ECOLE SUPERIEURE POLYTECHNIQUE

DEPARTEMENT GENIE INFORMATIQUE

Conteneurisation avec Docker

TITRE DU PROJET :

Lab_10 : Déploiement d'applications full-stack et orchestration

PRESENTÉ PAR :

NOM	PRÉNOM	CLASSE
GUEYE	Yacine	M1 - GLSI

1. Introduction à Docker et aux conteneurs

1. . QuŠest-ce quŠun conteneur ?

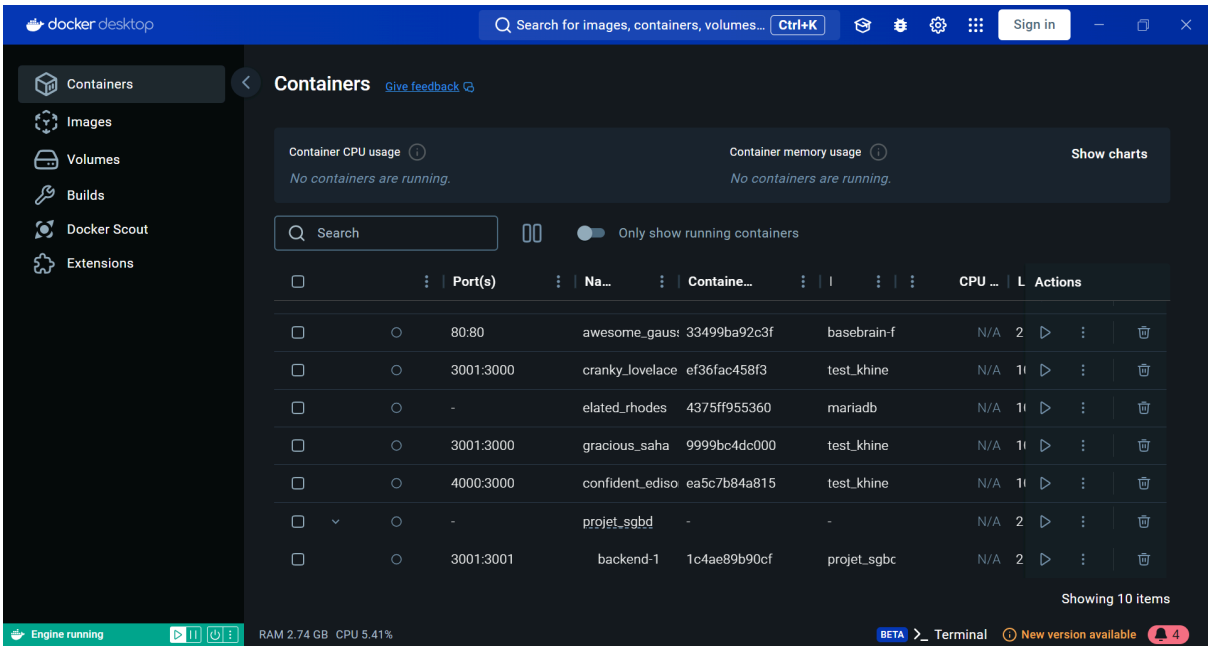
1.2 LŠarchitecture de Docker

1.3 Concepts clés de Docker

2. Mise en place de l'environnement de travail

2.1. Installation de Docker

Pour suivre ce cours-lab, nous aurons besoin d'installer Docker sur notre système. Puisque nous avons **Windows** comme système d'exploitation, nous allons utiliser docker desktop



2.2 Clonage des dépôts de code pour le TP

Pour réaliser les exercices pratiques, nous allons utiliser une application full-stack composée d'un frontend en React, d'un backend en Spring Boot et d'une base de données MariaDB

Nous allons procéder au clonage des dépôts de code source :

Ouvrons un terminal et exécutonsles commandes suivantes :

 # Création d'un dossier pour le projet

mkdir -p docker-fullstack-app

cd docker-fullstack-app

```
C:\Users\USER\Documents\M1-GLSI\Semestre 2\IPDL>mkdir -p docker-fullstack-app

C:\Users\USER\Documents\M1-GLSI\Semestre 2\IPDL>cd docker-fullstack-app

C:\Users\USER\Documents\M1-GLSI\Semestre 2\IPDL\docker-fullstack-app>
```

 # Clonage du backend

git clone https://github.com/elbachir67/tp-agl-backend-code.git backend

```
C:\Users\USER\Documents\M1-GLSI\Semestre 2\IPDL\docker-fullstack-app>git clone https://github.com/elbachir67/tp-agl-back
end-code.git backend
Cloning into 'backend'...
remote: Enumerating objects: 87, done.
remote: Total 87 (delta 0), reused 0 (delta 0), pack-reused 87 (from 1)
Receiving objects: 100% (87/87), 44.70 MiB | 2.18 MiB/s, done.
Resolving deltas: 100% (10/10), done.

C:\Users\USER\Documents\M1-GLSI\Semestre 2\IPDL\docker-fullstack-app>
```

 # Clonage du frontend

git clone https://github.com/elbachir67/tp-agl-frontend-code.git frontend

```
C:\Users\USER\Documents\M1-GLSI\Semestre 2\IPDL\docker-fullstack-app>git clone https://github.com/elbachir67/tp-agl-fron
tend-code.git  frontend
Cloning into 'frontend'...
remote: Enumerating objects: 29, done.
remote: Counting objects: 100% (29/29), done.
remote: Compressing objects: 100% (28/28), done.
remote: Total 29 (delta 1), reused 29 (delta 1), pack-reused 0 (from 0)
Receiving objects: 100% (29/29), 190.45 KiB | 815.00 KiB/s, done.
Resolving deltas: 100% (1/1), done.

C:\Users\USER\Documents\M1-GLSI\Semestre 2\IPDL\docker-fullstack-app>
```

Nous pouvons voir nos deux dossiers contenant le frontend et le backend

```
C:\Users\USER\Documents\M1-GLSI\Semestre 2\IPDL\docker-fullstack-app>dir
 Le volume dans le lecteur C n'a pas de nom.
 Le numéro de série du volume est 0C19-D8C0

 Répertoire de C:\Users\USER\Documents\M1-GLSI\Semestre 2\IPDL\docker-fullstack-app

05/05/2025  09:18    <DIR>          .
05/05/2025  09:18    <DIR>          ..
05/05/2025  09:16    <DIR>          backend
05/05/2025  09:18    <DIR>          frontend
               0 fichier(s)              0 octets
               4 Rép(s)  590 624 669 696 octets libres

C:\Users\USER\Documents\M1-GLSI\Semestre 2\IPDL\docker-fullstack-app>
```

3. Les commandes Docker essentielles

Avant de conteneuriser notre application complète, familiarisons-nous avec les commandes Docker fondamentales.

