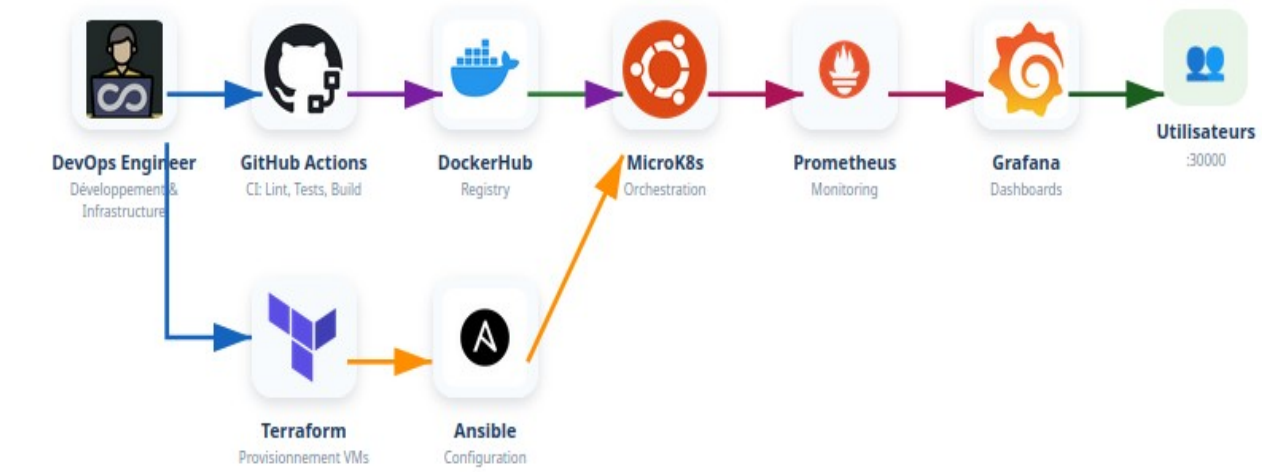
Vote App - Rapport Projet Fil Rouge DevOps

Mbainaissem Narcisse MEDION

Introduction

Ce projet illustre une chaîne DevOps complète : développement, déploiement, supervision, sécurité. L'objectif principal est de prouver la maîtrise de chaque brique moderne : CI/CD, IaC, orchestration, monitoring, et sécurité.

Architecture du Projet



Découpage et choix des technologies

- **Docker** : Containerisation universelle pour le backend, frontend, et Redis (portabilité, reproductibilité).
- **GitHub Actions** : Plateforme CI solide, native GitHub, facile à intégrer pour tests/lints/builds automatiques.
- **Terraform** : Provisionnement IaC, création dynamique des VMs (Proxmox), documenté, versionnable.
- **Ansible** : Configuration automatique, installation des systèmes/tools (Docker, MicroK8s), idéal pour clusters.
- **MicroK8s** : Orchestration Kubernetes locale, légère, facile à script Ansible.
- **Prometheus & Grafana** : Monitoring temps réel, dashboards, alertes, visualisation de tout le cluster.

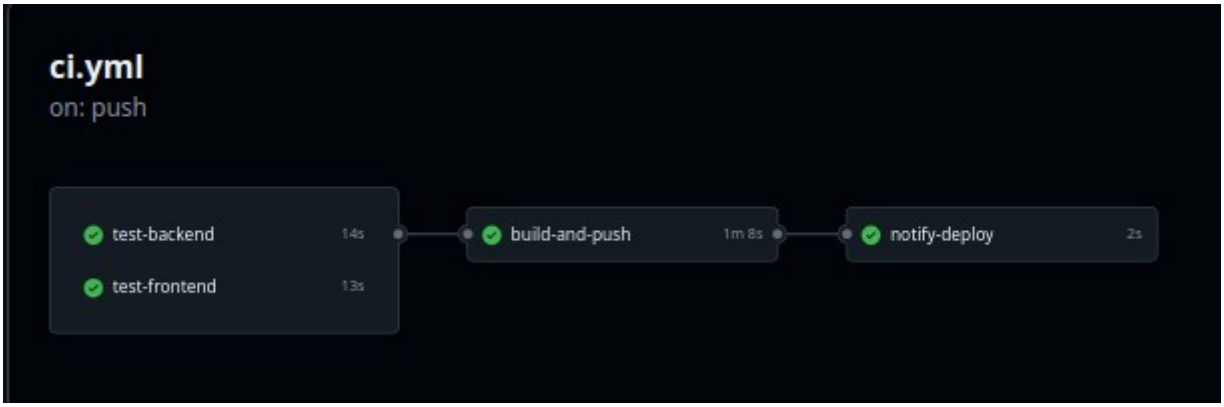
- **Scripts Bash** : permettent un CD manuel, sécurisé (sudo local, jamais ouvert au public).

