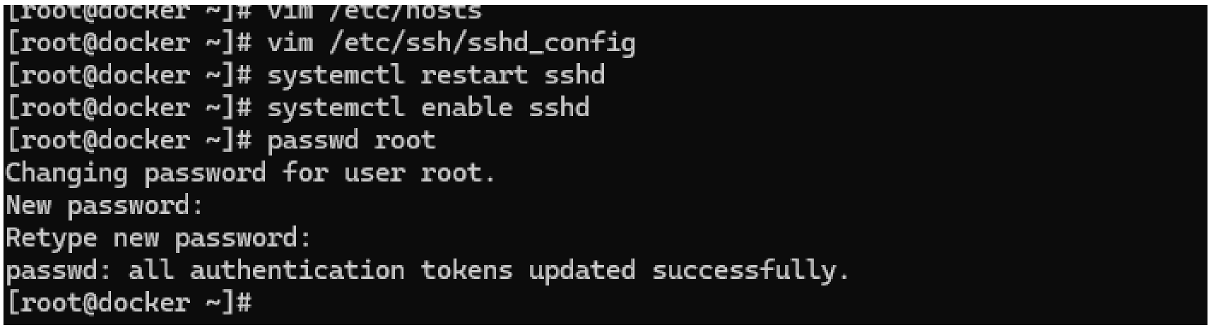


```
# Authentication:

#LoginGraceTime 2m
PermitRootLogin yes
#StrictModes yes
#MaxAuthTries 6
#MaxSessions 10

#PubkeyAuthentication yes
```

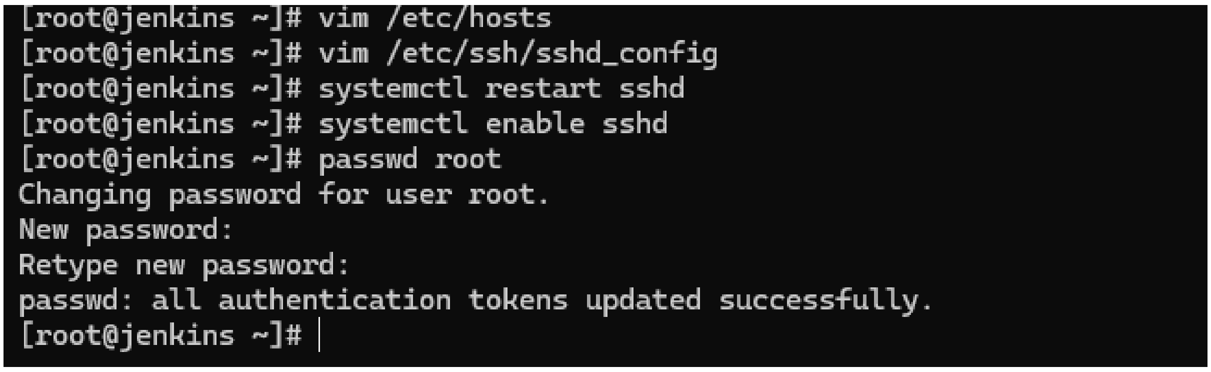


```
#HostbasedAuthentication no
# Change to yes if you don't trust ~/.ssh/known_hosts for
# HostbasedAuthentication
#IgnoreUserKnownHosts no
# Don't read the user's ~/.rhosts and ~/.shosts files
#IgnoreRhosts yes

# Explicitly disable PasswordAuthentication. By presetting it, we
# avoid the cloud-init set_passwords module modifying sshd_config and
# restarting sshd in the default instance launch configuration.
PasswordAuthentication yes
PermitEmptyPasswords no
```

Now restart sshd and give password in both Docker and Jenkins

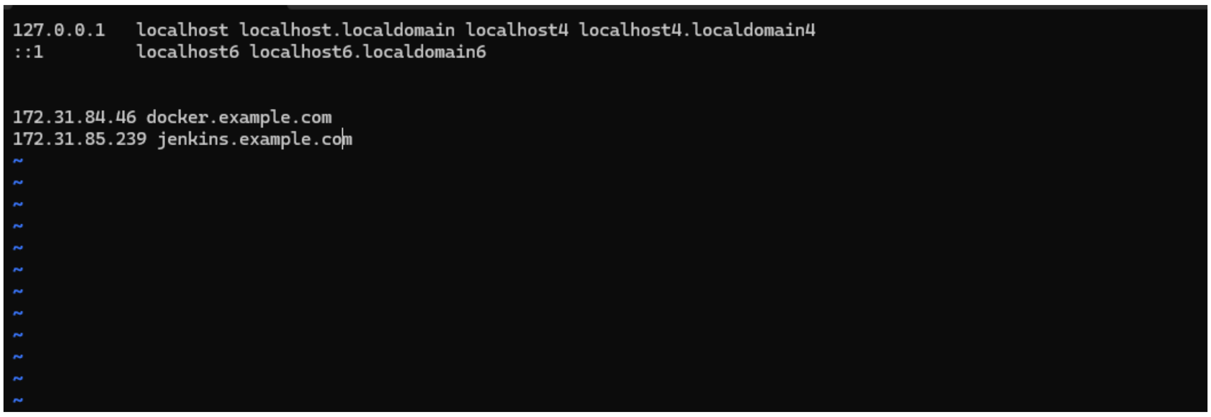


```
[root@docker ~]# vim /etc/hosts
[root@docker ~]# vim /etc/ssh/sshd_config
[root@docker ~]# systemctl restart sshd
[root@docker ~]# systemctl enable sshd
[root@docker ~]# passwd root
Changing password for user root.
New password:
Retype new password:
passwd: all authentication tokens updated successfully.
[root@docker ~]#
```



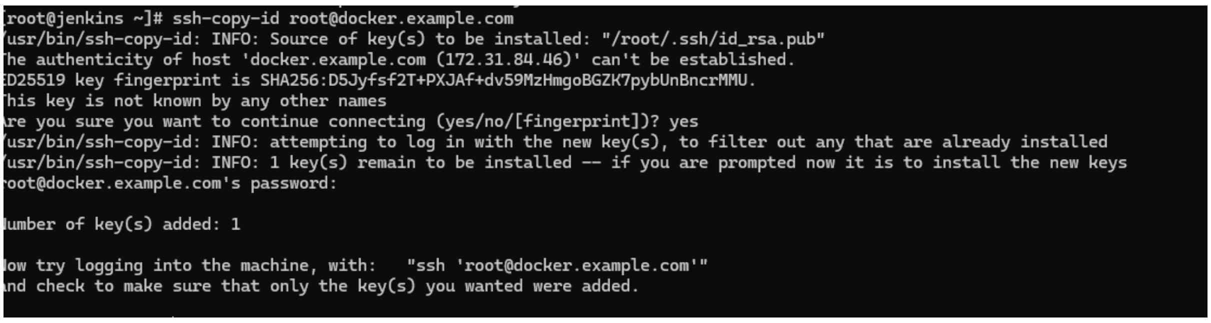
```
[root@jenkins ~]# vim /etc/hosts
[root@jenkins ~]# vim /etc/ssh/sshd_config
[root@jenkins ~]# systemctl restart sshd
[root@jenkins ~]# systemctl enable sshd
[root@jenkins ~]# passwd root
Changing password for user root.
New password:
Retype new password:
passwd: all authentication tokens updated successfully.
[root@jenkins ~]#
```

In both Docker and Jenkins, add private ip address and hostname of both docker and jenkins in /etc/hosts in both instances



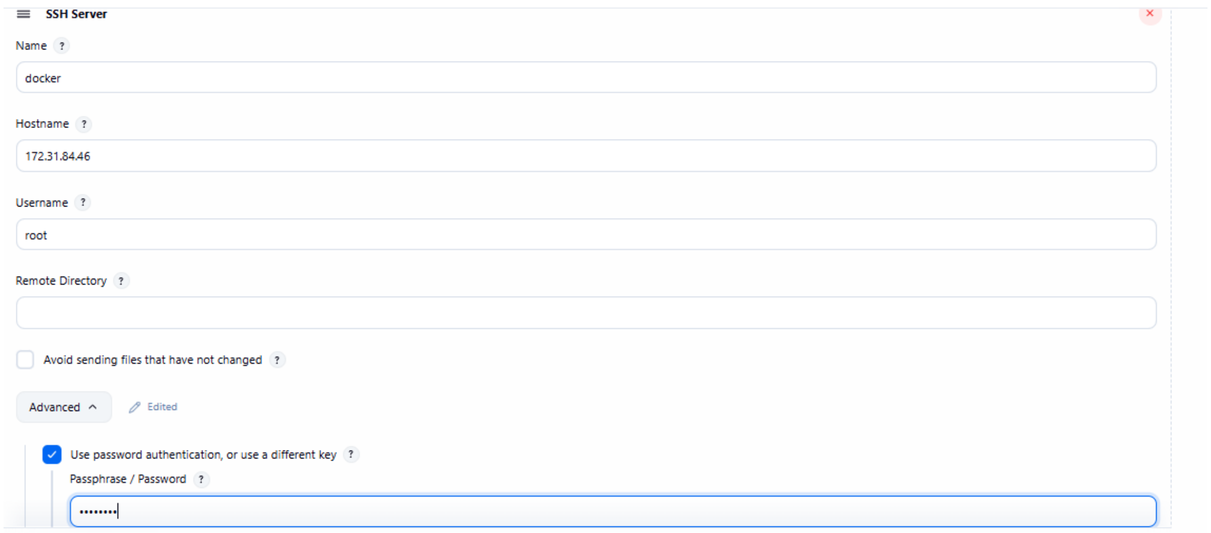
```
127.0.0.1   localhost localhost.localdomain localhost4 localhost4.localdomain4
::1         localhost6 localhost6.localdomain6


172.31.84.46 docker.example.com
172.31.85.239 jenkins.example.com
~
~
~
~
~
~
~
~
~
~
~
```

Copy the ssh key from Jenkins to docker



```
[root@jenkins ~]# ssh-copy-id root@docker.example.com
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/root/.ssh/id_rsa.pub"
The authenticity of host 'docker.example.com (172.31.84.46)' can't be established.
ED25519 key fingerprint is SHA256:D5Jyfsf2T+PXJAf+dv59MzHmgoBGZK7pybUnBncrMMU.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter out any that are already installed
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are prompted now it is to install the new keys
root@docker.example.com's password:

Number of key(s) added: 1

Now try logging into the machine, with:   "ssh 'root@docker.example.com'"
and check to make sure that only the key(s) you wanted were added.
```

