Fase 4 - Activitat 10.2: Gestio mínima de nodes i desplegaments d'un cluster amb microk8s.

0- Identificació del grup i activitat:

Curs: ASIX2
Projecte: PJ9 DevOps i Cloud Computing
Fase: 4
Activitat: 10.2
Grup/Individual: Individual
Membres/Alumne:

1- Objectius de l'activitat 10.2

a) Desinstal·lació de kubernetes
b) Afegint i eliminant nodes d'un cluster
c) Actualització de desplegaments
d) Rollbacks de desplegaments
e) Esborrament i comprovació de HA de Pods.
f) Esborramet de deployments, serveis i proxys ingress.

2- Desinstal·lació de microk8s i eliminació de clusters de kubernetes

a) Dins de **node1**, atura **microk8s**. Executa:

```
microk8s stop
```

b) A continuació, desinstal·la **microk8s** i esborra tots els arxius de configuració:

```
sudo snap remove microk8s --purge
```

c) Comprova que s'ha esborrat **/var/snap/microk8s** a on es desen les configuracions del programa i clusters.

d) Realitza la mateixa tasca a **node2** i **node3**.

3- Afegint i eliminant nodes d'un cluster

3.1- Unint un node a un cluster

a) Reinstal·la **snap** en **node1** i **node2**. Executa:

```
sudo snap install microk8s --classic
```

NOTA: L'opció --**classic** és necessària per donar a **microk8s** accés a recursos de sistema com per exemple la xarxa, el sistema d'emmagatzematge i gestió de processos.

b) Afegeix **node2** com a **worker** d'un cluster controlat per **node1** com a **control plane**. Executa:

- En **node1** → **microk8s add-node**
- En **node2** → L'ordre mostrada per **node1** per unir un node worker al cluster.

c) Comprova des de **node1** que els **2 nodes** estan funcionant (Ready) dins del cluster amb l'ordre:

```
microk8s kubectl get nodes
```

3.2- Eliminant un node d'un cluster

a) Dins de **node2**, executa:

```
micro8ks leave
```

b) Dins de **node1**, executa:

```
micro8ks kubectl get nodes
```

i comprova que l'estat de **node1** és **Ready** i l'estat de **node2** és **NotReady**.

3.3- Unit novament un node a un cluster

a) Torna a fer els mateixos passos que vas seguir a l'apartat **3.1** punt **b)** per unir **node2** al cluster.

b) Comprova que ara, l'estat de **node1** és **Ready** i l'estat de **node2** és **Ready** també.

4- Actualització (rollout) d'un desplegament

4.1- Reinstal·lació de microk8s i desplegament novament de l'aplicació ppv sobre node1

a) Torna a desplegar l'aplicació, el servei i ingress sobre **node1** com ho vam fer a l'activitat pj9f4a10.1.

b) Comprova que s'ha creat:
- El **deployment** de l'aplicació → **microk8s kubectl get deployments**
- Els **3 pods** amb l'aplicació, distribuïts entre els dos nodes → **microk8s kubectl get pods -o wide**
- Que s'ha creat el **service** de l'aplicació → **microk8s kubectl get services**
- Que s'ha creat el servei proxy **ingress** → **microk8s kubectl get ingress**

c) Comprova que l'aplicació funciona.

4.2- Creació d'una nova imatge de l'aplicació

a) Dins de **node1** accedeix a la carpeta **ppv** i modifica **Dockerfile** perquè el nou desplegament treballi amb la versió **8.4** de **PHP**.

b) Crea una imatge de contenidor de nom **ippv** i versió **1.1**.

c) Dins del directori **ppv** crea un **tag** de la imatge local **ippv**:**1.1** necessari per poder pujar-la al teu compte de **Docker Hub**.

d) Fes un **login** al teu compte de **Docker Hub**.
e) Ara puja la nova imatge **ippv**:**1.1** al teu dipòsit de **Docker Hub**.

f) Si s'has pogut pujar la imatge, ara ja pots fer un **logout** del teu compte de **Docker Hub**.

4.3- Modificació i actualització (rollout) automàtica del desplegament de l'aplicació

a) Modifica i **actualitza** automàticament el desplegament editant el fitxer **ppv-deployment.yaml** utilitzant la següent ordre:

```
KUBE_EDITOR="nano -I -c" microk8s kubectl edit deployment/depl-ppv
```

i canviant el paràmetre **image** de **ippv:1.0** a **ippv:1.1** dins de la línia **37**.

b) Comprova el sistema mostra el missage: **`deployment.apps/depl-ppv edited`**

c) Finalment, comprova que s'ha fet l'actualització (**rollout**) executant:

```
microk8s kubectl rollout status deployment/depl-ppv
```

i comprova que el missatge final quan finalita el procés d'actualització és aquest: **`deployment "depl-ppv" successfully rolled out`**

d) Comprova que el deployment **depl-ppv** funciona correctament sobre els **node1** executant:

```
microk8s kubectl get deployments
```

e) Executant l'ordre:

```
microk8s kubectl get pods
```

comprova que els **Pods**:

- Funcionen (Running) els **3 pods** sobre **node1** i **node2**.
- Han canviat de nom.
- Que el seu paràmetre **AGE** és ara un valor petit i segurament en segons (o uns pocs minuts).

f) Comprova l'estat d'un (qualsevol dels existents) dels **Pods** amb l'ordre:

Nom real del Pod

```
microk8s kubectl describe pod <nom_pod> | grep "Image:"
```

i comprova que la imatge que utilitza és la versió **1.1** de l'aplicació **ippv** descarregada des del teus dipòsit de **Docker Hub**.

