Projet

Rapport

Maddalena Matteo

Supervisé par
M. B. Charroux

UFR des Sciences fondamentales et biomédicales

Université Paris Cité

Février, 2024

Contents

List of Figures

1

1.1 | Description

1.1.1 | Création des Images Docker

Création de trois images Docker pour les services : frontend, api, et quote. Utilisation de fichiers Dockerfile pour chaque service afin de définir les dépendances, l'environnement, et les commandes d'exécution.

1.1.2 | Construction et Publication des Images Docker

Utilisation de la commande "docker build" pour construire les images Docker à partir des fichiers Dockerfile (Ici le "docker build" à été remplacé par un "docker-compose"). Attribution de tags aux images avec la commande docker tag. Publication des images sur Docker Hub avec la commande docker push.

1.1.3 | Création des Déploiements Kubernetes

Écriture de fichiers YAML pour définir les déploiements de chaque service (frontend, api, quote). Utilisation des déploiements pour spécifier le nombre de répliques, les conteneurs, les images.

1.1.4 | Création des Services Kubernetes

Écriture de fichiers YAML pour définir les services de chaque service (frontend, api, quote). Les services exposent les ports nécessaires pour communiquer avec les con-

teneurs.

1.1.5 | Configuration d'une Gateway avec Docker

Lancement de Minikube :
Utilisation de Minikube pour créer un cluster Kubernetes local. Démarrage du cluster avec la commande minikube start.

1.1.6 | Publication des Services avec Minikube

Utilisation de la commande minikube service pour exposer un service à l'extérieur du cluster. Récupération de l'URL exposée avec la commande minikube service myservice –url.

1.2 | Captures d'écran :