3.1 Gestion des images Docker

 # Télécharger une image depuis Docker Hub

docker pull nginx:latest

```
C:\Users\USER\Documents\M1-GLSI\Semestre 2\IPDL\docker-fullstack-app>docker pull nginx:latest
latest: Pulling from library/nginx
254e724d7786: Pull complete
913115292750: Pull complete
3e544d53ce49: Pull complete
4f21ed9ac0c0: Pull complete
d38f2ef2d6f2: Pull complete
40a6e9f4e456: Pull complete
d3dc5ec71e9d: Pull complete
Digest: sha256:c15da6c91de8d2f436196f3a768483ad32c258ed4e1beb3d367a27ed67253e66
Status: Downloaded newer image for nginx:latest
docker.io/library/nginx:latest

What's next:
    View a summary of image vulnerabilities and recommendations → docker scout quickview nginx:latest

C:\Users\USER\Documents\M1-GLSI\Semestre 2\IPDL\docker-fullstack-app>
```

 # Lister les images disponibles localement

docker images

```
C:\Users\USER\Documents\M1-GLSI\Semestre 2\IPDL\docker-fullstack-app>docker images
REPOSITORY                TAG       IMAGE ID       CREATED         SIZE
nginx                     latest    a830707172e8   2 weeks ago     192MB
projet_sgbd-backend       latest    d1c908567860   6 weeks ago     1.16GB
projet_sgbd-frontend      latest    bd8cdbe19bb6   6 weeks ago     55.4MB
<none>                    <none>    be4a27d8b9ed   6 weeks ago     1.21GB
basebrain-frontend        latest    4bf73a04e684   6 weeks ago     53.8MB
basebrain-backend         latest    19ad911c2e78   6 weeks ago     1.21GB
minio/minio               latest    2eaf94c71682   7 weeks ago     182MB
test_khine                latest    5644dc3d3657   9 months ago    2.2GB
yacineg/mon_image_docker  latest    5644dc3d3657   9 months ago    2.2GB
mariadb                   latest    4486d64c9c3b   10 months ago   406MB
```

 # Construire une image à partir d'un Dockerfile

docker build -t mon-app:1.0 .

```
C:\Users\USER\Documents\M1-GLSI\Semestre 2\IPDL\docker-fullstack-app>docker build -t mon-app:1.0 .
[+] Building 0.1s (1/1) FINISHED                                                                docker:desktop-linux
 => [internal] load build definition from Dockerfile                                                          0.0s
 => => transferring dockerfile: 2B                                                                            0.0s
ERROR: failed to solve: failed to read dockerfile: open Dockerfile: no such file or directory

View build details: docker-desktop://dashboard/build/desktop-linux/desktop-linux/t0nsju1waxd8a41fn2vpuuffw
```

Normal puisque dans ce dossier nous n'avons pas de Dockerfile

 # Supprimer une image

docker rmi nginx:latest

```
C:\Users\USER\Documents\M1-GLSI\Semestre 2\IPDL\docker-fullstack-app>docker rmi nginx:latest
Untagged: nginx:latest
Untagged: nginx@sha256:c15da6c91de8d2f436196f3a768483ad32c258ed4e1beb3d367a27ed67253e66
Deleted: sha256:a830707172e8069c09cf6c67a04e23e5a1a332d70a90a54999b76273a928b9ce
Deleted: sha256:f33f068b16dcd524bfab3d9c0fe569c586b498f882b3646575f8ca9dad28183b
Deleted: sha256:2ca2fe99d37993d2cd67b9459690ca6204d4d52cb4ed785e5db36f7a0dd962bd
Deleted: sha256:9219473716627829567af49238831efb419c04a3b0fb551e8b2403b01acc32c7
Deleted: sha256:f5fb61f1d156dfaab34b082893de038561147bda56dce0665a09aa45b8f0ebd0
Deleted: sha256:da1ec5e247ce7e2343b89c04cc7378a48a108ce052887856c8f768f94d41fcbe
Deleted: sha256:04179f3ea760289be096f97bd5c92241bf7322df00e17161655e83eca3b53859
Deleted: sha256:6c4c763d22d0c5f9b2c5901dfa667fbbc4713cee6869336b8fd5022185071f1c
```

 # Trouver des images sur Docker Hub

docker search ubuntu

```
C:\Users\USER\Documents\M1-GLSI\Semestre 2\IPDL\docker-fullstack-app>docker search ubuntu
NAME                                DESCRIPTION                                     STARS     OFFICIAL
ubuntu                              Ubuntu is a Debian-based Linux operating sys…   17564     [OK]
ubuntu/squid                        Squid is a caching proxy for the Web. Long-t…   113
ubuntu/nginx                        Nginx, a high-performance reverse proxy & we…   129
ubuntu/cortex                       Cortex provides storage for Prometheus. Long…   4
ubuntu/kafka                        Apache Kafka, a distributed event streaming …   53
ubuntu/prometheus                   Prometheus is a systems and service monitori…   71
ubuntu/zookeeper                    ZooKeeper maintains configuration informatio…   13
ubuntu/bind9                        BIND 9 is a very flexible, full-featured DNS…   104
ubuntu/apache2                      Apache, a secure & extensible open-source HT…   90
ubuntu/postgres                     PostgreSQL is an open source object-relation…   41
ubuntu/mysql                        MySQL open source fast, stable, multi-thread…   67
ubuntu/redis                        Redis, an open source key-value store. Long-…   23
ubuntu/dotnet-aspnet                Chiselled Ubuntu runtime image for ASP.NET a…   26
ubuntu/jre                          Distroless Java runtime based on Ubuntu. Lon…   20
ubuntu/grafana                      Grafana, a feature rich metrics dashboard & …   12
ubuntu/python                       A chiselled Ubuntu rock with the Python runt…   23
ubuntu/dotnet-deps                  Chiselled Ubuntu for self-contained .NET & A…   16
ubuntu/cassandra                    Cassandra, an open source NoSQL distributed …   2
ubuntu/memcached                    Memcached, in-memory keyvalue store for smal…   5
ubuntu/dotnet-runtime               Chiselled Ubuntu runtime image for .NET apps…   20
ubuntu/prometheus-alertmanager      Alertmanager handles client alerts from Prom…   10
ubuntu/mlflow                       MLFlow: for managing the machine learning li…   5
ubuntu/telegraf                     Telegraf collects, processes, aggregates & w…   4
ubuntu/loki                         Grafana Loki, a log aggregation system like …   2
ubuntu/chiselled-jre                [MOVED TO ubuntu/jre] Chiselled JRE: distrol…   3
```

