QBurst | A High AI-Q™ Company

Modernizing Infrastructure with EKS and GitOps

Transforming container orchestration with Kubernetes, GitOps, and self-service deployments to enhance scalability, reliability, and developer autonomy.

Overview

- Our engagement with a leading global fashion retailer involved migrating workloads from Amazon ECS to Amazon EKS, implementing Helm for application packaging, and adopting ArgoCD for GitOps-driven continuous delivery.
- Developers gained direct ownership of deployments through a self-service model, reducing ticket dependency.
- Lead time for container changes dropped by **90%,** major production incidents decreased by nearly **90%**, and regional inconsistencies were eliminated through a standardized, Kubernetes-first architecture.

A Leading Fashion Retailer Driven by Innovation

The client is a leading fashion retailer managing multiple brands across diverse markets. Their business requires fast, scalable, and reliable technology platforms to support peak demand during campaigns and sales events. To achieve this, they needed to modernize their infrastructure and reduce reliance on a vendor-specific orchestration platform.

The Challenge: ECS Limitations Slowing Agility

- ECS introduced vendor lock-in and limited flexibility
- Developers depended on the DevOps team for container changes, causing 2–4 week delays
- Manual processes and poor visibility often caused production incidents

- Regional inconsistencies (dedicated vs. shared architectures) increased operational complexity
- Scaling during peak events was difficult and unpredictable

QBurst Solution: Migrating to EKS with GitOps

We led a full migration of workloads from ECS to Amazon EKS, moving to an open Kubernetes-based model. Key solution components included:

- Helm for standardized application packaging and versioned deployments
- ArgoCD to enable GitOps-driven automation and continuous delivery
- Self-service deployment model giving developers direct control over scaling and configuration
- Separation of concerns between app teams (service deployment) and infra teams (cluster stability/security)
- Infrastructure as Code for consistent, auditable deployments

This modernized platform aligned operations with Kubernetes standards, reduced dependencies, and improved agility across environments.

Solution-Focused Approach: Scale and Transparency

- Self-service deployments with Kubernetes manifests and Helm
- Standardized architecture across multiple regions
- Kubernetes-native observability with Datadog and Fluent Bit
- Event-driven autoscaling with KEDA and node autoscaling with Karpenter
- Policy enforcement via Gatekeeper
- Cost monitoring with OpenCost

Outcome

- **90% faster container changes** (98.7 hours → 10 hours)
- **90% improvement in reliability** with major incidents nearly eliminated
- **30% fewer DevOps requests** saving ~ 80 hours per ticket
- Standardized global architecture reducing operational complexity
- Improved scalability during sales events with automated autoscaling
- Future-ready infrastructure portable across hybrid and multi-cloud setups