



# Projet

*Rapport*

**Maddalena Matteo**

Supervisé par  
M. B. Charroux

UFR des Sciences fondamentales et biomédicales  
Université Paris Cité

Février, 2024

---

## Contents

<b>1</b>		<b>1</b>
1.1	Description . . . . .	1
1.1.1	Création des Images Docker . . . . .	1
1.1.2	Construction et Publication des Images Docker . . . . .	1
1.1.3	Création des Déploiements Kubernetes . . . . .	1
1.1.4	Création des Services Kubernetes . . . . .	1
1.1.5	Configuration d'une Gateway avec Docker . . . . .	2
1.1.6	Publication des Services avec Minikube . . . . .	2
1.2	Captures d'écran : . . . . .	2
1.3	Conclusion : . . . . .	6

---

## List of Figures

1.1	Création des images et push grâce à docker . . . . .	2
1.2	Création de 6 fichiers pour le déploiement et les services . . . . .	3
1.3	Application de apply -f . . . . .	3
1.4	Affichage du bon fonctionnement des déploiements . . . . .	3
1.5	Application de minikube service myservice –url . . . . .	4
1.6	Création de 3 fichiers ingress . . . . .	4
1.7	Application de apply et get ingress . . . . .	5

## 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" a 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 :

```
(base) matteo@Air-de-Matteo simple-microservice-example-1 % docker tag 774f644ebfb47093a5681dcaeb7ef9b11e1e3532947b565b28d8db17073812 matteomaddalena/simple-microservice-example-1-quotes:latest
(base) matteo@Air-de-Matteo simple-microservice-example-1 % docker tag e89ec5b24d86d75726b50cdf75c9b0b5181e1652d38b98593ad41104348a5b2 matteomaddalena/simple-microservice-example-1-frontend:latest
(base) matteo@Air-de-Matteo simple-microservice-example-1 % docker push matteomaddalena/simple-microservice-example-1-api:latest

The push refers to repository [docker.io/matteomaddalena/simple-microservice-example-1-api]
0d599e45dccc: Mounted from matteomaddalena/simple-microservice-api
46babc6598ab: Mounted from matteomaddalena/simple-microservice-api
bb411d7dd5d4: Mounted from matteomaddalena/simple-microservice-api
5f70bf18a086: Mounted from matteomaddalena/app.dockerfile
02687a158043: Mounted from matteomaddalena/simple-microservice-api
3ab01e8988bf: Mounted from matteomaddalena/simple-microservice-api
c98dc9a94132: Mounted from matteomaddalena/simple-microservice-api
3ffdb7e28503: Mounted from matteomaddalena/simple-microservice-api
33ad93485756: Mounted from matteomaddalena/simple-microservice-api
00707a1c12b77: Mounted from matteomaddalena/simple-microservice-api
052174538f63: Mounted from matteomaddalena/simple-microservice-api
8abfe7e7c816: Mounted from matteomaddalena/simple-microservice-api
c8b886062a47: Mounted from matteomaddalena/simple-microservice-api
16fc2e3ca032: Mounted from matteomaddalena/simple-microservice-api
latest: digest: sha256:1643beda8863ab6d3227b2110804d47d110b4b9ce70e738138d4cd1442a7eaaa size: 3254
(base) matteo@Air-de-Matteo simple-microservice-example-1 % docker push matteomaddalena/simple-microservice-example-1-quotes:latest

The push refers to repository [docker.io/matteomaddalena/simple-microservice-example-1-quotes]
ec84837b6f60: Mounted from matteomaddalena/simple-microservice-quote
62be28d78dc5: Mounted from matteomaddalena/simple-microservice-quote
5f70bf18a086: Mounted from matteomaddalena/simple-microservice-example-1-api
a993da1cd329: Mounted from matteomaddalena/simple-microservice-quote
3dd1a7b3caf3: Mounted from matteomaddalena/simple-microservice-quote
349b0b22a493: Mounted from matteomaddalena/simple-microservice-quote
a07a24a37470: Mounted from matteomaddalena/simple-microservice-quote
84f540ade319: Mounted from matteomaddalena/simple-microservice-quote
9fe4e8a1862c: Mounted from matteomaddalena/simple-microservice-quote
909275a3eaa5: Mounted from matteomaddalena/simple-microservice-quote
f3f47b3309ca: Mounted from matteomaddalena/simple-microservice-quote
1a5fc1184c48: Mounted from matteomaddalena/simple-microservice-quote
latest: digest: sha256:9f909385f13c5785a27195177c2f01fd3d6a9f1009b63fe3bca6d734c64774c78 size: 2839
(base) matteo@Air-de-Matteo simple-microservice-example-1 % docker push matteomaddalena/simple-microservice-example-1-frontend:latest

The push refers to repository [docker.io/matteomaddalena/simple-microservice-example-1-frontend]
c02e81f4e570: Mounted from matteomaddalena/simple-microservice-front-end-app
5c0b7e307136: Mounted from matteomaddalena/simple-microservice-front-end-app
d9344a90d635: Mounted from matteomaddalena/simple-microservice-front-end-app
db9deeda8ac9: Mounted from matteomaddalena/simple-microservice-front-end-app
5d6cbe0dbcf9: Mounted from matteomaddalena/simple-microservice-front-end-app
latest: digest: sha256:74f58752be0bf1b2910858a98f144784fb86cdfa899dbc1247ff76fc5f8c2fe size: 1364
(base) matteo@Air-de-Matteo simple-microservice-example-1 %
```

