Demo Project: CD - Deploy to LKE cluster from Jenkins Pipeline

Project Description:

This project demonstrates how to deploy an application to a Linode Kubernetes Engine (LKE) cluster via a Jenkins Pipeline. Unlike AWS EKS, no additional platform-specific authentication is required. Instead, you'll create an LKE cluster through the Linode UI, configure access via a kubeconfig file, add LKE credentials in Jenkins, and update your Jenkinsfile to deploy to the LKE cluster.

Step 1: Create a Kubernetes Cluster on LKE

1. **Go to the Linode UI:**
   - Navigate to **Create → Kubernetes**.
2. **Configure the Cluster:**
   - **Cluster Label:** `test`
   - **Region:** Select the region closest to you.
   - **HA Control Plane:** No
   - **Node Pools:**

    - Select **Shared CPU → Linode 2GB**
    - Set the instance count to **1**
3. **Create the Cluster:**
    - Click **Create Cluster** and wait a few minutes for the node to reach the **Running** state.
4. **Download the kubeconfig File:**
    - Once the cluster is running, download the file (e.g., `test-kubeconfig.yaml`).

Step 2: Configure Local kubectl to Connect to the LKE Cluster

1. **Point kubectl to new cluster:** `export KUBECONFIG=test-kubeconfig.yaml`
2. **Verify Connection:** `kubectl get node`
    - You should see the LKE node(s) listed.

Step 3: Add LKE credentials on Jenkins

1. **In Jenkins:**
    - Navigate to your **Multibranch Pipeline** configuration.
    - Go to **Global Credentials** (or credentials scoped to your multibranch pipeline).
2. **Add a New Credential:**
    - **Kind:** Secret file
    - **Upload:** The `test-kubeconfig.yaml` file
    - **Name/ID:** `lke-credentials`
    - Click **Create**.

Step 4: Install Kubernetes CLI Plugin on Jenkins

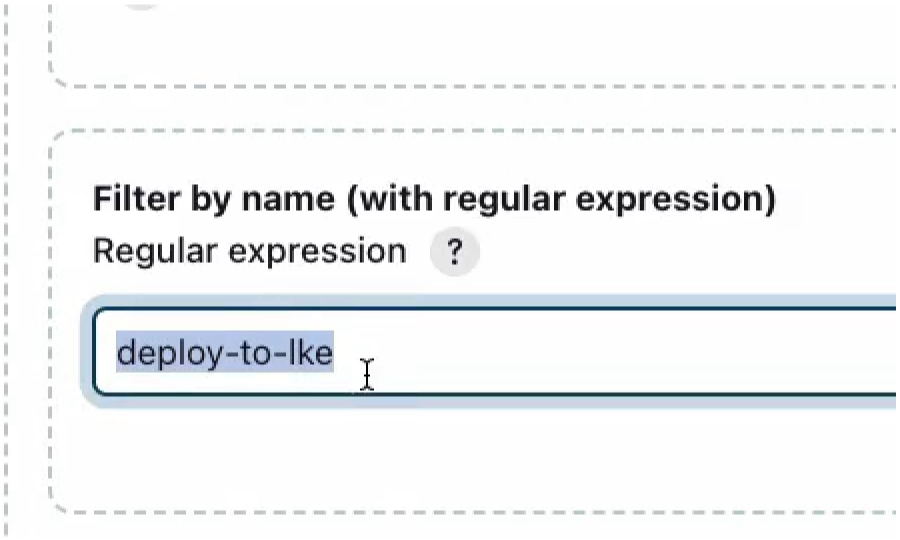
1. **Install the Plugin:**
    - In Jenkins, navigate to **Manage Jenkins → Plugins → Available Plugins**.
    - Search for and install the **Kubernetes CLI Plugin**.
2. **Restart Jenkins:**
    - Restart the Jenkins server to complete the installation.
    - *Note:* If Jenkins hangs on "Please wait while Jenkins is restarting...", SSH into the Jenkins server and check the container status:
        - `docker ps -a`
        - `docker start <container-id>`

Step 5: Configure the Jenkinsfile to deploy to LKE

1. **Create a New Branch for Deployment:**
    - From your existing deploy-on-k8s branch, create a new branch: `git checkout -b deploy-to-lke`
2. **Update the Jenkinsfile:**
    - Add the necessary configuration to connect to your EKS cluster by referencing your kubeconfig.
3. **Commit and Push Changes:**
    - Commit your updated Jenkinsfile to the `deploy-to-lke` branch and push it to your repository.

