AWS CICD PIPELINE

Amazon Web Services (AWS) offers multiple services like Elastic Beanstalk, ECS, and EKS for deploying Dockerized applications.

| **Property** | **Description** | **Example** |
| --- | --- | --- |
| **Code Deployment** | Use Elastic Beanstalk for simple setups or ECS/EKS for more complex applications. | eb init -p docker my-app <br> eb create my-app-env |
| **Framework Update** | Update Docker image and push to Amazon Elastic Container Registry (ECR). | docker build -t mydockerimage . <br> aws ecr create-repository --repository-name mydockerimage <br> docker push myaccountid.dkr.ecr.us-west-2.amazonaws.com/mydockerimage |
| **Database Timeout** | Configure RDS parameters and use connection pooling libraries. | aws rds create-db-instance --db-instance-identifier mydb --allocated-storage 20 --db-instance-class db.t2.micro --engine mysql --master-username myuser --master-user-password mypassword |
| **Disaster Recovery** | Use AWS Backup and enable Multi-AZ deployment for RDS. | aws backup create-backup-vault --backup-vault-name myBackupVault |
| **Bug Tracking** | Integrate with AWS CodePipeline and CloudWatch for CI/CD and monitoring. | aws codepipeline create-pipeline --cli-input-json file://pipeline.json |

Deploy in Amazon Elastic Kubernetes Service(EKS)

Download eksctl:using ubuntu

Install eksctl:

saikumar@DESKTOP-1JNARAI:~$ sudo curl --silent --location "https://github.com/weaveworks/eksctl/releases/latest/download/eksctl\_$( uname -s)\_amd64.tar.gz" | tar xz -C /tmp

saikumar@DESKTOP-1JNARAI:~$ sudo mv /tmp/eksctl /usr/local/bin

saikumar@DESKTOP-1JNARAI:~$ eksctl --help

saikumar@DESKTOP-1JNARAI:~$ sudo apt update && sudo apt upgrade -y

saikumar@DESKTOP-1JNARAI:~$ aws eks update-kubeconfig --region=ap-south-1 --name=my-eks-cluster3

saikumar@DESKTOP-1JNARAI:~$ sudo apt install -y eksctl

Verify the Installation:

saikumar@DESKTOP-1JNARAI:~$ eksctl version

create cluster:

saikumar@DESKTOP-1JNARAI:~$ eksctl create cluster --name=my-eks-cluster --region=ap-south-1 --nodes=3 --node-type=m5.large

saikumar@DESKTOP-1JNARAI:~$ eksctl create cluster --name=my-eks-cluster --region=ap-south-1 --nodes=3 --node-type=m5.large

2024-02-02 01:11:59 [ℹ] eksctl version 0.169.0

2024-02-02 01:11:59 [ℹ] using region ap-south-1

2024-02-02 01:12:01 [ℹ] setting availability zones to [ap-south-1a ap-south-1c ap-south-1b]

2024-02-02 01:12:01 [ℹ] subnets for ap-south-1a - public:192.168.0.0/19 private:192.168.96.0/19

2024-02-02 01:12:01 [ℹ] subnets for ap-south-1c - public:192.168.32.0/19 private:192.168.128.0/19

2024-02-02 01:12:01 [ℹ] subnets for ap-south-1b - public:192.168.64.0/19 private:192.168.160.0/19

2024-02-02 01:12:01 [ℹ] nodegroup "ng-15da3fa2" will use "" [AmazonLinux2/1.27]

2024-02-02 01:12:01 [ℹ] using Kubernetes version 1.27

2024-02-02 01:12:01 [ℹ] creating EKS cluster "my-eks-cluster" in "ap-south-1" region with managed nodes

2024-02-02 01:12:01 [ℹ] will create 2 separate CloudFormation stacks for cluster itself and the initial managed nodegroup

2024-02-02 01:12:01 [ℹ] if you encounter any issues, check CloudFormation console or try 'eksctl utils describe-stacks --region=ap-south-1 --cluster=my-eks-cluster'

2024-02-02 01:12:01 [ℹ] Kubernetes API endpoint access will use default of {publicAccess=true, privateAccess=false} for cluster "my-eks-cluster" in "ap-south-1"

