

# Lab 3 - Créer et déployer une représentation logique de Pods avec les Deployments

COMPTE RENDU

YOUSSEF BEN GHORBEL

# Table des matières

<b>Objectifs .....</b>	<b>2</b>
<b>Pré-requis .....</b>	<b>2</b>
<b>Étapes à suivre .....</b>	<b>2</b>
1. <b>Créer un Namespace.....</b>	<b>2</b>
2. <b>Créer un Deployment .....</b>	<b>2</b>
3. <b>Vérifier le Deployment .....</b>	<b>3</b>
4. <b>Configurer la montée en charge .....</b>	<b>3</b>
5. <b>Mettre à jour l'image Docker.....</b>	<b>4</b>
6. <b>Créer une nouvelle version du Deployment.....</b>	<b>4</b>
7. <b>Gérer les révisions .....</b>	<b>5</b>

## Objectifs

- Écrire un fichier de configuration Deployment.
- Réaliser une montée en charge horizontale via les ReplicaSet.
- Gérer une montée en version.

## Pré-requis

- Assurez-vous que le Namespace mynamespaceexercice1 soit supprimé

```
kubectl delete namespace mynamespaceexercice1
```

## Étapes à suivre

### 1. Créer un Namespace

- Créez un fichier mynamespaceexercice2.yaml

```
apiVersion: v1
kind: Namespace
metadata:
  name: mynamespaceexercice2
```

- Appliquer le fichier :

```
kubectl apply -f exercice2-deployment/mynamespaceexercice2.yaml
```

```
[root@localhost exercice2-deployment]#
namespace/mynamespaceexercice2 created
[root@localhost exercice2-deployment]#
```

### 2. Créer un Deployment

- Créez un fichier mydeployment.yaml

```
apiVersion:
apps/v1 kind:
Deployment
metadata:
  name:
mydeployment spec:
  selector:
    matchLabels:
      app: mypod
  template:
    metadata:
      labels:
```

```

    app: mypod
spec:
  containers:
  - name:
    mycontainer
    image:
    nginx:1.19
    ports:
    - containerPort: 80

```

- Appliquer le fichier :

```

kubectl apply -f exercice2-deployment/mydeployment.yaml -n
mynamespaceexercice2

```

```

[root@localhost exercice2-deployment]# kubectl apply -f mydeployment.yaml -n mynamespaceexercice2
deployment.apps/mydeployment created
[root@localhost exercice2-deployment]#

```

### 3. Vérifier le Deployment

- Vérifier le Pod

```

kubectl get pods -n mynamespaceexercice2 -o wide

```

```

[root@localhost exercice2-deployment]# kubectl get pods -n mynamespaceexercice2
NAME                                READY   STATUS    RESTARTS   AGE
mydeployment-57bb89d874-tftx6       1/1     Running   0           5m36s
[root@localhost exercice2-deployment]#

```

- Vérifier les déploiements :

```

kubectl get deployments.apps -n mynamespaceexercice2

```

```

[root@localhost exercice2-deployment]# kubectl get deployments.apps -n mynamespaceexercice2
NAME          READY   UP-TO-DATE   AVAILABLE   AGE
mydeployment  1/1     1            1           5m36s
[root@localhost exercice2-deployment]#

```

### 4. Configurer la montée en charge

- Modifiez mydeployment.yaml pour inclure les réplicas :

```

spec:
  replicas: 3

```

- Appliquez la nouvelle configuration :

```

kubectl apply -f exercice2-deployment/mydeployment.yaml -n
mynamespaceexercice2

```

```

[root@localhost exercice2-deployment]# kubectl apply -f mydeployment.yaml -n mynamespaceexercice2
deployment.apps/mydeployment configured
[root@localhost exercice2-deployment]#

```

## 5. Mettre à jour l'image Docker

- Modifier l'image à nginx:1.20 :

```
kubectl set image -n mynamespaceexercice2 deployment mydeployment  
mycontainer=nginx:1.20
```

```
[root@localhost exercice2-deployment]# kubectl  
deployment.apps/mydeployment image updated  
[root@localhost exercice2-deployment]#
```

- Annoter le changement :

```
kubectl annotate deployments.apps -n mynamespaceexercice2  
mydeployment kubernetes.io/change-cause="Image en 1.20"
```

```
[root@localhost exercice2-deployment]# kubectl a  
deployment.apps/mydeployment annotated  
[root@localhost exercice2-deployment]#
```

## 6. Créer une nouvelle version du Deployment

- Créez mydeployment121.yaml :

```
apiVersion: apps/v1  
kind: Deployment  
metadata:  
  name: mydeployment  
spec:  
  replicas: 3  
  selector:  
    matchLabels:  
      app: mypod  
  template:  
    metadata:  
      labels:  
        app: mypod  
    spec:  
      containers:  
      - name:  
        mycontainer  
        image:  
        nginx:1.21  
        ports:  
        - containerPort: 80
```

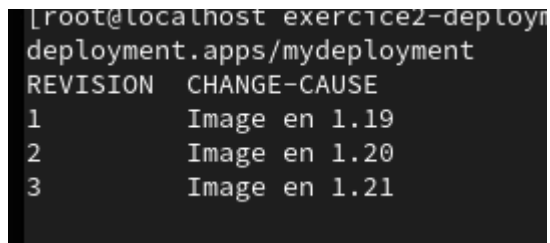
- Appliquez la nouvelle configuration :

```
kubectl apply -f exercice2-deployment/mydeployment121.yaml -n  
mynamespaceexercice2
```

## 7. Gérer les révisions

- Afficher l'historique des révisions :

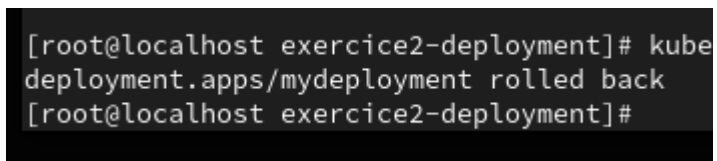
```
kubectl rollout history deployment -n mynamespaceexercice2  
mydeployment
```



```
[root@localhost exercice2-deployment]# kubectl rollout history  
deployment.apps/mydeployment  
REVISION  CHANGE-CAUSE  
1          Image en 1.19  
2          Image en 1.20  
3          Image en 1.21
```

- Revenir à une Révision Antérieure

```
kubectl rollout undo deployment mydeployment -n  
mynamespaceexercice2 --to-revision=1
```



```
[root@localhost exercice2-deployment]# kubectl rollout undo  
deployment.apps/mydeployment rolled back  
[root@localhost exercice2-deployment]#
```