Cost Optimization Overview for Flask SaaS App Deployment

This document describes how the Kubernetes deployment for the Flask SaaS app is cost optimized on DigitalOcean.

Cost Optimization Strategies

1. Basic Droplets

- The cluster uses basic (smallest recommended) Droplets for worker nodes, minimizing compute costs.
- No additional storage volumes or block storage are attached, reducing storage expenses.

2. Node Count and Autoscaling

- The default node count is set to three, providing high availability while keeping costs low.
- Cluster autoscaling is enabled:
  - Nodes automatically scale up to a maximum of five only when average CPU usage exceeds 70%.
  - This ensures you only pay for extra resources when truly needed, avoiding over-provisioning.

3. Resource Requests and Limits

- Pod resource requests and limits are set appropriately to prevent unnecessary resource allocation.
- This helps the scheduler pack pods efficiently, reducing wasted capacity.

4. Load Balancer Usage

- Only one LoadBalancer service is provisioned, minimizing cloud networking costs.

5. Stateless Application

- The Flask app is stateless, so no persistent storage is required.
- This allows for easy scaling and avoids costs associated with stateful sets or database hosting.

6. Monitoring and Alerts

- Resource usage is monitored, and alerts are set for abnormal spikes.
- This enables proactive scaling and cost management.

7. Image Optimization

- The Docker image uses a slim Python base, reducing image size and speeding up deployments, which can lower bandwidth and storage costs.

8. Idle Resource Management

• Unused resources (nodes, services) are regularly reviewed and decommissioned to avoid unnecessary charges.

Summary

This deployment is designed to balance reliability and cost:

• Three nodes by default for redundancy.

• Autoscaling up to five nodes only when needed.

• No extra storage or unnecessary services.

• Efficient resource allocation and monitoring.

These practices help ensure the Flask SaaS app runs reliably while keeping cloud costs predictable