Difficultés rencontrées

- **Restrictions réseau** : Terraform et Ansible impossibles à exécuter en CI GitHub car Proxmox & K8s sont sur un réseau local non exposé.
- **Sécurité des accès** : impossibilité de tout automatiser sans prise de risque (mot de passe sudo, non-exposition des VM), nécessité de scripts manuels.
- **Interopérabilité** : Compilation et tests sur plusieurs environnements, gestion des variables sensibles.
- **Monitoring** : Configuration avancée Prometheus/Grafana, tuning des alertes pour cluster homogène.

Déroulement détaillé

1. **Containerisation**
    - Dockerfile multistage, docker-compose pour toute la stack en mode dev.
    - Build d'images déployables sur DockerHub.
2. **CI (GitHub Actions)**
    - Lint, tests, build/push des images sur DockerHub à chaque commit.

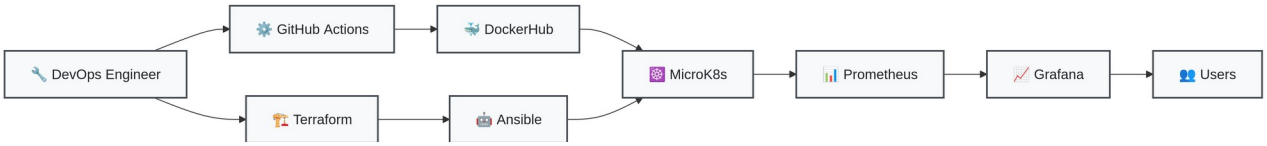


3. **IaC (Terraform)**
    - Création des VMs (1 master + 2 workers) via Proxmox.
    - Inventaire dynamique pour Ansible.
4. **Configuration (Ansible)**

- Installation Docker, MicroK8s, configuration système et users, activation des addons K8s.

5. **Déploiement Applicatif + Supervision**
    - Déploiement du backend, frontend, Redis via playbooks.
    - Installation auto de Grafana & Prometheus, création du dashboard cluster.
    - Activation du HPA (Horizontal Pod Autoscaler), test de charge.
6. **CD manuel (script bash sécurisé)**
    - Un script local permet de déployer toute la stack avec sécurité (mot de passe sudo, accès local uniquement).

Architecture visuelle



Monitoring & supervision

- Grafana montre :
    - Nodes, pods, CPU/RAM, redémarrages, Health Pods
    - Alertes configurables pour toute l’infra
- Prometheus collecte et expose toutes les métriques du cluster