3.2 Gestion des conteneurs

 # Créer et démarrer un conteneur

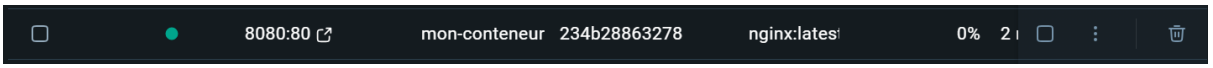
docker run --name mon-conteneur -p 8080:80 -d nginx:latest

```
C:\Users\USER\Documents\M1-GLSI\Semestre 2\IPDL\docker-fullstack-app>docker run --name mon-conteneur -p 8080:80 -d ngin
:latest
Unable to find image 'nginx:latest' locally
latest: Pulling from library/nginx
254e724d7786: Pull complete
913115292750: Pull complete
3e544d53ce49: Pull complete
4f21ed9ac0c0: Pull complete
d38f2ef2d6f2: Pull complete
40a6e9f4e456: Pull complete
d3dc5ec71e9d: Pull complete
Digest: sha256:c15da6c91de8d2f436196f3a768483ad32c258ed4e1beb3d367a27ed67253e66
Status: Downloaded newer image for nginx:latest
234b28863278d40d594c1bb62de1b10f6fe4ce371feaed0a348736bba2adc506

C:\Users\USER\Documents\M1-GLSI\Semestre 2\IPDL\docker-fullstack-app>
```

 # Lister les conteneurs en cours d'exécution

docker ps

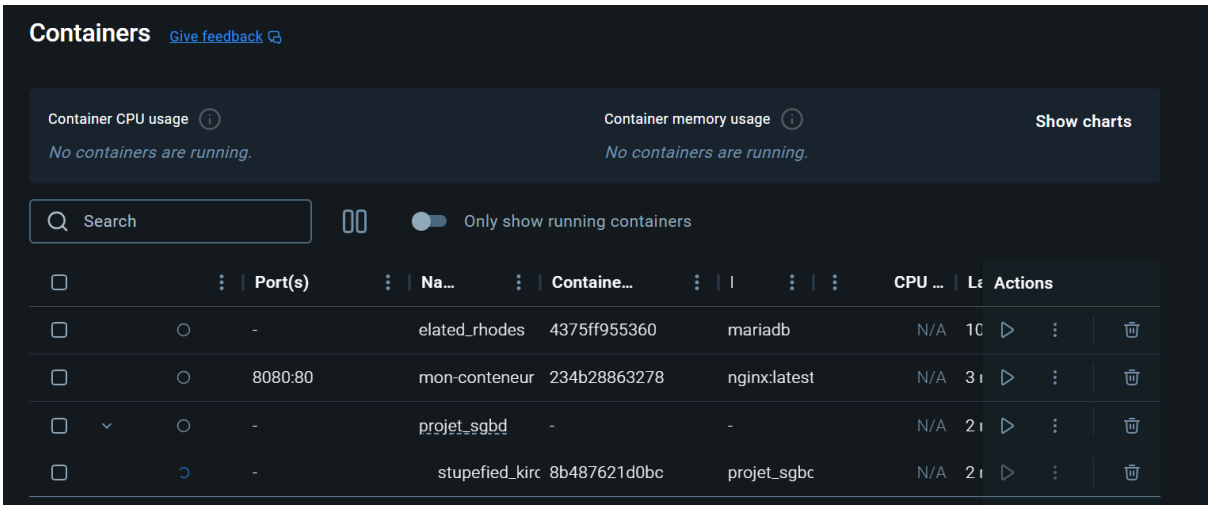
```
C:\Users\USER\Documents\M1-GLSI\Semestre 2\IPDL\docker-fullstack-app>docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED          STATUS          PORTS                  NAMES
234b28863278   nginx:latest   "/docker-entrypoint.…"   About a minute ago   Up About a minute   0.0.0.0:8080->80/tcp   mon-conteneur

C:\Users\USER\Documents\M1-GLSI\Semestre 2\IPDL\docker-fullstack-app>
```



 # Lister tous les conteneurs (y compris ceux arrêtés)

docker ps -a

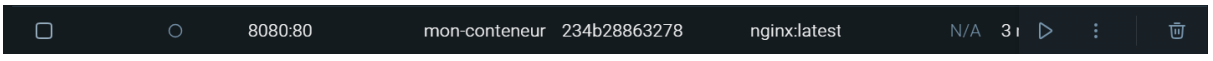
```
C:\Users\USER\Documents\M1-GLSI\Semestre 2\IPDL\docker-fullstack-app>docker ps -a
CONTAINER ID   IMAGE                          COMMAND                  CREATED          STATUS                       PORTS                  NAMES
234b28863278   nginx:latest                   "/docker-entrypoint.…"   2 minutes ago    Up 2 minutes                 0.0.0.0:8080->80/tcp   mon-conteneur
8b487621d0bc   projet_sgbd-frontend:latest    "/docker-entrypoint.…"   6 weeks ago      Exited (255) 22 minutes ago  80/tcp                 stupefied_kirch
4375ff955360   mariadb                        "docker-entrypoint.s…"   9 months ago     Exited (1) 9 months ago                             elated_rhodes

C:\Users\USER\Documents\M1-GLSI\Semestre 2\IPDL\docker-fullstack-app>
```



 # Arrêter un conteneur

docker stop mon-conteneur

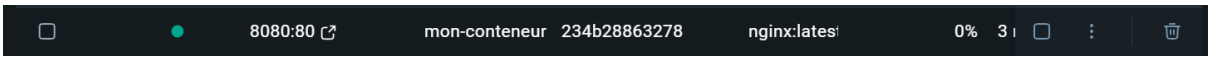
```
C:\Users\USER\Documents\M1-GLSI\Semestre 2\IPDL\docker-fullstack-app>docker stop mon-conteneur
mon-conteneur

C:\Users\USER\Documents\M1-GLSI\Semestre 2\IPDL\docker-fullstack-app>
```



 # Démarrer un conteneur existant

docker start mon-conteneur

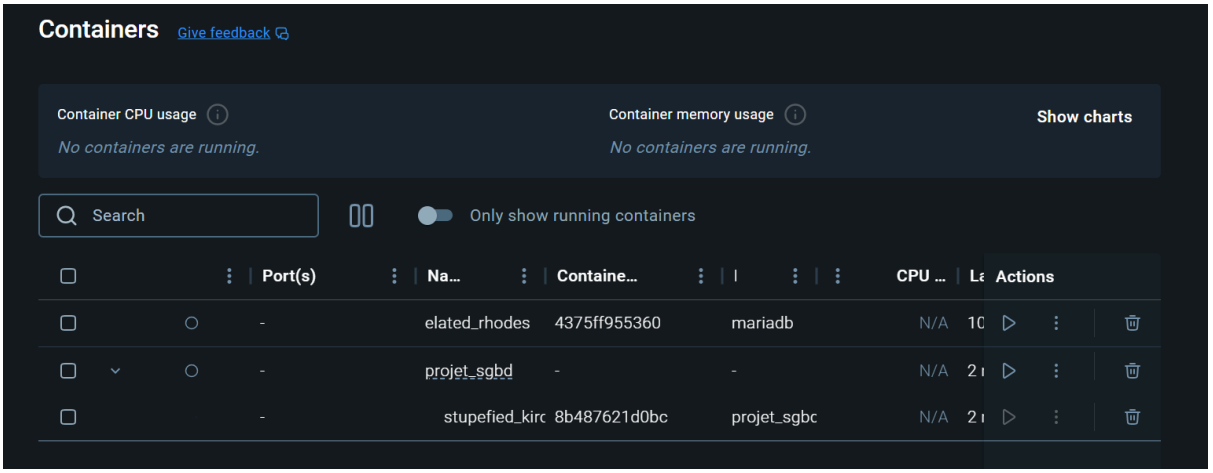
```
C:\Users\USER\Documents\M1-GLSI\Semestre 2\IPDL\docker-fullstack-app>docker start mon-conteneur
mon-conteneur

C:\Users\USER\Documents\M1-GLSI\Semestre 2\IPDL\docker-fullstack-app>
```