In Jenkins go to system, in that SSH Server add docker private ip and password



SSH Server

Name: docker

Hostname: 172.31.84.46

Username: root

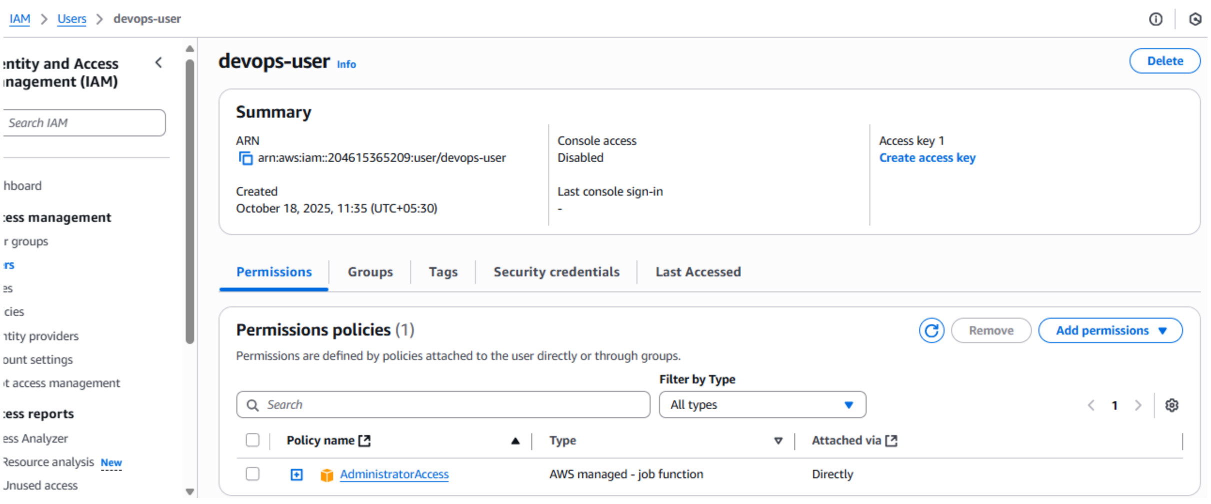
Remote Directory:

☐ Avoid sending files that have not changed

☑ Use password authentication, or use a different key

Passphrase / Password: ••••••••

Now create an IAM user and allow administrator access



IAM > Users > devops-user

ntity and Access
nagement (IAM)

Search IAM

hboard
ess management
r groups
rs
es
cies
tity providers
ount settings
t access management
ess reports
ess Analyzer
Resource analysis New
Jnused access

devops-user Info

Delete

Summary

ARN arn:aws:iam::204615365209:user/devops-user	Console access Disabled	Access key 1 Create access key
Created October 18, 2025, 11:35 (UTC+05:30)	Last console sign-in -	

Permissions | Groups | Tags | Security credentials | Last Accessed

Permissions policies (1)

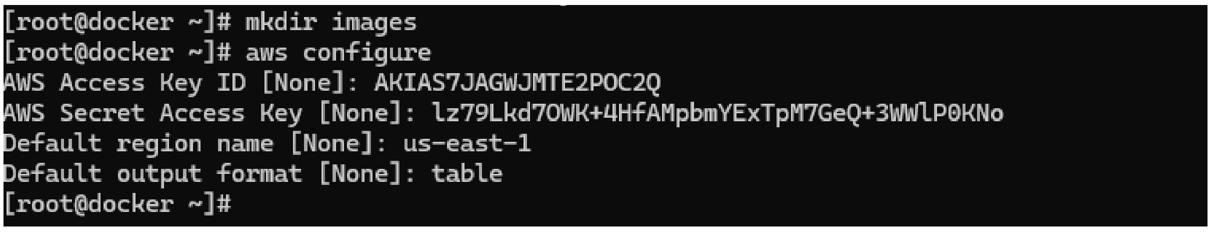
Remove | Add permissions

Permissions are defined by policies attached to the user directly or through groups.

Filter by Type: All types

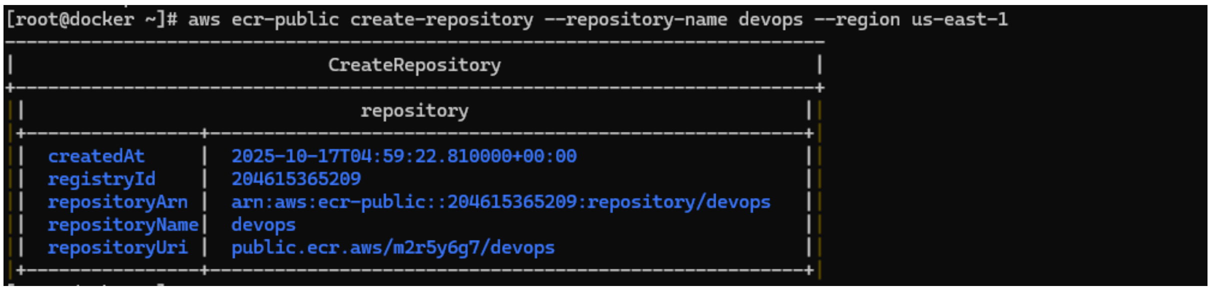
Policy name	Type	Attached via
AdministratorAccess	AWS managed - job function	Directly

Create credentials like access key and secret key for that IAM user and give them in docker instance for accessing of ECR



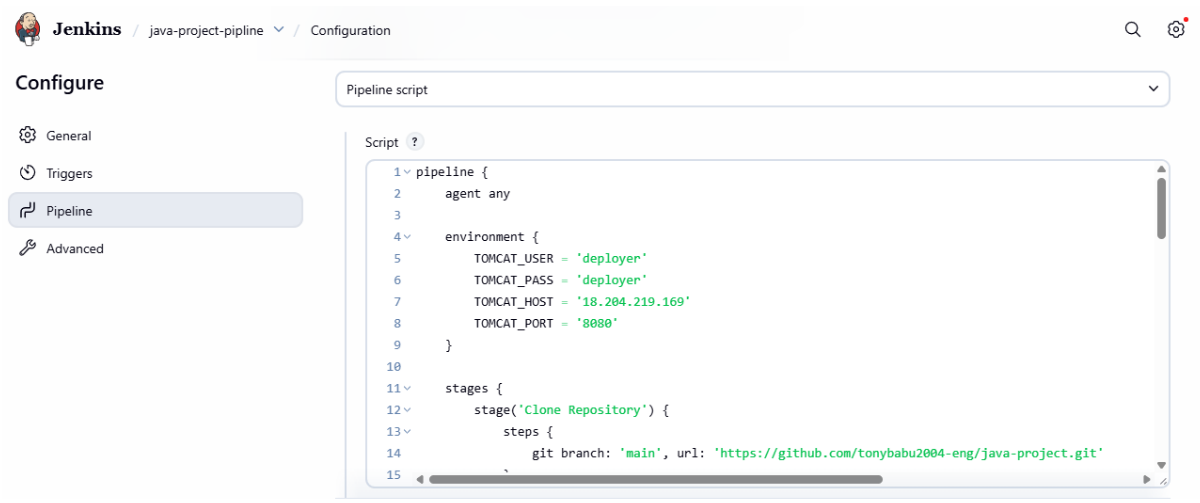
```
[root@docker ~]# mkdir images
[root@docker ~]# aws configure
AWS Access Key ID [None]: AKIAS7JAGWJMTE2POC2Q
AWS Secret Access Key [None]: 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
Default region name [None]: us-east-1
Default output format [None]: table
[root@docker ~]#
```

Now create public repository in ECR for storing images



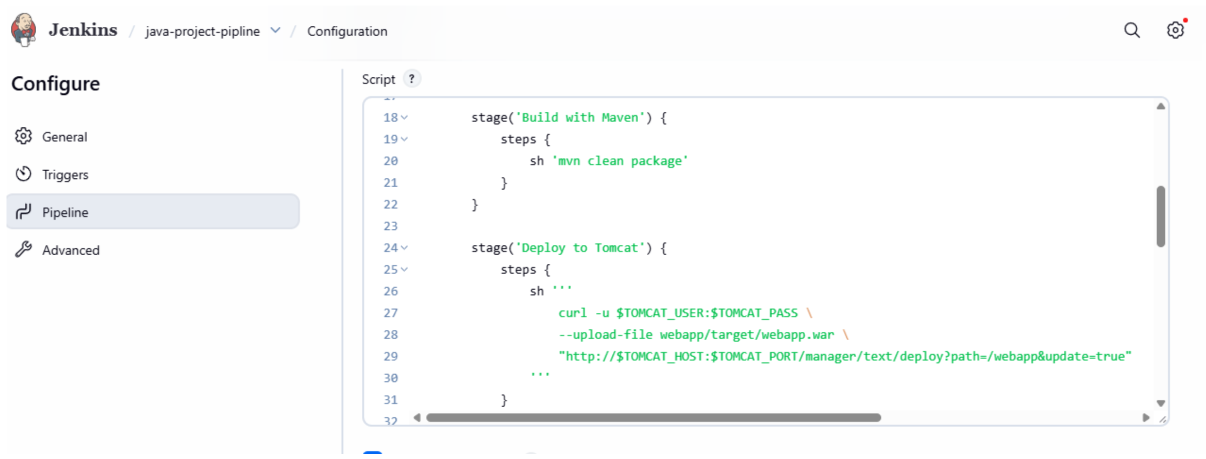
```
[root@docker ~]# aws ecr-public create-repository --repository-name devops --region us-east-1
------------------------------------------------------------------------------
|                              CreateRepository                              |
+----------------------------------------------------------------------------+
||                                repository                                ||
|+----------------+---------------------------------------------------------+|
||  createdAt     |  2025-10-17T04:59:22.810000+00:00                        ||
||  registryId    |  204615365209                                            ||
||  repositoryArn |  arn:aws:ecr-public::204615365209:repository/devops      ||
||  repositoryName|  devops                                                  ||
||  repositoryUri |  public.ecr.aws/m2r5y6g7/devops                          ||
|+----------------+---------------------------------------------------------+|
```

