

JEDY STARWARS KUBERNETES

Architecture Microservices & Orchestration

Cloud-Native

TP DevOps 2025

Contexte & Objectifs

Mission DevOps

Ce projet simule le cycle de vie complet d'une application moderne. L'objectif est de passer d'un code source à un déploiement en production automatisé, résilient et scalable.

Compétences Clés

- ✓ **Conteneurisation** : Dockerfiles optimisés.
- ✓ **Orchestration** : Déploiement Kubernetes complet.
- ✓ **Automatisation** : Pipeline CI/CD.
- ✓ **Résilience** : Self-healing et mise à l'échelle.

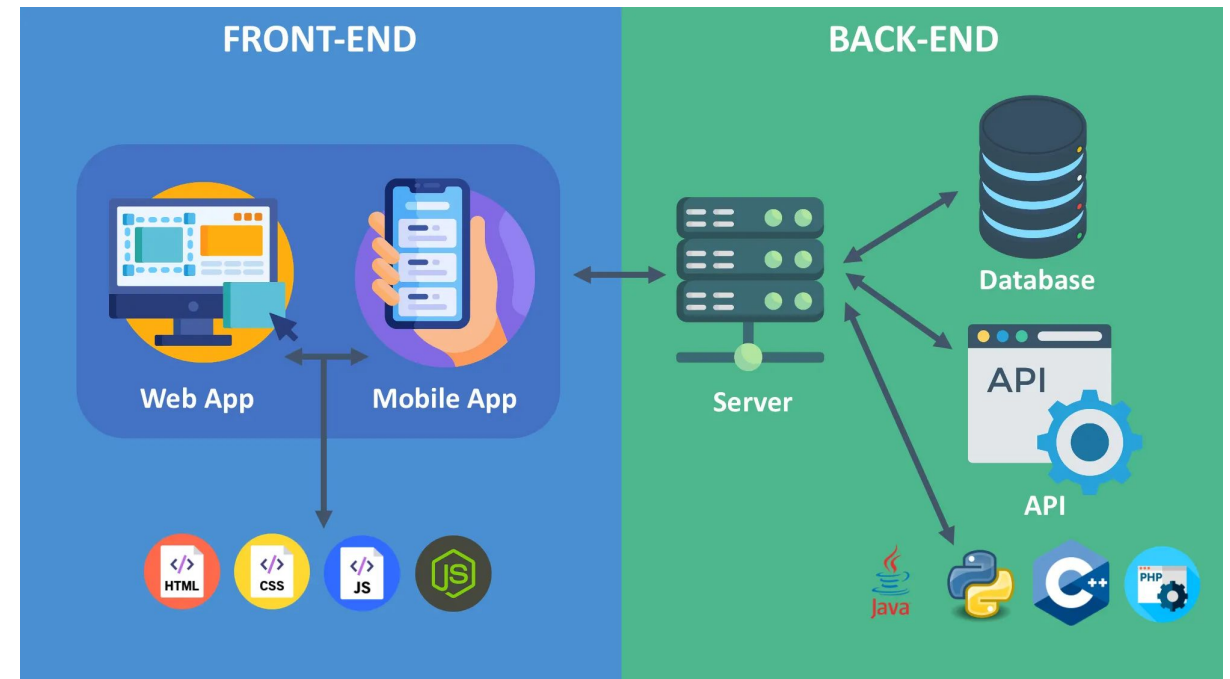
Architecture Microservices

L'application est découpée en services autonomes pour garantir modularité et scalabilité :

- ✓ **Frontend** : Interface utilisateur moderne (Astro/tailwind).
- ✓ **Jedi API** : Backend Python + IA générative.
- ✓ **Base de données** : Redis pour la performance.