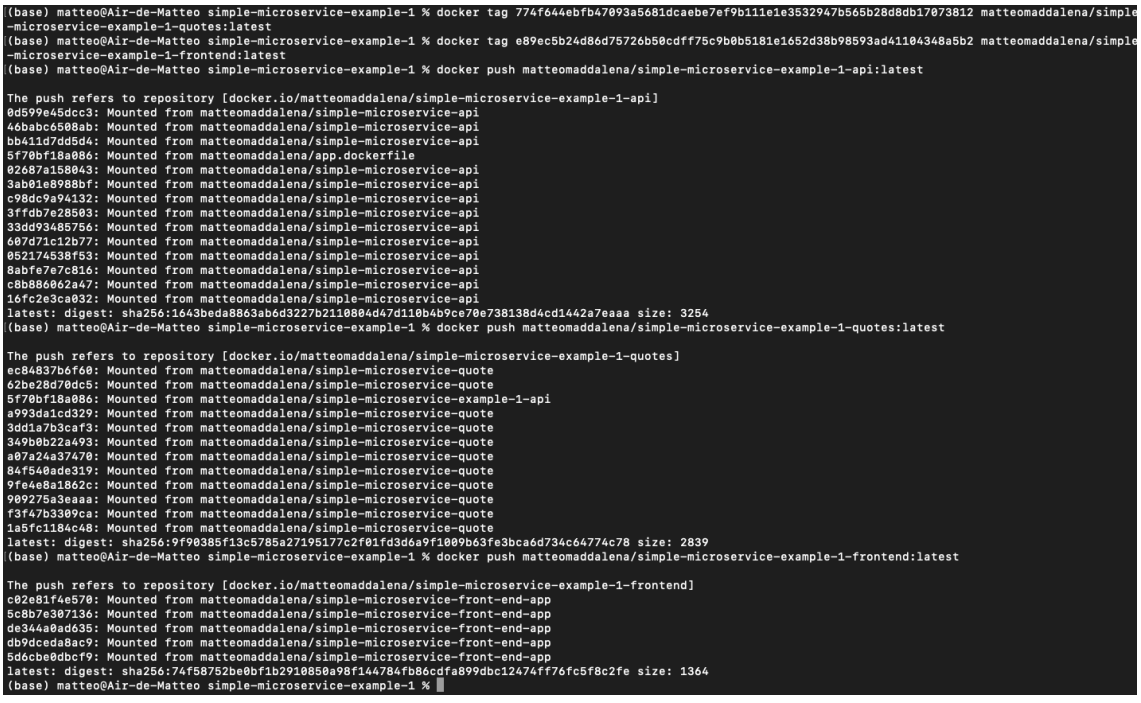

Figure 1.1: Création des images et push grâce à docker

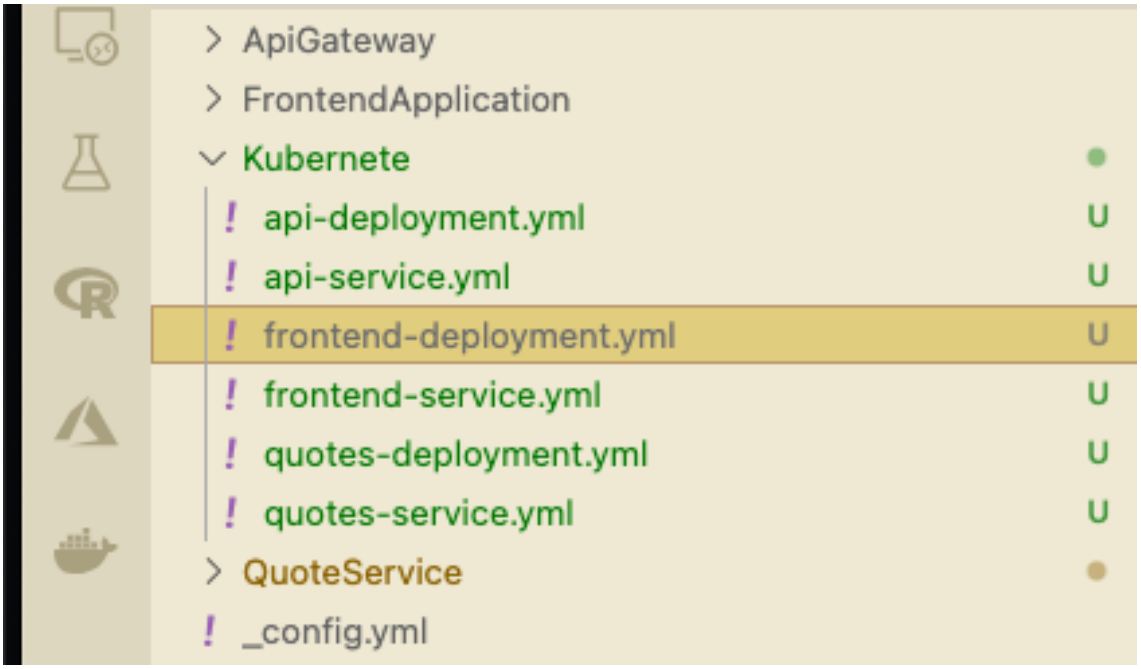

Figure 1.2: Création de 6 fichiers pour le déploiement et les services

```
(base) matteo@Air-de-Matteo simple-microservice-example-1 % cd Kubernete
(base) matteo@Air-de-Matteo Kubernete % ls
api-deployment.yml      api-service.yml         frontend-deployment.yml frontend-service.yml    quotes-deployment.yml   quotes-service.yml
(base) matteo@Air-de-Matteo Kubernete % kubectl apply -f api-deployment.yaml
error: the path "api-deployment.yaml" does not exist
(base) matteo@Air-de-Matteo Kubernete % kubectl apply -f api-deployment.yml
deployment.apps/api-deployment created
(base) matteo@Air-de-Matteo Kubernete % kubectl apply -f quotes-deployment.yml
deployment.apps/quote-deployment created
(base) matteo@Air-de-Matteo Kubernete % kubectl apply -f frontend-deployment.yml
deployment.apps/frontend-deployment created
(base) matteo@Air-de-Matteo Kubernete % kubectl apply -f frontend-service.yml
service/frontend-service created
(base) matteo@Air-de-Matteo Kubernete % kubectl apply -f api-service.yml
service/api-service created
(base) matteo@Air-de-Matteo Kubernete % kubectl apply -f quotes-service.yml
service/quote-service created
(base) matteo@Air-de-Matteo Kubernete %
```

