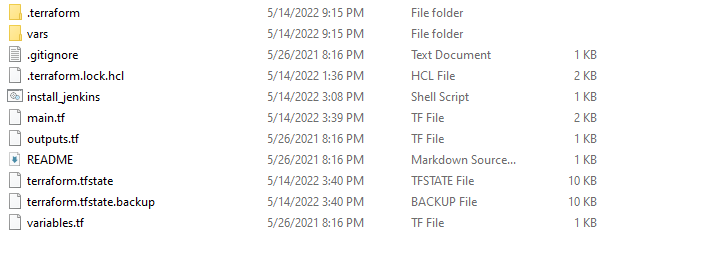
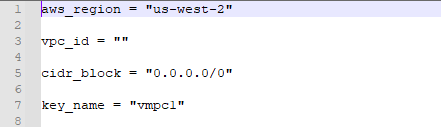
**Project**

**Part 1: Create Jenkins server environment in AWS using Terraform**

Terraform files:



Variable file:



Main.tf:

terraform {

required\_version = ">= 0.12"

}

provider "aws" {

region = var.aws\_region

}

resource "aws\_security\_group" "jenkins\_sg" {

name = "jenkins\_sg"

description = "Allow Jenkins Traffic"

vpc\_id = var.vpc\_id

ingress {

description = "Allow from Personal CIDR block"

from\_port = 8080

to\_port = 8080

protocol = "tcp"

cidr\_blocks = [var.cidr\_block]

}

ingress {

description = "Allow SSH from Personal CIDR block"

from\_port = 22

to\_port = 22

protocol = "tcp"

cidr\_blocks = [var.cidr\_block]

}

egress {

from\_port = 0

to\_port = 0

protocol = "-1"

cidr\_blocks = ["0.0.0.0/0"]

ipv6\_cidr\_blocks = ["::/0"]

}

tags = {

Name = "Jenkins SG"

}

}

data "aws\_ami" "ubuntu-linux-1804" {

most\_recent = true

owners = ["099720109477"] # Canonical

filter {

name = "name"

values = ["ubuntu/images/hvm-ssd/ubuntu-bionic-18.04-amd64-server-\*"]

}

filter {

name = "virtualization-type"

values = ["hvm"]

}

}

resource "aws\_instance" "web" {

ami = data.aws\_ami.ubuntu-linux-1804.id

instance\_type = "t2.micro"

key\_name = var.key\_name

security\_groups = [aws\_security\_group.jenkins\_sg.name]

tags = {

Name = "Jenkins"

}

user\_data = <<EOC

#!/bin/bash

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

sudo sh -c 'echo deb http://pkg.jenkins.io/debian-stable binary/ > /etc/apt/sources.list.d/jenkins.list'

sudo apt update

sudo apt install jenkins

sudo apt install maven -y

sudo apt install -y docker\*

sudo apt install gnupg2 pass

sudo systemctl daemon-reload

sudo systemctl start jenkins

sudo systemctl status jenkins

sudo chmod 666 /var/run/docker.sock

EOC

}

Terraform command:

**Terraform init**

**Terraform plan -var-file="vars/dev-west.tfvars"**

**Terraform apply -var-file="vars/dev-west.tfvars"**

**Output:**

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

following symbols:

+ create

Terraform will perform the following actions:

# aws\_instance.web will be created

+ resource "aws\_instance" "web" {

+ ami = "ami-0e8e7369d92f16d0a"

+ arn = (known after apply)

+ associate\_public\_ip\_address = (known after apply)

+ availability\_zone = (known after apply)

+ cpu\_core\_count = (known after apply)

+ cpu\_threads\_per\_core = (known after apply)

+ disable\_api\_termination = (known after apply)

+ ebs\_optimized = (known after apply)

+ get\_password\_data = false

+ host\_id = (known after apply)

+ id = (known after apply)

+ instance\_initiated\_shutdown\_behavior = (known after apply)

+ instance\_state = (known after apply)

+ instance\_type = "t2.micro"

+ ipv6\_address\_count = (known after apply)

+ ipv6\_addresses = (known after apply)

+ key\_name = "vmpc1"

+ monitoring = (known after apply)

+ outpost\_arn = (known after apply)

+ password\_data = (known after apply)

+ placement\_group = (known after apply)

+ placement\_partition\_number = (known after apply)

+ primary\_network\_interface\_id = (known after apply)

+ private\_dns = (known after apply)

+ private\_ip = (known after apply)

+ public\_dns = (known after apply)

+ public\_ip = (known after apply)

+ secondary\_private\_ips = (known after apply)

+ security\_groups = [

+ "jenkins\_sg",

]

+ source\_dest\_check = true

+ subnet\_id = (known after apply)

+ tags = {

+ "Name" = "Jenkins"

}

+ tags\_all = {

+ "Name" = "Jenkins"

}

+ tenancy = (known after apply)

+ user\_data = "e699b0b9e3200734532cce85c2111be5cd2bc320"

+ user\_data\_base64 = (known after apply)

+ user\_data\_replace\_on\_change = false

+ vpc\_security\_group\_ids = (known after apply)

+ capacity\_reservation\_specification {

+ capacity\_reservation\_preference = (known after apply)

+ capacity\_reservation\_target {

+ capacity\_reservation\_id = (known after apply)

+ capacity\_reservation\_resource\_group\_arn = (known after apply)

}

}

+ ebs\_block\_device {

+ delete\_on\_termination = (known after apply)

+ device\_name = (known after apply)

+ encrypted = (known after apply)

+ iops = (known after apply)

+ kms\_key\_id = (known after apply)

+ snapshot\_id = (known after apply)

+ tags = (known after apply)

+ throughput = (known after apply)

+ volume\_id = (known after apply)

+ volume\_size = (known after apply)

+ volume\_type = (known after apply)

}

+ enclave\_options {

+ enabled = (known after apply)

}

+ ephemeral\_block\_device {

+ device\_name = (known after apply)

+ no\_device = (known after apply)

+ virtual\_name = (known after apply)

}

+ maintenance\_options {

+ auto\_recovery = (known after apply)

}

+ metadata\_options {

+ http\_endpoint = (known after apply)

+ http\_put\_response\_hop\_limit = (known after apply)

+ http\_tokens = (known after apply)

+ instance\_metadata\_tags = (known after apply)

}

+ network\_interface {

+ delete\_on\_termination = (known after apply)

+ device\_index = (known after apply)

+ network\_card\_index = (known after apply)

+ network\_interface\_id = (known after apply)

}

+ root\_block\_device {

+ delete\_on\_termination = (known after apply)

+ device\_name = (known after apply)

+ encrypted = (known after apply)

+ iops = (known after apply)

+ kms\_key\_id = (known after apply)

+ tags = (known after apply)

+ throughput = (known after apply)

+ volume\_id = (known after apply)

+ volume\_size = (known after apply)

+ volume\_type = (known after apply)

}

}

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

Do you want to perform these actions?

Terraform will perform the actions described above.

Only 'yes' will be accepted to approve.

