**1. What is GitOps?**

GitOps is a **DevOps practice** that uses **Git as the single source of truth** for **infrastructure and application deployment**. It enables **continuous delivery (CD)** by using **version control, automation, and declarative infrastructure**.

**Key Principles of GitOps**

1. **Declarative Infrastructure**: Everything (infrastructure, application configs) is defined as code.
2. **Version Control (Git as a Source of Truth)**: Git repositories store the desired state.
3. **Automated Deployments**: Changes in Git trigger automated deployments.
4. **Continuous Reconciliation**: A GitOps tool continuously checks for deviations and corrects them.
5. **Auditability & Traceability**: Every change is tracked in Git with commit history.

**How GitOps Works?**

1. **Developers push changes** to Git (e.g., Kubernetes manifests, Helm charts).
2. **GitOps tool (e.g., ArgoCD)** detects changes and automatically applies them to the cluster.
3. **Cluster state is continuously monitored** to match the desired state.

**2. What is ArgoCD?**

ArgoCD is a **GitOps continuous delivery tool** designed for **Kubernetes**. It synchronizes **Kubernetes clusters** with the desired state stored in **Git repositories**.

**Key Features of ArgoCD**

✅ **Declarative & Automated**: Deploys Kubernetes applications automatically from Git.  
✅ **Continuous Syncing**: Monitors and reconciles cluster state to match Git.  
✅ **Rollback & History**: Supports rollbacks using Git commit history.  
✅ **Multi-Cluster Management**: Can deploy applications across multiple Kubernetes clusters.  
✅ **Web UI & CLI**: Provides a dashboard and CLI for management.  
✅ **Access Control & Security**: Integrates with SSO and RBAC policies.  
✅ **Supports Helm, Kustomize, Jsonnet**: Works with different configuration tools.

**3. ArgoCD Architecture**

ArgoCD consists of several components:

**1. API Server**

* Provides a REST API and GraphQL endpoint.
* Powers the Web UI and CLI.

**2. Repository Server**

* Fetches **Git repositories** and **Helm charts**.
* Parses **Kubernetes manifests**.

**3. Application Controller**

* Detects **differences** between Git and Kubernetes clusters.
* Reconciles state to match the Git repository.

**4. Redis**

* Caches application data for performance optimization.

**4. How ArgoCD Works?**

1. **Define Application Configuration in Git**
   * Store Kubernetes manifests, Helm charts, or Kustomize configurations in Git.
2. **ArgoCD Monitors the Repository**
   * Continuously watches for changes in Git.
3. **ArgoCD Synchronizes the State**
   * Applies new configurations to the Kubernetes cluster.
4. **Continuous Reconciliation**
   * If someone manually modifies resources, ArgoCD will revert them to match Git.
5. **5. Installing ArgoCD on Kubernetes**
6. **Step 1: Install ArgoCD**

kubectl create namespace argocd

kubectl apply -n argocd -f <https://raw.githubusercontent.com/argoproj/argo-cd/stable/manifests/install.yaml>

Step 2: Install ArgoCD CLI

curl -sSL -o /usr/local/bin/argocd https://github.com/argoproj/argo-cd/releases/latest/download/argocd-linux-amd64

chmod +x /usr/local/bin/argocd

Step 3: Access the ArgoCD UI

kubectl port-forward svc/argocd-server -n argocd 8080:443

 Open **https://localhost:8080** in a browser.

 Get the admin password:

kubectl get secret argocd-initial-admin-secret -n argocd -o jsonpath="{.data.password}" | base64 -d

Login with:

* Username: **admin**
* Password: **(retrieved from the command above)**

**7. ArgoCD Best Practices**

✅ **Use Separate Repositories** for application code and deployment configurations.  
✅ **Enable Webhooks** to trigger faster synchronization.  
✅ **Use Helm/Kustomize** for managing environment-specific configurations.  
✅ **Enable Role-Based Access Control (RBAC)** to restrict unauthorized changes.  
✅ **Use Notifications** to monitor application health and deployments.

**9. Use Cases of GitOps & ArgoCD**

🔹 **Continuous Deployment for Kubernetes Applications**  
🔹 **Multi-Cluster Kubernetes Management**  
🔹 **Disaster Recovery & Rollback Automation**  
🔹 **Scalability & Multi-Environment Configurations**  
🔹 **Security & Compliance Auditing**

**Conclusion**

🚀 **GitOps + ArgoCD** provide a **declarative, automated, and scalable** way to manage Kubernetes deployments. It ensures **stability, traceability, and fast rollbacks** while simplifying DevOps workflows.

ARGO CD installation steps

sudo apt update && sudo apt upgrade -y

curl -sfL https://get.k3s.io | sh -

systemctl status k3s

kubectl get nodes

export KUBECONFIG=/etc/rancher/k3s/k3s.yaml

echo 'export KUBECONFIG=/etc/rancher/k3s/k3s.yaml' >> ~/.bashrc

source ~/.bashrc

kubectl create namespace argocd

kubectl apply -n argocd -f https://raw.githubusercontent.com/argoproj/argo-cd/stable/manifests/install.yaml

kubectl get pods -n argocd

kubectl patch svc argocd-server -n argocd -p '{"spec": {"type": "NodePort"}}'

kubectl get svc -n argocd

https://<Ubuntu-Server-IP>:<NodePort>

kubectl get secret argocd-initial-admin-secret -n argocd -o jsonpath="{.data.password}" | base64 -d