Projet Kubernetes

Kabli Mehdi

5SRC2

Construction de l’image Docker

Orchestration avec Docker Compose

Dans notre fichier DockerFile nous commençons par mettre nos instructions de construction après avoir mis notre repo sur notre machine

```
  GNU nano 7.2                                    Dockerfile *
# Utilisation d'une image de base officielle pour OpenJDK 17
FROM amazoncorretto:17-alpine

# Définir le répertoire de travail
WORKDIR /app

# Copier le fichier JAR dans le conteneur
COPY target/paymybuddy.jar /app/paymybuddy.jar

# Exposer le port 8080
EXPOSE 8080

# Commande pour démarrer l'application Spring Boot
CMD ["java", "-jar", "paymybuddy.jar"]
```

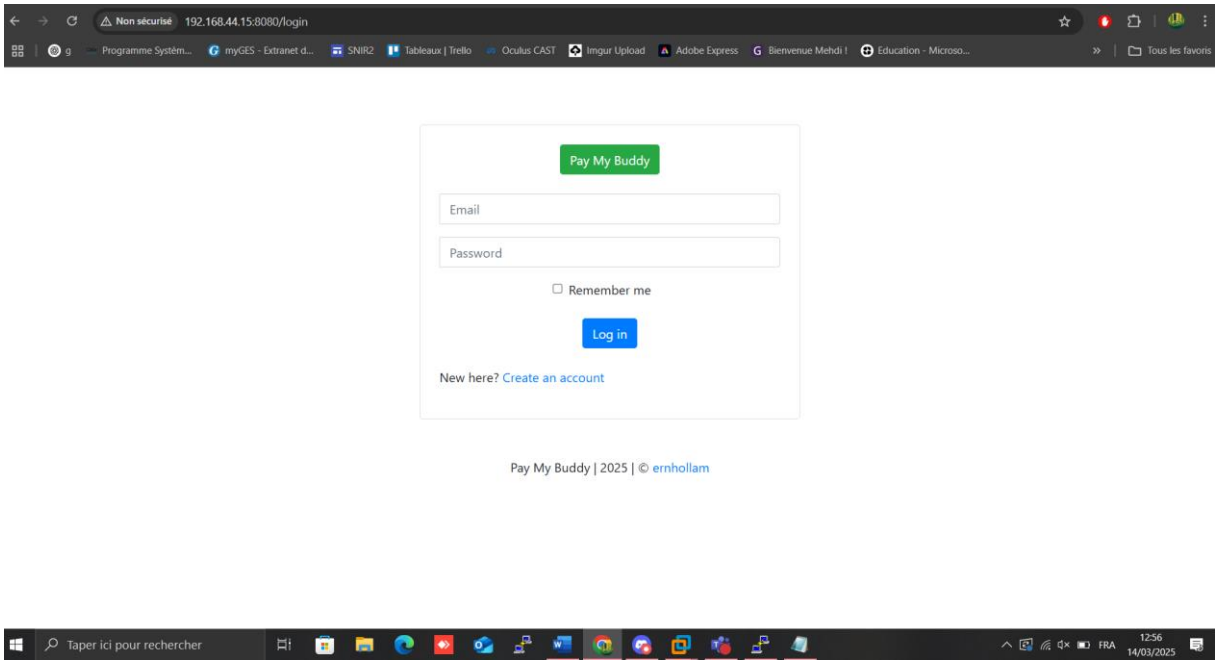
Maintenant nous définisson les services qui son nécessaire a l'application « Paymybuddy » dans le fichier docker-compose.yml

```yaml
version: '3.7'

services:
  # Service pour le backend
  paymybuddy-backend:
    build:
      context: .
      dockerfile: Dockerfile  # Utilise le Dockerfile dans le même dossier
    environment:
      SPRING_DATASOURCE_URL: jdbc:mysql://paymybuddy-db:3306/db_paymybuddy
      SPRING_DATASOURCE_USERNAME: root
      SPRING_DATASOURCE_PASSWORD: password
    ports:
      - "8080:8080"
    depends_on:
      - paymybuddy-db
    networks:
      - my_network

  # Service pour la base de données MySQL
  paymybuddy-db:
    image: mysql:5.7  # Utilise l'image officielle MySQL 5.7
    environment:
      MYSQL_ROOT_PASSWORD: password
      MYSQL_DATABASE: db_paymybuddy
    volumes:
      - mysql-data:/var/lib/mysql  # Volume pour persister les données de la DB
    ports:
      - "3306:3306"
    networks:
      - my_network

# Volumes persistants pour la base de données
volumes:
  mysql-data:

# Réseau personnalisé pour connecter les services entre eux
networks:
  my_network:
    driver: bridge
```