 # Supprimer un conteneur

docker rm mon-conteneur

```
C:\Users\USER\Documents\M1-GLSI\Semestre 2\IPDL\docker-fullstack-app>docker stop mon-conteneur
mon-conteneur

C:\Users\USER\Documents\M1-GLSI\Semestre 2\IPDL\docker-fullstack-app>docker rm mon-conteneur
mon-conteneur
```



 # Exécuter une commande dans un conteneur en cours d'exécution

docker exec -it mon-conteneur bash

```
C:\Users\USER\Documents\M1-GLSI\Semestre 2\IPDL\docker-fullstack-app>docker run --name mon-conteneur -p 8080:80 -d nginx:latest
7f17ff71303316da39be8eee26a3eaeddd9849746b6b8e7a2030f97f4fb6d350

C:\Users\USER\Documents\M1-GLSI\Semestre 2\IPDL\docker-fullstack-app>docker exec -it mon-conteneur bash
root@7f17ff713033:/#
```

 # Voir les logs d'un conteneur

docker logs mon-conteneur

```
C:\Users\USER\Documents\M1-GLSI\Semestre 2\IPDL\docker-fullstack-app>docker logs mon-conteneur
/docker-entrypoint.sh: /docker-entrypoint.d/ is not empty, will attempt to perform configuration
/docker-entrypoint.sh: Looking for shell scripts in /docker-entrypoint.d/
/docker-entrypoint.sh: Launching /docker-entrypoint.d/10-listen-on-ipv6-by-default.sh
10-listen-on-ipv6-by-default.sh: info: Getting the checksum of /etc/nginx/conf.d/default.conf
10-listen-on-ipv6-by-default.sh: info: Enabled listen on IPv6 in /etc/nginx/conf.d/default.conf
/docker-entrypoint.sh: Sourcing /docker-entrypoint.d/15-local-resolvers.envsh
/docker-entrypoint.sh: Launching /docker-entrypoint.d/20-envsubst-on-templates.sh
/docker-entrypoint.sh: Launching /docker-entrypoint.d/30-tune-worker-processes.sh
/docker-entrypoint.sh: Configuration complete; ready for start up
2025/05/05 09:39:22 [notice] 1#1: using the "epoll" event method
2025/05/05 09:39:22 [notice] 1#1: nginx/1.27.5
2025/05/05 09:39:22 [notice] 1#1: built by gcc 12.2.0 (Debian 12.2.0-14)
2025/05/05 09:39:22 [notice] 1#1: OS: Linux 5.15.167.4-microsoft-standard-WSL2
2025/05/05 09:39:22 [notice] 1#1: getrlimit(RLIMIT_NOFILE): 1048576:1048576
2025/05/05 09:39:22 [notice] 1#1: start worker processes
2025/05/05 09:39:22 [notice] 1#1: start worker process 29
2025/05/05 09:39:22 [notice] 1#1: start worker process 30
2025/05/05 09:39:22 [notice] 1#1: start worker process 31
2025/05/05 09:39:22 [notice] 1#1: start worker process 32
```

Dans Docker, la distinction entre démarrage et création est importante :

— `docker run` à la fois crée ET démarre un nouveau conteneur

— `docker start` redémarre un conteneur existant qui a été arrêté

Pensez à `docker run` comme à l'action de démarrer une voiture pour la première fois, tandis que `docker start` est comme redémarrer une voiture déjà utilisée.

3.3 Exercice pratique : Manipulation de base de Docker

Exercice 1 : Manipulations de base avec Docker

Suivons ces étapes pour nous familiariser avec les commandes Docker de base :

1. Téléchargeons l'image officielle de Nginx :

docker pull nginx:latest

```
C:\Users\USER\Documents\M1-GLSI\Semestre 2\IPDL\docker-fullstack-app>docker pull nginx:latest
latest: Pulling from library/nginx
Digest: sha256:c15da6c91de8d2f436196f3a768483ad32c258ed4e1beb3d367a27ed67253e66
Status: Image is up to date for nginx:latest
docker.io/library/nginx:latest

What's next:
    View a summary of image vulnerabilities and recommendations → docker scout quickview nginx:latest

C:\Users\USER\Documents\M1-GLSI\Semestre 2\IPDL\docker-fullstack-app>
```

2. Lançons un conteneur Nginx qui expose le port 80 sur le port 8080 de votre machine :

docker run --name test-nginx -p 8080:80 -d nginx

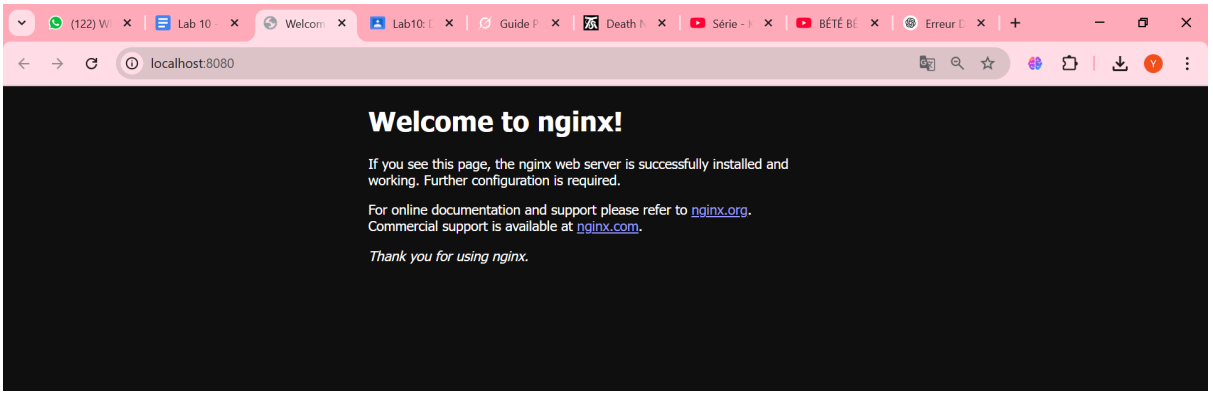
```
C:\Users\USER\Documents\M1-GLSI\Semestre 2\IPDL\docker-fullstack-app>docker run --name test-nginx -p 8080:80 -d nginx
56ccdf53acfba600f9e18d551ddf5869c9b11bff6e2aefbab4fff65f39dfe01b

C:\Users\USER\Documents\M1-GLSI\Semestre 2\IPDL\docker-fullstack-app>
```

