

<u>Nom Complet</u> : ALPHA SOULEYMANE BAH <u>Encadreur</u> : M. N’Gor Seck	TAF : Vous allez déployer l'application stock-ms dans minikube en mode ligne de commande dans un namespace nommé jee.	<u>Année</u> : 2024-2025 Groupe ISI Dakar (SN) <u>Niveau</u> : M2GL
---	--	---

Les étapes à suivre de a-z sont :

1. Prérequis

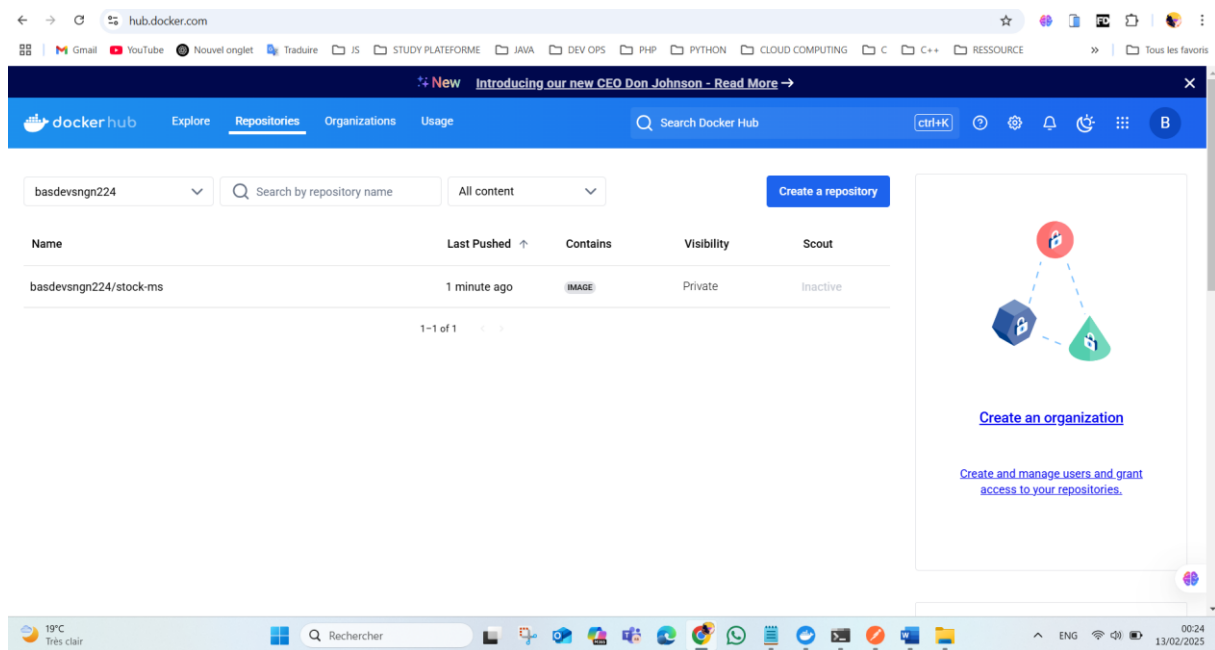
Assurez-vous d'avoir installé et configuré correctement :

- **Minikube** : Installation Minikube
- **Kubectl** : Installation kubectl
- **Docker Desktop** (déjà installé) avec **Kubernetes activé**
- **Hyper-V ou WSL 2** (si nécessaire pour Minikube)

```
PS C:\Users\LENOVO> minikube version
minikube version: v1.34.0
commit: 210b148df93a80eb872ecbeb7e35281b3c582c61
PS C:\Users\LENOVO>
PS C:\Users\LENOVO> kubectl version
Client Version: v1.31.4
Kustomize Version: v5.4.2
Server Version: v1.31.0
PS C:\Users\LENOVO>
```

2. Build de l'application et push sur docker hub

```
View build details: docker-desktop://dashboard/build/desktop-linux/desktop-linux/4qywc0jo8n9pyvu9wqr55m5m
PS D:\MASTER_II_GL\DEV_OPS\k8s_tp> docker push basdevsngn224
Using default tag: latest
The push refers to repository [docker.io/library/basdevsngn224]
An image does not exist locally with the tag: basdevsngn224
PS D:\MASTER_II_GL\DEV_OPS\k8s_tp> docker push basdevsngn224/stock-ms:latest
The push refers to repository [docker.io/basdevsngn224/stock-ms]
e3d1bc399bc1: Layer already exists
6f9ff1c5de04: Layer already exists
6be690267e47: Layer already exists
13a34b66fff78: Layer already exists
9c1b6dd6c1e6: Layer already exists
latest: digest: sha256:a50956dac165ea3207131d75c9f5f323248c1ea96aa0762e654801ff6ec5c92f size: 1371
PS D:\MASTER_II_GL\DEV_OPS\k8s_tp>
```