Now in Jenkins pipeline, writing stages in those steps in Jenkins file for cloning repository



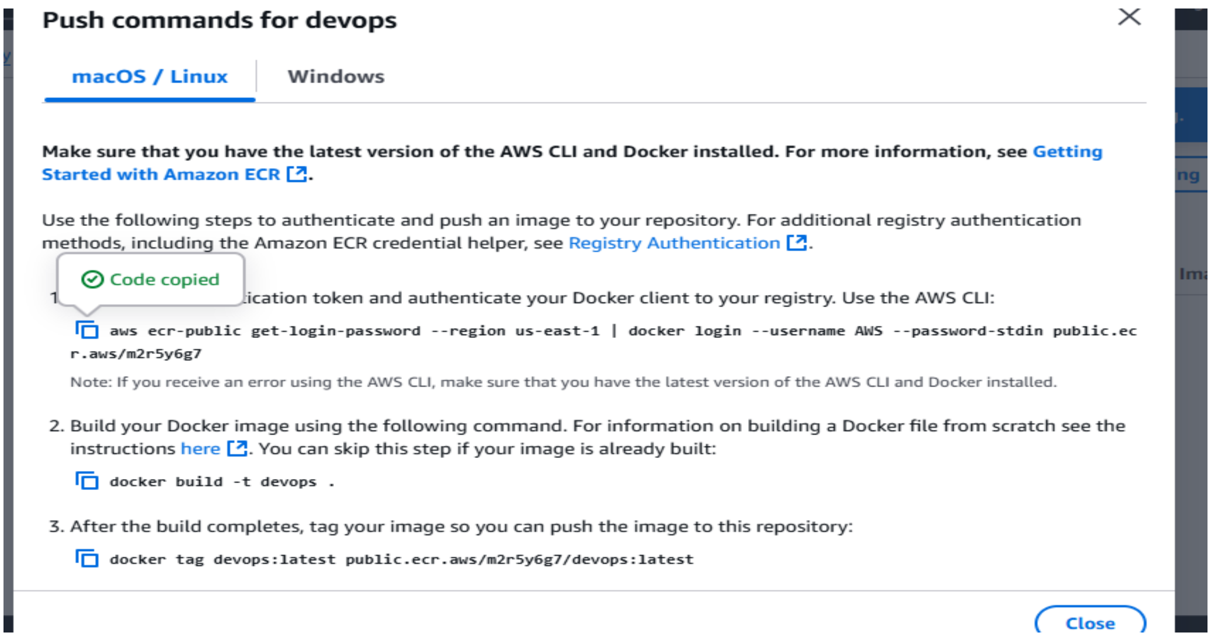
```
pipeline {
    agent any

    environment {
        TOMCAT_USER = 'deployer'
        TOMCAT_PASS = 'deployer'
        TOMCAT_HOST = '18.204.219.169'
        TOMCAT_PORT = '8080'
    }

    stages {
        stage('Clone Repository') {
            steps {
                git branch: 'main', url: 'https://github.com/tonybabu2004-eng/java-project.git'
```

Writing Jenkins file for building stage using maven and deploying stage war files to tomcat server



```
        stage('Build with Maven') {
            steps {
                sh 'mvn clean package'
            }
        }

        stage('Deploy to Tomcat') {
            steps {
                sh '''
                    curl -u $TOMCAT_USER:$TOMCAT_PASS \
                    --upload-file webapp/target/webapp.war \
                    "http://$TOMCAT_HOST:$TOMCAT_PORT/manager/text/deploy?path=/webapp&update=true"
                '''
            }
```

Now writing a stage in Jenkins file to transfer java project folder into images folder in docker and also executing commands that create latest image and push into public repository in ECR



Push commands for devops

macOS / Linux | Windows

Make sure that you have the latest version of the AWS CLI and Docker installed. For more information, see Getting Started with Amazon ECR.

Use the following steps to authenticate and push an image to your repository. For additional registry authentication methods, including the Amazon ECR credential helper, see Registry Authentication.

Code copied

1. ication token and authenticate your Docker client to your registry. Use the AWS CLI:

```
aws ecr-public get-login-password --region us-east-1 | docker login --username AWS --password-stdin public.ec
r.aws/m2r5y6g7
```

Note: If you receive an error using the AWS CLI, make sure that you have the latest version of the AWS CLI and Docker installed.

2. Build your Docker image using the following command. For information on building a Docker file from scratch see the instructions here. You can skip this step if your image is already built:

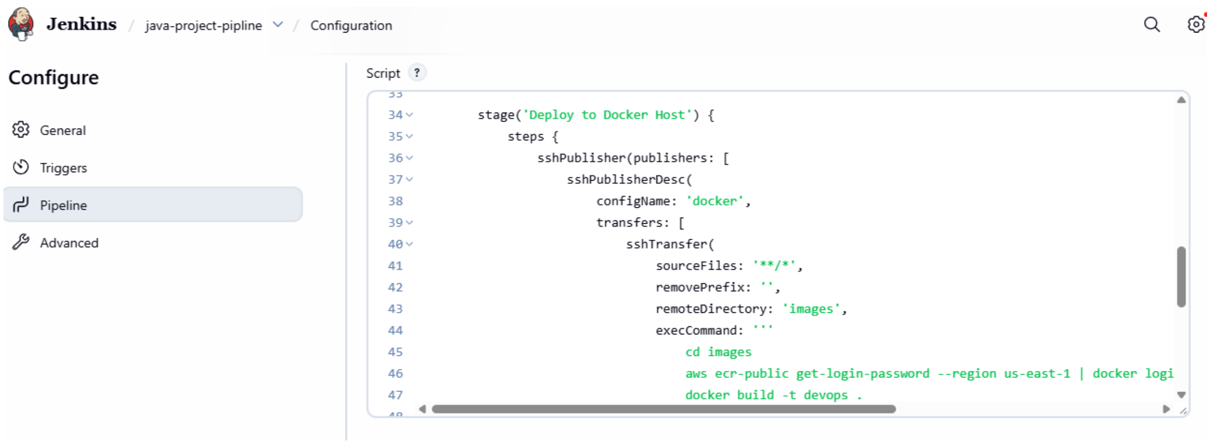
```
docker build -t devops .
```

3. After the build completes, tag your image so you can push the image to this repository:

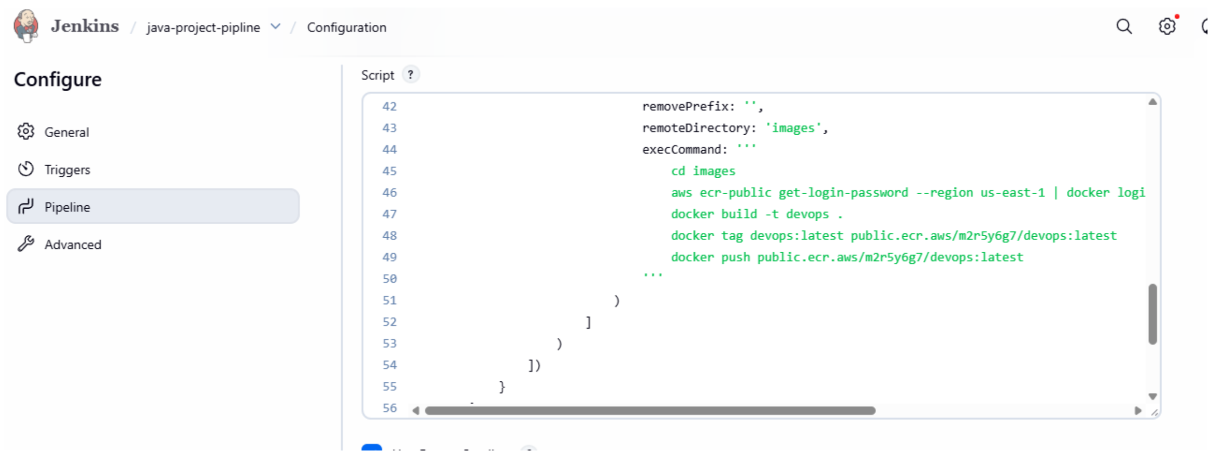
```
docker tag devops:latest public.ecr.aws/m2r5y6g7/devops:latest
```

Close

Pasting those push commands in the Jenkins file

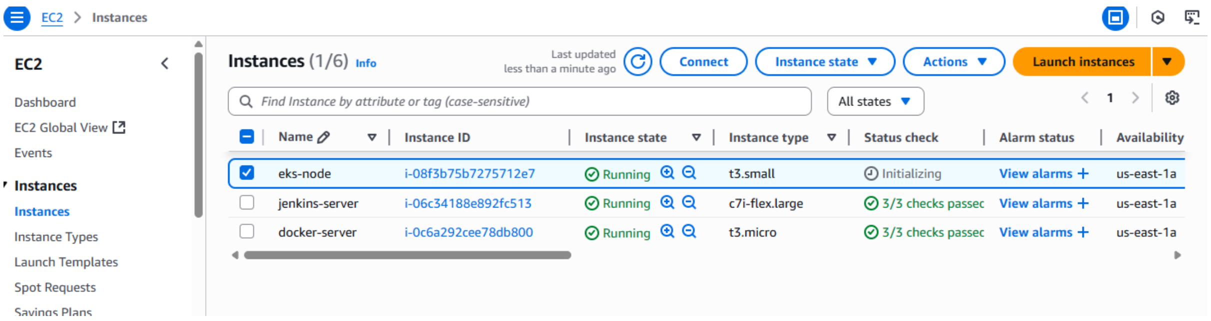


```
        stage('Deploy to Docker Host') {
            steps {
                sshPublisher(publishers: [
                    sshPublisherDesc(
                        configName: 'docker',
                        transfers: [
                            sshTransfer(
                                sourceFiles: '**/*',
                                removePrefix: '',
                                remoteDirectory: 'images',
                                execCommand: '''
                                    cd images
                                    aws ecr-public get-login-password --region us-east-1 | docker logi
                                    docker build -t devops .
```