Step 6: Update Jenkins Multi-Branch Pipeline Configuration

- **In Jenkins:**
  - Update the multibranch pipeline configuration to include the new branch (`deploy-to-lke`).



Filter by name (with regular expression)

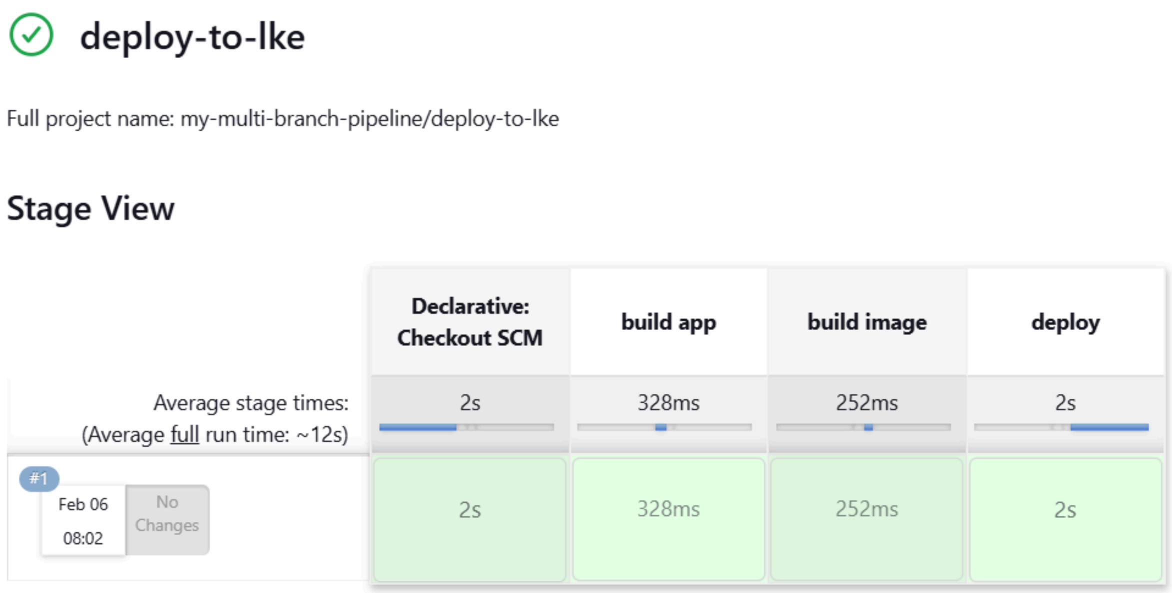
Regular expression ?

deploy-to-lke

  - Save the changes.

Step 7: Execute the Jenkins Pipeline

1. **Trigger the Pipeline:**
   - From the Jenkins UI, trigger the pipeline on the `deploy-to-lke` branch.
2. **Monitor Pipeline Output:**
   - Verify that the build, image build, and deployment stages complete successfully.


deploy-to-lke

Full project name: my-multi-branch-pipeline/deploy-to-lke

Stage View

	Declarative: Checkout SCM	build app	build image	deploy
Average stage times: (Average full run time: ~12s)	2s	328ms	252ms	2s
#1 Feb 06 08:02 No Changes	2s	328ms	252ms	2s

3. **Verify Deployment:** `kubectl get pod`
    - Confirm that the deployment (e.g., `nginx-deployment`) is running in the LKE cluster.
    - Example:

```
kubectl get pod
NAME                              READY   STATUS    RESTARTS   AGE
nginx-deployment-5959b5b5c9-q9lcb   1/1     Running   0          62s
```

Step 8: Clean Up

1. **Delete the Deployment (if needed):** `kubectl delete deployment nginx-deployment`
2. **Optionally, Delete the LKE Cluster:**
    - Use the Linode UI or the appropriate CLI command if desired.
