AEGIS

DEVOPS

GitOps Best Practices

Declarative infrastructure and application delivery with ArgoCD, Flux, and Kubernetes

GitOps Principles

Declarative

All infrastructure and applications defined as code. Desired state, not scripts.

Versioned

Git is the single source of truth. Full history, rollback, audit trail.

Approved

Pull requests for all changes. Code review before apply.

Reconciled

Agents continuously sync desired and actual state. Self-healing.

GitOps Tools Comparison

Tool	Type	Best For	Key Features
ArgoCD	Pull-based	Multi-cluster, visibility	Web UI, multi-tenancy, SSO
Flux	Pull-based	Helm, Kustomize native	Lightweight, GitOps toolkit
Rancher Fleet	Pull-based	Large-scale edge	1M+ cluster scale

Best Practices

✅ Do

- Separate app and infra repos
- Use Kustomize for overlays
- Seal/encrypt secrets
- Implement RBAC for repos
- Use signed commits

❌ Don't

- Store plain secrets in Git
- Mix config and app code
- Use mutable image tags
- Skip PR reviews
- Deploy without tests

Sync Strategies

Strategy	Description	Use Case
Manual	Human triggers sync	Production, sensitive envs
Auto Sync	Sync on Git change	Dev, staging environments
Auto Prune	Remove orphaned resources	Clean environments
Self Heal	Revert manual drift	Compliance, consistency

AEGIS

Implement GitOps

Let us help you implement GitOps practices for consistent, auditable, and reliable deployments.

Contact Us

info@aegisit.ai

(404) 490-0234

aegisit.ai