

Project 1: Containerized Web App with HPA on GKE

Objective:

Deploy a simple web application using GKE with autoscaling based on CPU utilization, all orchestrated via Terraform.

Requirements:

- Use a simple containerized web application (e.g., a Python Flask app serving “Hello, World!”).
- Provision:
 - GKE cluster (standard, not Autopilot).
 - Kubernetes Deployment for the app [preferably use Helm for deployment]
 - Kubernetes Service with a LoadBalancer type or use Ingress.
- Configure Horizontal Pod Autoscaler (HPA):
 - Target CPU utilization threshold for autoscaling.
 - Scale out when CPU > 50%, scale in when < 20%.
 - Min: 1 pod, Max: 3 pods.
- Set up Terraform configurations (can use community GKE modules or write your own).
- Simulate CPU load (e.g., using stress or hey) and demonstrate autoscaling behavior.
- Optional: Add logging/monitoring integration and a CI/CD pipeline.
- Provide an architecture diagram that illustrates the components and their interactions within the solution.

Project 2: Design a Kubernetes Operator for Zero-Downtime Node Cycling

Objective:

Design a simple Python-based Kubernetes Operator that safely moves pods to new nodes every 3 days with zero downtime. This operator will help teams refresh their infrastructure while maintaining application availability.

Requirements:

- Create a basic NodeRefresh CRD with fields for:
 - Target nodes (using labels)
 - Maximum pods to move at once
 - Minimum health threshold before proceeding
 - Refresh schedule (optional)
- Implement a Python-based controller using the Kubernetes Python client, Create a reconciliation loop that:
 - Identifies target nodes matching specified labels
 - Provisions a new node before draining an old one
 - Ensures pods are running successfully on new nodes before proceeding
 - Respects pod disruption budgets during migrations
- Core Features:
 - Implement safe pod eviction logic that checks application health
 - Add a basic retry mechanism for failed migrations
 - Include proper logging of all operations
 - Provide status updates through the CRD status field