Figure 1.3: Application de apply -f

```
(base) matteo@Air-de-Matteo Kubernete % kubectl get deployment
NAME                 READY   UP-TO-DATE   AVAILABLE   AGE
api-deployment       1/1     1            1           98s
frontend-deployment  1/1     1            1           108s
quote-deployment     0/1     1            0           3s
(base) matteo@Air-de-Matteo Kubernete % kubectl get pods
NAME                                 READY   STATUS    RESTARTS   AGE
api-deployment-8449bc7bbb-2j96r      1/1     Running   0          102s
frontend-deployment-97cf87589-879xl  1/1     Running   0          111s
quote-deployment-5b7f9c6bb9-rlgvx    1/1     Running   0          7s
(base) matteo@Air-de-Matteo Kubernete %
```

Figure 1.4: Affichage du bon fonctionnement des déploiements

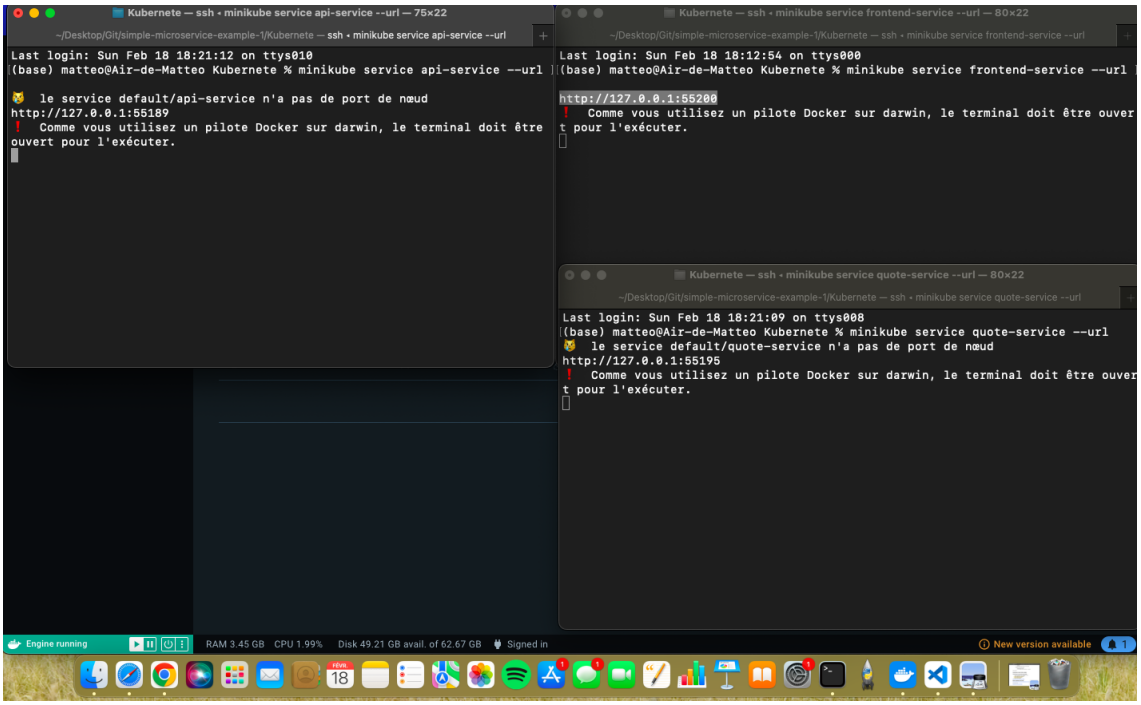

Figure 1.5: Application de minikube service myservice –url

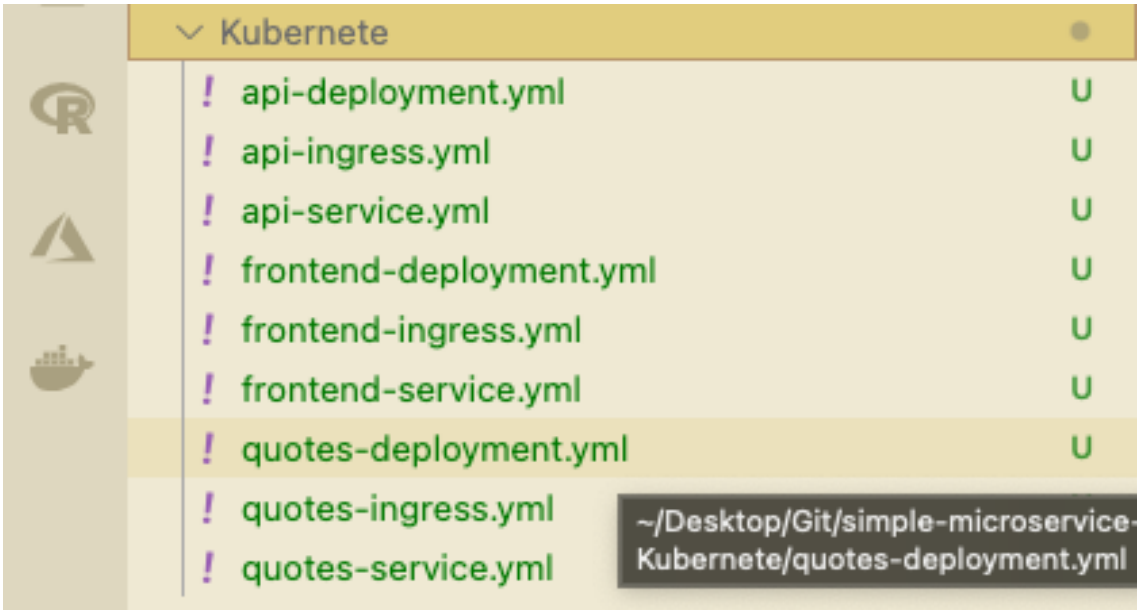

Figure 1.6: Création de 3 fichiers ingress

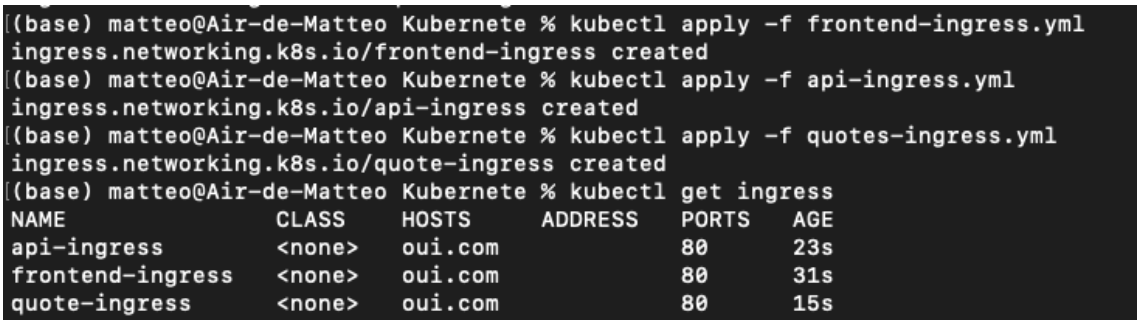

Figure 1.7: Application de apply et get ingress

1.3 | Conclusion :

J'ai réussi à créer des images Docker pour mes services, publié ces images sur Docker Hub, déployé des applications dans un cluster Kubernetes local avec Minikube, exposé des services à l'extérieur du cluster, et ajouté des règles Ingress pour gérer le trafic (je n'ai pas réussi à aller jusqu'au bout de ingress).