2024-02-02 01:12:01 [ℹ] CloudWatch logging will not be enabled for cluster "my-eks-cluster" in "ap-south-1"

2024-02-02 01:12:01 [ℹ] you can enable it with 'eksctl utils update-cluster-logging --enable-types={SPECIFY-YOUR-LOG-TYPES-HERE (e.g. all)} --region=ap-south-1 --cluster=my-eks-cluster'

2024-02-02 01:12:01 [ℹ]

2 sequential tasks: { create cluster control plane "my-eks-cluster",

2 sequential sub-tasks: {

wait for control plane to become ready,

create managed nodegroup "ng-15da3fa2",

}

}

2024-02-02 01:12:01 [ℹ] building cluster stack "eksctl-my-eks-cluster-cluster"

2024-02-02 01:12:04 [ℹ] deploying stack "eksctl-my-eks-cluster-cluster"

2024-02-02 01:12:34 [ℹ] waiting for CloudFormation stack "eksctl-my-eks-cluster-cluster"

2024-02-02 01:13:06 [ℹ] waiting for CloudFormation stack "eksctl-my-eks-cluster-cluster"

2024-02-02 01:14:08 [ℹ] waiting for CloudFormation stack "eksctl-my-eks-cluster-cluster"

2024-02-02 01:15:10 [ℹ] waiting for CloudFormation stack "eksctl-my-eks-cluster-cluster"

2024-02-02 01:16:12 [ℹ] waiting for CloudFormation stack "eksctl-my-eks-cluster-cluster"

2024-02-02 01:17:14 [ℹ] waiting for CloudFormation stack "eksctl-my-eks-cluster-cluster"

2024-02-02 01:18:17 [ℹ] waiting for CloudFormation stack "eksctl-my-eks-cluster-cluster"

2024-02-02 01:19:19 [ℹ] waiting for CloudFormation stack "eksctl-my-eks-cluster-cluster"

2024-02-02 01:21:34 [ℹ] building managed nodegroup stack "eksctl-my-eks-cluster-nodegroup-ng-15da3fa2"

2024-02-02 01:21:37 [ℹ] deploying stack "eksctl-my-eks-cluster-nodegroup-ng-15da3fa2"

2024-02-02 01:21:37 [ℹ] waiting for CloudFormation stack "eksctl-my-eks-cluster-nodegroup-ng-15da3fa2"

2024-02-02 01:22:09 [ℹ] waiting for CloudFormation stack "eksctl-my-eks-cluster-nodegroup-ng-15da3fa2"

2024-02-02 01:22:49 [ℹ] waiting for CloudFormation stack "eksctl-my-eks-cluster-nodegroup-ng-15da3fa2"

2024-02-02 01:23:58 [ℹ] waiting for CloudFormation stack "eksctl-my-eks-cluster-nodegroup-ng-15da3fa2"

2024-02-02 01:25:17 [ℹ] waiting for CloudFormation stack "eksctl-my-eks-cluster-nodegroup-ng-15da3fa2"

2024-02-02 01:26:55 [ℹ] waiting for CloudFormation stack "eksctl-my-eks-cluster-nodegroup-ng-15da3fa2"

2024-02-02 01:28:19 [ℹ] waiting for CloudFormation stack "eksctl-my-eks-cluster-nodegroup-ng-15da3fa2"

2024-02-02 01:29:25 [ℹ] waiting for CloudFormation stack "eksctl-my-eks-cluster-nodegroup-ng-15da3fa2"

2024-02-02 01:30:58 [ℹ] waiting for CloudFormation stack "eksctl-my-eks-cluster-nodegroup-ng-15da3fa2"

2024-02-02 01:32:49 [ℹ] waiting for CloudFormation stack "eksctl-my-eks-cluster-nodegroup-ng-15da3fa2"

2024-02-02 01:34:31 [ℹ] waiting for CloudFormation stack "eksctl-my-eks-cluster-nodegroup-ng-15da3fa2"

2024-02-02 01:36:18 [ℹ] waiting for CloudFormation stack "eksctl-my-eks-cluster-nodegroup-ng-15da3fa2"

2024-02-02 01:37:14 [ℹ] waiting for CloudFormation stack "eksctl-my-eks-cluster-nodegroup-ng-15da3fa2"

