

CI/CD Flask avec GitHub Actions, Docker Hub et Argo CD

Ce projet met en place un **pipeline CI/CD complet** permettant de :

- Construire automatiquement une image Docker d'une application Flask
 - Publier l'image sur Docker Hub avec un tag unique (SHA du commit)
 - Mettre à jour les manifests Kubernetes
 - Déployer automatiquement l'application sur un cluster Kubernetes via **Argo CD**
-

Architecture globale

Flux de déploiement :

1. Push sur la branche **main** (dossier **/deployment**)
 2. GitHub Actions :
 - Build de l'image Docker
 - Push sur Docker Hub (**latest** + **SHA**)
 - Mise à jour du tag d'image dans **deployment/deployment.yaml**
 - Commit automatique du manifest
 3. Argo CD :
 - Détecte le changement Git
 - Synchronise l'état du cluster Kubernetes
-

Structure du projet

```
.
├── .github/workflows/
│   └── ci-cd.yml
├── deployment/
│   ├── app.py
│   ├── Dockerfile
│   ├── deployment.yaml
│   ├── service.yaml
│   └── namespace.yaml
└── README.md
```

Pipeline GitHub Actions

Le workflow est déclenché à chaque modification dans **/deployment**.

Fonctionnalités :

- Authentification Docker Hub via secrets GitHub

- Tag de l'image basé sur le SHA du commit
- Mise à jour automatique du manifest Kubernetes
- Protection contre boucle CI (commit du bot ignoré)

Secrets requis :

- `DOCKERHUB_USERNAME`
- `DOCKERHUB_TOKEN`

Déploiement avec Argo CD

Argo CD surveille ce dépôt Git et synchronise automatiquement :

- le Deployment
- le Service

L'application est déployée dans le namespace `flask-app`.

Tester l'application

Le Service est de type `ClusterIP`. Pour tester localement :

```
kubectl -n flask-app port-forward svc/flask-app 8081:80
```

Puis ouvrir dans le navigateur :

```
http://localhost:8081
```

Statut

- CI/CD : fonctionnel
- Docker Hub : image versionnée automatiquement
- Argo CD : synchronisation automatique active
- Application : Healthy / Synced

Accéder à Argo

```
$pw_b64 = kubectl -n argocd get secret argocd-initial-admin-secret -o jsonpath="{.data.password}"  
[Text.Encoding]::UTF8.GetString([Convert]::FromBase64String($pw_b64))
```

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

```
https://localhost:8080
```

Auteur

Projet réalisé dans le cadre d'un travail CI/CD Kubernetes avec Argo CD.