3. Vérifions que le conteneur est en cours d'exécution :

docker ps

```
C:\Users\USER\Documents\M1-GLSI\Semestre 2\IPDL\docker-fullstack-app>docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED          STATUS          PORTS                  NAMES
56ccdf53acfb   nginx     "/docker-entrypoint.…"   26 seconds ago   Up 25 seconds   0.0.0.0:8080->80/tcp   test-nginx

C:\Users\USER\Documents\M1-GLSI\Semestre 2\IPDL\docker-fullstack-app>
C:\Users\USER\Documents\M1-GLSI\Semestre 2\IPDL\docker-fullstack-app>
```

4. Ouvrons notre navigateur et accédons à http://localhost:8080 pour voir la page par défaut de Nginx.



5. Examinons les logs du conteneur :

docker logs test-nginx

```
C:\Users\USER\Documents\M1-GLSI\Semestre 2\IPDL\docker-fullstack-app>docker logs test-nginx
/docker-entrypoint.sh: /docker-entrypoint.d/ is not empty, will attempt to perform configuration
/docker-entrypoint.sh: Looking for shell scripts in /docker-entrypoint.d/
/docker-entrypoint.sh: Launching /docker-entrypoint.d/10-listen-on-ipv6-by-default.sh
10-listen-on-ipv6-by-default.sh: info: Getting the checksum of /etc/nginx/conf.d/default.conf
10-listen-on-ipv6-by-default.sh: info: Enabled listen on IPv6 in /etc/nginx/conf.d/default.conf
/docker-entrypoint.sh: Sourcing /docker-entrypoint.d/15-local-resolvers.envsh
/docker-entrypoint.sh: Launching /docker-entrypoint.d/20-envsubst-on-templates.sh
/docker-entrypoint.sh: Launching /docker-entrypoint.d/30-tune-worker-processes.sh
/docker-entrypoint.sh: Configuration complete; ready for start up
2025/05/05 09:46:16 [notice] 1#1: using the "epoll" event method
2025/05/05 09:46:16 [notice] 1#1: nginx/1.27.5
2025/05/05 09:46:16 [notice] 1#1: built by gcc 12.2.0 (Debian 12.2.0-14)
2025/05/05 09:46:16 [notice] 1#1: OS: Linux 5.15.167.4-microsoft-standard-WSL2
2025/05/05 09:46:16 [notice] 1#1: getrlimit(RLIMIT_NOFILE): 1048576:1048576
2025/05/05 09:46:16 [notice] 1#1: start worker processes
2025/05/05 09:46:16 [notice] 1#1: start worker process 29
2025/05/05 09:46:16 [notice] 1#1: start worker process 30
2025/05/05 09:46:16 [notice] 1#1: start worker process 31
2025/05/05 09:46:16 [notice] 1#1: start worker process 32
172.17.0.1 - - [05/May/2025:09:47:14 +0000] "GET / HTTP/1.1" 200 615 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/135.0.0.0 Safari/537.36" "-"
2025/05/05 09:47:21 [error] 29#29: *1 open() "/usr/share/nginx/html/favicon.ico" failed (2: No such file or directory), client: 172.17.0.1, server: localho
st, request: "GET /favicon.ico HTTP/1.1", host: "localhost:8080", referrer: "http://localhost:8080/"
172.17.0.1 - - [05/May/2025:09:47:21 +0000] "GET /favicon.ico HTTP/1.1" 404 555 "http://localhost:8080/" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWe
bKit/537.36 (KHTML, like Gecko) Chrome/135.0.0.0 Safari/537.36" "-"

C:\Users\USER\Documents\M1-GLSI\Semestre 2\IPDL\docker-fullstack-app>
```

6. Modifions la page d'accueil de Nginx en exécutant une commande dans le conteneur :

docker exec -it test-nginx bash

echo '<h1>Hello Docker World!</h1>' > /usr/share/nginx/html/index.html

exit

```
C:\Users\USER\Documents\M1-GLSI\Semestre 2\IPDL\docker-fullstack-app>docker exec -it test-nginx bash
root@56ccdf53acfb:/# echo '<h1>Hello Docker World!</h1>' > /usr/share/nginx/html/index.html
root@56ccdf53acfb:/# exit
exit

What's next:
    Try Docker Debug for seamless, persistent debugging tools in any container or image → docker debug test-nginx
    Learn more at https://docs.docker.com/go/debug-cli/

C:\Users\USER\Documents\M1-GLSI\Semestre 2\IPDL\docker-fullstack-app>
```

7. Rafraîchissons notre navigateur pour voir les changements.

Hello Docker World!

8. Arrêtons et supprimons le conteneur :

docker stop test-nginx

docker rm test-nginx

```
C:\Users\USER\Documents\M1-GLSI\Semestre 2\IPDL\docker-fullstack-app>docker stop test-nginx
test-nginx

C:\Users\USER\Documents\M1-GLSI\Semestre 2\IPDL\docker-fullstack-app>docker rm test-nginx
test-nginx

C:\Users\USER\Documents\M1-GLSI\Semestre 2\IPDL\docker-fullstack-app>
```

4. Création d'images Docker avec Dockerfile

Concept fondamental

Un Dockerfile est un script contenant une série d'instructions qui décrivent comment construire une image Docker. Chaque instruction crée une nouvelle couche dans l'image, permettant une construction efficace et incrémentale.

4.1 Structure d'un Dockerfile

Structure de base d'un Dockerfile

```
# Image de base
FROM node:14-alpine

# Métadonnées
LABEL maintainer="votre.email@exemple.com"
LABEL version="1.0"

# Variables d'environnement
ENV NODE_ENV=production

# Répertoire de travail
WORKDIR /app

# Copie des fichiers
COPY package*.json ./
COPY src/ ./src/

# Exécution de commandes
RUN npm install

# Exposition de ports
EXPOSE 3000

# Commande de démarrage
CMD ["npm", "start"]
```

4.2 Bonnes pratiques pour la création de Dockerfile

Point clé à retenir

Meilleures pratiques pour des images Docker efficaces :

- **Utiliser des images de base légères** : Privilégier les variantes Alpine ou Slim
- **Combiner les instructions RUN** : Réduire le nombre de couches en utilisant `&&` pour enchaîner les commandes
- **Supprimer les fichiers inutiles** dans la même instruction RUN
- **Utiliser .dockerignore** pour exclure les fichiers non nécessaires
- **Optimiser l'ordre des couches** : Placer les instructions qui changent peu au début
- **Utiliser des arguments de construction** (ARG) pour les valeurs variables
- **Spécifier des versions précises** des images de base plutôt que `latest`
- **Ne pas exécuter les conteneurs en tant que root** quand possible

