**Creating CI/CD pipeline with Jenkins & integrating EKS Cluster where hosting Springboot java application.**

**Jenkins pipeline flow:**

**- Cloning from Github**

**- Automating builds using Jenkins**

**- Automating Docker image creation**

**- Automating Docker image upload into AWS ECR**

**- Automating Docker Containers Deployments to Kubernetes Cluster**

Diagram

Description automatically generated

1. **Install Jenkins, Maven , git , & Docker.**

**Change Host Name to Jenkins**

sudo hostname Jenkins

**Perform update first**

sudo apt update

**Install the maven first and it will install along with jdk .**

sudo apt install maven -y

**you can type** mvn –version & java -version

**Install the Jenkins packages**

**Add Repository key to the system**

wget -q -O - https://pkg.jenkins.io/debian/jenkins.io.key | sudo apt-key add -

**Append Debian/ubuntu package repo address to the system**

echo deb http://pkg.jenkins.io/debian-stable binary/ | sudo tee /etc/apt/sources.list.d/jenkins.list

**Update Ubuntu package**

sudo apt update

**Install Jenkins**

sudo apt install jenkins -y

Try to access the Jenkins using the Publicip:8080

**Install GIT**

apt install git -y

**Install the Docker packages**

**Docker installation steps using default repository from Ubuntu**

**Update local packages by executing below command:**

sudo apt-get update

**Install the below packages**

sudo apt install gnupg2 pass -y

**gnupg2 is tool for secure communication and data storage. It can be used to encrypt data and to create digital signatures**

**Install docker**

sudo apt install docker.io -y

**Add Ubuntu user to Docker group**

sudo usermod -aG docker $USER

**We need to reload shell in order to have new group settings applied. Now you need to logout and log back in command line or execute the below command:**