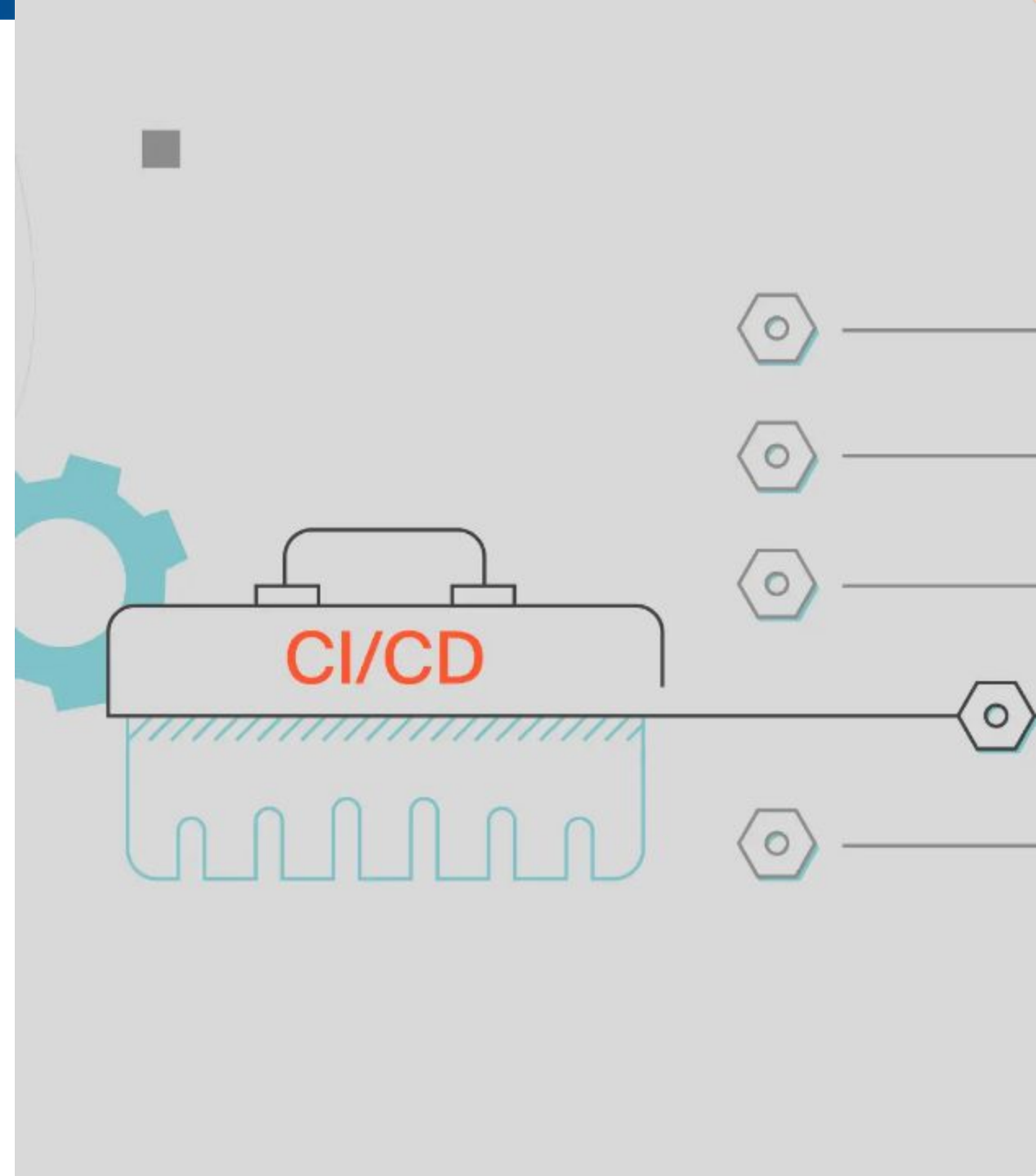
Communication via API REST internes au cluster.



Workflow CI/CD

Un pipeline automatisé garantit que chaque modification de code est testée et déployée sans intervention manuelle.

- ✓ **Build** : Construction des images Docker.
- ✓ **Test** : Vérification de l'intégrité.
- ✓ **Push** : Envoi vers le registre d'images.
- ✓ **Deploy** : Mise à jour Kubernetes.



Déploiement Kubernetes



Deployments

Gestion des Réplicas et Rolling Updates
pour la haute disponibilité.



Services

Exposition via ClusterIP (interne) et
Gateway (externe).



Volumes (PVC)

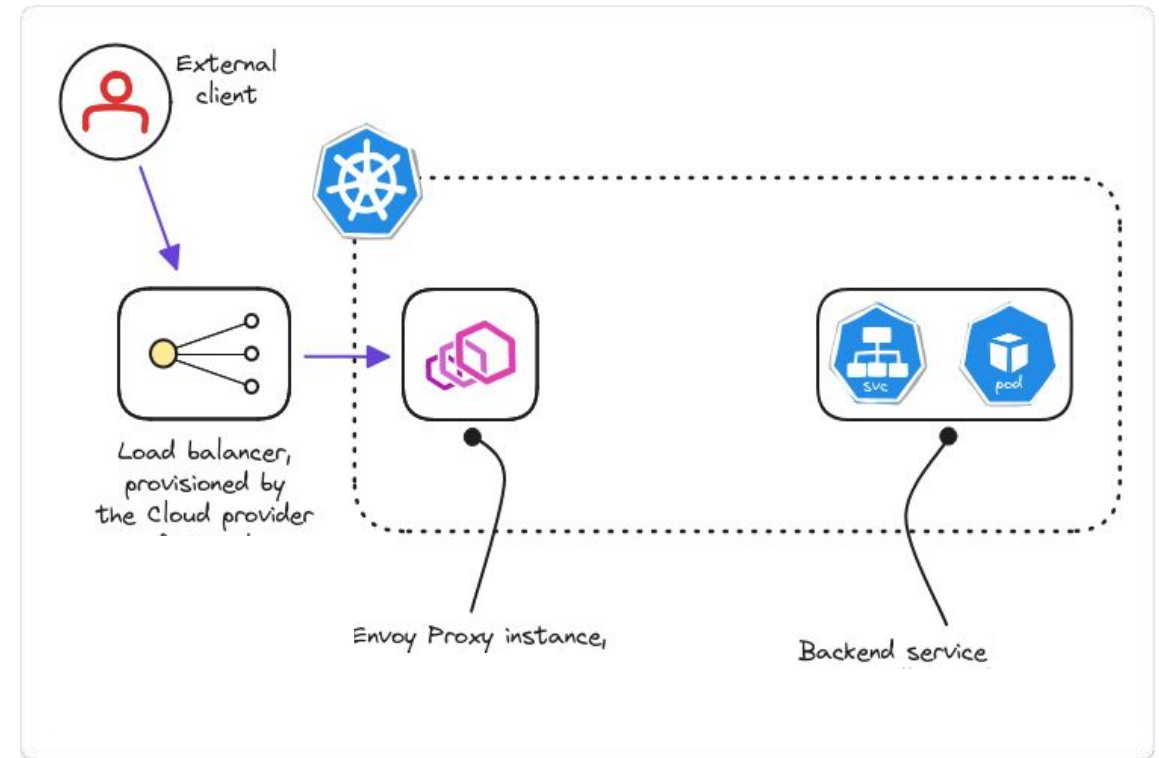
Persistence des données critiques pour
survivre aux redémarrages.