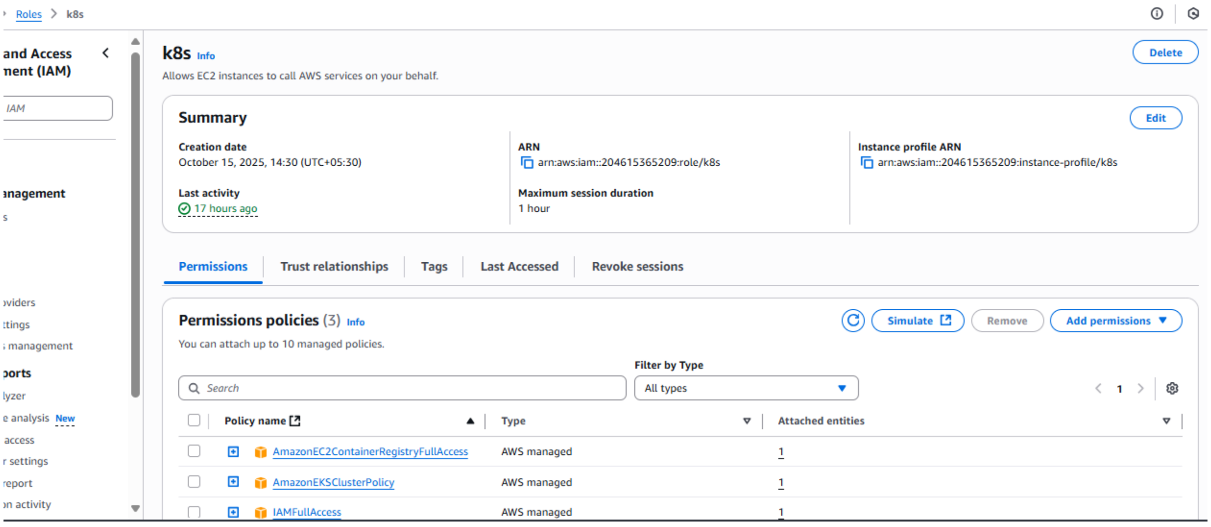
```
                                    removePrefix: '',
                                    remoteDirectory: 'images',
                                    execCommand: '''
                                        cd images
                                        aws ecr-public get-login-password --region us-east-1 | docker logi
                                        docker build -t devops .
                                        docker tag devops:latest public.ecr.aws/m2r5y6g7/devops:latest
                                        docker push public.ecr.aws/m2r5y6g7/devops:latest
                                    '''
                                )
                            ]
                        )
                    ])
                }
```

Creating an instance for EKS node for creation of EKS cluster



Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability
eks-node	i-08f3b75b7275712e7	Running	t3.small	Initializing	View alarms +	us-east-1a
jenkins-server	i-06c34188e892fc513	Running	c7i-flex.large	3/3 checks passec	View alarms +	us-east-1a
docker-server	i-0c6a292cee78db800	Running	t3.micro	3/3 checks passec	View alarms +	us-east-1a

Adding IAM role to EKS-node instance which has access to IAMfull access, EKS full access, ECR full access



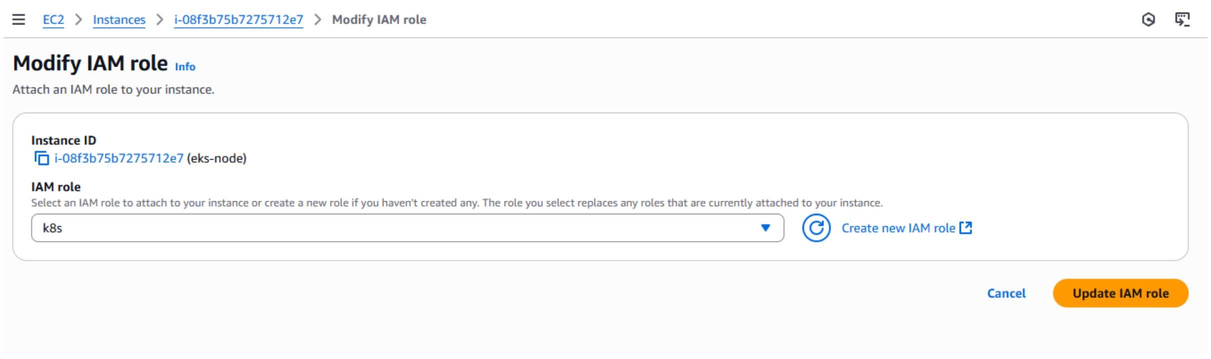
k8s Info
Allows EC2 instances to call AWS services on your behalf.

Summary

Creation date: October 15, 2025, 14:30 (UTC+05:30)
ARN: arn:aws:iam::204615365209:role/k8s
Instance profile ARN: arn:aws:iam::204615365209:instance-profile/k8s
Last activity: 17 hours ago
Maximum session duration: 1 hour

Permissions policies (3) Info
You can attach up to 10 managed policies.
Filter by Type: All types

Policy name	Type	Attached entities
AmazonEC2ContainerRegistryFullAccess	AWS managed	1
AmazonEKSClusterPolicy	AWS managed	1
IAMFullAccess	AWS managed	1



EC2 > Instances > i-08f3b75b7275712e7 > Modify IAM role

Modify IAM role

Attach an IAM role to your instance.

Instance ID

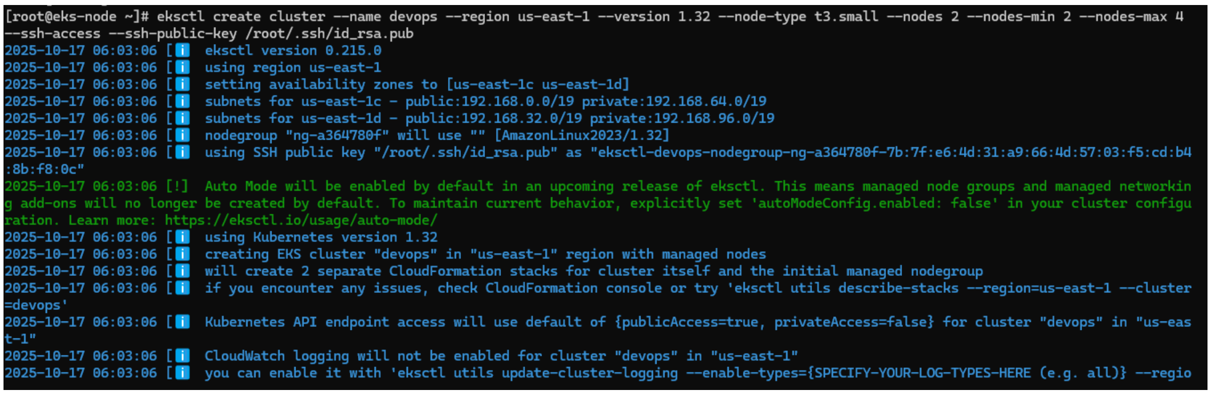
i-08f3b75b7275712e7 (eks-node)

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.

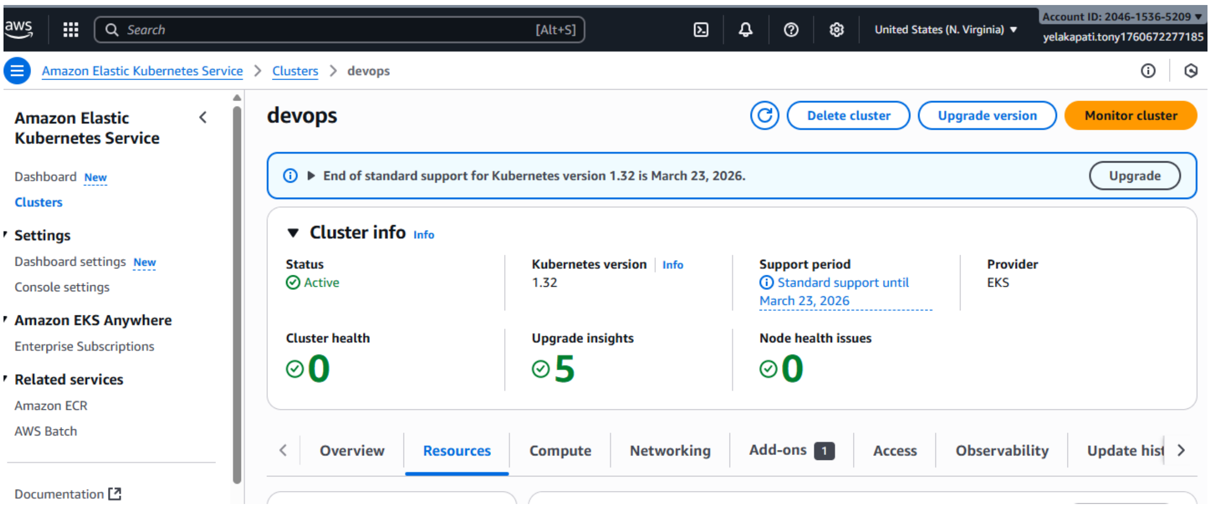
k8s

Install kubectl, eksctl, aws cli in eks-node instance and create cluster



```
[root@eks-node ~]# eksctl create cluster --name devops --region us-east-1 --version 1.32 --node-type t3.small --nodes 2 --nodes-min 2 --nodes-max 4
--ssh-access --ssh-public-key /root/.ssh/id_rsa.pub
2025-10-17 06:03:06 [ℹ  eksctl version 0.215.0
2025-10-17 06:03:06 [ℹ  using region us-east-1
2025-10-17 06:03:06 [ℹ  setting availability zones to [us-east-1c us-east-1d]
2025-10-17 06:03:06 [ℹ  subnets for us-east-1c - public:192.168.0.0/19 private:192.168.64.0/19
2025-10-17 06:03:06 [ℹ  subnets for us-east-1d - public:192.168.32.0/19 private:192.168.96.0/19
2025-10-17 06:03:06 [ℹ  nodegroup "ng-a364780f" will use "" [AmazonLinux2023/1.32]
2025-10-17 06:03:06 [ℹ  using SSH public key "/root/.ssh/id_rsa.pub" as "eksctl-devops-nodegroup-ng-a364780f-7b:7f:e6:4d:31:a9:66:4d:57:03:f5:cd:b4
:8b:f8:0c"
2025-10-17 06:03:06 [!]  Auto Mode will be enabled by default in an upcoming release of eksctl. This means managed node groups and managed networkin
g add-ons will no longer be created by default. To maintain current behavior, explicitly set 'autoModeConfig.enabled: false' in your cluster configu
ration. Learn more: https://eksctl.io/usage/auto-mode/
2025-10-17 06:03:06 [ℹ  using Kubernetes version 1.32
2025-10-17 06:03:06 [ℹ  creating EKS cluster "devops" in "us-east-1" region with managed nodes
2025-10-17 06:03:06 [ℹ  will create 2 separate CloudFormation stacks for cluster itself and the initial managed nodegroup
2025-10-17 06:03:06 [ℹ  if you encounter any issues, check CloudFormation console or try 'eksctl utils describe-stacks --region=us-east-1 --cluster
=devops'
2025-10-17 06:03:06 [ℹ  Kubernetes API endpoint access will use default of {publicAccess=true, privateAccess=false} for cluster "devops" in "us-eas
t-1"
2025-10-17 06:03:06 [ℹ  CloudWatch logging will not be enabled for cluster "devops" in "us-east-1"
2025-10-17 06:03:06 [ℹ  you can enable it with 'eksctl utils update-cluster-logging --enable-types={SPECIFY-YOUR-LOG-TYPES-HERE (e.g. all)} --regio
```