4.3 Exercice pratique : Création d'un Dockerfile simple

Tâche à réaliser

Exercice 2 : Création d'un Dockerfile pour une application Node.js simple

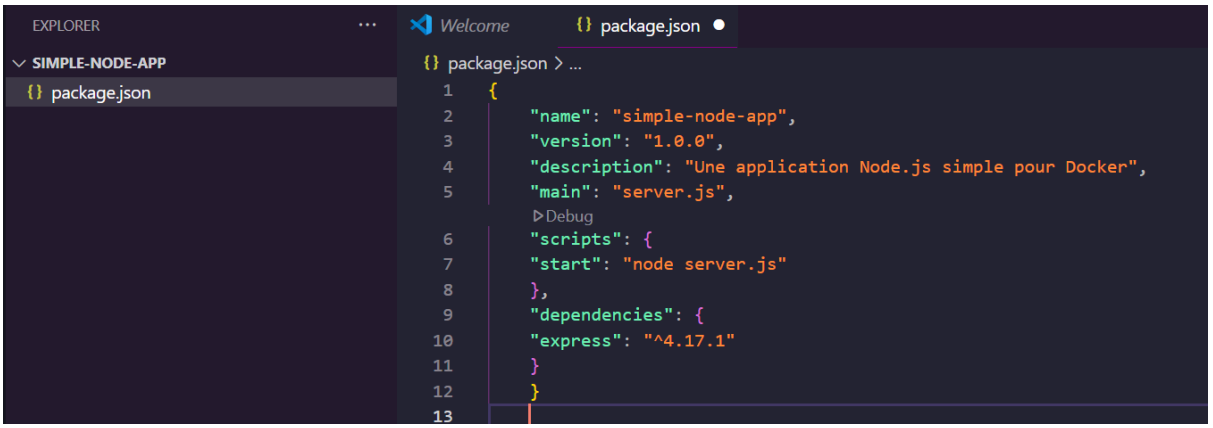
Dans cet exercice, nous allons créer une image Docker pour une application Node.js simple.

1. Créons un nouveau dossier et les fichiers suivants :

```
C:\Users\USER\Documents\M1-GLSI\Semestre 2\IPDL>mkdir simple-node-app

C:\Users\USER\Documents\M1-GLSI\Semestre 2\IPDL>
```

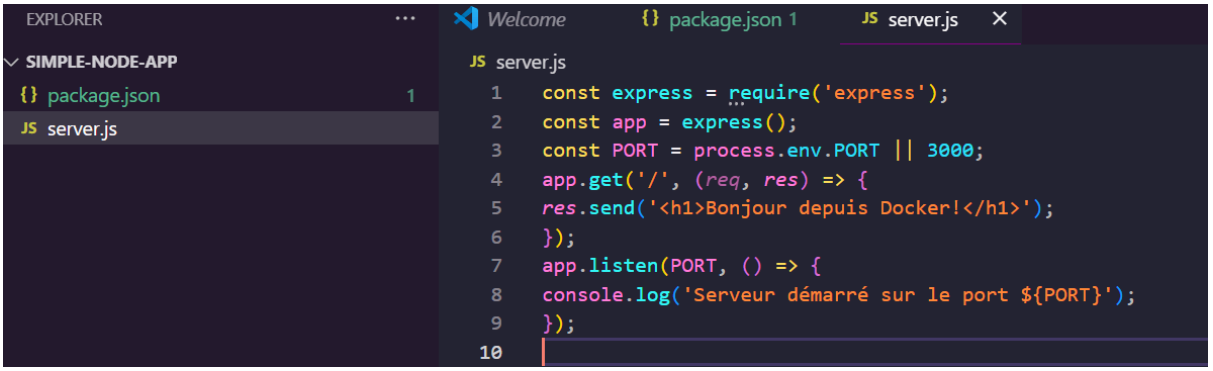
Dossier : **simple-node-app**
Fichier : **package.json**

```
{
"name": "simple-node-app",
"version": "1.0.0",
"description": "Une application Node.js simple pour Docker",
"main": "server.js",
"scripts": {
"start": "node server.js"
},
"dependencies": {
"express": "^4.17.1"
}
}
```

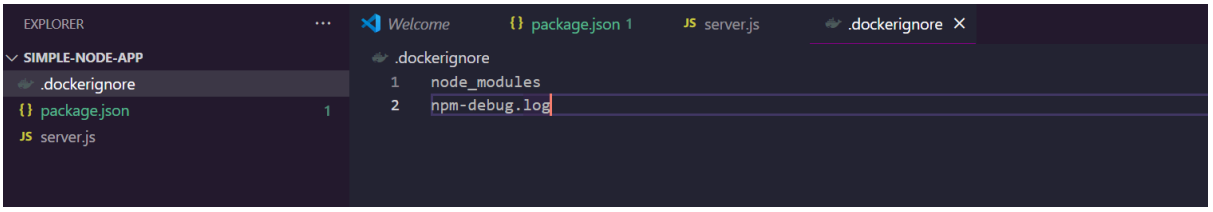


Fichier : **server.js**

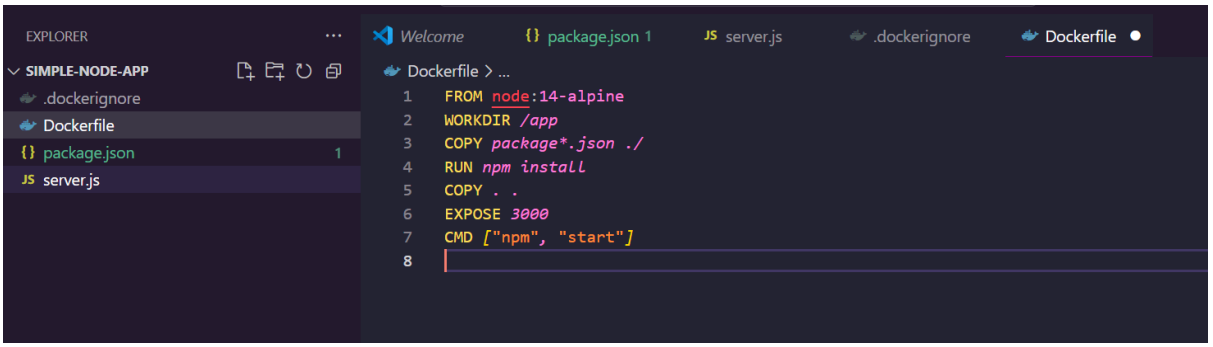
```
const express = require('express');
const app = express();
const PORT = process.env.PORT || 3000;
app.get('/', (req, res) => {
res.send('<h1>Bonjour depuis Docker!</h1>');
});
app.listen(PORT, () => {
console.log('Serveur démarré sur le port ${PORT}');
});
```