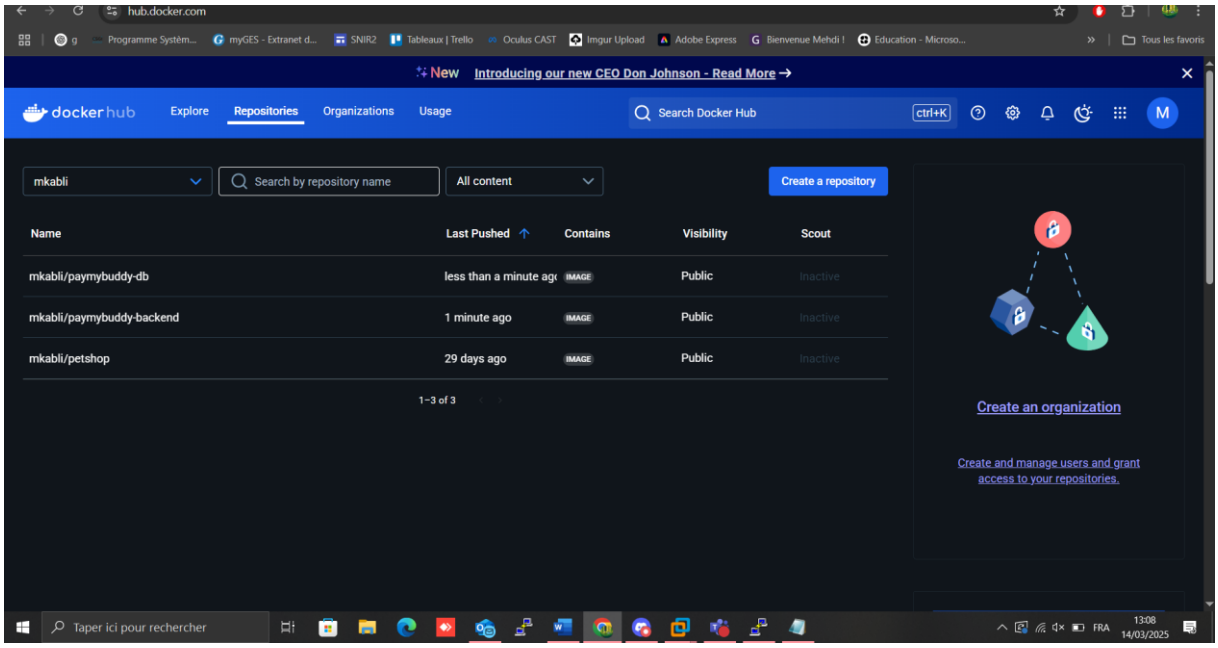
Ensuite nous lançons nos conteneurs avec docker-compose

```
root@ubuntuserv:/etc/TPkubernetes/projet/projet-esgi# docker ps
CONTAINER ID   IMAGE                              COMMAND                  CREATED
d9b209985451   projet-esgi_paymybuddy-backend   "java -jar paymybudd…"   15 minutes ago
end_1
c71e9dd8b86d   mysql:5.7                          "docker-entrypoint.s…"   15 minutes ago
```

Nous pouvons maintenant accéder à notre application :



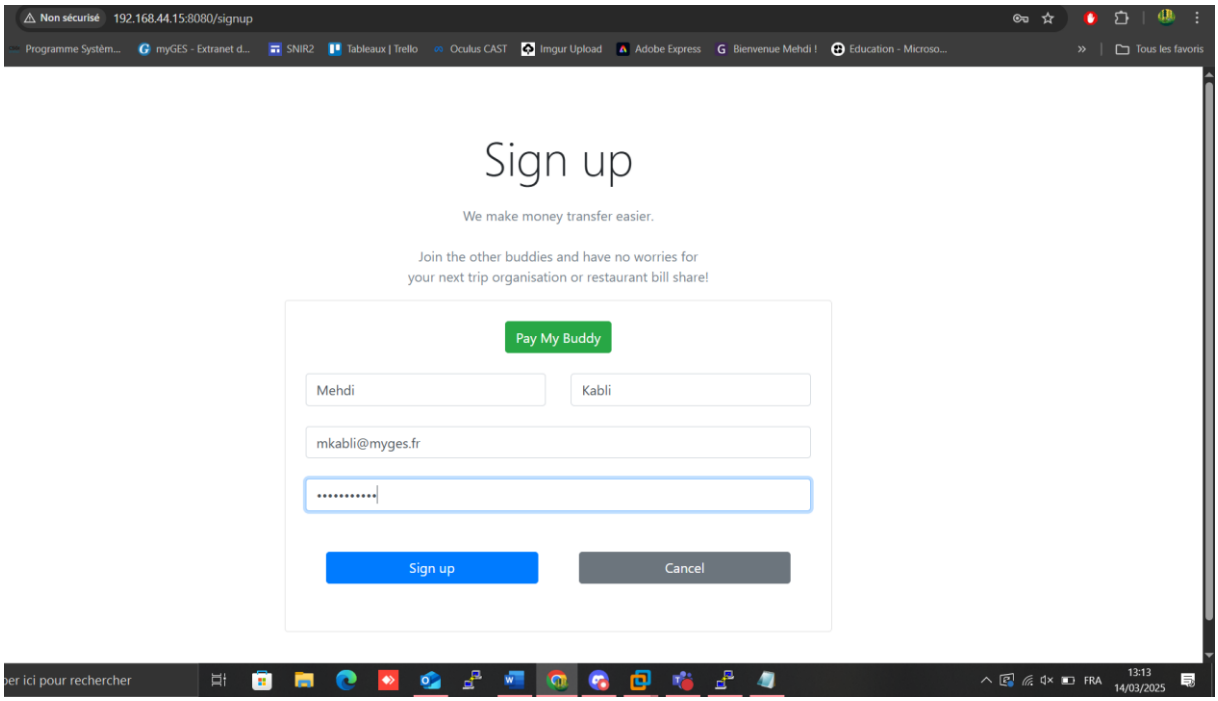
Ensuite nous devons tagger les images Docker pour qu'elles correspondent au format requis pour Docker Hub