Cluster devops created in AWS EKS



devops

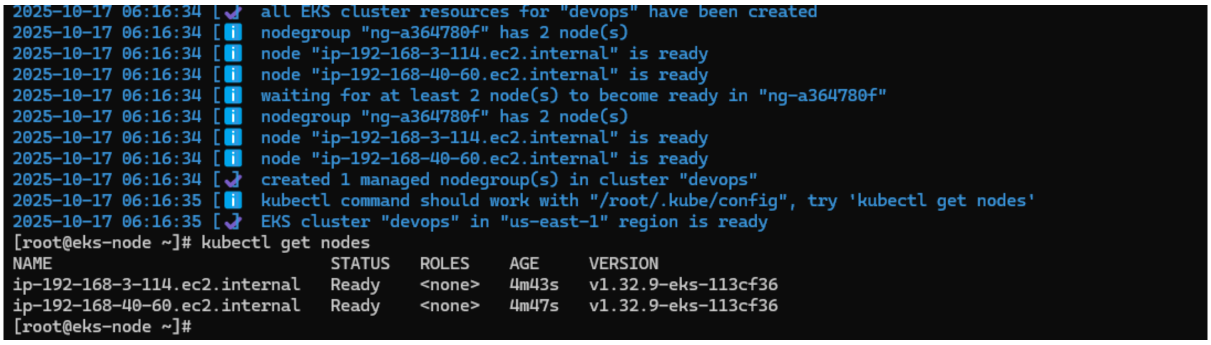
End of standard support for Kubernetes version 1.32 is March 23, 2026.

Cluster info Info

Status	Kubernetes version	Support period	Provider
Active	1.32	Standard support until March 23, 2026	EKS

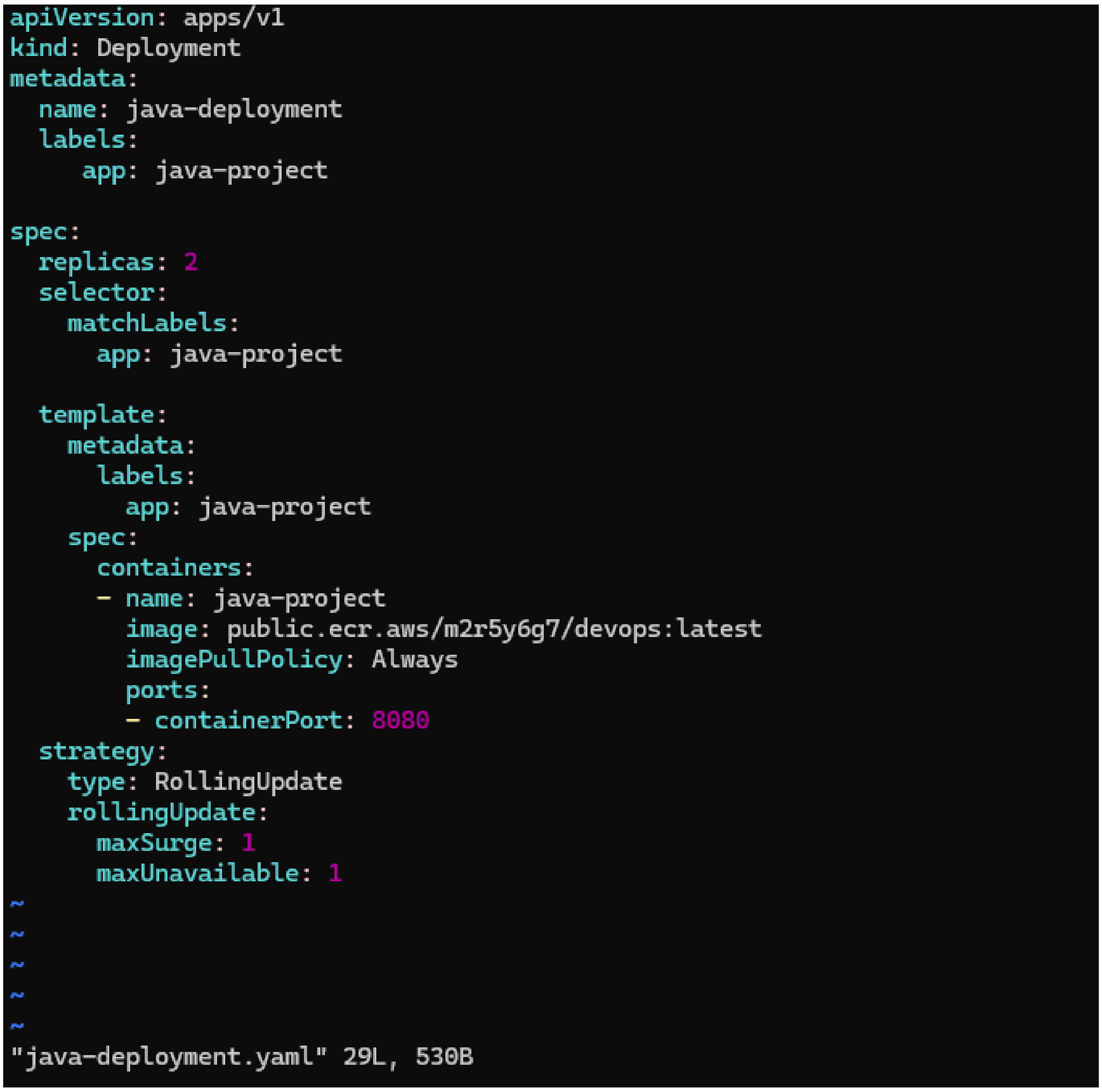
Cluster health	Upgrade insights	Node health issues
0	5	0

We get two target nodes



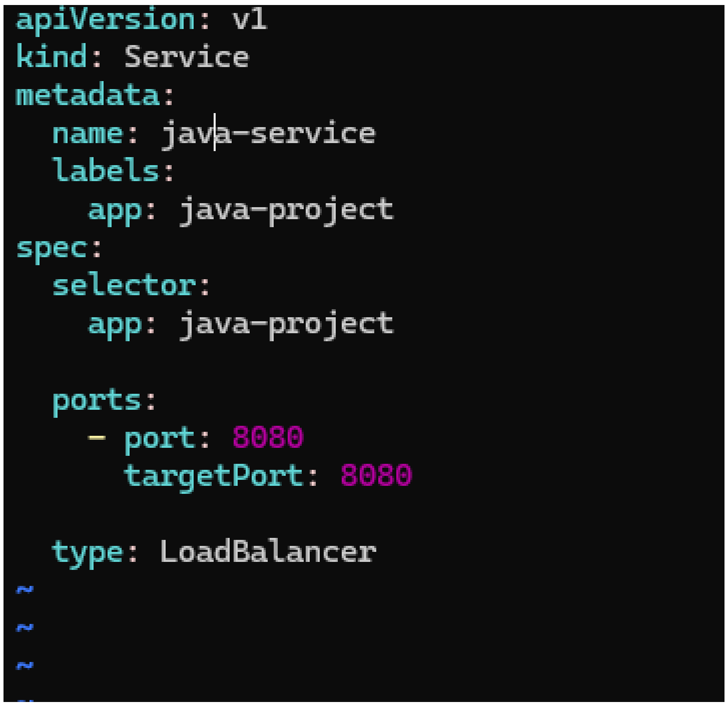
```
2025-10-17 06:16:34 [✓]  all EKS cluster resources for "devops" have been created
2025-10-17 06:16:34 [ℹ]  nodegroup "ng-a364780f" has 2 node(s)
2025-10-17 06:16:34 [ℹ]  node "ip-192-168-3-114.ec2.internal" is ready
2025-10-17 06:16:34 [ℹ]  node "ip-192-168-40-60.ec2.internal" is ready
2025-10-17 06:16:34 [ℹ]  waiting for at least 2 node(s) to become ready in "ng-a364780f"
2025-10-17 06:16:34 [ℹ]  nodegroup "ng-a364780f" has 2 node(s)
2025-10-17 06:16:34 [ℹ]  node "ip-192-168-3-114.ec2.internal" is ready
2025-10-17 06:16:34 [ℹ]  node "ip-192-168-40-60.ec2.internal" is ready
2025-10-17 06:16:34 [✓]  created 1 managed nodegroup(s) in cluster "devops"
2025-10-17 06:16:35 [ℹ]  kubectl command should work with "/root/.kube/config", try 'kubectl get nodes'
2025-10-17 06:16:35 [✓]  EKS cluster "devops" in "us-east-1" region is ready
[root@eks-node ~]# kubectl get nodes
NAME                            STATUS   ROLES    AGE     VERSION
ip-192-168-3-114.ec2.internal   Ready    <none>   4m43s   v1.32.9-eks-113cf36
ip-192-168-40-60.ec2.internal   Ready    <none>   4m47s   v1.32.9-eks-113cf36
[root@eks-node ~]#
```