Figure 1.1: Création des images et push grâce à docker

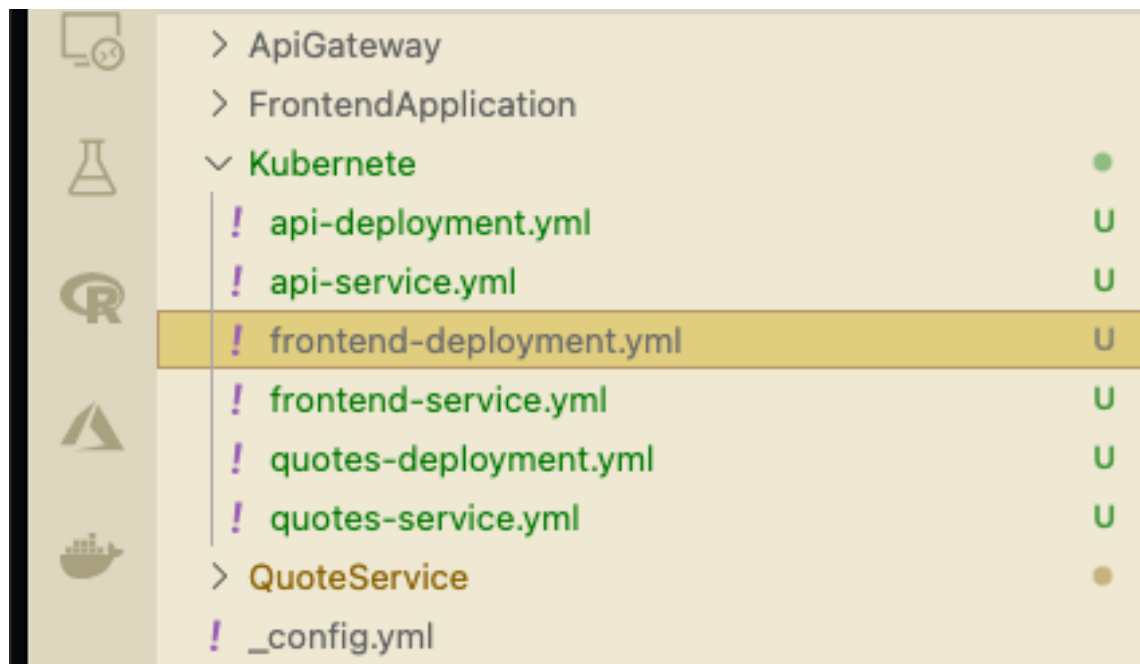


Figure 1.2: Création de 6 fichiers pour le déploiement et les services

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

Figure 1.3: Application de apply -f

```
(base) matteo@Air-de-Matteo Kubernetes % 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 Kubernetes % kubectl get pods
NAME                READY   STATUS    RESTARTS   AGE
api-deployment-8449bc7bbb-2j96r  1/1     Running   0           102s
frontend-deployment-97cf87589-879x1  1/1     Running   0           111s
quote-deployment-5b7f9c6bb9-rlgvx  1/1     Running   0            7s
(base) matteo@Air-de-Matteo Kubernetes %
```