newgrp docker

**The Docker service needs to be setup to run at startup. To do so, type in each command followed by enter:**

sudo systemctl start docker

sudo systemctl enable docker

sudo systemctl status docker

**Next Steps:**

**Add jenkins user to Docker group**

sudo usermod -a -G docker jenkins

**Restart Jenkins service**

sudo service jenkins restart

**Reload system daemon files**

sudo systemctl daemon-reload

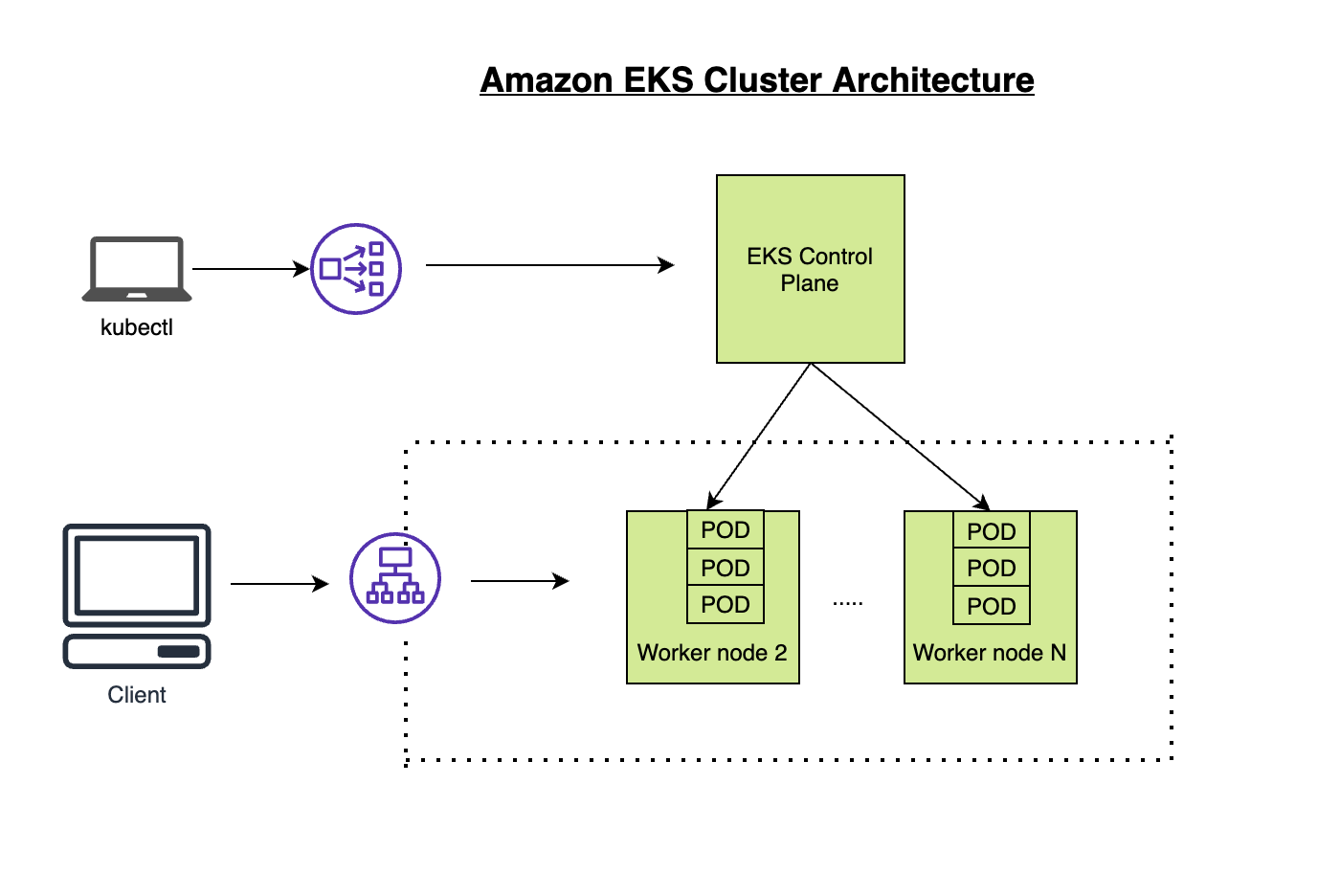
**Restart Docker service as well**

sudo service docker stop

sudo service docker start

1. **Create EKS cluster & install required service for the set up.**

**-**Amazon EKS is a fully managed container orchestration service. EKS allows you to quickly deploy a production ready Kubernetes cluster in AWS, deploy and manage containerized applications more easily with a fully managed Kubernetes service.



**\*Before starting kindly create 1 role for Administrator access, AmazonEC2ContainerRegistryFullAccess policies and attach to your EC2 instance\***

**-Install AWS CLI – Command line tools for working with AWS services, including Amazon EKS.**

curl "https://awscli.amazonaws.com/awscli-exe-linux-x86\_64.zip" -o "awscliv2.zip"

sudo apt install unzip

sudo unzip awscliv2.zip

sudo ./aws/install

aws --version

**Install eksctl – A command line tool for working with EKS clusters that automates many individual tasks.**

**Download and extract the latest release of eksctl with the following command.**

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

**Move the extracted binary to /usr/local/bin.**

sudo mv /tmp/eksctl /usr/local/bin

eksctl version

A computer screen capture

Description automatically generated with low confidence

**-Install kubectl – A command line tool for working with Kubernetes clusters.**

sudo curl -o kubectl <https://s3.us-west-2.amazonaws.com/amazon-eks/1.23.7/2022-06-29/bin/linux/amd64/kubectl>

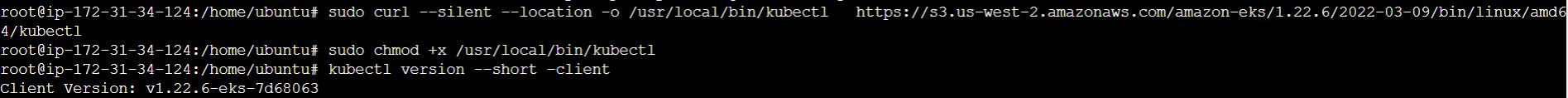
chmod +x ./kubectl

mkdir -p $HOME/bin && cp ./kubectl $HOME/bin/kubectl && export PATH=$PATH:$HOME/bin

echo 'export PATH=$PATH:$HOME/bin' >> ~/.bashrc

**Verify if kubectl got installed**

kubectl version --short –client



1. **Create EKS Cluster with two worker nodes using eksctl**

**Create jenkins user first**

useradd jenkins

passwd jenkins

**Let's add jenkins user as an administrator and also ass NOPASSWD so that during the pipeline run it will not ask for root password.**