3. Lançons Minikube et Vérification

```
Windows PowerShell
Copyright (C) Microsoft Corporation. Tous droits réservés.

Installez la dernière version de PowerShell pour de nouvelles fonctionnalités et améliorations ! https://aka.ms/PSWindows

PS C:\Users\LENOVO> minikube start
* minikube v1.34.0 sur Microsoft Windows 11 Pro 10.0.26100.3037 Build 26100.3037
* Utilisation du pilote docker basé sur le profil existant
* Démarrage du nœud "minikube" primary control-plane dans le cluster "minikube"
* Extraction de l'image de base v0.0.45...
* Redémarrage du docker container existant pour "minikube" ...
* minikube 1.35.0 est disponible ! Téléchargez-le ici : https://github.com/kubernetes/minikube/releases/tag/v1.35.0
* Pour désactiver cette notification, exécutez : 'minikube config set WantUpdateNotification false'

! Failing to connect to https://registry.k8s.io/ from inside the minikube container
* Pour extraire de nouvelles images externes, vous devrez peut-être configurer un proxy : https://minikube.sigs.k8s.io/docs/reference/networking/proxy/
* Préparation de Kubernetes v1.31.0 sur Docker 27.2.0...
* Vérification des composants Kubernetes...
  - Utilisation de l'image docker.io/kubernetes/dashboard:v2.7.0
  - Utilisation de l'image gcr.io/k8s-minikube/storage-provisioner:v5
  - Utilisation de l'image docker.io/kubernetes/metrics-scraper:v1.0.8
* Après que le module est activé, veuillez exécuter "minikube tunnel" et vos ressources ingress seront disponibles à "127.0.0.1"
  - Utilisation de l'image registry.k8s.io/ingress-nginx/kube-webhook-certgen:v1.4.3
  - Utilisation de l'image registry.k8s.io/ingress-nginx/controller:v1.11.2
  - Utilisation de l'image registry.k8s.io/ingress-nginx/kube-webhook-certgen:v1.4.3
* Certaines fonctionnalités du tableau de bord nécessitent le module complémentaire metrics-server. Pour activer toutes les fonctionnalités, veuillez exécuter :
  minikube addons enable metrics-server

* Vérification du module ingress...
* Modules activés: storage-provisioner, dashboard, default-storageclass, ingress
* Terminé ! kubectl est maintenant configuré pour utiliser "minikube" cluster et espace de noms "default" par défaut.
PS C:\Users\LENOVO> minikube status
minikube
type: Control Plane
host: Running
kubeadm: Running
apiserver: Running
kubeconfig: Configured
```

3.2. Créer le Namespace jee et Vérification

```
PS C:\Users\LENOVO> kubectl get namespaces
NAME                STATUS    AGE
default             Active   7d
devops              Active  35h
ingress-nginx       Active  35h
jee                 Active  13h
kube-node-lease     Active   7d
kube-public         Active   7d
kube-system         Active   7d
kubernetes-dashboard Active   7d
PS C:\Users\LENOVO> kubectl delete namespaces jee
namespace "jee" deleted
PS C:\Users\LENOVO> kubectl create namespace jee
```

```
PS C:\Users\LENOVO> kubectl get namespaces
NAME                STATUS    AGE
default             Active   7d
devops              Active  35h
ingress-nginx       Active  35h
jee                 Active  21m
kube-node-lease     Active   7d
kube-public         Active   7d
kube-system         Active   7d
kubernetes-dashboard Active   7d
PS C:\Users\LENOVO>
```

4. Création d'une imagePullSecret pour Docker Hub et Vérification

```
PS C:\Users\LENOVO> kubectl create secret docker-registry docker-hub-secret `
>> --docker-server=https://index.docker.io/v1/ `
>> --docker-username=basdevsngn224 `
>> --docker-password=dckr_pat_pyrgBvuL9d74k-lAHIVMWpf4AoY `
>> --docker-email=bahalphasouleymane2019@gmail.com `
>> --namespace=jee
secret/docker-hub-secret created
```

```
PS C:\Users\LENOVO> kubectl get secrets -n jee
NAME                TYPE                                DATA    AGE
docker-hub-secret   kubernetes.io/dockerconfigjson    1        23m
PS C:\Users\LENOVO>
```

