End-to-End CI/CD Pipeline with Docker, ECR, Kubernetes (EKS), CodeBuild, and CodePipeline

# 1. Overview

This document explains how to set up an end-to-end CI/CD pipeline:  
• Dockerize the application  
• Push Docker images to AWS ECR  
• Deploy to Kubernetes (AWS EKS)  
• Automate builds & deployments using CodeBuild and CodePipeline  
• Monitor with CloudWatch

# 2. Prerequisites

• AWS account  
• AWS CLI v2 configured (`aws configure`)  
• kubectl & eksctl installed  
• Docker installed  
• GitHub repository containing application code

# 3. Dockerfile

Example Dockerfile for static app (dist/):  
  
FROM nginx:alpine  
COPY dist/ /usr/share/nginx/html  
EXPOSE 80  
CMD ["nginx", "-g", "daemon off;"]

# 4. Kubernetes YAML Files

deployment.yaml:  
  
apiVersion: apps/v1  
kind: Deployment  
metadata:  
 name: trend  
spec:  
 replicas: 2  
 selector:  
 matchLabels:  
 app: trend  
 template:  
 metadata:  
 labels:  
 app: trend  
 spec:  
 containers:  
 - name: trend  
 image: <aws\_account\_id>.dkr.ecr.ap-south-1.amazonaws.com/trend:latest  
 ports:  
 - containerPort: 80  
  
service.yaml:  
  
apiVersion: v1  
kind: Service  
metadata:  
 name: trend-service  
spec:  
 selector:  
 app: trend  
 ports:  
 - protocol: TCP  
 port: 80  
 targetPort: 80  
 type: LoadBalancer

# 5. AWS ECR Setup

1. Create repo:  
aws ecr create-repository --repository-name trend --region ap-south-1  
  
2. Authenticate Docker:  
aws ecr get-login-password --region ap-south-1 | docker login --username AWS --password-stdin <account\_id>.dkr.ecr.ap-south-1.amazonaws.com  
  
3. Build & Push:  
docker build -t trend:latest .  
docker tag trend:latest <account\_id>.dkr.ecr.ap-south-1.amazonaws.com/trend:latest  
docker push <account\_id>.dkr.ecr.ap-south-1.amazonaws.com/trend:latest

# 6. buildspec.yml

version: 0.2  
  
env:  
 variables:  
 AWS\_REGION: "ap-south-1"  
 ECR\_REPO: "<account\_id>.dkr.ecr.ap-south-1.amazonaws.com/trend"  
phases:  
 pre\_build:  
 commands:  
 - $(aws ecr get-login-password --region $AWS\_REGION | docker login --username AWS --password-stdin $ECR\_REPO)  
 build:  
 commands:  
 - docker build -t trend:latest .  
 - docker tag trend:latest $ECR\_REPO:latest  
 post\_build:  
 commands:  
 - docker push $ECR\_REPO:latest  
 - aws eks update-kubeconfig --region $AWS\_REGION --name <eks-cluster>  
 - kubectl apply -f k8s/deployment.yaml  
 - kubectl apply -f k8s/service.yaml

# 7. IAM Permissions

CodeBuild role must have:  
• AmazonEC2ContainerRegistryPowerUser  
• AmazonEKSClusterPolicy  
• CloudWatchLogsFullAccess  
  
Also map the role to aws-auth in EKS ConfigMap for kubectl access.

# 8. CodeBuild Setup

• Source: GitHub  
• Environment: Managed Ubuntu, privileged enabled  
• Role: Attach above IAM role  
• Buildspec: buildspec.yml

# 9. CodePipeline Setup

Stages:  
• Source: GitHub repo  
• Build: AWS CodeBuild project  
• Deploy: (Optional) CodeBuild runs kubectl

# 10. Monitoring

• CodeBuild & CodePipeline logs available in CloudWatch  
• Application logs: kubectl logs <pod>  
• Optional: Install CloudWatch agent DaemonSet for full cluster logs

# 11. Verification

kubectl get pods -l app=trend  
kubectl get svc trend-service  
kubectl logs -f <pod-name>