**Open the file /etc/sudoers in**vi**mode**

sudo vi /etc/sudoers

**Add the following line at the end of the file**

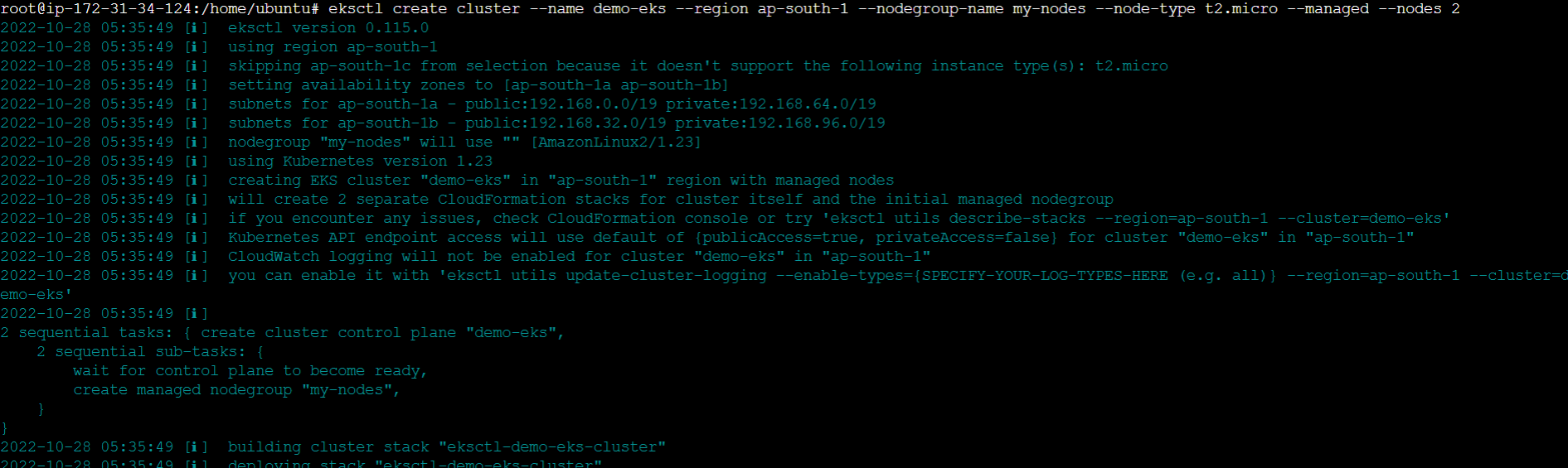
jenkins ALL=(ALL) NOPASSWD: ALL

**switch to Jenkins user**

sudo su - jenkins

**Commands to create EKS cluster**

eksctl create cluster --name demo-eks-2 --region ap-south-1 --nodegroup-name my-nodes2 --node-type t2.micro --managed --nodes 2



the above command should create a EKS cluster in AWS, it might take 15 to 20 mins. The ***eksctl*** tool uses CloudFormation under the hood, creating one stack for the EKS master control plane and another stack for the worker nodes.

**Verification step:**

eksctl get cluster --name demo-eks2 --region ap-south-1

**Update Kube config by entering below command:**

aws eks update-kubeconfig --name demo-eks2 --region ap-south-1

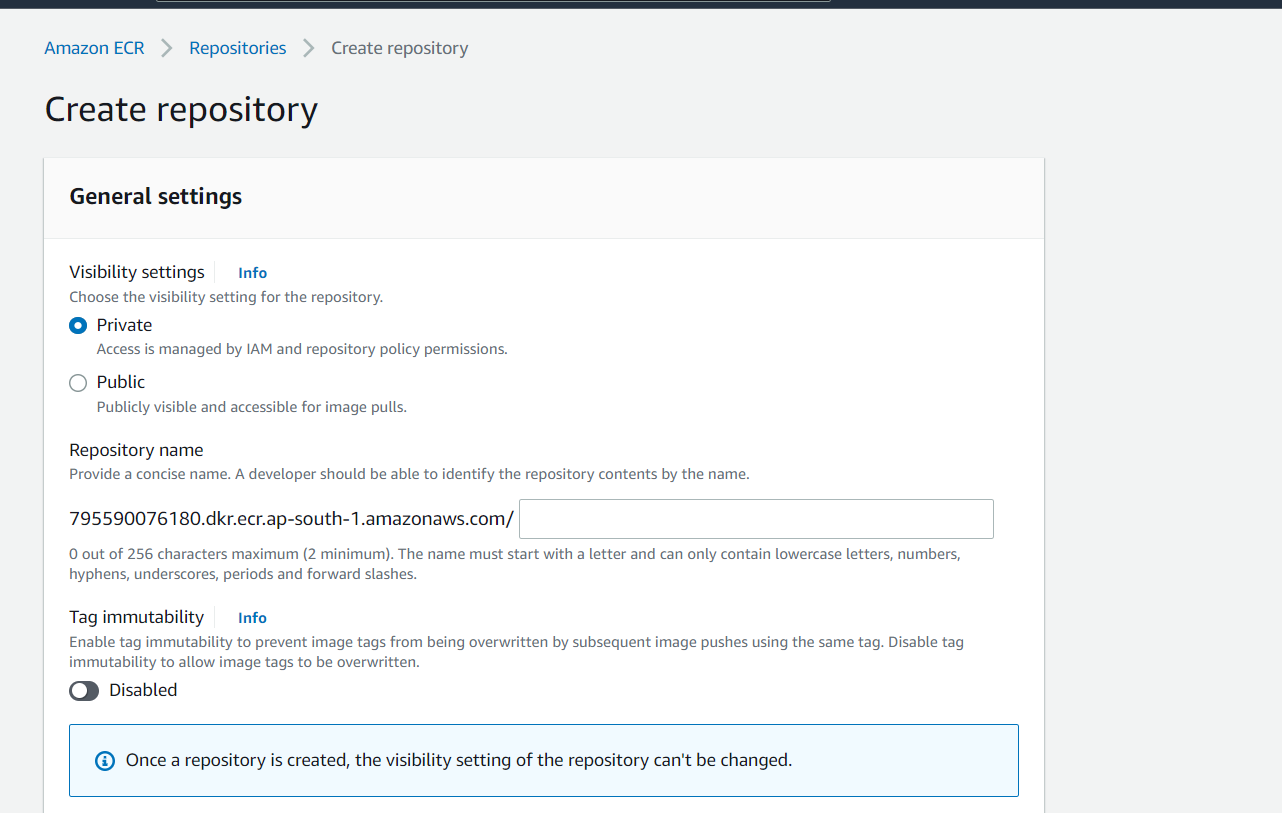
**kubeconfig file be updated under /var/lib/jenkins/.kube folder.you can view the kubeconfig file by entering the below command:**

