TrendStore End-to-End DevOps Project Documentation

# 1. Overview

This document describes the complete CI/CD pipeline and monitoring setup for the 'TrendStore' application.  
The project demonstrates how to automate build, push, and deployment processes using Jenkins, Docker, and AWS EKS,  
and monitor application health using Prometheus and Grafana.

Technologies Used: AWS, Jenkins, Docker, Kubernetes (EKS), Prometheus, Grafana, CloudWatch

# 2. Jenkins Setup

- Install Jenkins on an EC2 instance.  
- Install required plugins: Docker, Git, Kubernetes, Pipeline.  
- Configure credentials for DockerHub and GitHub.  
- Connect Jenkins to EKS using kubectl and kubeconfig.  
- Integrate Jenkins with GitHub webhook for automatic build triggers.

# 3. Jenkinsfile (CI/CD Pipeline Script)

pipeline {  
 agent any  
  
 environment {  
 DOCKERHUB\_CREDENTIALS = credentials('dockerhub-creds')  
 DOCKER\_IMAGE = "karthickm13799/trendstore:latest"  
 }  
  
 stages {  
 stage('Checkout Code') {  
 steps {  
 git branch: 'main', url: 'https://github.com/karthicmariappan/trendstore.git'  
 }  
 }  
  
 stage('Build Docker Image') {  
 steps {  
 sh 'docker build -t $DOCKER\_IMAGE .'  
 }  
 }  
  
 stage('Push to DockerHub') {  
 steps {  
 sh 'echo $DOCKERHUB\_CREDENTIALS\_PSW | docker login -u $DOCKERHUB\_CREDENTIALS\_USR --password-stdin'  
 sh 'docker push $DOCKER\_IMAGE'  
 }  
 }  
  
 stage('Deploy to Kubernetes (EKS)') {  
 steps {  
 sh 'kubectl apply -f k8s/deployment.yaml'  
 sh 'kubectl apply -f k8s/service.yaml'  
 }  
 }  
 }  
}

# 4. Kubernetes Deployment Files

deployment.yaml

apiVersion: apps/v1  
kind: Deployment  
metadata:  
 name: trendstore-deployment  
 labels:  
 app: trendstore  
spec:  
 replicas: 2  
 selector:  
 matchLabels:  
 app: trendstore  
 template:  
 metadata:  
 labels:  
 app: trendstore  
 spec:  
 containers:  
 - name: trendstore  
 image: karthickm13799/trendstore:latest  
 ports:  
 - containerPort: 3000

service.yaml

apiVersion: v1  
kind: Service  
metadata:  
 name: trendstore-service  
spec:  
 selector:  
 app: trendstore  
 ports:  
 - protocol: TCP  
 port: 80  
 targetPort: 3000  
 type: LoadBalancer

# 5. Deployment Validation

Use the following commands to validate your deployment:  
kubectl get pods  
kubectl get svc  
Then copy the EXTERNAL-IP from the LoadBalancer service and open it in your browser.

# 6. Monitoring Setup (Prometheus + Grafana)

Prometheus Setup:  
helm repo add prometheus-community https://prometheus-community.github.io/helm-charts  
helm repo update  
helm install prometheus prometheus-community/prometheus  
  
Grafana Setup:  
helm repo add grafana https://grafana.github.io/helm-charts  
helm install grafana grafana/grafana  
kubectl get svc  
Access Grafana using LoadBalancer IP and login with admin/prom-operator.  
Add Prometheus as a data source and create dashboards for CPU, Memory, and Pod metrics.

# 7. AWS CloudWatch Integration

Enable CloudWatch logging for EKS cluster using Fluent Bit:  
kubectl apply -f https://raw.githubusercontent.com/aws/amazon-cloudwatch-container-insights/main/k8s-deployment-manifest-templates/deployment-mode/daemonset/container-insights-monitoring/fluent-bit/fluent-bit-daemonset.yaml  
  
You can now view metrics in CloudWatch Logs group: /aws/containerinsights/trendstore/

# 8. CI/CD & Monitoring Flow

Flow: GitHub → Jenkins → DockerHub → EKS → Prometheus/Grafana → CloudWatch