5- Rollback de deployment

a) FesMostra el **nom** del deployment executant

```
microk8s kubectl get deployments
```

i comprova que el nom és **depl-ppv.**

b) Fes un **Roollback** del deployment executant:

```
microk8s  kubectl rollout undo deployment/depl-ppv
```

d) Executant l'ordre:

```
microk8s kubectl get pods
```

comprova que els **Pods**:

- Funcionen (Running) els **3 pods** sobre el **node1** i **node2**.
- Han canviat de nom.
- Que el seu paràmetre **AGE** és ara un valor petit i segurament en segons (o uns pocs minuts).

e) Comprova l'estat d'un (qualsevol dels existents) dels **Pods** amb l'ordre:

Nom real del Pod

```
microk8s kubectl describe pod <nom_pod> | grep "Image:"
```

i comprova que la imatge que utilitza és la versió **1.0** de l'aplicació **ippv** descarregada des del teus dipòsit de **Docker Hub**.

6- Drenant un node de Pods abans d'aturar-lo o deixar el cluster

a) Dins de **node1**, **acordona** (no permets afegir nous pods) el node2. Executa:

```
microk8s kubectl cordon node2
```

i comprova que el sistema mostra el missatge: **`node/node2 cordoned`**

b) Comprova a continuació que l'estat del **node2** és: **Ready,SchedulingDisabled** executant:

```
microk8s kubectl get nodes
```

c) Després de **drena** (fa que tots els pods passin a un altre node) **node2**:

```
microk8s kubectl drain --ignore-daemonsets node2
```

i comprova que el sistema mostra el missatge final: **`node2 drained`**

d) Comprova que tots els **pods** es troben dins de **node1** executant:

```
microk8s kubectl get pods -o wide
```

e) Des de **node2** deixa el cluster i atura **microk8s**. Executa:

```
microk8s leave
microk8s stop
```

f) Des de **node1**, comprova que **node2** ara està en mode **NotReady,SchedulingDisable**. Executa:

```
microk8s kubectl get nodes
```

7- Esborrament d'un deployment, servei i proxy amb ingress

7.1- Esborrament d'un ingress

a) Comprova que tens en funcionament un **ingress** de nom **proxy-ppv**. Executa:

```
microk8s kubectl get ingress
```

b) Esborra **proxy-ppv** executant:

```
microk8s kubectl delete ingress proxy-ppv
```

i comprova que surt el missatge de sistema: **`ingress.networking.k8s.io "proxy-ppv" deleted`**

c) torna a mostrar la llista d'**ingress** en funcionament per comprovar que **proxy-ppv** ha estat esborrat.

7.2- Esborrament d'un service

a) Comprova que tens en funcionament un **service** de nom **serv-ppv**. Executa:

```
microk8s kubectl get services
```

b) Esborra **serv-ppv** executant:

```
microk8s kubectl delete service serv-ppv
```

i comprova que surt el missatge de sistema: `service "serv-ppv" deleted`

c) torna a mostrar la llista de **service** en funcionament per comprovar que **serv-ppv** ha estat esborrat.

7.3- Esborrament d'un deployment

a) Comprova que tens en funcionament un **deployment** de nom **depl-ppv**. Executa:

```
microk8s kubectl get deployments
```

b) Esborra **depl-ppv** executant:

```
microk8s kubectl delete deployment depl-ppv
```

i comprova que surt el missatge de sistema: `deployment.apps "depl-ppv" deleted`

c) torna a mostrar la llista de **deployments** en funcionament per comprovar que **depl-ppv** ha estat esborrat.

Lliurament de l'activitat

- Prova 1: Demostra que **microk8s** i els **clusters** han estat esborrats de tots els nodes.
- Prova 2: Mostra que **node1** i **node2** tornen formar part d'un cluster.
- Prova 3: Elimina **node2** del cluster. Mostra que **node2** no és part del cluster.
- Prova 4: Uneix **node2** del cluster.
- Prova 5: Mostra que **node2** és part del cluster.
- Prova 6: Mostra que has **actualitzat** el **deployment** a la versió **1.1** de la imatge de l'aplicació **ippv**.
- Prova 7: Mostra que has fet un **Rollback** del **deployment** i treballes novament amb la versió **1.0**.
- Prova 8: Mostra que has **acordonat**, **drenat** i **fet marxar** del cluster node2.
- Prova 9: Esborra completament el **deployment**, **service** i **ingress** de l'aplicació **ippv**.
- **Data Limit:** Només s'accepta fins el divendres dia **13-2-26** fins a les **17.45h**.