Enter a value: yes

aws\_instance.web: Creating...

aws\_instance.web: Still creating... [10s elapsed]

aws\_instance.web: Still creating... [20s elapsed]

aws\_instance.web: Still creating... [30s elapsed]

aws\_instance.web: Still creating... [40s elapsed]

aws\_instance.web: Creation complete after 41s [id=i-0dc43eb15a19e4f44]

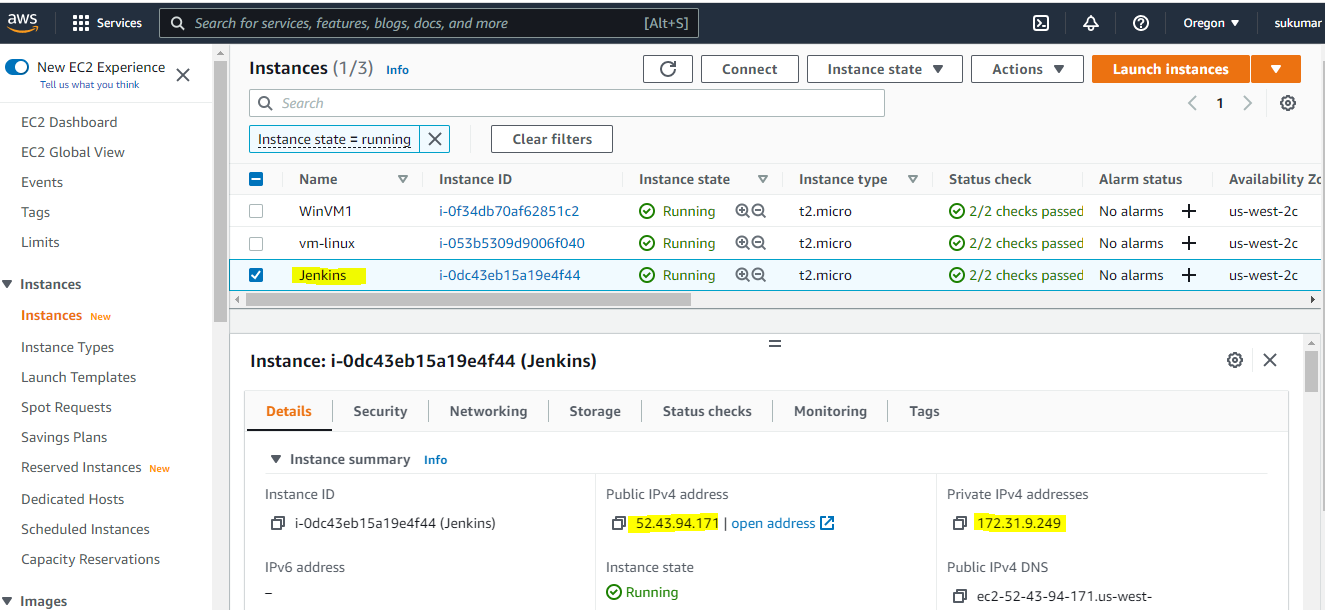
Apply complete! Resources: 1 added, 0 changed, 0 destroyed.

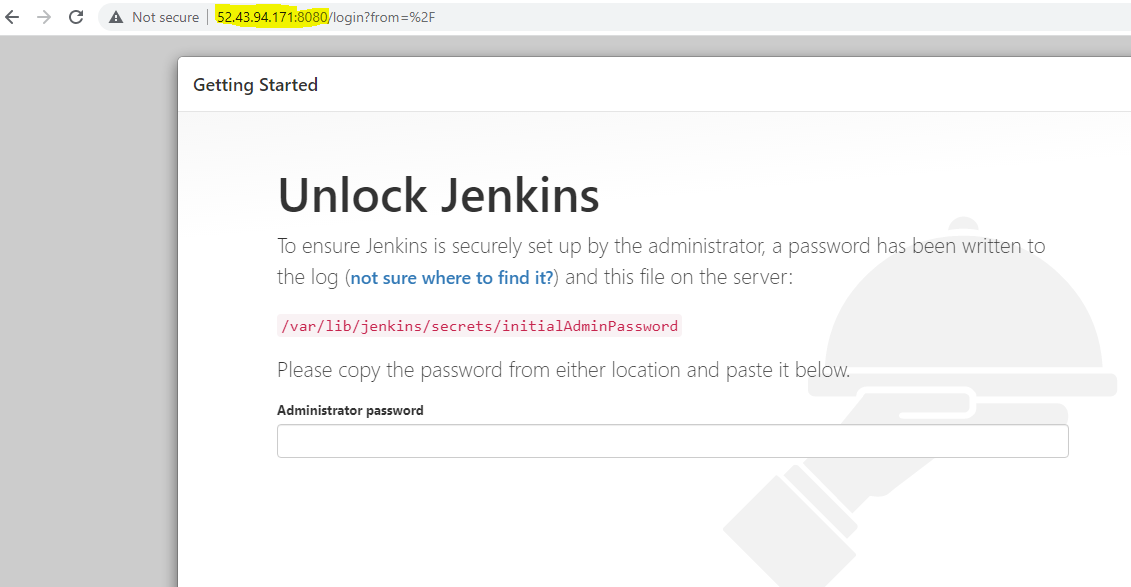
Outputs:

output\_name = "some resource"

