README

Workloads Project (StatefulSet + DaemonSet + Job + CronJob) Goal Create one practical Kubernetes scenario where these workloads are **interlinked** by writing into a **common shared location**: - `StatefulSet` - `DaemonSet` - `Job` - `CronJob` This helps you verify behavior from one place. Project structure `text workloads/ container/ Dockerfile writer.sh manifests/ 00-namespace.yaml 01-pv-pvc.yaml 02-statefulset-headless-svc.yaml 03-statefulset.yaml 04-daemonset.yaml 05-job.yaml 06-cronjob.yaml 07-reader-pod.yaml` Scenario design All workloads run the same small custom image (`vinodhconnects/shared-writer:latest`). Each pod writes log lines into: - `/data/shared/activity.log` (combined log) - `/data/shared/<prefix>.log` (per-workload log) Prefixes used: - `statefulset` - `daemonset` - `job` - `cronjob` Prerequisites A running Kubernetes cluster (`kubectl` works) Docker or compatible image build tool For local learning, this setup is best on **single-node Minikube/KIND** Note: Shared volume here uses a `hostPath`-based `PersistentVolume` for demo simplicity. Step 1: Build the container image From repo root: `bash cd workloads/container docker build -t vinodhconnects/shared-writer:latest .` If using Minikube Option A (build directly inside Minikube Docker): `bash eval $(minikube docker-env) docker build -t vinodhconnects/shared-writer:latest .` Option B (build locally, then load): `bash docker build -t vinodhconnects/shared-writer:latest . minikube image load vinodhconnects/shared-writer:latest` If using KIND `bash docker build -t vinodhconnects/shared-writer:latest . kind load docker-image vinodhconnects/shared-writer:latest` Step 2: Apply manifests From repo root: `bash kubectl apply -f workloads/manifests/00-namespace.yaml kubectl apply -f workloads/manifests/01-pv-pvc.yaml kubectl apply -f workloads/manifests/02-statefulset-headless-svc.yaml kubectl apply -f workloads/manifests/03-statefulset.yaml kubectl apply -f workloads/manifests/04-daemonset.yaml kubectl apply -f workloads/manifests/05-job.yaml kubectl apply -f workloads/manifests/06-cronjob.yaml kubectl apply -f workloads/manifests/07-reader-pod.yaml` Quick status check: `bash kubectl get all -n workloads kubectl get pv,pvc -n workloads` Step 3: Verify interlinked output from one common place Use the reader pod to inspect shared files: `bash kubectl exec -n workloads shared-reader -- ls -l /data/shared kubectl exec -n workloads shared-reader -- tail -n 40 /data/shared/activity.log kubectl exec -n workloads shared-reader -- tail -n 20 /data/shared/statefulset.log kubectl exec -n workloads shared-reader -- tail -n 20 /data/shared/daemonset.log kubectl exec -n workloads shared-reader -- tail -n 20 /data/shared/job.log kubectl exec -n workloads shared-reader -- tail -n 20 /data/shared/cronjob.log` You should see lines like: `text 2026-02-24T10:30:00Z mode=loop prefix=statefulset ns=workloads pod=stateful-writer-0 node=minikube 2026-02-24T10:30:10Z mode=loop prefix=daemonset ns=workloads pod=daemon-writer-abcde node=minikube 2026-02-24T10:30:15Z mode=once prefix=job ns=workloads pod=one-time-writer-xxxxx node=minikube 2026-02-24T10:32:00Z mode=once prefix=cronjob ns=workloads pod=scheduled-writer-xxxxx node=minikube` How each workload contributes `StatefulSet` (`replicas: 2`): continuously writes with stable pod identity (`stateful-writer-0`, `stateful-writer-1`) `DaemonSet`: one pod per node writes node-level activity `Job`: writes one-time completion record `CronJob`: writes scheduled records every 2 minutes All records land in the same shared path, making cross-workload verification easy. Useful commands `bash kubectl get statefulset,daemonset,job,cronjob -n workloads kubectl logs -n workloads statefulset/stateful-writer --all-pods=true --tail=20 kubectl logs -n workloads daemonset/daemon-writer --tail=20 kubectl describe cronjob -n workloads scheduled-writer` Cleanup `bash kubectl delete -f workloads/manifests/07-reader-pod.yaml kubectl delete -f`

`workloads/manifests/06-cronjob.yaml kubectl delete -f workloads/manifests/05-job.yaml kubectl delete -f workloads/manifests/04-daemonset.yaml kubectl delete -f workloads/manifests/03-statefulset.yaml kubectl delete -f workloads/manifests/02-statefulset-headless-svc.yaml kubectl delete -f workloads/manifests/01-pv-pvc.yaml kubectl delete -f workloads/manifests/00-namespace.yaml` If you want a clean host path too (node-level path), remove `/tmp/workloads-shared` from the cluster node. Notes for real environments This project uses `hostPath` PV for easy local learning. In production, prefer proper network storage (RWX capable CSI/NFS/etc.) for truly shared multi-node writes. `hostPath` behavior differs across multi-node clusters.