YouTube Clone on Kubernetes: Deploying with Terraform, Jenkins, and Real-Time CI/CD Alerts via Slack

Step 1: Infrastructure Provisioning with Terraform

1. ***Set up Terraform project:***
    - ***Create a directory for your Terraform configuration files.***

Define a `main.tf` file with the necessary cloud provider configuration.
Example for AWS:

```
provider "aws" {
  region = "us-east-1"
}

resource "aws_eks_cluster" "youtube_cluster" {
  name     = "youtube-clone-cluster"
  role_arn = aws_iam_role.eks_cluster_role.arn
  vpc_config {
    subnet_ids = [aws_subnet.public_a.id, aws_subnet.public_b.id]
  }
```

    - ***}***

Include networking, IAM roles, and storage resources as needed.

2. ***Initialize Terraform:***
   terraform init

Plan and Apply:
terraform plan

3. ***terraform apply***
   This step provisions your Kubernetes cluster.

Step 2: Deploy Kubernetes Resources

1. ***Prepare Kubernetes manifests:***
   - ***Create YAML files for:***
     - ***Deployment: Backend, frontend, and database services.***
     - ***Service: Load Balancer for external access.***
     - ***ConfigMaps and Secrets: For environment variables and sensitive data.***

Example `deployment.yaml` for backend:

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: backend
spec:
  replicas: 3
  selector:
    matchLabels:
      app: backend
  template:
    metadata:
      labels:
        app: backend
    spec:
      containers:
      - name: backend
        image: your-backend-image:latest
        ports:
```

- ***- containerPort: 8080***

Apply Kubernetes manifests:
kubectl apply -f deployment.yaml

2. ***kubectl apply -f service.yaml***

Verify deployment:
kubectl get pods

3. ***kubectl get svc***

Step 3: Set Up Jenkins for CI/CD

1. ***Install Jenkins:***

Use a Kubernetes Helm chart to deploy Jenkins in your cluster:
helm repo add jenkins https://charts.jenkins.io

   - ***helm install jenkins jenkins/jenkins***
   - ***Retrieve the admin password:***
     kubectl exec --namespace default -it svc/jenkins -c jenkins -- /bin/cat /run/secrets/additional/chart-admin-password
2. ***Configure Jenkins:***
   - ***Install necessary plugins: Kubernetes, Slack Notification, Pipeline.***
   - ***Create credentials for Docker and Kubernetes.***
3. ***Create a Jenkins pipeline:***

Use a `Jenkinsfile` to define your pipeline. Example:

```
pipeline {
  agent any
  stages {
    stage('Build') {
      steps {
        sh 'docker build -t your-backend-image:latest .'
      }
```

```
    }
    stage('Push') {
      steps {
        withDockerRegistry([credentialsId: 'docker-cred', url:
'https://index.docker.io/v1/']) {
          sh 'docker push your-backend-image:latest'
        }
      }
    }
    stage('Deploy') {
      steps {
        sh 'kubectl apply -f deployment.yaml'
      }
    }
  }
  post {
    always {
      slackSend (channel: '#alerts', message: "Pipeline
${currentBuild.currentResult}: ${env.JOB_NAME}
#${env.BUILD_NUMBER}")
    }
  }

}
```

4. ***Trigger the pipeline:***
    - ***Commit and push code to trigger the build.***

Step 4: Set Up Slack Alerts

1. ***Create a Slack webhook:***
    - ***Go to your Slack workspace and create an Incoming Webhook.***
    - ***Copy the webhook URL.***

2. ***Configure Jenkins Slack plugin:***
    - ***Go to Jenkins > Manage Jenkins > Configure System.***
    - ***Add the Slack webhook URL and default channel.***
3. ***Test Slack notifications:***
    - ***Trigger a pipeline build and verify Slack notifications in your configured channel.***

Step 5: Monitoring and Maintenance

Monitor Kubernetes resources:
kubectl get pods -n your-namespace

1. ***kubectl logs -f pod-name***
2. ***Scale application:***
    - ***To scale pods:***
      kubectl scale deployment backend --replicas=5
3. ***Update application:***
    - ***Modify the `Jenkinsfile` or Kubernetes manifests for new features and redeploy.***