PS C:\project\terraform-jenkins-master\terraform-jenkins-master>

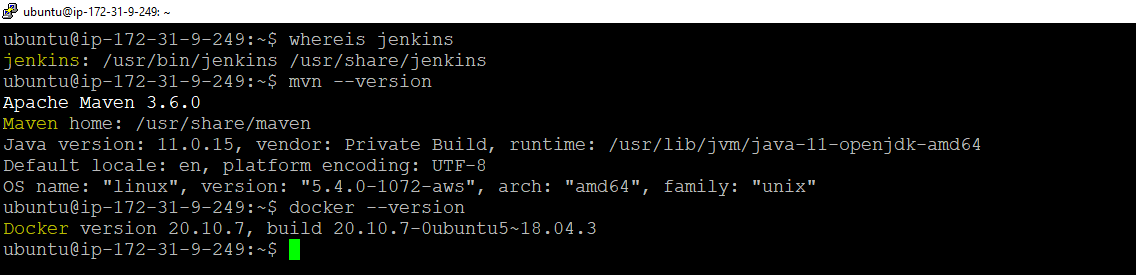
**AWS Console:**





**Tools Installed using Terraform:**

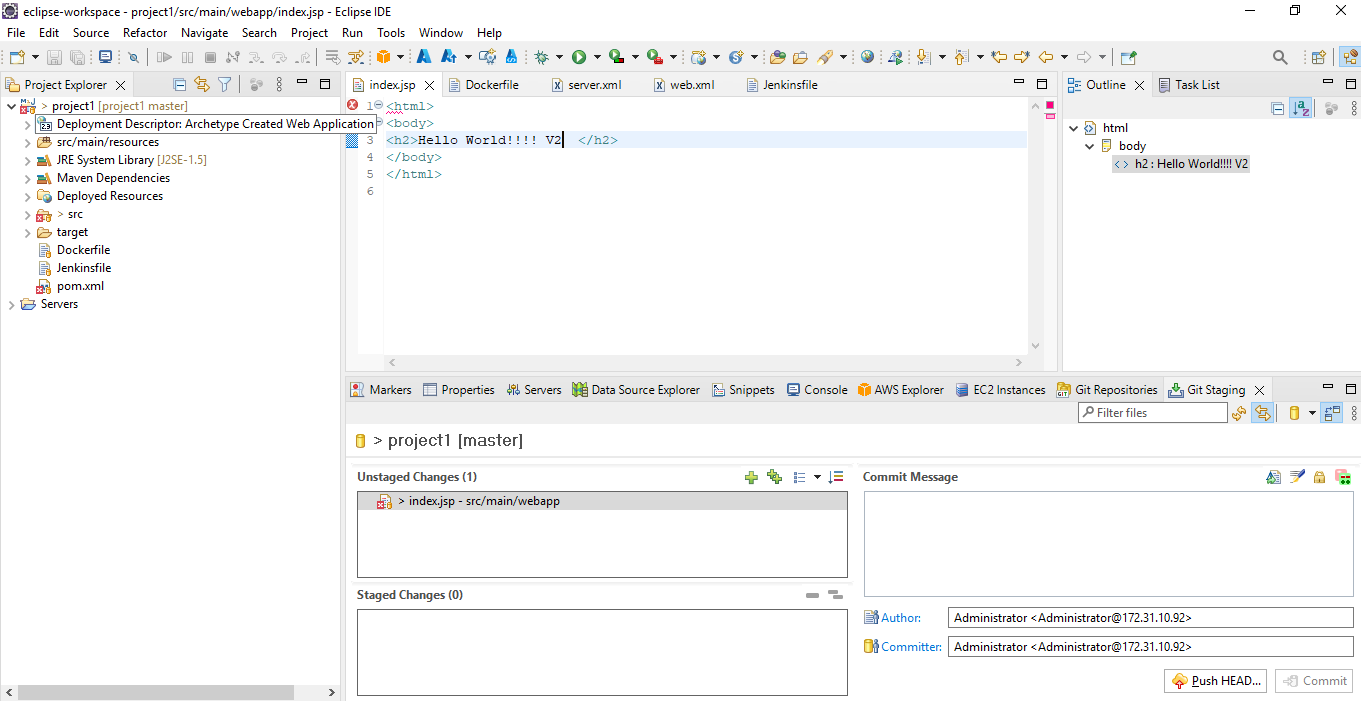
**Java, Maven, Jenkins, Docker:**



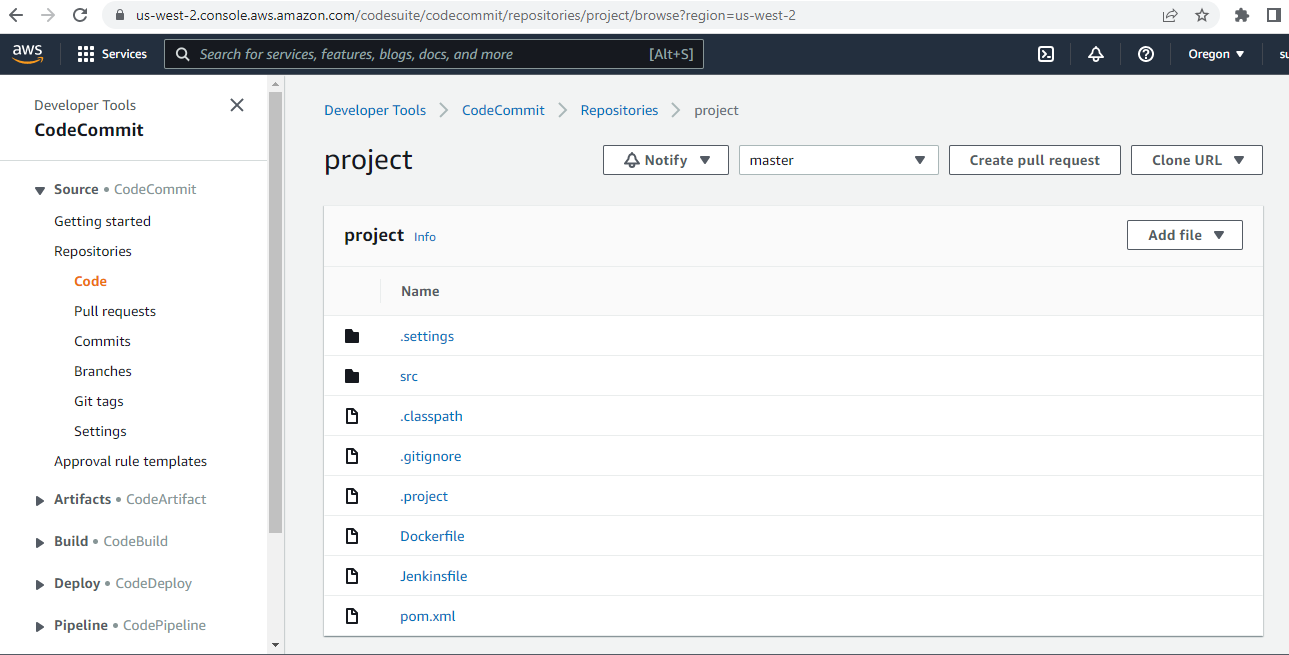
**PART 2:**

1: Simple Java WebApplication

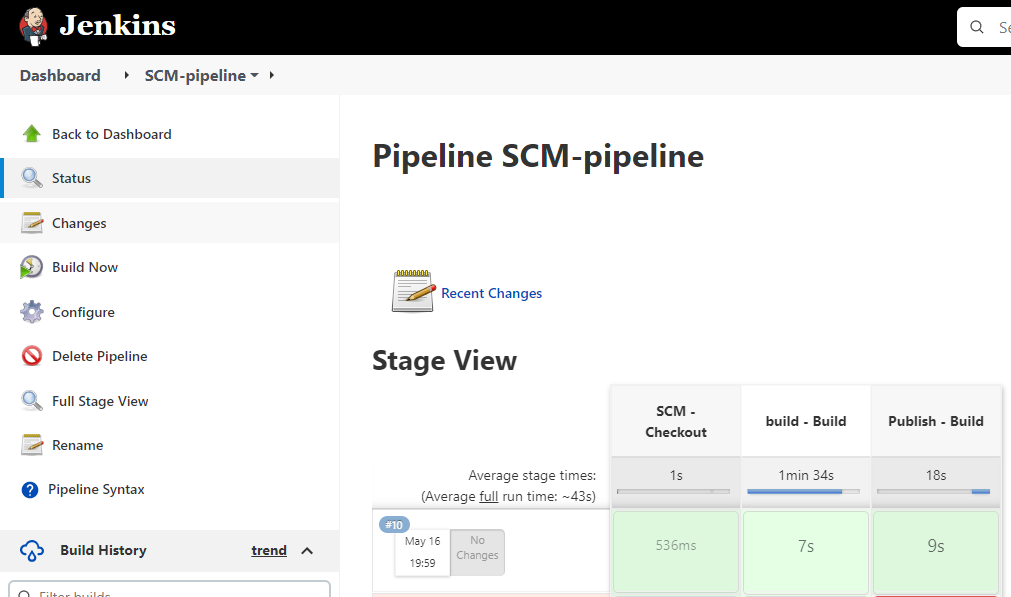
2: Containerize the webapplication using Dockerfile



