Setup Guide

Overview

This guide walks through deploying a scalable and cost-efficient Flask-based web application on DigitalOcean Kubernetes (DOKS). This app is containerized in Docker, deployed via Kubernetes manifests, exposed with a load balancer, and horizontally scaled using CPU-based autoscaling.

Prerequisites

The following tools must be installed and configured:

- Docker
- kubectl
- doctl
- git

Instructions

Step 1: Clone the repository

```
git clone https://github.com/YOUR-USERNAME/birthday-reminder-doks.git
```

```
cd birthday-reminder-doks
```

Step 2: Build and Push the Docker Image

1. Log into your DOCR registry:

   ```
   doctl registry login
   ```

2. Tag and push the image:

   ```
   docker build -t birthday-app .
   docker tag birthday-app registry.digitalocean.com/<your-registry-name>/birthday-app:latest
   docker push registry.digitalocean.com/<your-registry-name>/birthday-app:latest
   ```

Step 3: Provision a Kubernetes Cluster on DOKS

1. Create .env as a Kubernetes Secret:
   `kubectl create secret generic birthday-app-secrets --from-env-file=.env`
2. Create a new cluster:

   ```
   doctl kubernetes cluster create birthday-cluster \
     --region nyc3 \
     --version 1.33.1-do.0 \
     --count 3 \
     --size s-2vcpu-4gb
   ```

   NOTE: This action can also be performed on the DigitalOcean Dashboard

3. Save cluster credentials:

   ```
   doctl kubernetes cluster kubeconfig save birthday-cluster
   kubectl get nodes
   ```

Step 4: Allow Private Image Pulls from DOCR

```
doctl registry kubernetes-manifest --cluster birthday-cluster | kubectl apply -f -
```

Step 5: Deploy the Application

Apply the following Kubernetes manifests:

```
kubectl apply -f k8s/deployment.yaml
kubectl apply -f k8s/service.yaml
kubectl apply -f k8s/hpa.yaml
```

Step 6: Access the Application

1. Get the external IP for the load balancer:
   `kubectl get svc`
2. Visit this URL in a web browser:
   `https://<external-ip>`
3. At this point, a basic web UI containing mock birthday reminders should appear.

Step 7: Verify Horizontal Pod Autoscaling (optional)

1. Watch the current state of HPA autoscaling

```
kubectl get hpa -w
```

2. In a separate console, simulate CPU load:
   `ab -n 10000 -c 100 http://<external-ip>/`
3. Return to the original terminal, then wait and confirm that pods are scaling up properly