2024-02-02 01:38:12 [ℹ] waiting for CloudFormation stack "eksctl-my-eks-cluster-nodegroup-ng-15da3fa2"

2024-02-02 01:39:12 [ℹ] waiting for CloudFormation stack "eksctl-my-eks-cluster-nodegroup-ng-15da3fa2"

2024-02-02 01:39:13 [ℹ] waiting for the control plane to become ready

2024-02-02 01:39:13 [!] failed to determine authenticator version, leaving API version as default v1alpha1: failed to retrieve authenticator version: failed to run aws-iam-authenticator version command: exec: "aws-iam-authenticator": executable file not found in $PATH

2024-02-02 01:39:13 [✔] saved kubeconfig as "/home/saikumar/.kube/config"

2024-02-02 01:39:13 [ℹ] no tasks

2024-02-02 01:39:13 [✔] all EKS cluster resources for "my-eks-cluster" have been created

2024-02-02 01:39:16 [ℹ] nodegroup "ng-15da3fa2" has 4 node(s)

2024-02-02 01:39:16 [ℹ] node "ip-192-168-19-165.ap-south-1.compute.internal" is ready

2024-02-02 01:39:16 [ℹ] node "ip-192-168-58-214.ap-south-1.compute.internal" is ready

2024-02-02 01:39:16 [ℹ] node "ip-192-168-9-110.ap-south-1.compute.internal" is ready

2024-02-02 01:39:16 [ℹ] node "ip-192-168-94-118.ap-south-1.compute.internal" is ready

2024-02-02 01:39:16 [ℹ] waiting for at least 3 node(s) to become ready in "ng-15da3fa2"

2024-02-02 01:39:16 [ℹ] nodegroup "ng-15da3fa2" has 4 node(s)

2024-02-02 01:39:16 [ℹ] node "ip-192-168-19-165.ap-south-1.compute.internal" is ready

2024-02-02 01:39:16 [ℹ] node "ip-192-168-58-214.ap-south-1.compute.internal" is ready

2024-02-02 01:39:16 [ℹ] node "ip-192-168-9-110.ap-south-1.compute.internal" is ready

2024-02-02 01:39:16 [ℹ] node "ip-192-168-94-118.ap-south-1.compute.internal" is ready

2024-02-02 01:39:16 [✖] could not find any of the authenticator commands: aws-iam-authenticator, aws

2024-02-02 01:39:16 [ℹ] cluster should be functional despite missing (or misconfigured) client binaries

2024-02-02 01:39:16 [✔] EKS cluster "my-eks-cluster" in "ap-south-1" region is ready

ECR(Elastic Container Repository):

dockerlinuxservice.yml:

apiVersion: v1

kind: Service

metadata:

name: dockerlinux2

spec:

selector:

app: dockerlinux2

ports:

- protocol: TCP

portqu: 80

targetPort: 80

type: LoadBalancer

dokcerdeployment.yml:

apiVersion: apps/v1

kind: Deployment

metadata:

name: dockerlinux2

labels:

app: dockerlinux2

spec:

replicas: 3

selector:

matchLabels:

app: dockerlinux2

template:

metadata:

labels:

app: dockerlinux2

spec:

containers:

- name: dockerlinux2

image: 242909465937.dkr.ecr.ap-south-1.amazonaws.com/dockerlinux2:dockerlinux2

ports:

- containerPort: 80

ubuntu:18

saikumar@DESKTOP-1JNARAI:~$ eksctl version

0.169.0

saikumar@DESKTOP-1JNARAI:~$ eksctl create cluster --name=my-eks-cluster4 --region=ap-south-1 --nodes=3 --node-type=t2.small

// aws iam authenticator:

Install Windows Subsystem for Linux (WSL):

Download link for AWSCLI TOOL

https://awscli.amazonaws.com/AWSCLIV2.msi

Open PowerShell as Administrator and run the following command:

powershell

Copy code

dism.exe /online /enable-feature /featurename:Microsoft-Windows-Subsystem-Linux /all /norestart

curl -o aws-iam-authenticator https://amazon-eks.s3.us-west-2.amazonaws.com/1.23.0/2021-07-05/bin/linux/amd64/aws-iam-authenticator

chmod +x ./aws-iam-authenticator

sudo mv aws-iam-authenticator /usr/local/bin/

//kubectl:

PS C:\Windows\system32> kubectl version

Client Version: v1.29.0-eks-5e0fdde

Kustomize Version: v5.0.4-0.20230601165947-6ce0bf390ce3

Server Version: v1.28.5-eks-5e0fdde

PS C:\Windows\system32> kubectl get nodes

NAME STATUS ROLES AGE VERSION

ip-192-168-20-13.ap-south-1.compute.internal Ready <none> 110m v1.27.9-eks-5e0fdde

ip-192-168-41-102.ap-south-1.compute.internal Ready <none> 110m v1.27.9-eks-5e0fdde

ip-192-168-57-250.ap-south-1.compute.internal Ready <none> 110m v1.27.9-eks-5e0fdde

PS C:\Windows\system32> kubectl get deployments

No resources found in default namespace.

PS C:\Windows\system32> kubectl get services

NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE

kubernetes ClusterIP 10.100.0.1 <none> 443/TCP 118m

PS C:\Windows\system32>

PS D:\csharpprojects\DockerLinux> kubectl get nodes

NAME STATUS ROLES AGE VERSION

ip-192-168-20-13.ap-south-1.compute.internal Ready <none> 115m v1.27.9-eks-5e0fdde

ip-192-168-41-102.ap-south-1.compute.internal Ready <none> 115m v1.27.9-eks-5e0fdde

ip-192-168-57-250.ap-south-1.compute.internal Ready <none> 115m v1.27.9-eks-5e0fdde

PS D:\csharpprojects\DockerLinux> kubectl apply -f DockerLinux/dockerlinuxservice.yml

service/dockerlinux2 created

PS D:\csharpprojects\DockerLinux> kubectl apply -f DockerLinux/dockerlineapp.deployment.yml

deployment.apps/dockerlinux2 created

PS D:\csharpprojects\DockerLinux> kubectl get all

NAME READY STATUS RESTARTS AGE

pod/dockerlinux2-847494f775-6bgzr 1/1 Running 0 16s

pod/dockerlinux2-847494f775-cg6tq 1/1 Running 0 16s

pod/dockerlinux2-847494f775-m5b97 1/1 Running 0 16s

NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE

service/dockerlinux2 LoadBalancer 10.100.37.101 ad897811161e64381a3edb02864beda8-1215962931.ap-south-1.elb.amazonaws.com 80:31872/TCP 92s

service/kubernetes ClusterIP 10.100.0.1 <none> 443/TCP 126m

NAME READY UP-TO-DATE AVAILABLE AGE

deployment.apps/dockerlinux2 3/3 3 3 18s

NAME DESIRED CURRENT READY AGE

replicaset.apps/dockerlinux2-847494f775 3 3 3 18s

PS D:\csharpprojects\DockerLinux> kubectl get services

NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE

dockerlinux2 LoadBalancer 10.100.37.101 ad897811161e64381a3edb02864beda8-1215962931.ap-south-1.elb.amazonaws.com 80:31872/TCP 2m11s

kubernetes ClusterIP 10.100.0.1 <none> 443/TCP 126m

PS D:\csharpprojects\DockerLinux> kubectl get deployments

NAME READY UP-TO-DATE AVAILABLE AGE

dockerlinux2 3/3 3 3 72s

Diployed web application link:

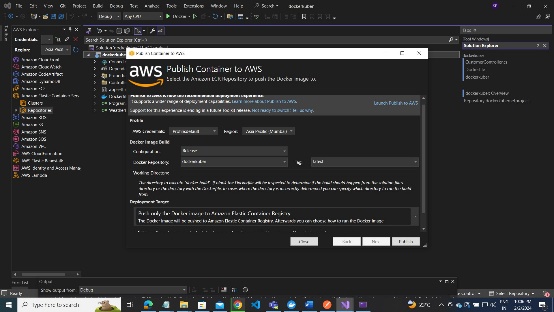
http://ad897811161e64381a3edb02864beda8-1215962931.ap-south-1.elb.amazonaws.com/weatherforecast

1.create asp.net core project:

2.create ECR Repository:

Project name = Repository name

3.publish ECR To cluster



Select only above picture:push only docker image into Elastic Container Repository.

4. create deployment.yml

5.create service.yml

6.create eksctl cluster.