Pushing Code and Dockerfile to Code commit



**Jenkins pipeline creation:**



**Jenkins pipeline code:**

**timestamps {**

**node () {**

**stage ('SCM - Checkout') {**

**checkout([$class: 'GitSCM', branches: [[name: '\*/master']], doGenerateSubmoduleConfigurations: false, extensions: [], submoduleCfg: [], userRemoteConfigs: [[credentialsId: 'awscodecommit', url: 'https://git-codecommit.us-west-2.amazonaws.com/v1/repos/project']]])**

**}**

**stage ('build - Build') {**

**sh """**

**mvn clean install test package**

**"""**

**}**

**stage ('Publish - Build') {**

**// Shell build step**

**sh """**

**docker build -t project1 .**

**docker tag project1:latest public.ecr.aws/n3p8x4e4/project1:latest**

**#docker tag project:latest 248183845077.dkr.ecr.us-west-2.amazonaws.com/project:latest**

**docker push public.ecr.aws/n3p8x4e4/project1:latest**

**#docker push 248183845077.dkr.ecr.us-west-2.amazonaws.com/project:latest**

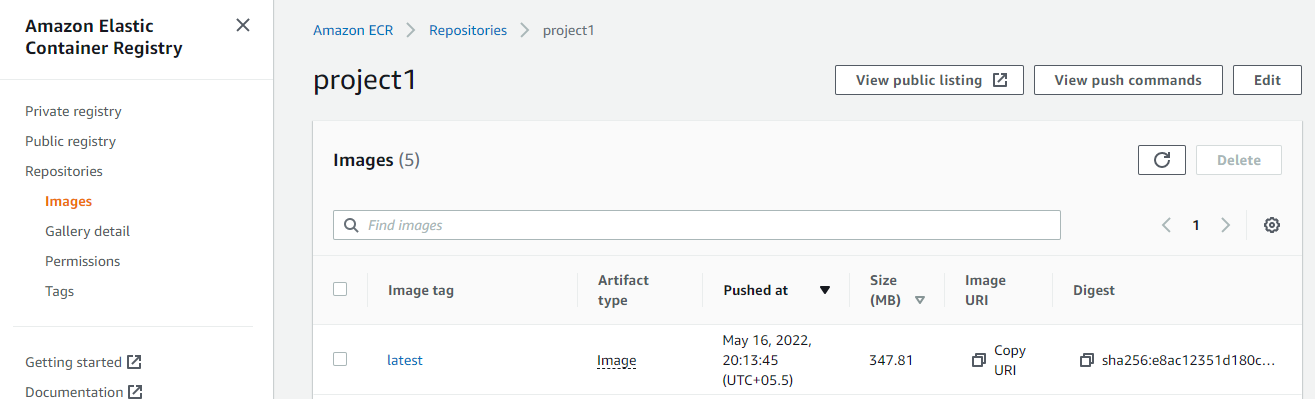
**"""**

**}**

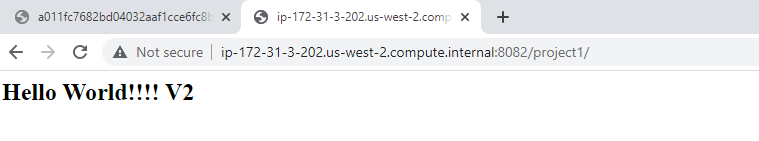
**}**

**}**

**Docker images in ECR**



**Once Deployed able to browse:**



**Automated Creation of EKS Cluster using eksctl Cloud Formation:**

**Commands: eksctl create cluster -f cluster.yaml**

**Cluster.yaml file:**

**apiVersion: eksctl.io/v1alpha5**

**kind: ClusterConfig**

**metadata:**

**name: project-demo #cluster name**

**region: us-west-2 #desired region**

**nodeGroups:**

**- name: ng-1 #cluster node group name**

**instanceType: t2.medium #desired instance type**

**desiredCapacity: 3 #desired nodes count / capacity**

**ssh:**

**allow: false # if true - will use ~/.ssh/id\_rsa.pub as the default ssh key**

**#publicKeyPath: ~/.ssh/ec2\_id\_rsa.pub #you can specify the public key path likr this as well**

**Output of the command:**

**2022-05-16 08:05:27 [ℹ] eksctl version 0.97.0**

**2022-05-16 08:05:27 [ℹ] using region us-west-2**

**2022-05-16 08:05:27 [ℹ] setting availability zones to [us-west-2d us-west-2b us-west-2c]**

**2022-05-16 08:05:27 [ℹ] subnets for us-west-2d - public:192.168.0.0/19 private:192.168.96.0/19**

**2022-05-16 08:05:27 [ℹ] subnets for us-west-2b - public:192.168.32.0/19 private:192.168.128.0/19**

**2022-05-16 08:05:27 [ℹ] subnets for us-west-2c - public:192.168.64.0/19 private:192.168.160.0/19**

**2022-05-16 08:05:27 [ℹ] nodegroup "ng-1" will use "ami-0cc74d90d5e3f4a46" [AmazonLinux2/1.22]**

**2022-05-16 08:05:27 [ℹ] using Kubernetes version 1.22**

**2022-05-16 08:05:27 [ℹ] creating EKS cluster "cicd-demo" in "us-west-2" region with un-managed nodes**

**2022-05-16 08:05:27 [ℹ] 1 nodegroup (ng-1) was included (based on the include/exclude rules)**

**2022-05-16 08:05:27 [ℹ] will create a CloudFormation stack for cluster itself and 1 nodegroup stack(s)**

**2022-05-16 08:05:27 [ℹ] will create a CloudFormation stack for cluster itself and 0 managed nodegroup stack(s)**

**2022-05-16 08:05:27 [ℹ] if you encounter any issues, check CloudFormation console or try 'eksctl utils describe-stacks --region=us-west-2 --cluster=cicd-demo'**

**2022-05-16 08:05:27 [ℹ] Kubernetes API endpoint access will use default of {publicAccess=true, privateAccess=false} for cluster "cicd-demo" in "us-west-2"**

**2022-05-16 08:05:27 [ℹ] CloudWatch logging will not be enabled for cluster "cicd-demo" in "us-west-2"**

**2022-05-16 08:05:27 [ℹ] you can enable it with 'eksctl utils update-cluster-logging --enable-types={SPECIFY-YOUR-LOG-TYPES-HERE (e.g. all)} --region=us-west-2 --cluster=cicd-demo'**

