Voici les Dockerfiles à créer pour mettre en place l'infrastructure Docker décrite dans le projet :

Dockerfile pour le loadbalancer

FROM haproxy:latest

# Install SSL/TLS certificates

RUN apt-get update && apt-get install -y certbot

# Generate a new SSL/TLS certificate

RUN certbot certonly --standalone --agree-tos --no-eff-email --domain www.example.com

# Configure HaProxy

ADD haproxy.cfg /etc/haproxy/haproxy.cfg

# Start HaProxy

CMD haproxy -f /etc/haproxy/haproxy.cfg

Dockerfile pour le serveur web1

FROM nginx

# Copy application files

COPY . /usr/share/nginx/html

# Start nginx

CMD nginx -g "daemon off;"

Dockerfile pour le serveur web2

FROM nginx

# Copy application files

COPY . /usr/share/nginx/html

# Start nginx

CMD nginx -g "daemon off;"

Dockerfile pour le serveur de base de données

FROM mysql:latest

# Configure database

ENV MYSQL\_DATABASE my\_database

ENV MYSQL\_USER my\_user

ENV MYSQL\_PASSWORD my\_password

# Start MySQL

CMD mysqld --user=root --skip-networking

Explications

* Le Dockerfile pour le loadbalancer installe HaProxy et génère un nouveau certificat SSL/TLS. Il configure également HaProxy pour écouter sur les ports 80 et 443.
* Le Dockerfile pour le serveur web1 copie les fichiers de l'application dans le conteneur nginx. Il démarre ensuite nginx en mode daemon.
* Le Dockerfile pour le serveur web2 est créé de la même manière que le Dockerfile pour le serveur web1.
* Le Dockerfile pour le serveur de base de données configure la base de données avec les paramètres suivants :
  + Nom de la base de données : my\_database
  + Utilisateur : my\_user
  + Mot de passe : my\_password
* Le serveur de base de données démarre MySQL en mode non connecté. Cela permet de ne pas exposer la base de données à Internet.

Comment créer les conteneurs

Une fois les Dockerfiles créés, vous pouvez créer les conteneurs en exécutant les commandes suivantes :

docker build -t loadbalancer .

docker run -d -p 80:80 -p 443:443 --name loadbalancer loadbalancer

docker build -t web1 .

docker run -d -p 80:80 --name web1 web1

docker build -t web2 .

docker run -d -p 80:80 --name web2 web2

docker build -t mysql .

docker run -d --name mysql mysql

Comment accéder aux applications

Une fois les conteneurs créés, vous pouvez accéder aux applications en accédant aux adresses suivantes :

* Adresse du loadbalancer : http://localhost
* Adresse du serveur web1 : http://localhost:80/
* Adresse du serveur web2 : http://localhost:80/

Comment mettre à jour les conteneurs

Pour mettre à jour les conteneurs, vous pouvez simplement modifier les Dockerfiles et exécuter les commandes de création de conteneurs. Les conteneurs existants seront remplacés par les nouveaux conteneurs.

Comment automatiser la création des conteneurs

Vous pouvez automatiser la création des conteneurs en créant un script qui exécute les commandes suivantes :

docker build -t loadbalancer .

docker run -d -p 80:80 -p 443:443 --name loadbalancer loadbalancer

docker build -t web1 .

docker run -d -p 80:80 --name web1 web1

docker build -t web2 .

docker run -d -p 80:80 --name web2 web2

docker build -t mysql .

docker run -d --name mysql mysql

Ce script peut être exécuté à chaque fois que vous souhaitez mettre à jour les applications.