Manipulation de Kubernetes

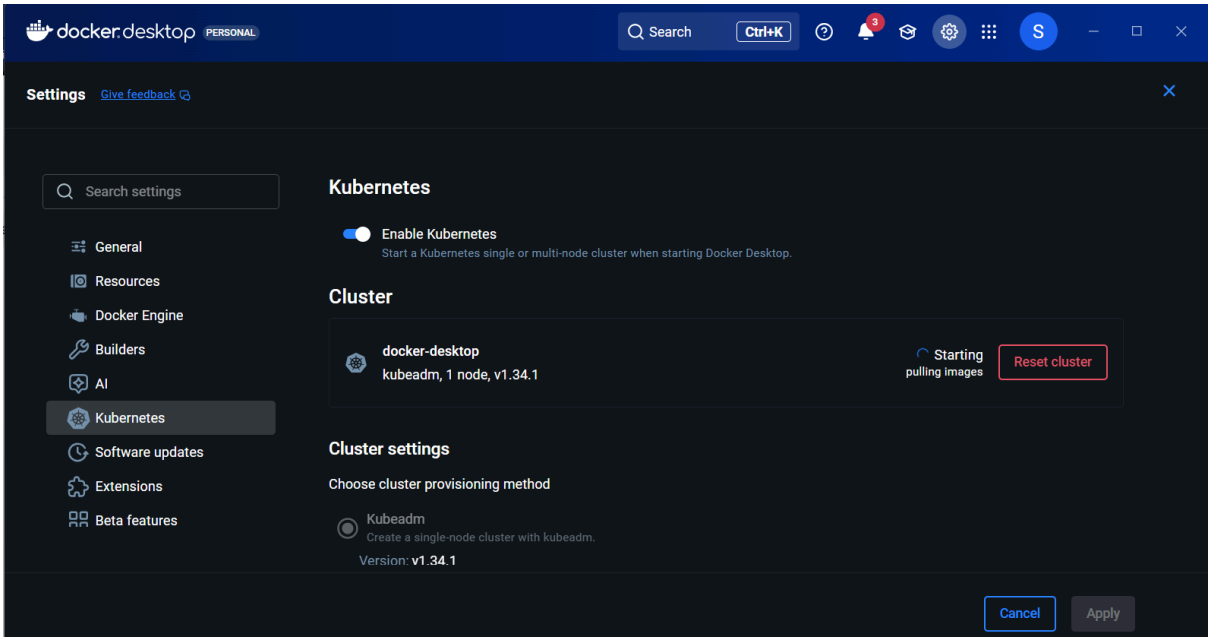
Prérequis :

1. Kubectl : Le client/command-line pour contrôler Kubernetes

kubectl est l'outil en ligne de commande qui sert à interagir avec un cluster Kubernetes (Minikube, un cluster cloud, etc.)
C'est l'équivalent de :

- git → pour Git
- docker → pour Docker
- kubectl → pour Kubernetes

Docker Desktop inclut un cluster Kubernetes intégré. On l'installe dans les paramètres de Docker Desktop.



On vérifie dans le Powershell :

```
PS C:\Users\cyrin> kubectl version --client
Client Version: v1.34.1
Kustomize Version: v5.7.1
```

- Cette commande vérifie que l'outil kubectl est installé et quelle version tu as côté client.

```
PS C:\Users\cyrin> kubectl cluster-info
Kubernetes control plane is running at https://kubernetes.docker.internal:6443
CoreDNS is running at https://kubernetes.docker.internal:6443/api/v1/namespaces/kube-system/services/kube-dns:dns/proxy

To further debug and diagnose cluster problems, use 'kubectl cluster-info dump'.
```

- Cette commande vérifie si un cluster Kubernetes est accessible.

2. Installation de Minikube : Le mini-cluster Kubernetes local

Minikube est un outil qui permet d'installer et faire tourner **un vrai cluster Kubernetes** sur l'ordinateur (Windows, Mac ou Linux).
Minikube est utilisé pour :

- Faire tourner Kubernetes en local
- Tester des déploiements Kubernetes sans serveur distant
- Développer, expérimenter, simuler des pannes, scaler, etc.
- Suivre des TP / projets Kubernetes sans cluster cloud

```
PS C:\Users\cyrin> minikube version
minikube version: v1.37.0
commit: 65318f4cfff9c12cc87ec9eb8f4cdd57b25047f3
PS C:\Users\cyrin>
```

Manipulation:

- Démarrer Minikube avec Docker

```
PS C:\Users\cyrin> minikube start --driver=docker
* minikube v1.37.0 sur Microsoft Windows 11 Home Single Language 10.0.26200.7309 Build 26200.7309
* Utilisation du pilote docker basé sur la configuration de l'utilisateur
* Utilisation du pilote Docker Desktop avec le privilège root
* Démarrage du nœud "minikube" primary control-plane dans le cluster "minikube"
```