**2022-05-16 08:05:27 [ℹ]**

**2 sequential tasks: { create cluster control plane "cicd-demo",**

**2 sequential sub-tasks: {**

**wait for control plane to become ready,**

**create nodegroup "ng-1",**

**}**

**}**

**2022-05-16 08:05:27 [ℹ] building cluster stack "eksctl-cicd-demo-cluster"**

**2022-05-16 08:05:27 [ℹ] deploying stack "eksctl-cicd-demo-cluster"**

**2022-05-16 08:05:57 [ℹ] waiting for CloudFormation stack "eksctl-cicd-demo-cluster"**

**2022-05-16 08:06:27 [ℹ] waiting for CloudFormation stack "eksctl-cicd-demo-cluster"**

**2022-05-16 08:07:27 [ℹ] waiting for CloudFormation stack "eksctl-cicd-demo-cluster"**

**2022-05-16 08:08:27 [ℹ] waiting for CloudFormation stack "eksctl-cicd-demo-cluster"**

**2022-05-16 08:09:27 [ℹ] waiting for CloudFormation stack "eksctl-cicd-demo-cluster"**

**2022-05-16 08:10:27 [ℹ] waiting for CloudFormation stack "eksctl-cicd-demo-cluster"**

**2022-05-16 08:11:27 [ℹ] waiting for CloudFormation stack "eksctl-cicd-demo-cluster"**

**2022-05-16 08:12:28 [ℹ] waiting for CloudFormation stack "eksctl-cicd-demo-cluster"**

**2022-05-16 08:13:28 [ℹ] waiting for CloudFormation stack "eksctl-cicd-demo-cluster"**

**2022-05-16 08:14:28 [ℹ] waiting for CloudFormation stack "eksctl-cicd-demo-cluster"**

**2022-05-16 08:15:28 [ℹ] waiting for CloudFormation stack "eksctl-cicd-demo-cluster"**

**2022-05-16 08:16:28 [ℹ] waiting for CloudFormation stack "eksctl-cicd-demo-cluster"**

**2022-05-16 08:18:28 [ℹ] building nodegroup stack "eksctl-cicd-demo-nodegroup-ng-1"**

**2022-05-16 08:18:28 [ℹ] --nodes-min=3 was set automatically for nodegroup ng-1**

**2022-05-16 08:18:28 [ℹ] --nodes-max=3 was set automatically for nodegroup ng-1**

**2022-05-16 08:18:29 [ℹ] deploying stack "eksctl-cicd-demo-nodegroup-ng-1"**

**2022-05-16 08:18:29 [ℹ] waiting for CloudFormation stack "eksctl-cicd-demo-nodegroup-ng-1"**

**2022-05-16 08:18:59 [ℹ] waiting for CloudFormation stack "eksctl-cicd-demo-nodegroup-ng-1"**

**2022-05-16 08:19:40 [ℹ] waiting for CloudFormation stack "eksctl-cicd-demo-nodegroup-ng-1"**

**2022-05-16 08:20:25 [ℹ] waiting for CloudFormation stack "eksctl-cicd-demo-nodegroup-ng-1"**

**2022-05-16 08:22:15 [ℹ] waiting for CloudFormation stack "eksctl-cicd-demo-nodegroup-ng-1"**

**2022-05-16 08:23:34 [ℹ] waiting for CloudFormation stack "eksctl-cicd-demo-nodegroup-ng-1"**

**2022-05-16 08:23:34 [ℹ] waiting for the control plane availability...**

**2022-05-16 08:23:35 [✔] saved kubeconfig as "/home/ubuntu/.kube/config"**

**2022-05-16 08:23:35 [ℹ] no tasks**

**2022-05-16 08:23:35 [✔] all EKS cluster resources for "cicd-demo" have been created**

**2022-05-16 08:23:35 [ℹ] adding identity "arn:aws:iam::248183845077:role/eksctl-cicd-demo-nodegroup-ng-1-NodeInstanceRole-PU6BTM0KP734" to auth ConfigMap**

**2022-05-16 08:23:35 [ℹ] nodegroup "ng-1" has 0 node(s)**

**2022-05-16 08:23:35 [ℹ] waiting for at least 3 node(s) to become ready in "ng-1"**

**2022-05-16 08:24:38 [ℹ] nodegroup "ng-1" has 3 node(s)**

**2022-05-16 08:24:38 [ℹ] node "ip-192-168-53-195.us-west-2.compute.internal" is ready**

**2022-05-16 08:24:38 [ℹ] node "ip-192-168-63-173.us-west-2.compute.internal" is ready**

**2022-05-16 08:24:38 [ℹ] node "ip-192-168-94-249.us-west-2.compute.internal" is ready**

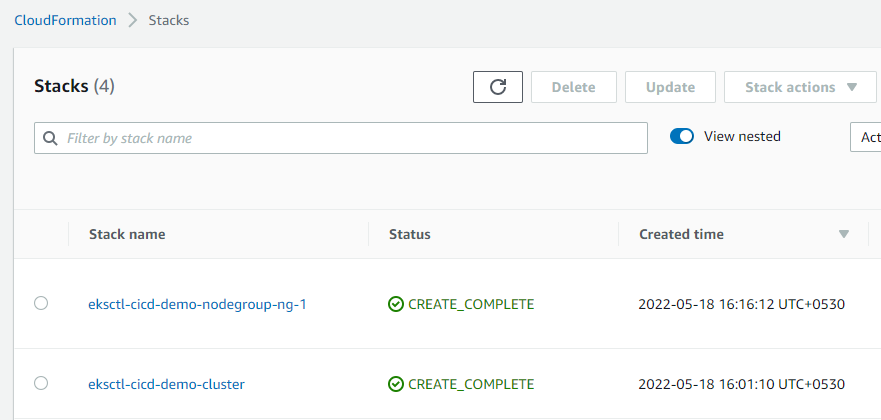
**2022-05-16 08:24:38 [✖] parsing kubectl version string (upstream error: error: exec plugin: invalid apiVersion "client.authentication.k8s.io/v1alpha1"**

**) / "0.0.0": Version string empty**

**2022-05-16 08:24:38 [ℹ] cluster should be functional despite missing (or misconfigured) client binaries**

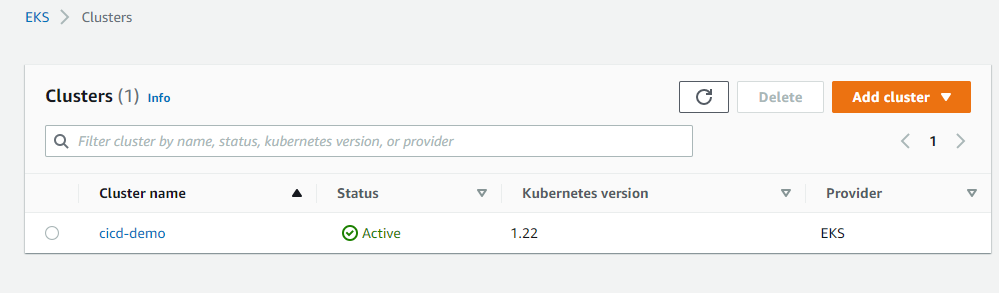
**2022-05-16 08:24:38 [✔] EKS cluster "cicd-demo" in "us-west-2" region is ready**

**Cloud Formation Stack:**

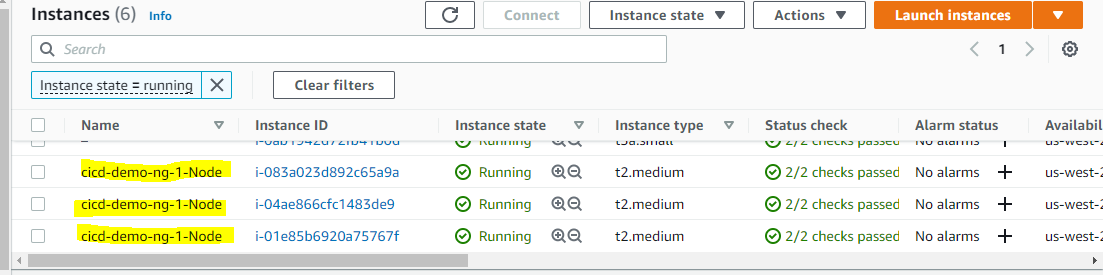


**EKS Cluster:**

EKS created with name cicd-demo:



**EC2: Worker nodes:**



Code pipeline creation:

Buildspec\_eks.yml file:

To check all the installations: AWS CLI, Kubectl, eksctl and other pre-requisites:

version: 0.2

run-as: root

phases:

install:

commands:

- echo Installing app dependencies...