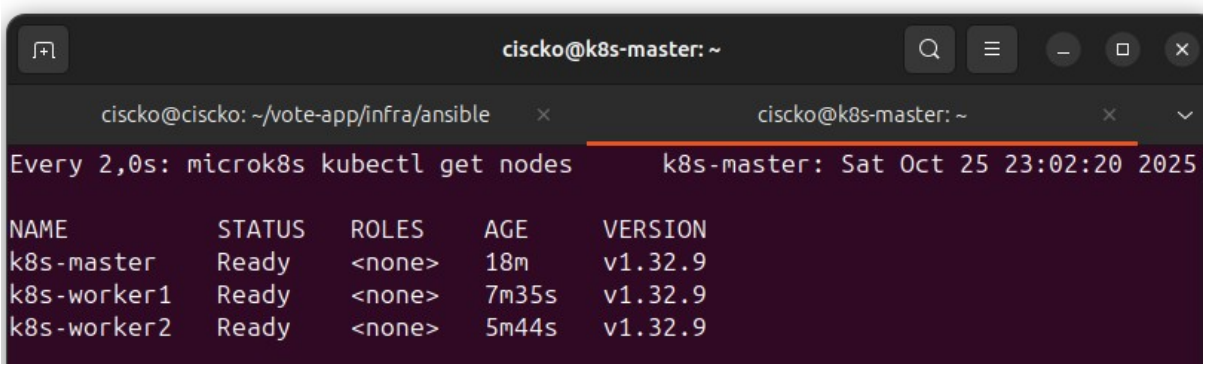


Figure 1: Nodes du cluster

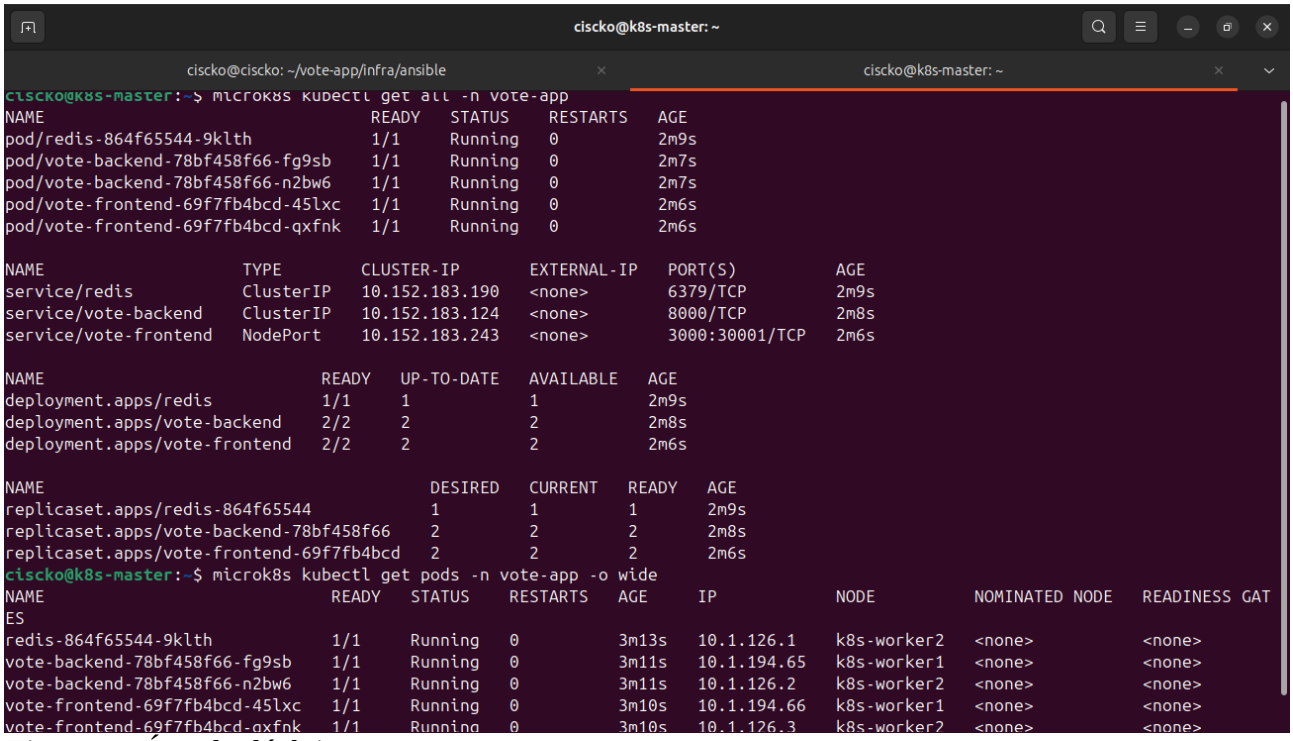


Figure 2 : État du déploiement

```
ciscko@k8s-master:~$ microk8s kubectl get hpa -n vote-app --watch
NAME               REFERENCE                   TARGETS       MINPODS   MAXPODS   REPLICAS   AGE
vote-backend-hpa   Deployment/vote-backend   cpu: 2%/50%   2         5         2          63m
vote-backend-hpa   Deployment/vote-backend   cpu: 45%/50%   2         5         2          63m
vote-backend-hpa   Deployment/vote-backend   cpu: 74%/50%   2         5         2          63m
vote-backend-hpa   Deployment/vote-backend   cpu: 72%/50%   2         5         3          63m
vote-backend-hpa   Deployment/vote-backend   cpu: 51%/50%   2         5         3          64m
vote-backend-hpa   Deployment/vote-backend   cpu: 52%/50%   2         5         3          64m
vote-backend-hpa   Deployment/vote-backend   cpu: 51%/50%   2         5         3          65m
vote-backend-hpa   Deployment/vote-backend   cpu: 52%/50%   2         5         3          66m
vote-backend-hpa   Deployment/vote-backend   cpu: 50%/50%   2         5         3          66m
vote-backend-hpa   Deployment/vote-backend   cpu: 52%/50%   2         5         3          66m
vote-backend-hpa   Deployment/vote-backend   cpu: 50%/50%   2         5         3          66m
vote-backend-hpa   Deployment/vote-backend   cpu: 52%/50%   2         5         3          67m
vote-backend-hpa   Deployment/vote-backend   cpu: 51%/50%   2         5         3          67m
vote-backend-hpa   Deployment/vote-backend   cpu: 52%/50%   2         5         3          67m
vote-backend-hpa   Deployment/vote-backend   cpu: 51%/50%   2         5         3          67m
vote-backend-hpa   Deployment/vote-backend   cpu: 53%/50%   2         5         3          68m
vote-backend-hpa   Deployment/vote-backend   cpu: 24%/50%   2         5         3          68m
vote-backend-hpa   Deployment/vote-backend   cpu: 2%/50%    2         5         3          68m
vote-backend-hpa   Deployment/vote-backend   cpu: 2%/50%    2         5         3          73m
vote-backend-hpa   Deployment/vote-backend   cpu: 3%/50%    2         5         2          73m
```