Create deployment file in eks-node instance



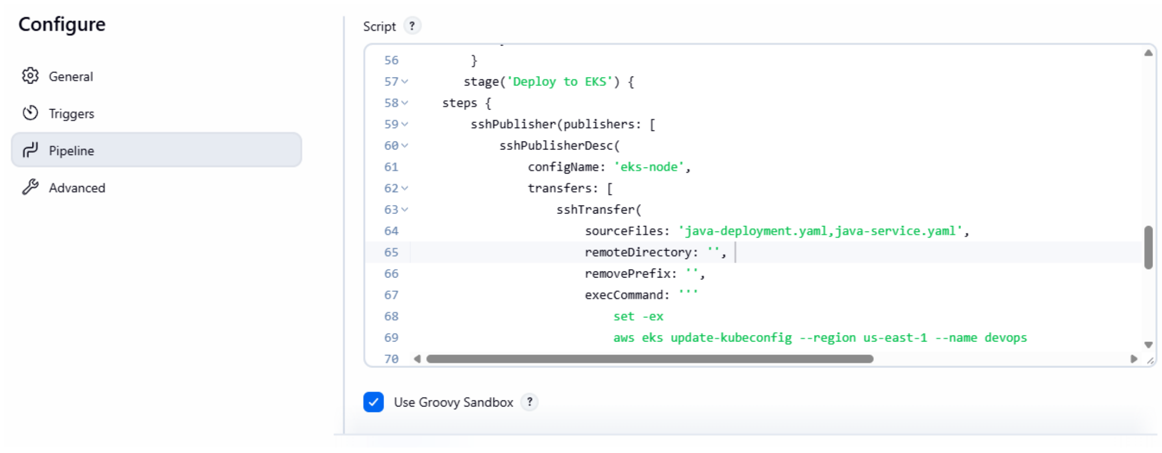
```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: java-deployment
  labels:
     app: java-project

spec:
  replicas: 2
  selector:
    matchLabels:
      app: java-project

  template:
    metadata:
      labels:
        app: java-project
    spec:
      containers:
      - name: java-project
        image: public.ecr.aws/m2r5y6g7/devops:latest
        imagePullPolicy: Always
        ports:
        - containerPort: 8080
  strategy:
    type: RollingUpdate
    rollingUpdate:
      maxSurge: 1
      maxUnavailable: 1
~
~
~
~
~
"java-deployment.yaml" 29L, 530B
```

Create service file in eks-node instance

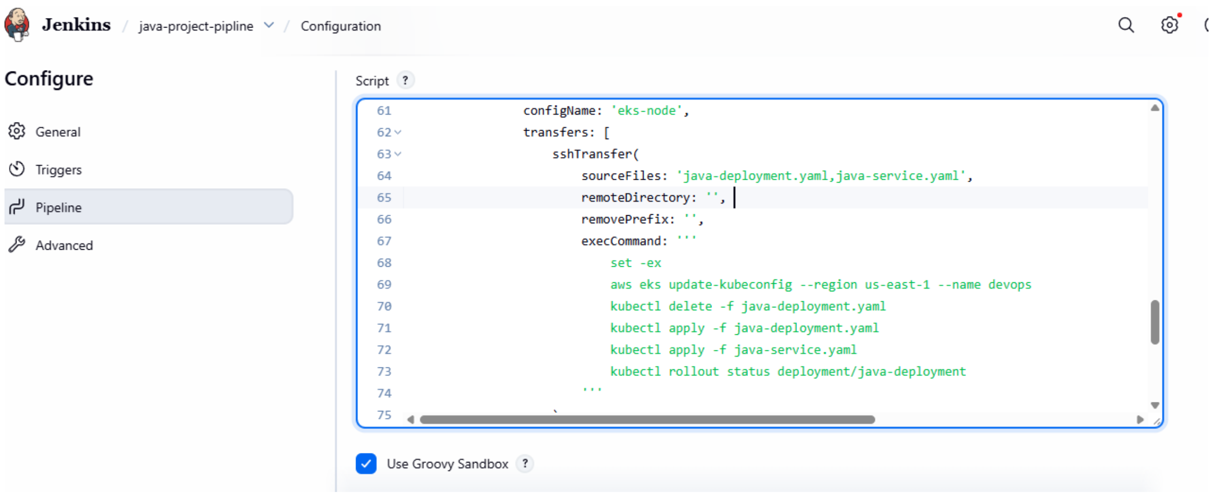


```
apiVersion: v1
kind: Service
metadata:
  name: java-service
  labels:
    app: java-project
spec:
  selector:
    app: java-project

  ports:
    - port: 8080
      targetPort: 8080

  type: LoadBalancer
~
~
~
```

Now writing a stage in Jenkins file to apply deployment and service file in eks-node



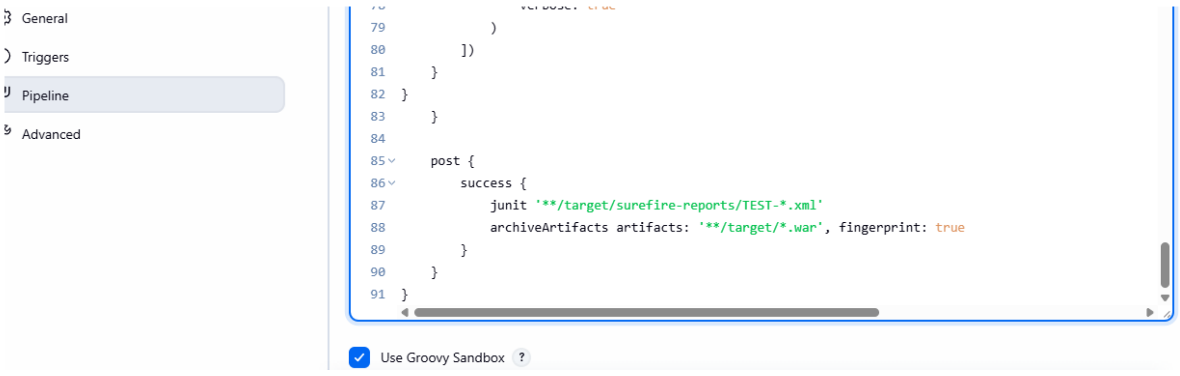
```
        }
       stage('Deploy to EKS') {
    steps {
        sshPublisher(publishers: [
            sshPublisherDesc(
                configName: 'eks-node',
                transfers: [
                    sshTransfer(
                        sourceFiles: 'java-deployment.yaml,java-service.yaml',
                        remoteDirectory: '',
                        removePrefix: '',
                        execCommand: '''
                            set -ex
                            aws eks update-kubeconfig --region us-east-1 --name devops
```



```
                configName: 'eks-node',
                transfers: [
                    sshTransfer(
                        sourceFiles: 'java-deployment.yaml,java-service.yaml',
                        remoteDirectory: '',
                        removePrefix: '',
                        execCommand: '''
                            set -ex
                            aws eks update-kubeconfig --region us-east-1 --name devops
                            kubectl delete -f java-deployment.yaml
                            kubectl apply -f java-deployment.yaml
                            kubectl apply -f java-service.yaml
                            kubectl rollout status deployment/java-deployment
                        '''
```

[✓] Use Groovy Sandbox

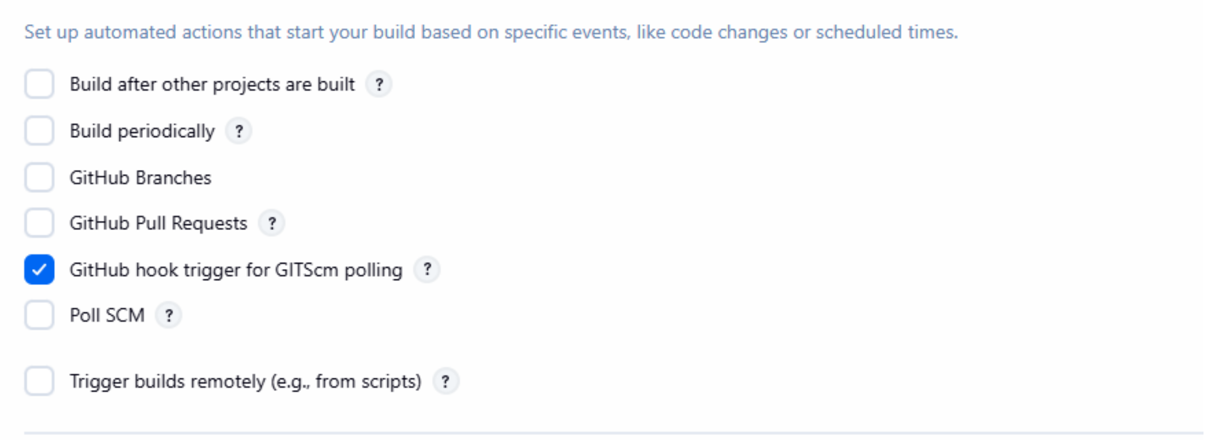
Now for post build stage print success for pipeline works fine



```
                        verbose: true
                    )
                ])
            }
        }
    }

    post {
        success {
            junit '**/target/surefire-reports/TEST-*.xml'
            archiveArtifacts artifacts: '**/target/*.war', fingerprint: true
        }
    }
}
```

[✓] Use Groovy Sandbox

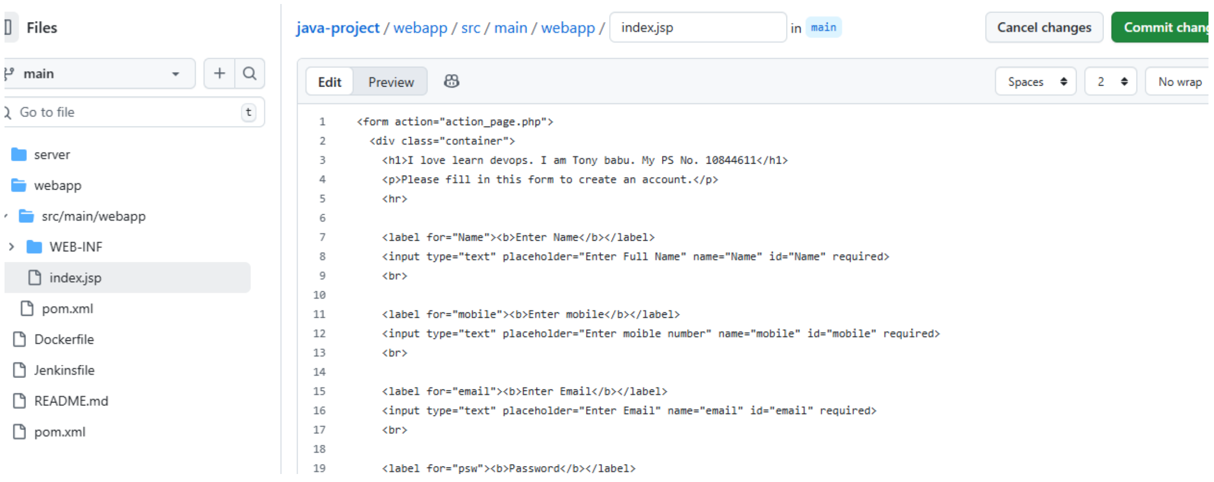
Selecting github scm trigger checkbox to check for changes



Set up automated actions that start your build based on specific events, like code changes or scheduled times.