- curl -o kubectl https://amazon-eks.s3.us-west-2.amazonaws.com/1.18.9/2020-11-02/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

- source ~/.bashrc

- echo 'Check kubectl version'

- kubectl version --short --client

- chmod +x eks\_cicd/prereqs.sh

- sh eks\_cicd/prereqs.sh

pre\_build:

commands:

- echo Logging in to Amazon EKS...

- aws eks --region $AWS\_DEFAULT\_REGION update-kubeconfig --name $AWS\_CLUSTER\_NAME

- echo check config

- kubectl config view --minify

- echo check kubectl access

- kubectl get svc

- echo Logging in to Amazon ECR...

- aws ecr get-login-password --region $AWS\_DEFAULT\_REGION | docker login --username AWS --password-stdin $AWS\_ACCOUNT\_ID.dkr.ecr.$AWS\_DEFAULT\_REGION.amazonaws.com

- REPOSITORY\_URI=$AWS\_ACCOUNT\_ID.dkr.ecr.$AWS\_DEFAULT\_REGION.amazonaws.com/$IMAGE\_REPO\_NAME

- docker pull $REPOSITORY\_URI:$IMAGE\_TAG

build:

commands:

- echo Build started on `date`

- echo Building the Docker image...

- docker build --cache-from $REPOSITORY\_URI:$IMAGE\_TAG -t $IMAGE\_REPO\_NAME:$IMAGE\_TAG .

- docker tag $IMAGE\_REPO\_NAME:$IMAGE\_TAG $AWS\_ACCOUNT\_ID.dkr.ecr.$AWS\_DEFAULT\_REGION.amazonaws.com/$IMAGE\_REPO\_NAME:$IMAGE\_TAG

post\_build:

commands:

- echo Build completed on `date`

- echo Pushing the Docker image...

- docker push $AWS\_ACCOUNT\_ID.dkr.ecr.$AWS\_DEFAULT\_REGION.amazonaws.com/$IMAGE\_REPO\_NAME:$IMAGE\_TAG

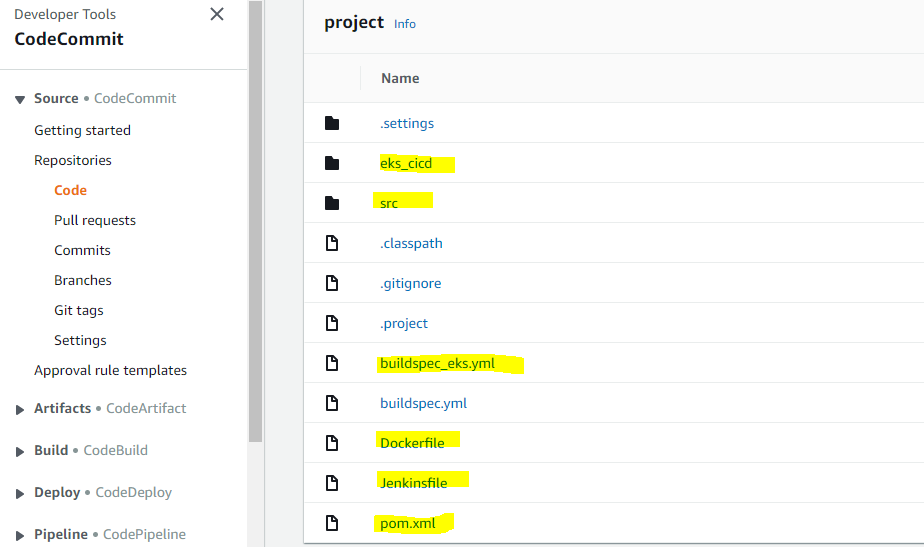
- echo Push the latest image to cluster

- kubectl apply -f eks\_cicd/deployment.yaml

- kubectl rollout restart -f eks\_cicd/deployment.yaml

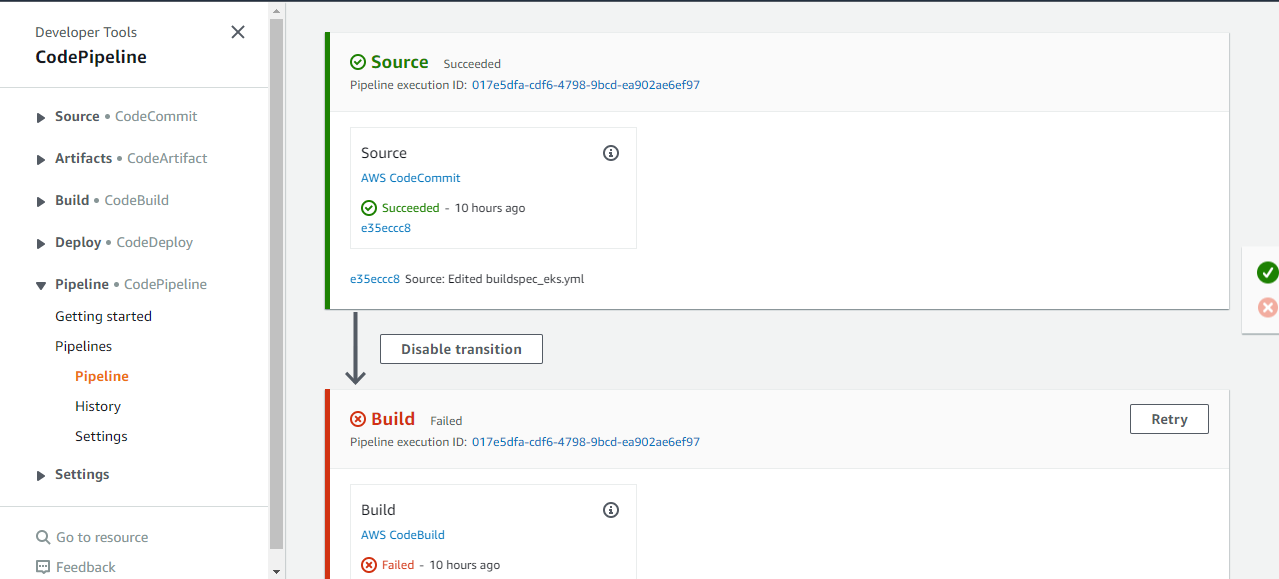
Code pipeline with two stages:

AWS code commit:



AWS code pipeline:

Code pipeline contains: SCM ( Code commit ) checkout and Code Build to deploy the image which in ECR container to the pods: ( Please note: I am getting below error while deploying taking help from Paparao sir fix that so Build is getting failed )