Figure 3: Test de HPA

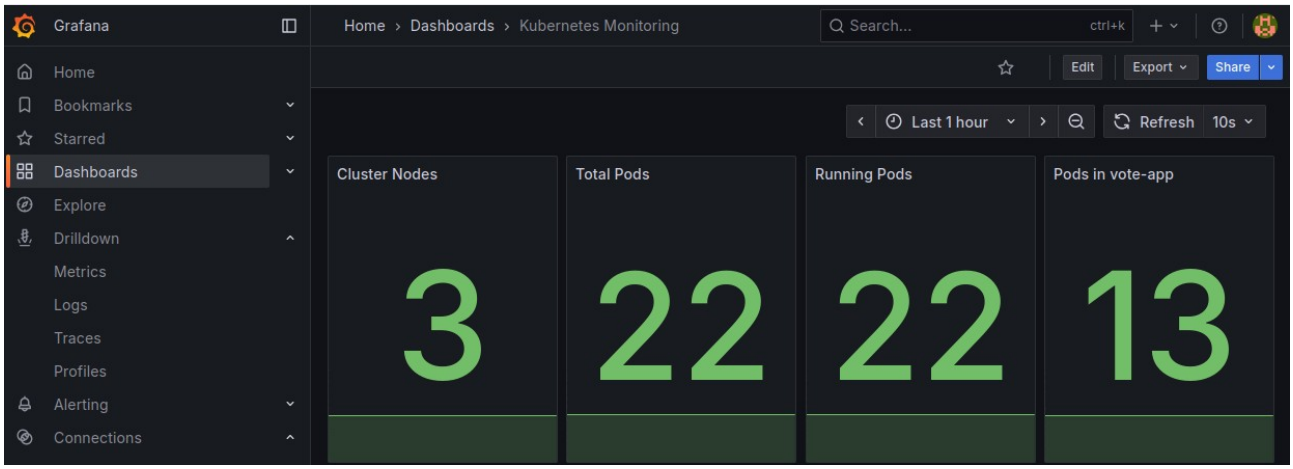


Figure 4: Nombre de pods

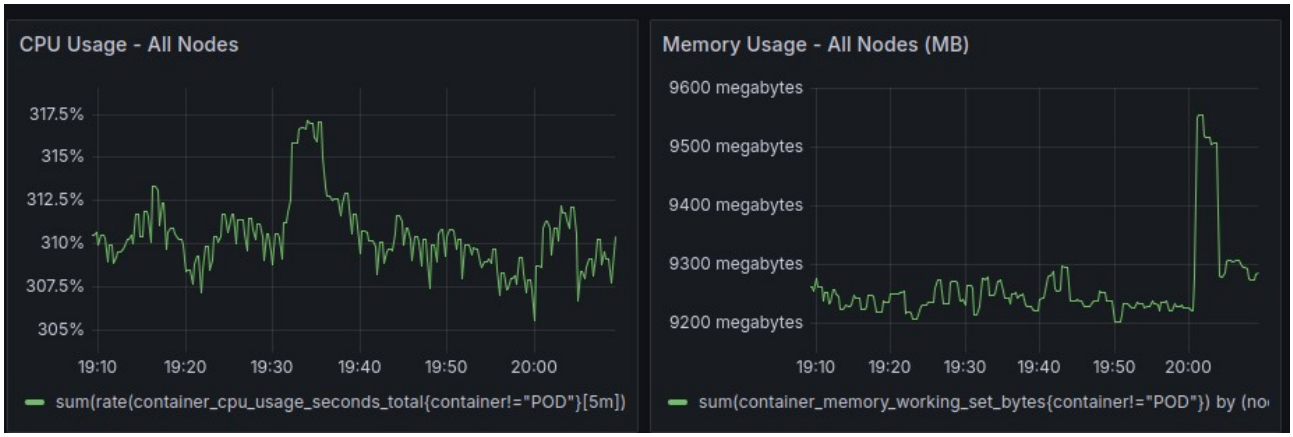


Figure 5: Taux d'utilisations du CPU et mémoires des Nodes

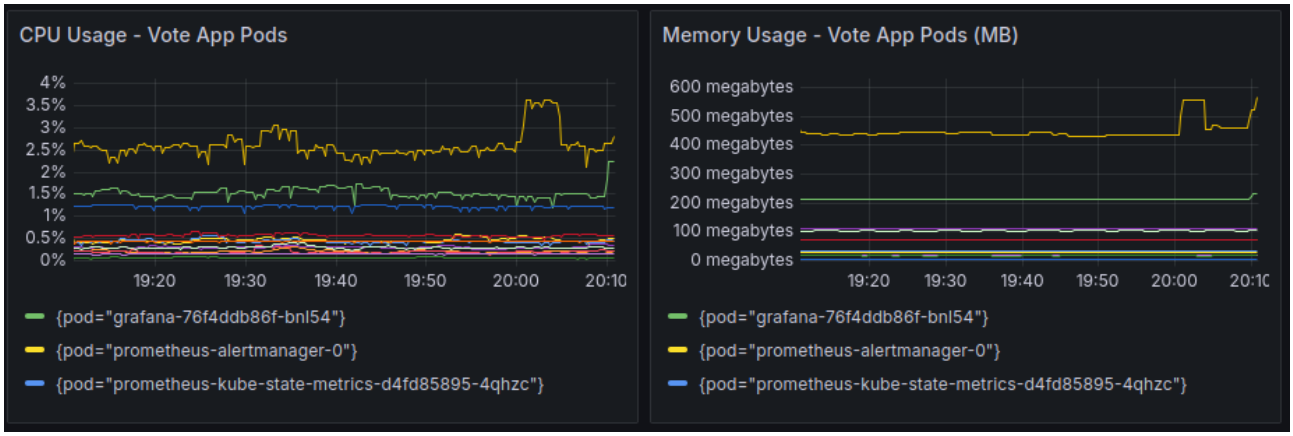


Figure 6: Taux d'utilisations du CPU et mémoires des Pods

Sécurité et évolutivité

- Jamais d'accès root ou sudo automatisés sans prompt; tout reste local pour la partie critique.
- Extension possible vers cluster cloud (AWS) ou runner GitHub Action auto-hébergé pour une CI/CD full auto.
- Ajout possible de scans Trivy, SonarQube, Kube-bench, Vault à court terme.