Lab 14: Kubernetes Monitoring and Init Containers

Overview

In this lab, you will explore Kubernetes cluster monitoring using Prometheus with the Kube Prometheus Stack. Additionally, you'll delve into the concept of Init Containers in Kubernetes.

Task 1: Kubernetes Cluster Monitoring with Prometheus

6 Points:

1. This lab was tested on a specific version of components:
   - Minikube v1.33.0
   - Minikube kubectl v1.28.3
   - kube-prometheus-stack-57.2.0 v0.72.0
   - the minikube start command - `minikube start --driver=docker --container-runtime=containerd`
2. Read about `Kube Prometheus Stack`:
   - Helm chart with installation guide
   - Explanation of components
3. Describe Components:
   - Create `14.md` and detail the components of the Kube Prometheus Stack, explaining their roles and functions. Avoid direct copy-pasting; provide a personal understanding.
4. Install Helm Charts:
   - Install the Kube Prometheus Stack to your Kubernetes cluster.
   - Install your app's Helm chart.
   - Provide the output of the `kubectl get po,sts,svc,pvc,cm` command in the report and explain each part.
5. Utilize Grafana Dashboards:
   - Access Grafana using `minikube service monitoring-grafana`.
   - Explore existing dashboards to find information about your cluster:
     1. Check CPU and Memory consumption of your StatefulSet.
     2. Identify Pods with higher and lower CPU usage in the default namespace.
     3. Monitor node memory usage in percentage and megabytes.
     4. Count the number of pods and containers managed by the Kubelet service.
     5. Evaluate network usage of Pods in the default namespace.
     6. Determine the number of active alerts; also check the Web UI with `minikube service monitoring-kube-prometheus-alertmanager`.
   - Provide answers to all these points in the report.

Task 2: Init Containers

4 Points:

1. Read about `Init Containers`:
   - Concept
   - Tutorial
2. Implement Init Container:
   - Create a new Volume.
   - Implement an Init container to download any file using `wget` (you can use a site from the example).
   - Provide proof of success, e.g., `kubectl exec pod/demo-0 -- cat /test.html`.

List of Requirements:

- Detailed explanation of monitoring stack components in `14.md`.
- Output and explanation of `kubectl get po,sts,svc,pvc,cm`.
- Answers to all 6 questions from point 4 in `14.md`.
- Implementation of Init Container.
- Proof of Init Container downloading a file.

Bonus Task: App Metrics & Multiple Init Containers

2.5 Points:

1. App Metrics:
   - Fetch metrics from your app and provide proof.
2. Init Container Queue:
   - Create a queue of three Init containers, with any logic like adding new lines to the same file.
   - Provide proof using the `cat` tool.

Guidelines

- Ensure clear and organized documentation.
- Use appropriate naming conventions for files and folders.
- For your repository PR, ensure it's from the `lab14` branch to the main branch.

> Note: Demonstrate successful implementation and understanding of Kubernetes monitoring and Init Containers. Take your time to explore the bonus tasks for additional learning opportunities.