Build output with Error:

|  |
| --- |
| [Container] 2022/05/18 05:52:01 Waiting for agent ping |
| 2 | [Container] 2022/05/18 05:52:02 Waiting for DOWNLOAD\_SOURCE |
| 3 | [Container] 2022/05/18 05:52:03 Phase is DOWNLOAD\_SOURCE |
| 4 | [Container] 2022/05/18 05:52:03 CODEBUILD\_SRC\_DIR=/codebuild/output/src034418456/src |
| 5 | [Container] 2022/05/18 05:52:03 YAML location is /codebuild/output/src034418456/src/buildspec\_eks.yml |
| 6 | [Container] 2022/05/18 05:52:03 Processing environment variables |
| 7 | [Container] 2022/05/18 05:52:03 No runtime version selected in buildspec. |
| 8 | [Container] 2022/05/18 05:52:05 Run-as user: root |
| 9 | [Container] 2022/05/18 05:52:05 Moving to directory /codebuild/output/src034418456/src |
| 10 | [Container] 2022/05/18 05:52:05 Configuring ssm agent with target id: codebuild:9fbb3a89-5b7a-4d07-bb3a-51f65b732087 |
| 11 | [Container] 2022/05/18 05:52:05 Successfully updated ssm agent configuration |
| 12 | [Container] 2022/05/18 05:52:05 Registering with agent |
| 13 | [Container] 2022/05/18 05:52:05 Phases found in YAML: 4 |
| 14 | [Container] 2022/05/18 05:52:05 PRE\_BUILD: 14 commands |
| 15 | [Container] 2022/05/18 05:52:05 BUILD: 1 commands |
| 16 | [Container] 2022/05/18 05:52:05 POST\_BUILD: 7 commands |
| 17 | [Container] 2022/05/18 05:52:05 INSTALL: 10 commands |
| 18 | [Container] 2022/05/18 05:52:05 Phase complete: DOWNLOAD\_SOURCE State: SUCCEEDED |
| 19 | [Container] 2022/05/18 05:52:05 Phase context status code: Message: |
| 20 | [Container] 2022/05/18 05:52:05 Entering phase INSTALL |
| 21 | [Container] 2022/05/18 05:52:05 Running command echo Installing app dependencies... |
| 22 | Installing app dependencies... |
| 23 |  |
| 24 | [Container] 2022/05/18 05:52:06 Running command curl -o kubectl https://amazon-eks.s3.us-west-2.amazonaws.com/1.18.9/2020-11-02/bin/linux/amd64/kubectl |
| 25 | % Total % Received % Xferd Average Speed Time Time Time Current |
| 26 | Dload Upload Total Spent Left Speed |
| 27 |  |
| 28 | 0 0 0 0 0 0 0 0 --:--:-- --:--:-- --:--:-- 0 |
| 29 | 7 57.2M 7 4605k 0 0 54.5M 0 0:00:01 --:--:-- 0:00:01 54.1M |
| 30 | 100 57.2M 100 57.2M 0 0 90.0M 0 --:--:-- --:--:-- --:--:-- 89.9M |
| 31 |  |
| 32 | [Container] 2022/05/18 05:52:07 Running command chmod +x ./kubectl |
| 33 |  |
| 34 | [Container] 2022/05/18 05:52:07 Running command mkdir -p $HOME/bin && cp ./kubectl $HOME/bin/kubectl && export PATH=$PATH:$HOME/bin |
| 35 |  |
| 36 | [Container] 2022/05/18 05:52:07 Running command echo 'export PATH=$PATH:$HOME/bin' >> ~/.bashrc |
| 37 |  |
| 38 | [Container] 2022/05/18 05:52:07 Running command source ~/.bashrc |
| 39 |  |
| 40 | [Container] 2022/05/18 05:52:07 Running command echo 'Check kubectl version' |
| 41 | Check kubectl version |
| 42 |  |
| 43 | [Container] 2022/05/18 05:52:07 Running command kubectl version --short --client |
| 44 | Client Version: v1.16.8-eks-e16311 |
| 45 |  |
| 46 | [Container] 2022/05/18 05:52:09 Running command chmod +x eks\_cicd/prereqs.sh |
| 47 |  |
| 48 | [Container] 2022/05/18 05:52:09 Running command sh eks\_cicd/prereqs.sh |
| 49 | kubectl already installed |
| 50 | aws-iam-authenticator already installed |
| 51 | AWS CLI already installed |
| 52 | eksctl already installed |
| 53 | Cloning into '/opt/kubectx'... |
| 54 | kubectx installed |
| 55 | { |
| 56 | "UserId": "AROATTSHI6DKV3C5EBQ4E:AWSCodeBuild-9fbb3a89-5b7a-4d07-bb3a-51f65b732087", |
| 57 | "Account": "248183845077", |
| 58 | "Arn": "arn:aws:sts::248183845077:assumed-role/CodeBuildKubectlRole/AWSCodeBuild-9fbb3a89-5b7a-4d07-bb3a-51f65b732087" |
| 59 | } |
| 60 |  |
| 61 | [Container] 2022/05/18 05:52:15 Phase complete: INSTALL State: SUCCEEDED |
| 62 | [Container] 2022/05/18 05:52:15 Phase context status code: Message: |
| 63 | [Container] 2022/05/18 05:52:15 Entering phase PRE\_BUILD |
| 64 | [Container] 2022/05/18 05:52:15 Running command aws sts get-caller-identity |
| 65 | { |
| 66 | "UserId": "AROATTSHI6DKV3C5EBQ4E:AWSCodeBuild-9fbb3a89-5b7a-4d07-bb3a-51f65b732087", |
| 67 | "Account": "248183845077", |
| 68 | "Arn": "arn:aws:sts::248183845077:assumed-role/CodeBuildKubectlRole/AWSCodeBuild-9fbb3a89-5b7a-4d07-bb3a-51f65b732087" |
| 69 | } |
| 70 |  |
| 71 | [Container] 2022/05/18 05:52:16 Running command echo Logging in to Amazon EKS... |
| 72 | Logging in to Amazon EKS... |
| 73 |  |
| 74 | [Container] 2022/05/18 05:52:16 Running command aws eks --region us-west-2 update-kubeconfig --name cicd-demo --role-arn arn:aws:iam::248183845077:role/CodeBuildKubectlRole |
| 75 | Added new context arn:aws:eks:us-west-2:248183845077:cluster/cicd-demo to /root/.kube/config |
| 76 |  |
| 77 | [Container] 2022/05/18 05:52:17 Running command aws eks update-kubeconfig --name cicd-demo --region us-west-2 --role-arn arn:aws:iam::248183845077:role/CodeBuildKubectlRole |
| 78 | Updated context arn:aws:eks:us-west-2:248183845077:cluster/cicd-demo in /root/.kube/config |
| 79 |  |
| 80 | [Container] 2022/05/18 05:52:17 Running command echo check config |
| 81 | check config |
| 82 |  |
| 83 | [Container] 2022/05/18 05:52:17 Running command kubectl config view --minify |
| 84 | apiVersion: v1 |
| 85 | clusters: |
| 86 | - cluster: |
| 87 | certificate-authority-data: DATA+OMITTED |
| 88 | server: https://48E12EEDF98C79498E739B1A1E7790B6.gr7.us-west-2.eks.amazonaws.com |
| 89 | name: arn:aws:eks:us-west-2:248183845077:cluster/cicd-demo |
