# LAB | Multi-Tier Guestbook App on Kubernetes

Author: Ferdinando (MTFerdi)  
Date: August 2025

## Overview

In this lab, we deployed the classic multi-tier PHP Guestbook app on Kubernetes. The stack consists of a Redis Leader (write operations), Redis Followers (read operations), and a PHP Frontend. We also set up CI with GitHub Actions to validate manifests, and added a .gitignore for repo hygiene.

## Repo Structure

LAB-Multi-Tier-Guestbook-App-on-K8s/  
├─ k8s/  
│ ├─ redis-leader-deployment.yaml  
│ ├─ redis-leader-service.yaml  
│ ├─ redis-follower-deployment.yaml  
│ ├─ redis-follower-service.yaml  
│ ├─ frontend-deployment.yaml  
│ └─ frontend-service.yaml  
├─ .github/workflows/validate-k8s.yml  
├─ .gitignore  
└─ README.md

## Steps Completed

### 1. Prerequisites

- Running Kubernetes cluster (Minikube or cloud).  
- kubectl installed and configured.  
- Git + GitHub account.  
- DockerHub (optional, if re-tagging images).

### 2. Deploy Redis Leader

Created Deployment & Service for Redis leader:

kubectl apply -f k8s/redis-leader-deployment.yaml

kubectl apply -f k8s/redis-leader-service.yaml

kubectl get pods && kubectl get svc

### 3. Deploy Redis Followers

kubectl apply -f k8s/redis-follower-deployment.yaml

kubectl apply -f k8s/redis-follower-service.yaml

### 4. Deploy Frontend

kubectl apply -f k8s/frontend-deployment.yaml

kubectl apply -f k8s/frontend-service.yaml

### 5. Access the Guestbook

kubectl port-forward svc/frontend 8080:80

Open http://localhost:8080 in browser.

### 6. Scaling Frontend

kubectl scale deployment frontend --replicas=5

kubectl scale deployment frontend --replicas=2

### 7. Cleanup

kubectl delete deployment -l app=redis  
kubectl delete service -l app=redis  
kubectl delete deployment frontend  
kubectl delete service frontend  
kubectl get pods

### 8. GitHub Repo Setup

- Initialized repo in VS Code.  
- Connected to GitHub as `MTFerdi/LAB-Multi-Tier-Guestbook-App-on-K8s`.  
- Pushed all manifests and README.

### 9. CI with GitHub Actions

Workflow file: `.github/workflows/validate-k8s.yml`  
  
Runs kubeconform on all YAML manifests under `k8s/`:  
  
name: Validate K8s Manifests  
on:  
 workflow\_dispatch: {}  
 push:  
 branches: [ "main" ]  
 pull\_request:  
 branches: [ "main" ]  
  
jobs:  
 validate:  
 runs-on: ubuntu-latest  
 steps:  
 - uses: actions/checkout@v4  
 - name: Install kubeconform  
 run: |  
 curl -L https://github.com/yannh/kubeconform/releases/download/v0.6.7/kubeconform-linux-amd64.tar.gz | tar xz  
 sudo mv kubeconform /usr/local/bin/kubeconform  
 - name: Validate manifests  
 run: |  
 kubeconform -summary -strict -schema-location default -schema-location "https://raw.githubusercontent.com/yannh/kubernetes-json-schema/master/{{.NormalizedKubernetesVersion}}-standalone/{{.ResourceKind}}.json" $(find k8s -type f -name '\*.yaml' -o -name '\*.yml')

### 10. .gitignore

# OS / Editor  
.DS\_Store  
Thumbs.db  
.vscode/  
.idea/  
  
# Logs / temp  
\*.log  
\*.tmp  
\*.swp  
  
# Python cache  
\_\_pycache\_\_/  
\*.pyc  
  
# Node  
node\_modules/  
npm-debug.log  
yarn-error.log  
  
# Build / dist  
dist/  
build/  
coverage/  
  
# Env / secrets  
.env  
\*.key  
\*.pem  
secrets/

### 11. Clean Old Workflow Runs

We learned to purge runs older than 3 days using GitHub CLI:  
  
# Dry-run list  
gh run list -R MTFerdi/LAB-Multi-Tier-Guestbook-App-on-K8s --limit 200 --json databaseId,createdAt,displayTitle  
  
# Delete runs older than 3 days  
gh run list -R MTFerdi/LAB-Multi-Tier-Guestbook-App-on-K8s --limit 200 --json databaseId,createdAt | jq -r --arg cutoff "$(date -u -d '3 days ago' +%Y-%m-%dT%H:%M:%SZ)" '.[] | select(.createdAt < $cutoff) | .databaseId' | xargs -r -n1 gh run delete -R MTFerdi/LAB-Multi-Tier-Guestbook-App-on-K8s --yes

## Conclusion

We successfully deployed and managed a multi-tier Guestbook app on Kubernetes, set up GitHub CI validation, and learned good repo hygiene practices. This lab builds strong foundations for production-ready workflows.