On peu voir que les images son bien pousser sur mon compte dockerhub

Nous testons maintenant l’accès à l’application

Pour cela nous devons crée notre premier compte



Orchestration Kubernetes

Dans un premier temps nous créons notre namespace mybuddy

```
mkabli@ubuntuserv:/etc/TPkubernetes/projet/projet-esgi/k8s$ ls
namespace-paymybuddy.yaml
```

```
  GNU nano 7.2                                  namespace-paymybuddy.yaml
apiVersion: v1
kind: Namespace
metadata:
  name: paymybuddy
```

Nous démarrons ensuite notre minikube

```
mkabli@ubuntuserv:/etc/TPkubernetes/projet/projet-esgi/k8s$ minikube status
minikube
type: Control Plane
host: Running
kubelet: Running
apiserver: Running
kubeconfig: Configured
```

Ensuite nous configurer notre fichier backend

```
  GNU nano 7.2                                                backend-deployment.yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: paymybuddy-backend
  namespace: paymybuddy
spec:
  replicas: 1
  selector:
    matchLabels:
      app: paymybuddy-backend
  template:
    metadata:
      labels:
        app: paymybuddy-backend
    spec:
      containers:
      - name: backend
        image: paymybuddy-backend:latest  # Remplace par ton image Docker si nécessaire
        imagePullPolicy: Never
        env:
        - name: SPRING_DATASOURCE_URL
          value: "jdbc:mysql://paymybuddy-db:3306/paymybuddy?useSSL=false"
        - name: SPRING_DATASOURCE_USERNAME
          valueFrom:
            configMapKeyRef:
              name: paymybuddy-config
              key: SPRING_DATASOURCE_USERNAME
        - name: SPRING_DATASOURCE_PASSWORD
          valueFrom:
            secretKeyRef:
              name: paymybuddy-secret
              key: SPRING_DATASOURCE_PASSWORD
        ports:
        - containerPort: 8080
---
apiVersion: v1
kind: Service
metadata:
  name: paymybuddy-backend
  namespace: paymybuddy
spec:
  selector:
    app: paymybuddy-backend
  ports:
    - protocol: TCP
      port: 80
```

Ce fichier est un **manifest Kubernetes** qui définit comment l'application **backend** (dans ce cas **PayMyBuddy**) doit être déployée sur ton cluster Kubernetes.

Nous configurons ensuite notre config map qui auras pour but de **stocker et gérer** des **configurations** sous forme de paires **clé-valeur**, sans avoir à modifier l'image Docker ou le code de l'application.

```
  GNU nano 7.2                                                  configmap-secrets.yaml
apiVersion: v1
kind: ConfigMap
metadata:
  name: paymybuddy-config
  namespace: paymybuddy
data:
  SPRING_DATASOURCE_URL: "jdbc:mysql://paymybuddy-db:3306/paymybuddy"
  SPRING_DATASOURCE_USERNAME: "paymybuddy"

---
apiVersion: v1
kind: Secret
metadata:
  name: paymybuddy-secret
  namespace: paymybuddy
type: Opaque
data:
  SPRING_DATASOURCE_PASSWORD: "cGFzc3dvcmQ="  # Base64 de "password"
```

Nous configurons maintenant notre fichier mysql deployment qui permet de **déployer MySQL dans Kubernetes**

```
  GNU nano 7.2                                                  mysql-deployment.yaml
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: mysql-pvc
  namespace: paymybuddy
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 1Gi

---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: paymybuddy-db
  namespace: paymybuddy
spec:
  selector:
    matchLabels:
      app: paymybuddy-db
  template:
    metadata:
      labels:
        app: paymybuddy-db
    spec:
      containers:
      - name: mysql
        image: mysql:5.7
        env:
        - name: MYSQL_ROOT_PASSWORD
          valueFrom:
            secretKeyRef:
              name: paymybuddy-secret
              key: SPRING_DATASOURCE_PASSWORD
        - name: MYSQL_DATABASE
          value: "paymybuddy"
        - name: MYSQL_USER
          valueFrom:
            configMapKeyRef:
              name: paymybuddy-config
              key: SPRING_DATASOURCE_USERNAME
        - name: MYSQL_PASSWORD
          valueFrom:
            secretKeyRef:
                                                                [ Read 71 lines ]
```