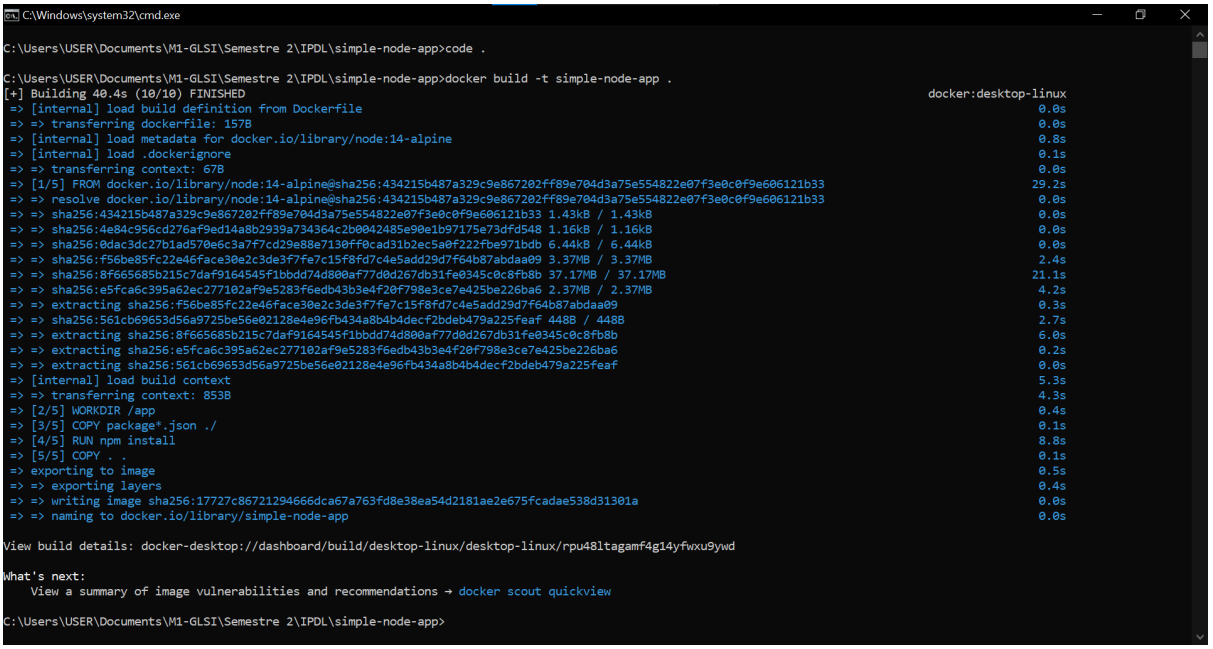
Fichier : **.dockerignore**
node_modules
npm-debug.log



2. Créons un Dockerfile avec le contenu suivant :
FROM node:14-alpine
WORKDIR /app
COPY package*.json ./
RUN npm install
COPY . .
EXPOSE 3000
CMD ["npm", "start"]



3. Construisons l'image Docker :
docker build -t simple-node-app .



4. Exécutons l'application dans un conteneur :

docker run --name node-app -p 3001:3000 -d simple-node-app

```
C:\Users\USER\Documents\M1-GLSI\Semestre 2\IPDL\simple-node-app>docker run --name node-app -p 3001:3000 -d simple-node-app
73985e3789336e5143238961e64ba4069980a458fa7600860165b3c9d9d5824f

C:\Users\USER\Documents\M1-GLSI\Semestre 2\IPDL\simple-node-app>
```

5. Testons l'application en ouvrant http://localhost:3001 dans notre navigateur.

Bonjour depuis Docker!

6. Arrêtons et supprimons le conteneur lorsque nous avons terminé :

docker stop node-app

docker rm node-app

```
C:\Users\USER\Documents\M1-GLSI\Semestre 2\IPDL\simple-node-app>docker stop node-app
node-app

C:\Users\USER\Documents\M1-GLSI\Semestre 2\IPDL\simple-node-app>docker rm node-app
node-app
```

5. Dockerisation d'une application full-stack

Concept fondamental

Conteneuriser une application full-stack implique de créer des images Docker distinctes pour chaque composant (frontend, backend, base de données) et de les configurer pour qu'ils puissent communiquer entre eux. Cette approche modulaire est au cœur de l'architecture microservices et facilite le déploiement, la mise à l'échelle et la maintenance.

5.1 Mise en place de la base de données MariaDB .

Nous commencerons par déployer une instance MariaDB pour notre application.

Créľation et déľmarrage d'un conteneur MariaDB

 # Téléchargement de l'image officielle MariaDB

docker pull mariadb:latest

```
C:\Users\USER\Documents\M1-GLSI\Semestre 2\IPDL\simple-node-app>docker pull mariadb:latest
latest: Pulling from library/mariadb
2726e237d1a3: Pull complete
0b86886c6aaa: Pull complete
2b221cf763a8: Pull complete
5e4180757702: Pull complete
43028b9f5f8e: Pull complete
bbef7eafa75b: Pull complete
ab732728101f: Pull complete
0c9f57c1bb30: Pull complete
Digest: sha256:81e893032978c4bf8ad43710b7a979774ed90787fa32d199162148ce28fe3b76
Status: Downloaded newer image for mariadb:latest
docker.io/library/mariadb:latest

What's next:
    View a summary of image vulnerabilities and recommendations → docker scout quickview mariadb:latest

C:\Users\USER\Documents\M1-GLSI\Semestre 2\IPDL\simple-node-app>
```

 # Vérification que l'image a bien été téléchargée

docker images

```
C:\Users\USER\Documents\M1-GLSI\Semestre 2\IPDL\simple-node-app>docker images
REPOSITORY                  TAG       IMAGE ID       CREATED         SIZE
simple-node-app             latest    17727c867212   8 minutes ago   124MB
nginx                       latest    a830707172e8   2 weeks ago     192MB
projet_sgbd-backend         latest    d1c908567860   6 weeks ago     1.16GB
projet_sgbd-frontend        latest    bd8cdbe19bb6   6 weeks ago     55.4MB
<none>                      <none>    be4a27d8b9ed   6 weeks ago     1.21GB
basebrain-frontend          latest    4bf73a04e684   6 weeks ago     53.8MB
basebrain-backend           latest    19ad911c2e78   6 weeks ago     1.21GB
minio/minio                 latest    2eaf94c71682   7 weeks ago     182MB
mariadb                     latest    9f3d79eba61e   2 months ago    328MB
test_khine                  latest    5644dc3d3657   9 months ago    2.2GB
yacineg/mon_image_docker    latest    5644dc3d3657   9 months ago    2.2GB
mariadb                     <none>    4486d64c9c3b   10 months ago   406MB

C:\Users\USER\Documents\M1-GLSI\Semestre 2\IPDL\simple-node-app>
```

 # Création et démarrage du conteneur MariaDB

docker run --name cardb -p 3306:3306 -e MARIADB_ROOT_PASSWORD=root -e MARIADB_DATABASE=cardb -d mariadb

```
C:\Users\USER\Documents\M1-GLSI\Semestre 2\IPDL\simple-node-app>docker run --name cardb -p 3306:3306 -e MARIADB_ROOT_PASSWORD=root -e MARIADB_DATABASE=cardb -d mariadb
ffdae82ee141c0ba7b72bb09a9d8fc374f96b775a251e7b91f8bc79d14c898e6

C:\Users\USER\Documents\M1-GLSI\Semestre 2\IPDL\simple-node-app>
```

 # Vérification que le conteneur est en cours d'exécution

docker ps

```
C:\Users\USER\Documents\M1-GLSI\Semestre 2\IPDL\simple-node-app>docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED          STATUS          PORTS                    NAMES
ffdae82ee141   mariadb   "docker-entrypoint.s…"   47 seconds ago   Up 47 seconds   0.0.0.0:3306->3306/tcp   cardb

C:\Users\USER\Documents\M1-GLSI\Semestre 2\IPDL\simple-node-app>
```

5.2 Configuration d'un réseau Docker

Pour permettre la communication entre les conteneurs, nous allons créer un réseau Docker dédié.

