Log Monitoring System

Project Overview

- **Log Generator:** Simulates app logs
- **Log Collector:** Collects logs from mounted volumes (basic Python script or Filebeat)
- **Visualization:** Grafana or Kibana
- **Deployment:** Automated via shell scripts
- **Kubernetes:** Manages Pods, Services, Volumes

Architecture Overview:

```
 [ log-generator Pods ] ---> write logs to Persistent Volume (PV)
                                  |
[ log-collector Pod ] ---> reads logs from PV
                                  |
[ Loki ] <-- stores logs from collector
                                  |
[ Grafana ] <--- queries Loki to visualize logs
```

Step 1: Create Directory Structure

```
mkdir -p log-monitoring-devops/{docker/log-generator,docker/log-collector,k8s,scripts}
cd log-monitoring-devops
```

Step 2: Create Log Collector (Python-based)

docker/log-collector/collector.py

```
import time
import os

log_path = "/logs/access.log"

while True:
```

```
    if os.path.exists(log_path):
        with open(log_path, "r") as f:
            lines = f.readlines()
            print("".join(lines[-10:]))
    else:
        print("Waiting for logs...")
    time.sleep(5)
```

docker/log-collector/Dockerfile

```
FROM python:3.9-slim
WORKDIR /app
COPY collector.py .
VOLUME ["/logs"]
CMD ["python", "collector.py"]
```

Step 3: Build Docker Image

scripts/build.sh

```
#!/bin/bash
eval $(minikube docker-env)

docker build -t log-collector:latest ./docker/log-collector

echo "Built log-collector image"
```

Run it:

```
chmod +x scripts/build.sh
./scripts/build.sh
```

Step 4: Create Persistent Volume & PVC

k8s/pv.yaml

```
apiVersion: v1
```

```
kind: PersistentVolume
metadata:
  name: log-pv
spec:
  capacity:
    storage: 1Gi
  accessModes:
    - ReadWriteMany
  hostPath:
    path: /tmp/logdata

---
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: log-pvc
spec:
  accessModes:
    - ReadWriteMany
  resources:
    requests:
      storage: 1Gi
```

Step 5: NGINX Log Generator Deployment

k8s/log-generator.yaml

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: log-generator
spec:
  replicas: 2
  selector:
    matchLabels:
      app: log-generator
  template:
    metadata:
      labels:
        app: log-generator
    spec:
```

```
    containers:
      - name: nginx
        image: nginx:alpine
        volumeMounts:
          - name: log-volume
            mountPath: /var/log/nginx
    volumes:
      - name: log-volume
        persistentVolumeClaim:
          claimName: log-pvc
```

Step 6: Log Collector Pod

k8s/log-collector.yaml

```
apiVersion: v1
kind: Pod
metadata:
  name: log-collector
spec:
  containers:
    - name: collector
      image: log-collector:latest
      volumeMounts:
        - name: log-volume
          mountPath: /logs
  volumes:
    - name: log-volume
      persistentVolumeClaim:
        claimName: log-pvc
```

Step 7: Apply All YAMLs

scripts/deploy.sh

```
#!/bin/bash

kubectl apply -f k8s/pv.yaml
kubectl apply -f k8s/log-generator.yaml
```

```
kubectl apply -f k8s/log-collector.yaml

echo "Deployed PV, NGINX, and Collector"
kubectl get pods
```

Run it:

```
chmod +x scripts/deploy.sh
./scripts/deploy.sh
```

Check logs:

```
kubectl logs log-collector
```

Step 8: Deploy Grafana + Loki (via Helm)

Install Helm :

Install Helm for Windows

- Go to: https://github.com/helm/helm/releases/latest
- Download helm-v3.x.x-windows-amd64.zip
- Extract helm.exe and put it in a folder like C:\bin
- Add C:\bin to your system PATH

Then verify:

```
helm version
```

Next Steps: Install Loki + Grafana in Minikube

1. Add the Grafana Helm repo:

```
helm repo add grafana https://grafana.github.io/helm-charts
helm repo update
```

2. Install Loki Stack with Grafana and Promtail enabled:

```
helm install loki grafana/loki-stack \
  --set grafana.enabled=true \
  --set promtail.enabled=true
```

This will install:

- Grafana (visualization)
- Loki (log aggregation)
- Promtail (log forwarder from your pods)

3. Check the services:

```
kubectl get svc
```

Look for a service like loki-grafana or similar.

4. Expose Grafana to view in browser:

```
minikube service loki-grafana --url
```

Open the URL in your browser — default credentials:

- Username: admin
- Password: prom-operator (or check using: kubectl get secret --namespace default loki-grafana -o jsonpath="{.data.admin-password}" | base64 --decode)

Confirm log generation

If you already deployed your Flask app or log generator pods, Promtail should be picking up logs and sending them to Loki.

In Grafana:

1. Go to “Explore”
2. Select the “Loki” data source
3. Try this query:

```
{app="log-generator"}
```

You should start seeing logs appear!

Important Notes

- Promtail must be deployed as DaemonSet and must access container logs.
- Loki and Promtail pods must be healthy (kubectl get pods).
- If you stop/start Minikube, Helm installations persist, but you must re-run minikube service to re-expose services.

Validation

- Logs appear in Grafana
- Pods are running and generating logs
- Grafana dashboard is accessible via browser

Optional Enhancements

- Add log-collector pod
- Set up Persistent Volumes
- Use Ingress for stable access to Grafana
- Add alerting in Grafana based on log patterns