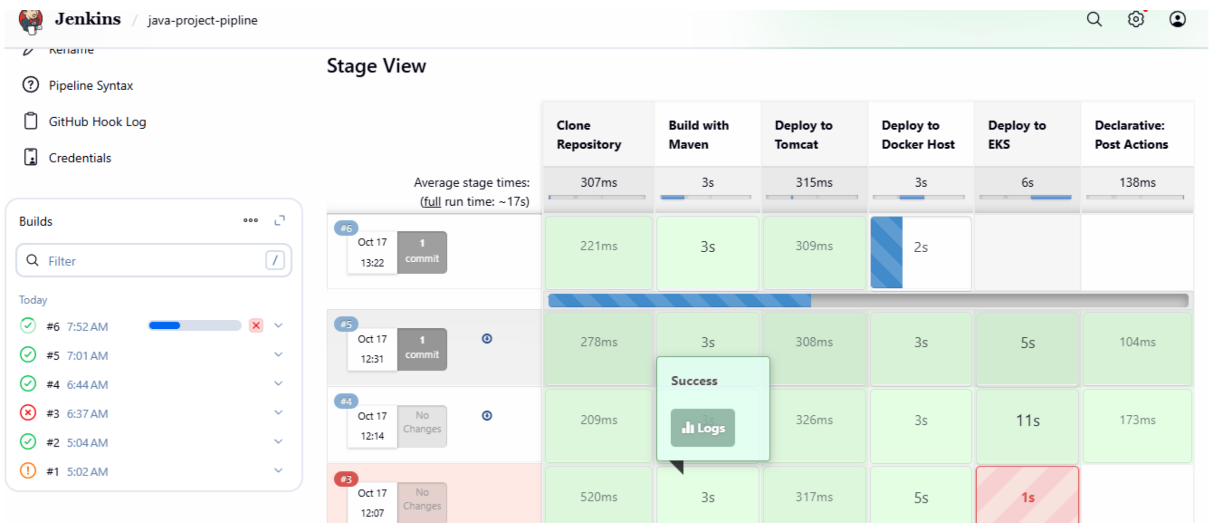
- [ ] Build after other projects are built ?
- [ ] Build periodically ?
- [ ] GitHub Branches
- [ ] GitHub Pull Requests ?
- [✓] GitHub hook trigger for GITScm polling ?
- [ ] Poll SCM ?
- [ ] Trigger builds remotely (e.g., from scripts) ?

Making changes in github repository and committing changes



```
<form action="action_page.php">
  <div class="container">
    <h1>I love learn devops. I am Tony babu. My PS No. 10844611</h1>
    <p>Please fill in this form to create an account.</p>
    <hr>

    <label for="Name"><b>Enter Name</b></label>
    <input type="text" placeholder="Enter Full Name" name="Name" id="Name" required>
    <br>

    <label for="mobile"><b>Enter mobile</b></label>
    <input type="text" placeholder="Enter moible number" name="mobile" id="mobile" required>
    <br>

    <label for="email"><b>Enter Email</b></label>
    <input type="text" placeholder="Enter Email" name="email" id="email" required>
    <br>

    <label for="psw"><b>Password</b></label>
```

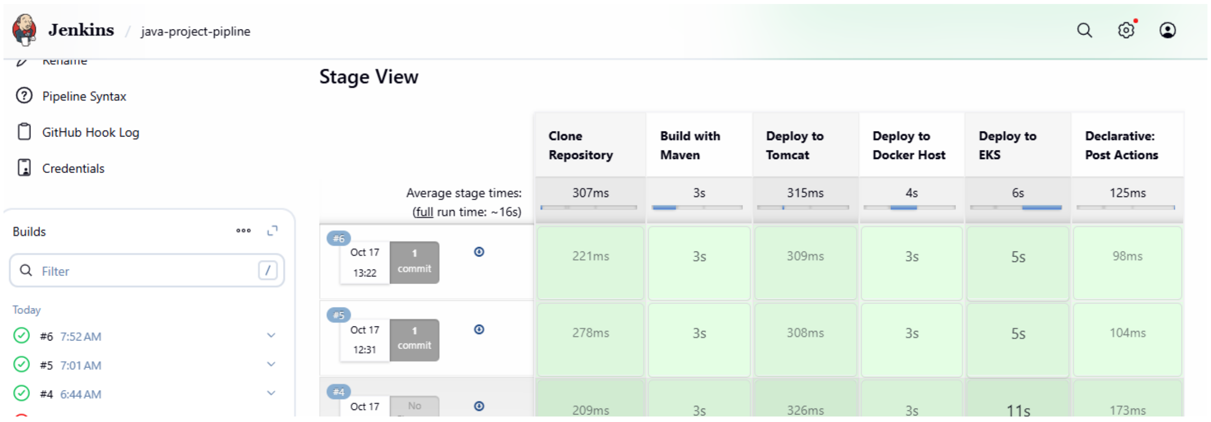
Build automatically triggers



#6	7:52 AM	
#5	7:01 AM	
#4	6:44 AM	
#3	6:37 AM	
#2	5:04 AM	
#1	5:02 AM	

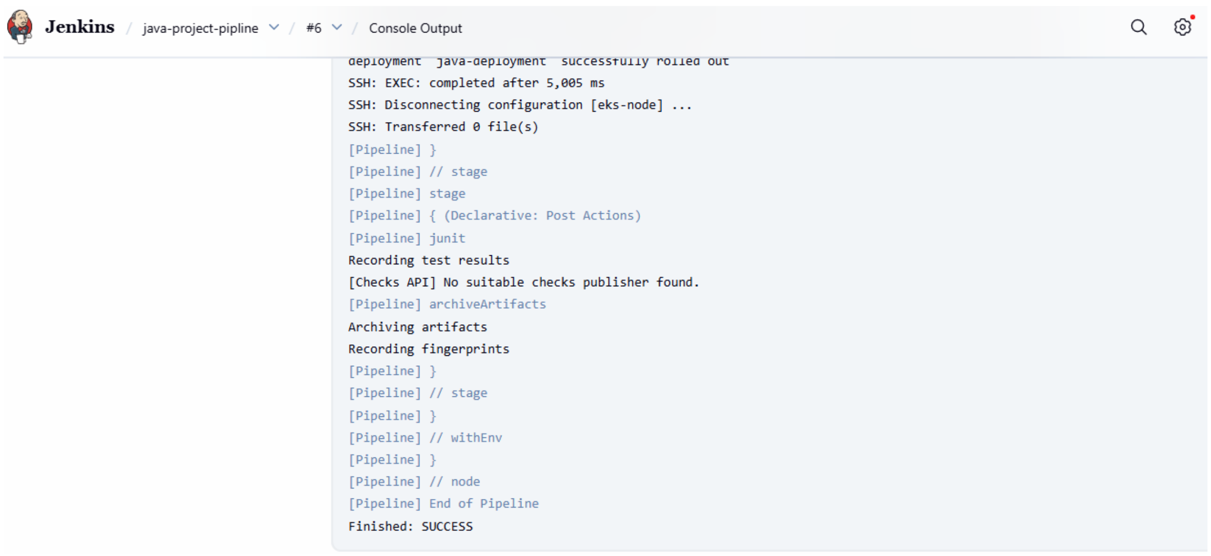
	Clone Repository	Build with Maven	Deploy to Tomcat	Deploy to Docker Host	Deploy to EKS	Declarative: Post Actions
Average stage times: (full run time: ~17s)	307ms	3s	315ms	3s	6s	138ms
#6 Oct 17 13:22 1 commit	221ms	3s	309ms	2s		
#5 Oct 17 12:31 1 commit	278ms	3s	308ms	3s	5s	104ms
#4 Oct 17 12:14 No Changes	209ms		326ms	3s	11s	173ms
#3 Oct 17 12:07 No Changes	520ms	3s	317ms	5s	1s	

Build success



Stage View

	Clone Repository	Build with Maven	Deploy to Tomcat	Deploy to Docker Host	Deploy to EKS	Declarative: Post Actions
Average stage times: (full run time: ~16s)	307ms	3s	315ms	4s	6s	125ms
#6 Oct 17 13:22 1 commit	221ms	3s	309ms	3s	5s	98ms
#5 Oct 17 12:31 1 commit	278ms	3s	308ms	3s	5s	104ms
#4 Oct 17 No	209ms	3s	326ms	3s	11s	173ms



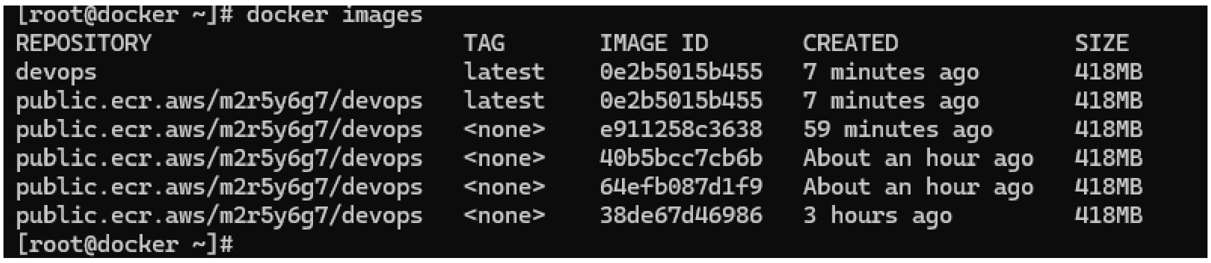
```
deployment "java-deployment" successfully rolled out
SSH: EXEC: completed after 5,005 ms
SSH: Disconnecting configuration [eks-node] ...
SSH: Transferred 0 file(s)
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (Declarative: Post Actions)
[Pipeline] junit
Recording test results
[Checks API] No suitable checks publisher found.
[Pipeline] archiveArtifacts
Archiving artifacts
Recording fingerprints
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```

Groovy script:-