  On met ce flag car Docker Desktop est ce qui fera tourner tes machines virtuelles Kubernetes.
- La commande ci-dessous permet de vérifier que mon cluster Kubernetes fonctionne.

```
PS C:\Users\cyrin> kubectl get nodes
NAME       STATUS   ROLES           AGE   VERSION
minikube   Ready    control-plane   62m   v1.34.0
PS C:\Users\cyrin>
```

- Créer un serveur Nginx :
  - Créer un déploiement

```
PS C:\Users\cyrin> kubectl create deployment my-nginx --image=nginx
deployment.apps/my-nginx created
```

  - Vérifier les pods

```
PS C:\Users\cyrin> kubectl get pods
NAME                        READY   STATUS              RESTARTS   AGE
my-nginx-54fc6798c5-f8c99   0/1     ContainerCreating   0          52s
```

- Scaler / augmenter le nombre de pods

```
PS C:\Users\cyrin> kubectl scale deployment my-nginx --replicas=4
deployment.apps/my-nginx scaled
```

```
PS C:\Users\cyrin> kubectl get pods
NAME                        READY   STATUS    RESTARTS   AGE
my-nginx-54fc6798c5-b2958   1/1     Running   0          25s
my-nginx-54fc6798c5-f8c99   1/1     Running   0          2m28s
my-nginx-54fc6798c5-q4xnv   1/1     Running   0          25s
my-nginx-54fc6798c5-t88sp   1/1     Running   0          25s
```

- Exposer le déploiement
  On crée un service accessible depuis l'extérieur :

```
PS C:\Users\cyrin> kubectl expose deployment my-nginx --type=NodePort --port=80
service/my-nginx exposed
```

On vérifie par la suite les services :

```
PS C:\Users\cyrin> kubectl get services
NAME         TYPE        CLUSTER-IP     EXTERNAL-IP   PORT(S)        AGE
kubernetes   ClusterIP   10.96.0.1      <none>        443/TCP        68m
my-nginx     NodePort    10.98.180.70   <none>        80:30551/TCP   28s
```

- Nettoyer / Libérer les ressources
  Supprimer le service :

```
PS C:\Users\cyrin> kubectl delete service my-nginx
service "my-nginx" deleted from default namespace
```

Supprimer le déploiement :

```
PS C:\Users\cyrin> kubectl delete deployment my-nginx
deployment.apps "my-nginx" deleted from default namespace
```

- Simuler une panne
  Suppression d'un pod ⇒ Kubernetes va recréer automatiquement un nouveau pod.

```
PS C:\Users\cyrin> kubectl get pods
NAME                        READY   STATUS              RESTARTS   AGE
my-nginx-54fc6798c5-6dcx7   0/1     ContainerCreating   0          5s
my-nginx-54fc6798c5-kt7g7   1/1     Running             0          26s
my-nginx-54fc6798c5-r8k2r   1/1     Running             0          5s
my-nginx-54fc6798c5-rcv9q   1/1     Running             0          5s
PS C:\Users\cyrin> kubectl delete pod my-nginx-54fc6798c5-r8k2r
pod "my-nginx-54fc6798c5-r8k2r" deleted from default namespace
PS C:\Users\cyrin> kubectl get pods
NAME                        READY   STATUS              RESTARTS   AGE
my-nginx-54fc6798c5-4jkrz   0/1     ContainerCreating   0          5s
my-nginx-54fc6798c5-6dcx7   1/1     Running             0          42s
my-nginx-54fc6798c5-kt7g7   1/1     Running             0          63s
my-nginx-54fc6798c5-rcv9q   1/1     Running             0          42s
```

Il y a encore 4 pods, car Kubernetes maintient le nombre demandé (replicas=4).

Conclusion sur Kubernetes :

Kubernetes (K8s) est une **plateforme d'orchestration de conteneurs**.
Il sert à déployer, gérer et scaler automatiquement des applications conteneurisées (Docker).

Pods
- Un Pod = la plus petite unité déployable
- Contient 1 ou plusieurs conteneurs (souvent 1)
- Partagent IP, stockage, namespace
- Kubernetes ne déploie jamais directement un conteneur → toujours un Pod

Deployment
- Un Deployment gère un ensemble de Pods identiques.
- Fonctions :
  - Mise à l'échelle (scaling)
  - Auto-réparation (si un Pod tombe, un autre redémarre)
  - Mise à jour continue (rolling update)

Services
- Un Service expose un Pod ou un Deployment via un réseau stable.
- Le Pod peut changer (auto-heal), mais le Service garde une IP fixe.

Minikube
- C'est un Kubernetes local qui permet d'avoir un petit cluster sur ton PC.

kubectl (le client Kubernetes)
- C'est l'outil CLI pour interagir avec Kubernetes.

Cycle de travail typique en kubernetes
1. Lancer minikube
2. Créer un déploiement
3. Observer les pods
4. Scaler l'application
5. Exposer l'application
6. Accéder dans le navigateur

Avantages de Kubernetes
- Scalabilité automatique
- Redondance / haute disponibilité
- Gestion réseau
- Déploiements propres sans downtime
- Fonctionne sur cloud, bare metal, local

  ⇒ Kubernetes est utilisé dans toutes les grandes infrastructures modernes.