Démonstration : Envoy Gateway

Acheminement du trafic

L'équilibrage de charge répartit les requêtes entrantes, entre plusieurs services backend afin d'améliorer la disponibilité, la réactivité et la scalabilité.

Aidant les services à rester rapides et fiables sous pression.



Défis & Solutions

Gestion des Secrets

Problème : Ne pas commiter les clés API dans Git.

Solution : Utilisation des *Secrets* injectés comme variables d'environnement au runtime pour les données sensibles.

Persistance

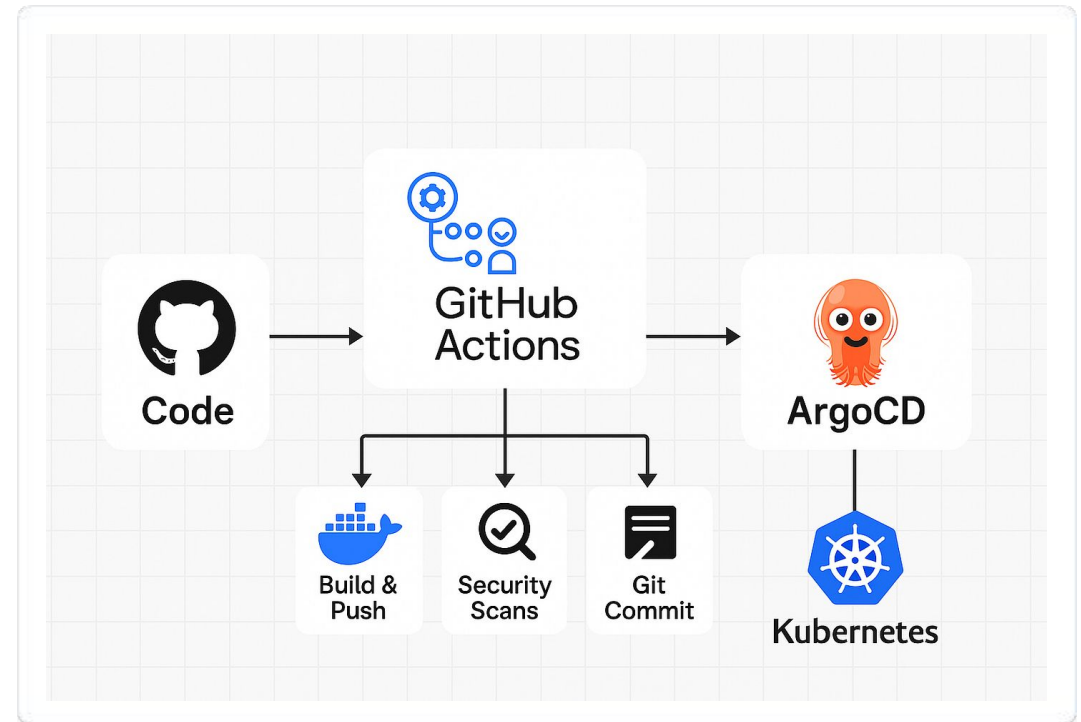
Problème : Les conteneurs sont éphémères.

Solution : Mise en place de volumes pour Docker compose. Implémentation de *PersistentVolumeClaims (PVC)* liés au cycle de vie du cluster pour Kubernetes.

Axes d'Amélioration

Pour durcir l'infrastructure de production :

- ✓ **Monitoring Avancé** : Prometheus et Grafana pour les métriques temps réel.
- ✓ **Sécurité (DevSecOps)** : Scan automatique des images avec **Trivy**.
- ✓ **GitOps** : Gestion des déploiements de manière déclarative avec **ArgoCD**.
- ✓ **Service Mesh** : Istio pour la sécurité mTLS.





Questions ?

Merci de votre attention.

<https://github.com/mjcc30/Jedy-StarWarsKubernetes>