Imago DevOps & Infrastructure Engineer – Challenge C3

Prepared by: Joyce Mwangi
Date: 17th August 2025
Role: DevOps & Infrastructure Engineer

Objective

This document presents my solution for the Imago DevOps & Infrastructure Engineer Challenge C3. It outlines my design, implementation steps, and reasoning for creating a scalable, secure, and observable infrastructure for the client-facing media search and download API.

Solution Links

GitHub Repository: https://github.com/joycemwangi/automation-pipeline-challenge-c3
CodeSandbox Project: https://codesandbox.io/p/github/joycemwangi/automation-pipeline-challenge-c3/main
Live Demo (/healthz): https://kncqz4-8080.csb.app/healthz
Pipeline Diagram: https://github.com/joycemwangi/automation-pipeline-challenge-c3/blob/main/docs/automation-pipeline.png

Key Implementations

• I implemented infrastructure as code using Ansible with modular roles (nginx, zabbix_agent, maintenance).
• Configured Nginx with HTTPS and automated SSL certificate handling.
• Integrated Zabbix Agent for observability across Server A and Server B, fronted by a load balancer.
• Automated OS patching and reboots for secure maintenance.
• Designed a GitLab CI/CD pipeline for testing, build, deploy, and rollback.
• Addressed secrets management using GitLab Vault/Kubernetes Secrets.
• Used Node.js demo API with a load balancer setup to simulate traffic routing.

Assumptions & Limitations

• The demo API is mock-only and does not connect to real Elasticsearch/storage.
• Certificates are self-signed for demo purposes, but production would use Let's Encrypt/ACME.
• CI/CD focuses on workflow structure rather than production-grade Helm charts.
• Observability is shown with Zabbix Agent, but in production, I would integrate Prometheus/Grafana.

Detailed Implementation – Ansible Playbooks

I created modular Ansible roles to ensure reusability and maintainability across infrastructure components.

• nginx role: Installs NGINX, deploys HTTPS configuration with SSL certificates, and enables the service.
• zabbix_agent role: Installs and configures Zabbix Agent with templates for active checks.

• maintenance role: Applies OS patches, triggers reboot if required, and ensures post-patch service health.

CI/CD Pipeline Design

The GitLab CI/CD pipeline automates testing, build, deployment, smoke testing, promotion, and rollback.

• validate – Run yamllint, ansible-lint, and syntax checks.
• build – Build and push Docker image of the API.
• deploy – Helm upgrade/install to Kubernetes cluster.
• smoke – Test /healthz endpoint after deployment.
• promote – Manual approval for staging → production.
• rollback – Helm rollback to previous release on failure.

Secrets Management & Environment Configuration

Secrets are stored securely in GitLab CI/CD variables or Kubernetes Secrets. Environment-specific configuration is handled with Helm values files (values-dev.yaml, values-staging.yaml, values-prod.yaml). This ensures separation of concerns and secure handling of sensitive values.

Submission Checklist

• A README outlining thoughts and solution design.
• GitHub repository with Ansible, CI/CD, and API demo code.
• CodeSandbox for executable demo.
• Clickable pipeline diagram.
• Instructions for local run provided in README.

Imago Challenge Brief

Hello! ■■ Thank you for taking on this task. Our goal is to better understand your hands-on development skills and workflow. The following task is designed to evaluate key skills we consider essential for this position. There is no single correct solution, so feel free to approach it in the way you believe is best. Use your judgment and what you know about IMAGO so far. We're particularly interested in your thought process, so please support your ideas with clear and well-reasoned arguments. We kindly ask you to submit your solution as a PDF. You're welcome to include links to tools like CodeSandbox, GitHub Pages, or any other relevant platform. The deadline for submission is 7 working days from the date you receive this challenge. If you need more time, don't hesitate to contact us to request an extension. We estimate this task should take approximately 4-6 hours to complete. We hope you enjoy the challenge. Best of luck! ■ --- IMAGO’s clients rely on a seamless experience when searching for and downloading media files from our platform. As our media library and customer base grow, ensuring the infrastructure that powers these client-facing services is robust, scalable, and observable is critical. Design and partially implement an infrastructure solution that supports a scalable, secure, and observable client-facing media search and download service. Focus on automation, reliability, and operational excellence. Scenario: - Our media search and download API is used by thousands of clients daily. It queries Elasticsearch for metadata, retrieves media files from storage, and serves them over HTTPS. - The core services run on Linux, with a mix of dedicated servers and cloud instances. - We are transitioning legacy infrastructure to a modern, Linux-centric, containerized environment. - Reliability, performance, and visibility into

system health are top priorities. Requirements: We are planning to deploy a new infrastructure platform to host an API service. The platform will consist of a load balancer and two Linux servers (Linux distribution can be chosen freely). - Load balancer: 192.168.10.2 - Server A: 192.168.10.3 - Server B: 192.168.10.4 - Zabbix Agent Server: 192.168.10.5 Tasks: Write an Ansible playbook to perform the following actions on Server A and Server B: 1. Install Nginx and Zabbix monitoring agent. 2. Configure Nginx with HTTPS and SSL certificates. 3. Automate OS patching and reboots. 4. Outline a CI/CD GitLab pipeline (testing, build, deployment, rollback, secrets, environment config). Deliverables: - README outlining thoughts, considerations, assumptions, and limitations. - GitHub repo or zip file with executable code. - Deployed solution link, CodeSandbox, or run instructions. Thank you for your time! ■