```
pipeline {
  agent any
    environment {
     TOMCAT_USER = 'deployer'
     TOMCAT_PASS = 'deployer'
     TOMCAT_HOST = '34.226.247.144'
     TOMCAT_PORT = '8080'
  }
  stages {
     stage('Clone Repository') {
        steps {
           git branch: 'main', url: 'https://github.com/tonybabu2004-eng/java-project.git'
        }
     }
     stage('Build with Maven') {
        steps {
          sh 'mvn clean package -Dmaven.test.failure.ignore=true'
        }
      }
      stage('Deploy to Tomcat') {
        steps {
          sh '''
            curl -u $TOMCAT_USER:$TOMCAT_PASS \
            --upload-file webapp/target/webapp.war \
            "http://$TOMCAT_HOST:$TOMCAT_PORT/manager/text/deploy?path=/webap
p&update=true"
          '''
        }
      }
      stage('Deploy to Docker Host') {
        steps {
          sshPublisher(publishers: [
            sshPublisherDesc(
              configName: 'docker-host',
              transfers: [
                sshTransfer(
                  sourceFiles: 'java-deployment.yaml,java-service.yaml',
                  removePrefix: '',
                  remoteDirectory: 'images',
                  execCommand: '''
                    cd images
                    aws ecr-public get-login-password --region us-east-1 | docker login --
username AWS --password-stdin public.ecr.aws/m2r5y6g7
                    docker build -t project .
                    docker tag project:latest public.ecr.aws/m2r5y6g7/project:latest
                    docker push public.ecr.aws/m2r5y6g7/project:latest
                  '''
                )
              ]
            )
          ])
        }
      }
    }
    post {
      success {
        junit '**/target/surefire-reports/TEST-*.xml'
        archiveArtifacts artifacts: '**/target/*.war', fingerprint: true
      }
```

```
   }
}
```

Image created and pushed to ECR



```
[root@docker ~]# docker images
REPOSITORY                        TAG       IMAGE ID       CREATED            SIZE
devops                            latest    0e2b5015b455   7 minutes ago      418MB
public.ecr.aws/m2r5y6g7/devops    latest    0e2b5015b455   7 minutes ago      418MB
public.ecr.aws/m2r5y6g7/devops    <none>    e911258c3638   59 minutes ago     418MB
public.ecr.aws/m2r5y6g7/devops    <none>    40b5bcc7cb6b   About an hour ago  418MB
public.ecr.aws/m2r5y6g7/devops    <none>    64efb087d1f9   About an hour ago  418MB
public.ecr.aws/m2r5y6g7/devops    <none>    38de67d46986   3 hours ago        418MB
[root@docker ~]#
```

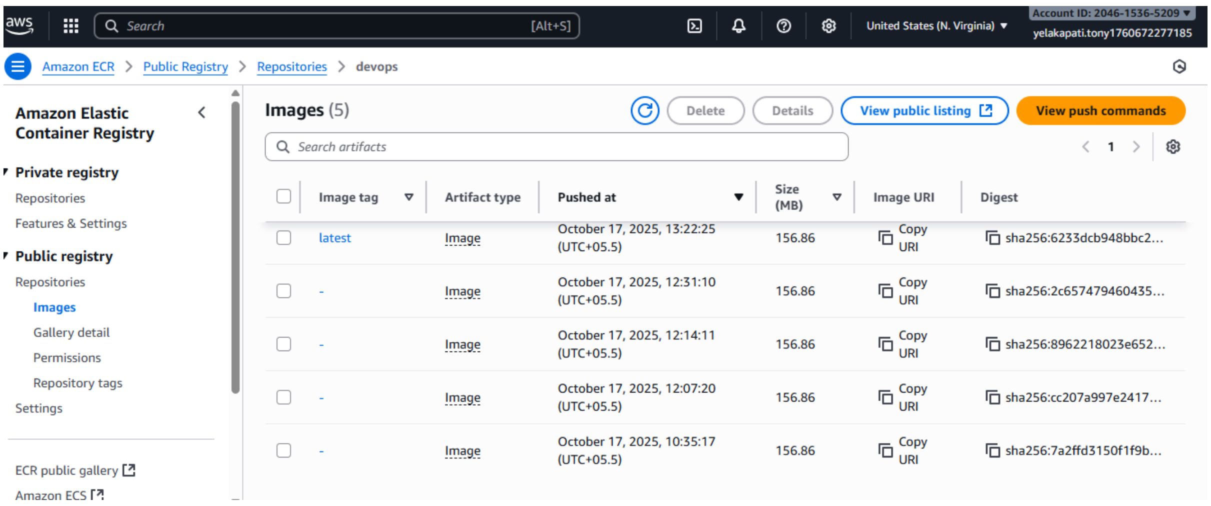
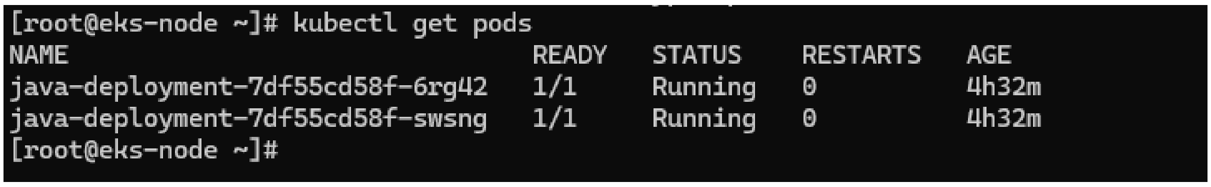


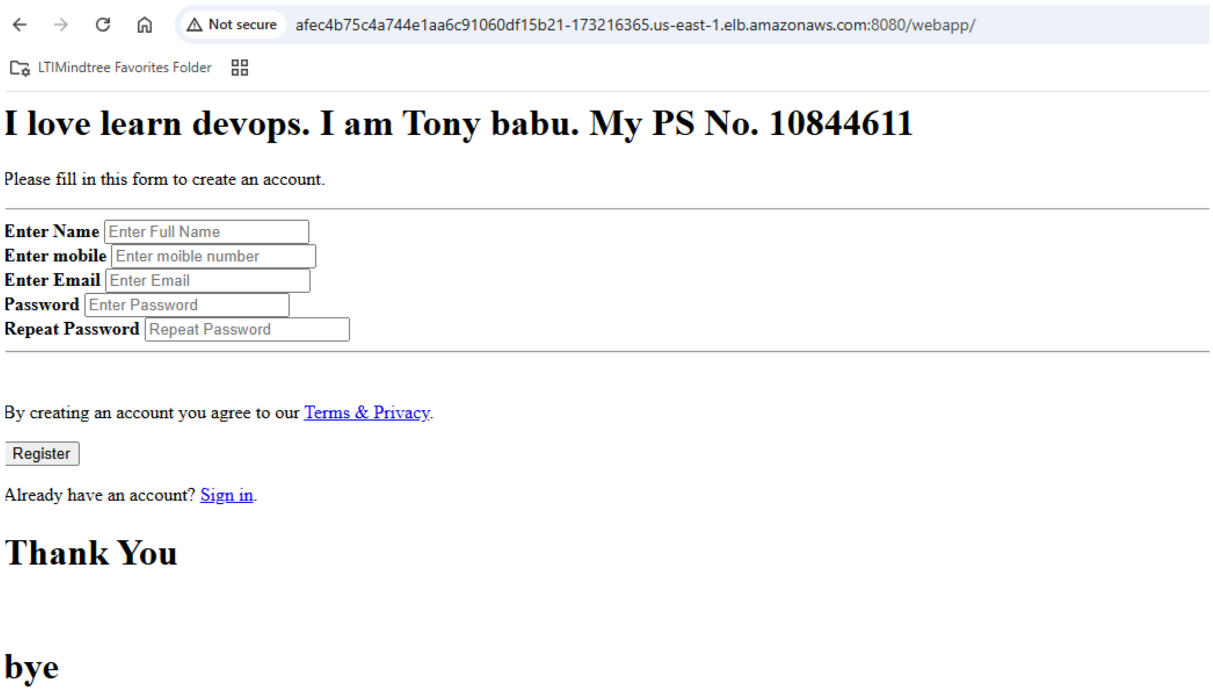
Image tag	Artifact type	Pushed at	Size (MB)	Image URI	Digest
latest	Image	October 17, 2025, 13:22:25 (UTC+05.5)	156.86	Copy URI	sha256:6233dcb948bbc2...
-	Image	October 17, 2025, 12:31:10 (UTC+05.5)	156.86	Copy URI	sha256:2c657479460435...
-	Image	October 17, 2025, 12:14:11 (UTC+05.5)	156.86	Copy URI	sha256:8962218023e652...
-	Image	October 17, 2025, 12:07:20 (UTC+05.5)	156.86	Copy URI	sha256:cc207a997e2417...
-	Image	October 17, 2025, 10:35:17 (UTC+05.5)	156.86	Copy URI	sha256:7a2ffd3150f1f9b...

Deployment file in eks-node pulls latest image and creates deployment and container in pods with latest image



```
[root@eks-node ~]# kubectl get pods
NAME                                  READY   STATUS    RESTARTS   AGE
java-deployment-7df55cd58f-6rg42      1/1     Running   0          4h32m
java-deployment-7df55cd58f-swsng      1/1     Running   0          4h32m
[root@eks-node ~]#
```

Paste service Ip in browser and changes reflected



Not secure afec4b75c4a744e1aa6c91060df15b21-173216365.us-east-1.elb.amazonaws.com:8080/webapp/

LTIMindtree Favorites Folder

I love learn devops. I am Tony babu. My PS No. 10844611

Please fill in this form to create an account.

Enter Name Enter Full Name
Enter mobile Enter moible number
Enter Email Enter Email
Password Enter Password
Repeat Password Repeat Password

By creating an account you agree to our Terms & Privacy.

Register

Already have an account? Sign in.

Thank You

bye

Done By –

Yelakapati Tony babu

Ps No. 10844611