5. Déployer l'application stock-ms

```
PS C:\Users\LENOVO> kubectl create deployment stock-ms --image=basdevsngn224/stock-ms:latest --namespace=jee
deployment.apps/stock-ms created
```

5.1. Ajoutons 2 répliques :

```
PS C:\Users\LENOVO> kubectl scale deployment stock-ms --replicas=2 -n jee
deployment.apps/stock-ms scaled
```

6. Associons le imagePullSecret au déploiement

```
PS C:\Users\LENOVO> kubectl patch deployment stock-ms -n jee --type='json' -p='[
>> {
>>   "op": "add",
>>   "path": "/spec/template/spec/imagePullSecrets",
>>   "value": [{ "name": "docker-hub-secret" }]
>> }
>> ]'
deployment.apps/stock-ms patched
```

7. Exposons l'application via un service et Verification

On expose le service sur le port 80 avec NodePort et target-port :8088 :

```
PS C:\Users\LENOVO> kubectl expose deployment stock-ms --type=NodePort --port=80 --target-port=8088 --name=stock-ms-service -
--namespace=jee
```

```
PS C:\Users\LENOVO> kubectl get svc -n jee
```

NAME	TYPE	CLUSTER-IP	EXTERNAL-IP	PORT(S)	AGE
stock-ms-service	NodePort	10.97.250.109	<none>	80:30888/TCP	28m

```
PS C:\Users\LENOVO>
```

8. Vérification des pods

```
PS C:\Users\LENOVO> kubectl get pods -n jee
```

NAME	READY	STATUS	RESTARTS	AGE
stock-ms-68b76c8878-kdvmm	1/1	Running	0	29m
stock-ms-68b76c8878-s6dkq	1/1	Running	0	29m