| 90 | contexts: |
| 91 | - context: |
| 92 | cluster: arn:aws:eks:us-west-2:248183845077:cluster/cicd-demo |
| 93 | user: arn:aws:eks:us-west-2:248183845077:cluster/cicd-demo |
| 94 | name: arn:aws:eks:us-west-2:248183845077:cluster/cicd-demo |
| 95 | current-context: arn:aws:eks:us-west-2:248183845077:cluster/cicd-demo |
| 96 | kind: Config |
| 97 | preferences: {} |
| 98 | users: |
| 99 | - name: arn:aws:eks:us-west-2:248183845077:cluster/cicd-demo |
| 100 | user: |
| 101 | exec: |
| 102 | apiVersion: client.authentication.k8s.io/v1alpha1 |
| 103 | args: |
| 104 | - --region |
| 105 | - us-west-2 |
| 106 | - eks |
| 107 | - get-token |
| 108 | - --cluster-name |
| 109 | - cicd-demo |
| 110 | - --role |
| 111 | - arn:aws:iam::248183845077:role/CodeBuildKubectlRole |
| 112 | command: aws |
| 113 | env: null |
| 114 |  |
| 115 | [Container] 2022/05/18 05:52:17 Running command cd eks\_cicd |
| 116 |  |
| 117 | [Container] 2022/05/18 05:52:17 Running command pwd |
| 118 | /codebuild/output/src034418456/src/eks\_cicd |
| 119 |  |
| 120 | [Container] 2022/05/18 05:52:18 Running command ls -l |
| 121 | total 28 |
| 122 | -rw-r--r-- 1 root root 1899 May 18 05:52 aws-auth.yaml |
| 123 | -rw-r--r-- 1 root root 481 May 18 05:52 cluster.yaml |
| 124 | -rw-r--r-- 1 root root 478 May 18 05:52 cluster.yaml.bak |
| 125 | -rw-r--r-- 1 root root 1263 May 18 05:52 create\_iam\_role.sh |
| 126 | -rw-r--r-- 1 root root 606 May 18 05:52 deployment.yaml |
| 127 | -rwxr-xr-x 1 root root 1893 May 18 05:52 prereqs.sh |
| 128 | -rw-r--r-- 1 root root 403 May 18 05:52 service.yaml |
| 129 |  |
| 130 | [Container] 2022/05/18 05:52:18 Running command cat aws-auth.yaml |
| 131 | apiVersion: v1 |
| 132 | data: |
| 133 | mapRoles: | |
| 134 | - groups: |
| 135 | - system:bootstrappers |
| 136 | - system:nodes |
| 137 | rolearn: arn:aws:iam::248183845077:role/eksctl-cicd-demo-nodegroup-ng-1-NodeInstanceRole-16C6Y2HYTPM57 |
| 138 | username: system:node:{{EC2PrivateDNSName}} |
| 139 | - groups: |
| 140 | - system:masters |
| 141 | rolearn: arn:aws:iam::248183845077:role/service-role/codebuild-projectdemo-service-role |
| 142 | username: codebuild-projectdemo-service-role |
| 143 | - groups: |
| 144 | - system:masters |
| 145 | rolearn: arn:aws:iam::248183845077:role/CodeBuildKubectlRole |
| 146 | username: CodeBuildKubectlRole |
| 147 | - groups: |
| 148 | - system:masters |
| 149 | rolearn: arn:aws:iam::248183845077:role/role-name |
| 150 | username: role-name |
| 151 | - system:masters |
| 152 | mapUsers: | |
| 153 | [] |
| 154 | kind: ConfigMap |
| 155 | metadata: |
| 156 | annotations: |
| 157 | kubectl.kubernetes.io/last-applied-configuration: | |
| 158 | {"apiVersion":"v1","data":{"mapRoles":"- groups:\n - system:bootstrappers\n - system:nodes\n rolearn: arn:aws:iam::248183845077:role/eksctl-cicd-demo-nodegroup-ng-1-NodeInstanceRole-16C6Y2HYTPM57\n username: system:node:{{EC2PrivateDNSName}}\n- groups:\n - system:masters\n rolearn: arn:aws:iam::248183845077:role/CodeBuildKubectlRole\n username: CodeBuildKubectlRole\n- groups:\n - system:masters\n rolearn: arn:aws:iam::248183845077:role/CodeBuildKubectlRole\n username: CodeBuildKubectlRole\n - groups:\n - system:masters\n rolearn: arn:aws:iam::248183845077:role/role-name\n username: role-name\n - system:masters\n","mapUsers":"[]\n"},"kind":"ConfigMap","metadata":{"annotations":{},"creationTimestamp":"2022-05-16T19:03:04Z","name":"aws-auth","namespace":"kube-system","resourceVersion":"15984","uid":"90720530-cdc8-4232-921c-9555828942fd"}} |
| 159 | creationTimestamp: "2022-05-16T19:03:04Z" |
| 160 | name: aws-auth |
| 161 | namespace: kube-system |
| 162 | resourceVersion: "176094" |
| 163 | uid: 90720530-cdc8-4232-921c-9555828942fd |
| 164 | [Container] 2022/05/18 05:52:18 Running command aws ecr get-login-password --region us-west-2 | docker login --username AWS --password-stdin 248183845077.dkr.ecr.us-west-2.amazonaws.com |
| 165 | WARNING! Your password will be stored unencrypted in /root/.docker/config.json. |
| 166 | Configure a credential helper to remove this warning. See |
| 167 | https://docs.docker.com/engine/reference/commandline/login/#credentials-store |
| 168 |  |
| 169 | Login Succeeded |
| 170 |  |
| 171 | [Container] 2022/05/18 05:52:18 Running command echo check kubectl access |
| 172 | check kubectl access |
| 173 |  |
| 174 | [Container] 2022/05/18 05:52:18 Running command aws sts get-caller-identity |
| 175 | { |
| 176 | "UserId": "AROATTSHI6DKV3C5EBQ4E:AWSCodeBuild-9fbb3a89-5b7a-4d07-bb3a-51f65b732087", |
| 177 | "Account": "248183845077", |
| 178 | "Arn": "arn:aws:sts::248183845077:assumed-role/CodeBuildKubectlRole/AWSCodeBuild-9fbb3a89-5b7a-4d07-bb3a-51f65b732087" |
| 179 | } |
| 180 |  |
| 181 | [Container] 2022/05/18 05:52:19 Running command kubectl get svc |
| 182 | error: You must be logged in to the server (Unauthorized) |
| 183 |  |
| 184 | [Container] 2022/05/18 05:52:24 Command did not exit successfully kubectl get svc exit status 1 |
| 185 | [Container] 2022/05/18 05:52:24 Phase complete: PRE\_BUILD State: FAILED |
| 186 | [Container] 2022/05/18 05:52:24 Phase context status code: COMMAND\_EXECUTION\_ERROR Message: Error while executing command: kubectl get svc. Reason: exit status 1 |
| 187 |  |

Feedback

Looking for language selection? Find it in the new [Unified Settings](https://us-west-2.console.aws.amazon.com/settings/home?region=us-west-2)