Figure 1.4: Affichage du bon fonctionnement des déploiements

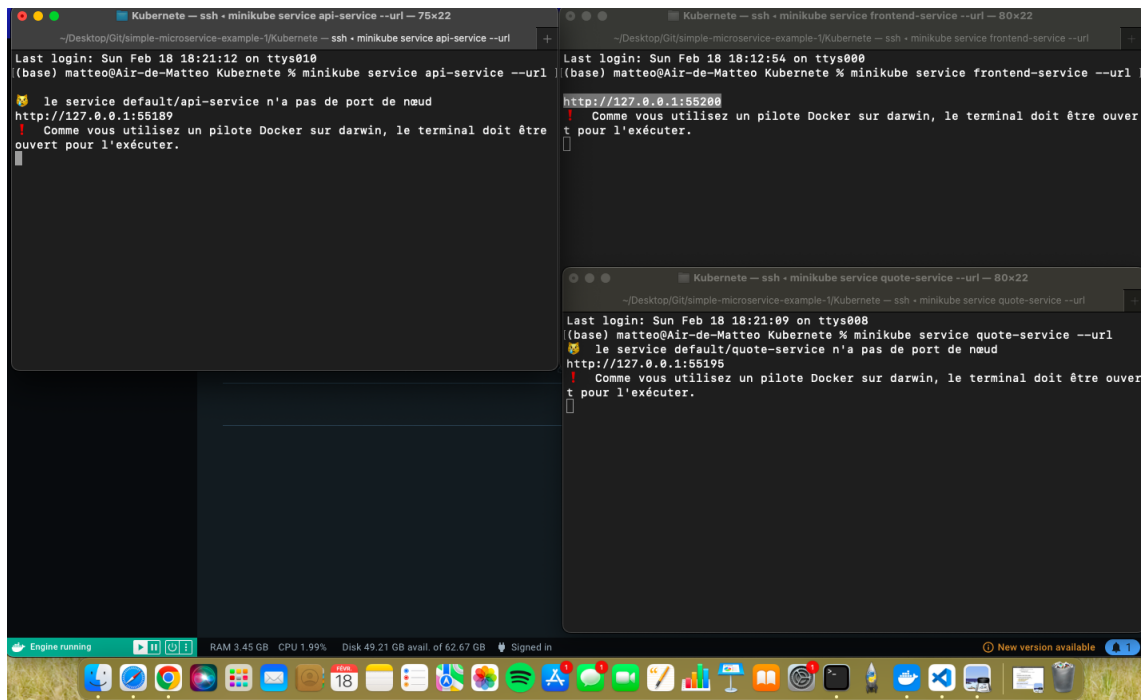


Figure 1.5: Application de minikube service myservice --url

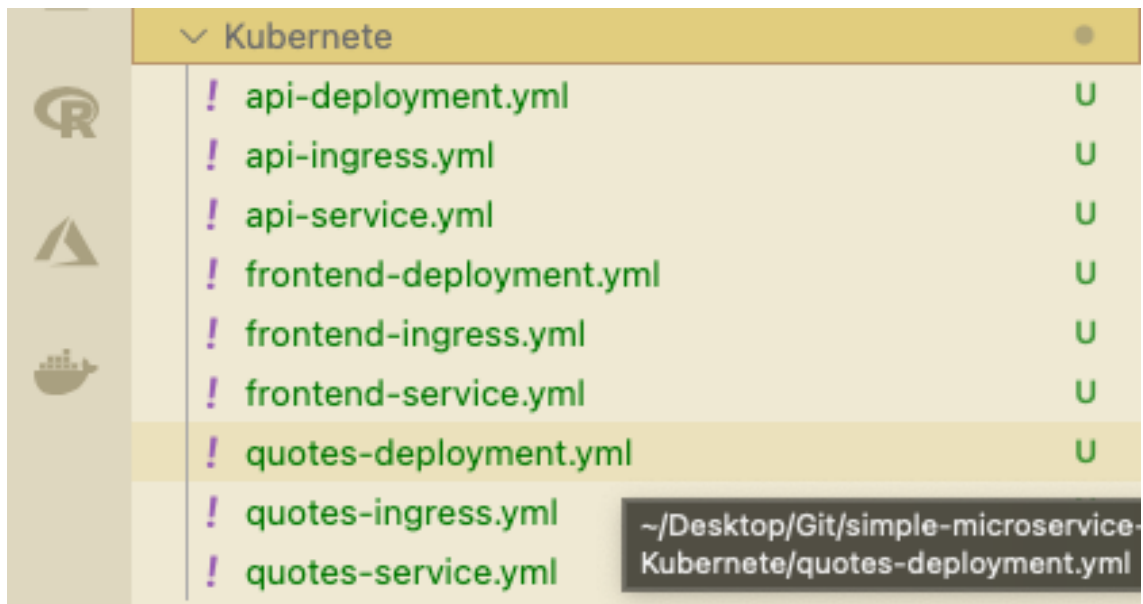


Figure 1.6: Création de 3 fichiers ingress

```
((base) matteo@Air-de-Matteo Kubernete % kubectl apply -f frontend-ingress.yml
ingress.networking.k8s.io/frontend-ingress created
((base) matteo@Air-de-Matteo Kubernete % kubectl apply -f api-ingress.yml
ingress.networking.k8s.io/api-ingress created
((base) matteo@Air-de-Matteo Kubernete % kubectl apply -f quotes-ingress.yml
ingress.networking.k8s.io/quote-ingress created
((base) matteo@Air-de-Matteo Kubernete % kubectl get ingress
NAME                CLASS    HOSTS    ADDRESS    PORTS    AGE
api-ingress          <none>   oui.com  80         23s
frontend-ingress     <none>   oui.com  80         31s
quote-ingress        <none>   oui.com  80         15s
```

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).