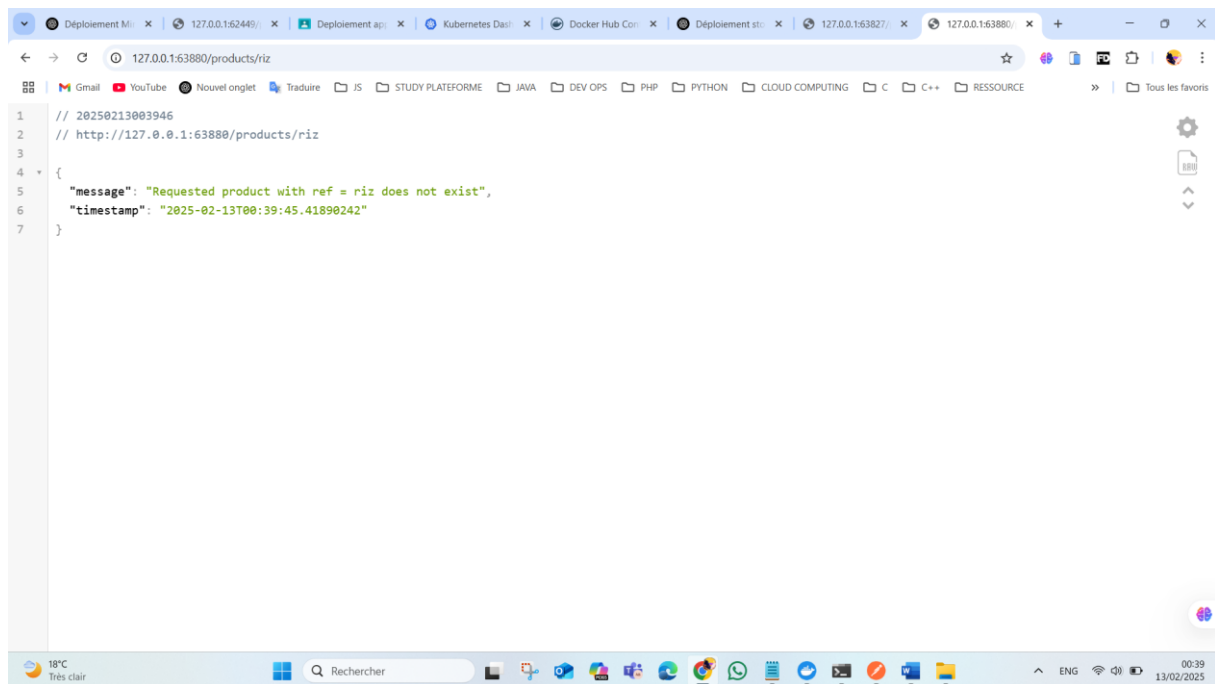
```
PS C:\Users\LENOVO>
```

9. Ouverture et Test de l'application

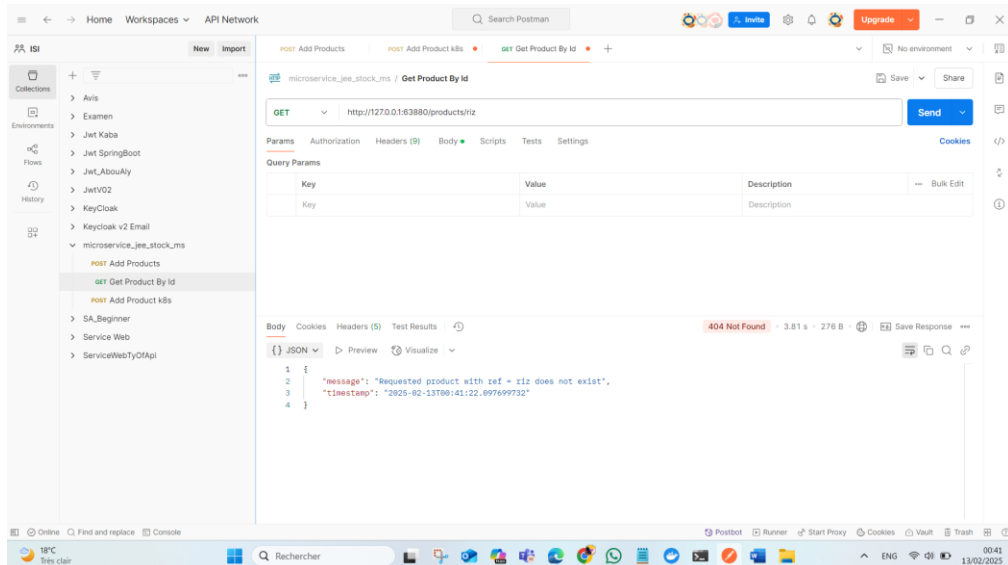
```
PS C:\Users\LENOVO> minikube service stock-ms-service -n jee --url
http://127.0.0.1:63880
! Comme vous utilisez un pilote Docker sur windows, le terminal doit être ouvert pour l'exécuter.
```

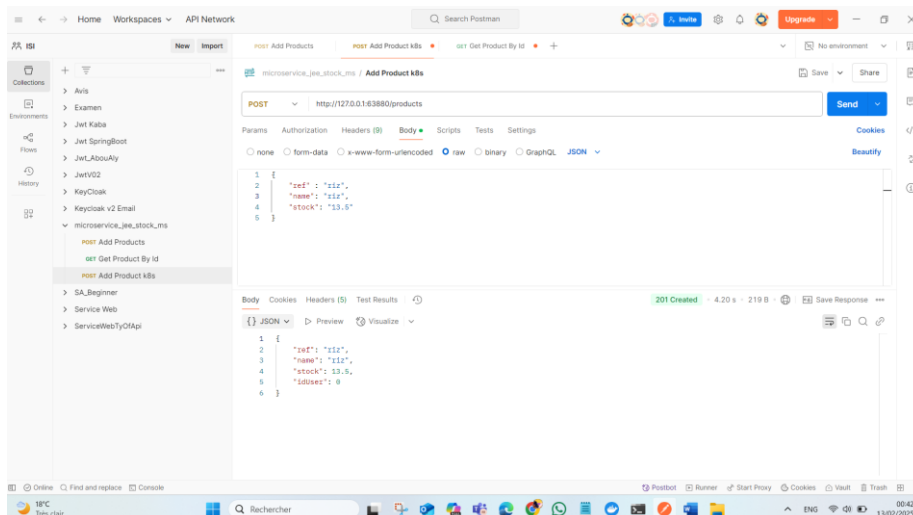
✓ Sur le navigateur avant insertion sur postman