cat /var/lib/jenkins/.kube/config

**Connect to EKS cluster using kubectl commands.To view the list of worker nodes as part of EKS cluster.**

kubectl get nodes

1. Create ECR **private** repository to store your docker images.



Graphical user interface, text, application

Description automatically generated

1. Set up Jenkins, git, Maven & Docker
2. Install the required plugin with name

-Docker, Docker pipeline, Maven and Kubernetes CLI plug-ins

1. Set up the Jenkins environment & integrate other tools like Maven & Kubernetes

Set up the variable for maven

Go to Global tool configurations

* Maven 🡪 Provide Name & Maven\_Home path

Table

Description automatically generated

**🡪(Kubernetes uses a YAML file called kubeconfig to store cluster authentication information for kubectl . kubeconfig contains a list of contexts to which kubectl refers when running commands. By default, the file is saved at $HOME/. kube/config . A context is a group of access parameters)**

Create Credentials for connecting to Kubernetes Cluster using **kubeconfig**. Click on Add Credentials, use Kubernetes configuration from drop down.

You will be able to find kubeconfig on the below mention path .

Go to **cat /var/lib/jenkins/.kube/config** or **cat /root/.kube/config**

**Copy the config file & paste into notepad. Save as kubeconfig**

**Go to global credentials & add the credential**

Graphical user interface, application

Description automatically generated

Graphical user interface, application, Teams

Description automatically generated

**Add kubeconfig & set ID as “K8S”Graphical user interface, application, Teams

Description automatically generated**

**7.Create Jenkins pipeline & perform the below steps**

**Graphical user interface, text, application, email

Description automatically generated**

**Graphical user interface, text, application, email

Description automatically generated**

**Edit the branch specifier to main branch**

**Graphical user interface, application

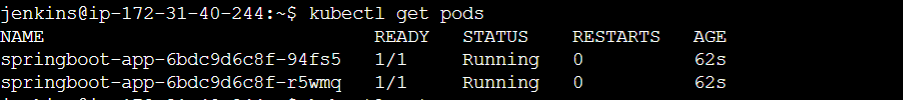
Description automatically generated**

**8.Run the pipeline**

**Table

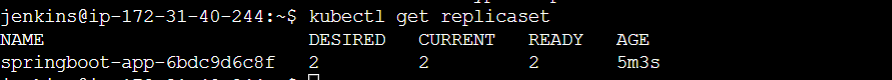
Description automatically generated**

**Once the pipeline has successfully run then go to your ec2 machine and check the creation of pods:**

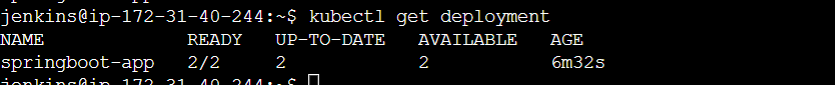
****

**Verification steps:**

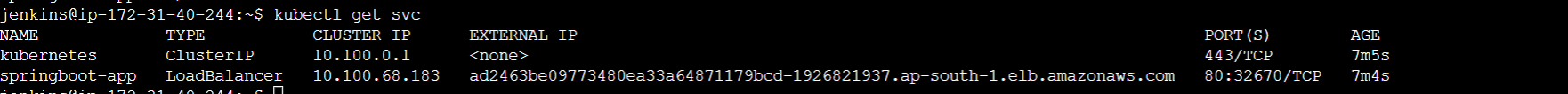
**As mention in our manifest file, 2 replica’s are running as per the requirement**

****

**Check the deployment**

****

**Check the Service type as load balancer is working or not. If it’s working fine then copy the External IP of load balancer & hit the Ip link on your interner browser.**

****

**Final Output:**

**Graphical user interface

Description automatically generated**