Ref=riz

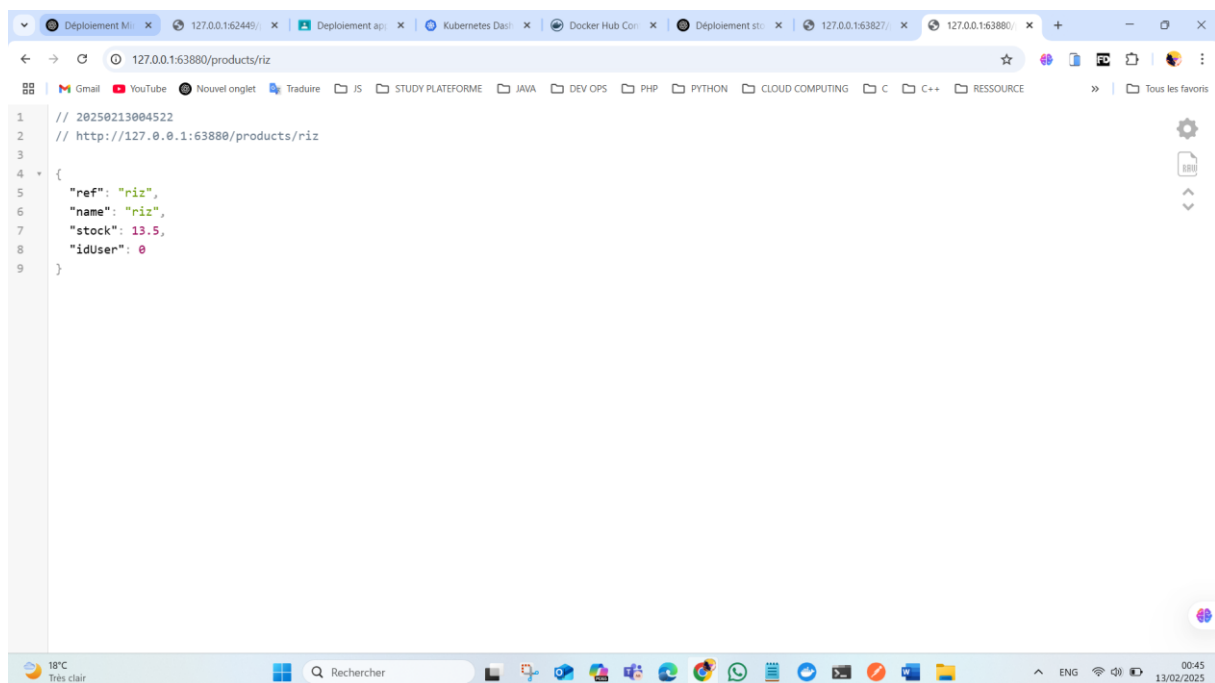


✓ **Insertion sur postman et test**
Sans succès car la ref = riz n'existe





Ajout avec succès



Récupération avec succès

Et enfin

Minikube Dashboard

Charges de travail

Statut des charges de travail

Running 1 Déploiements Running 2 Pods Running 2 Replica Sets

Déploiements

Nom	Images	Étiquettes	Pods	Date de création
stock-ms	basdevsng224/stock-ms:latest	-	2 / 2	8.58r.890

Pods

Nom	Images	Étiquettes	Node	Statut	Replaniments	Utilisation CPU (cores)	Utilisation mémoire (octets)	Date de création
stock-ms-5f688d79-n4vzv	basdevsng224/stock-ms:latest	app: stock-ms pod-template-hash: 5f688d79	minikube	Running	3	-	-	8.58r.890
stock-ms-5f688d79-n4pvr	basdevsng224/stock-ms:latest	app: stock-ms pod-template-hash: 5f688d79	minikube	Running	3	-	-	8.58r.890

Replica Sets

Nom	Images	Étiquettes	Pods	Date de création
stock-ms	basdevsng224/stock-ms:latest	-	2 / 2	8.58r.890

Espaces de noms

Espaces de noms

Nom	Étiquettes	Phase	Date de création
jee	kubernetes.io/metadata.name: jee	Active	42.81r.890
ingress-nginx	app.kubernetes.io/instance: ingress-nginx app.kubernetes.io/name: ingress-nginx kubernetes.io/metadata.name: ingress-nginx	Active	8.58r.890
devops	kubernetes.io/metadata.name: devops	Active	8.58r.890
kubernetes-dashboard	addonmanager.kubernetes.io/mode: Reconcile kubernetes.io/metadata.name: kubernetes-dashboard kubernetes.io/minikube-addon: dashboard	Active	7.58r.890
default	kubernetes.io/metadata.name: default	Active	7.58r.890
kube-node-lease	kubernetes.io/metadata.name: kube-node-lease	Active	7.58r.890
kube-public	kubernetes.io/metadata.name: kube-public	Active	7.58r.890
kube-system	kubernetes.io/metadata.name: kube-system	Active	7.58r.890

Merci !!!!