Création d'un réseau Docker et connexion du conteneur MariaDB

 # Création d'un réseau Docker nommé car-net

docker network create car-net

```
C:\Users\USER\Documents\M1-GLSI\Semestre 2\IPDL\simple-node-app>docker network create car-net
2e38e4e89063adab381d7aca26424dfb9063936d5820d861b7c4bd8add100d43

C:\Users\USER\Documents\M1-GLSI\Semestre 2\IPDL\simple-node-app>
```

 # Connexion du conteneur MariaDB au réseau

docker network connect car-net cardb

```
C:\Users\USER\Documents\M1-GLSI\Semestre 2\IPDL\simple-node-app>docker network connect car-net cardb

C:\Users\USER\Documents\M1-GLSI\Semestre 2\IPDL\simple-node-app>
```

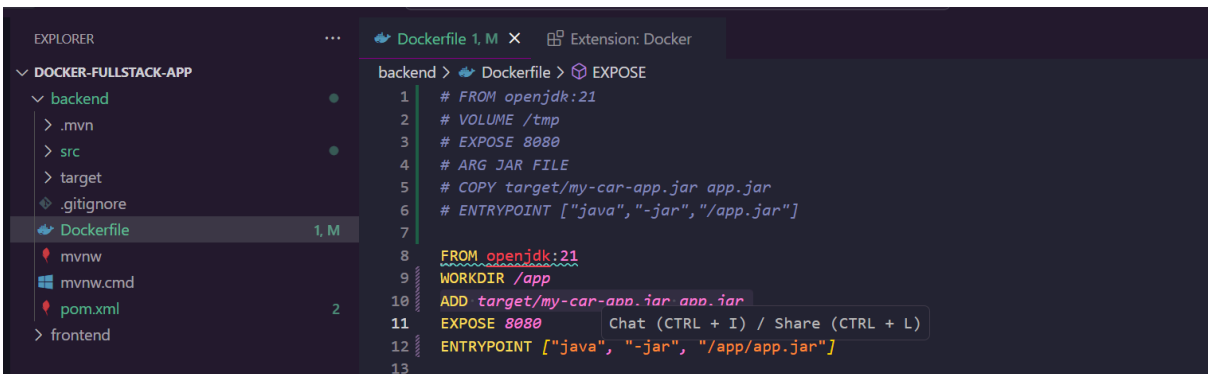
 # Vérification de la configuration du réseau

docker network inspect car-net

```
C:\Users\USER\Documents\M1-GLSI\Semestre 2\IPDL\simple-node-app>docker network inspect car-net
[
    {
        "Name": "car-net",
        "Id": "2e38e4e89063adab381d7aca26424dfb9063936d5820d861b7c4bd8add100d43",
        "Created": "2025-05-05T10:19:16.585263697Z",
        "Scope": "local",
        "Driver": "bridge",
        "EnableIPv6": false,
        "IPAM": {
            "Driver": "default",
            "Options": {},
            "Config": [
                {
                    "Subnet": "172.18.0.0/16",
                    "Gateway": "172.18.0.1"
                }
            ]
        },
        "Internal": false,
        "Attachable": false,
        "Ingress": false,
        "ConfigFrom": {
            "Network": ""
        },
        "ConfigOnly": false,
        "Containers": {
            "ffdae82ee141c0ba7b72bb09a9d8fc374f96b775a251e7b91f8bc79d14c898e6": {
                "Name": "cardb",
                "EndpointID": "b85ac1def22764d11db56153fb41efeea4cb6d765ef20a97ebc39a27c020656d",
                "MacAddress": "02:42:ac:12:00:02",
                "IPv4Address": "172.18.0.2/16",
                "IPv6Address": ""
            }
        },
        "Options": {},
        "Labels": {}
    }
]


C:\Users\USER\Documents\M1-GLSI\Semestre 2\IPDL\simple-node-app>
```

5.3 Création de l'image pour le backend Spring Boot

Maintenant, nous allons créer un Dockerfile pour notre application backend Spring Boot.



Construction de l'image du backend :
Naviguons dans le dossier du backend et exécutons la commande suivante :
cd backend
docker build -t carbackend .

```
C:\Users\USER\Documents\M1-GLSI\Semestre 2\IPDL\docker-fullstack-app>cd backend

C:\Users\USER\Documents\M1-GLSI\Semestre 2\IPDL\docker-fullstack-app\backend>docker build -t carbackend .
[+] Building 6.0s (8/8) FINISHED                                                                    docker:desktop-linux
 => [internal] load build definition from Dockerfile                                                               0.1s
 => => transferring dockerfile: 309B                                                                               0.0s
 => [internal] load metadata for docker.io/library/openjdk:21                                                      1.3s
 => [internal] load .dockerignore                                                                                  0.0s
 => => transferring context: 2B                                                                                    0.0s
 => [internal] load build context                                                                                  3.8s
 => => transferring context: 52.24MB                                                                               3.8s
 => CACHED [1/3] FROM docker.io/library/openjdk:21@sha256:af9de795d1f8d3b6172f6c55ca9ba1c5768baa11bb2dc8af7045c7d  0.0s
 => [2/3] WORKDIR /app                                                                                             0.1s
 => [3/3] ADD target/my-car-app.jar app.jar                                                                        0.2s
 => exporting to image                                                                                             0.4s
 => => exporting layers                                                                                            0.3s
 => => writing image sha256:ad7c9a3781e5f6965190c15f16f86f2da75d17b2c0dff875542c6c435d560e81                       0.0s
 => => naming to docker.io/library/carbackend                                                                      0.0s

View build details: docker-desktop://dashboard/build/desktop-linux/desktop-linux/ihh4jdcpbdp8g9vdh3o3eiq8u

What's next:
    View a summary of image vulnerabilities and recommendations → docker scout quickview

C:\Users\USER\Documents\M1-GLSI\Semestre 2\IPDL\docker-fullstack-app\backend>
```

Vérifiez que l'image a été correctement créée :
docker images

```
C:\Users\USER\Documents\M1-GLSI\Semestre 2\IPDL\docker-fullstack-app\backend>docker images
REPOSITORY        TAG       IMAGE ID       CREATED          SIZE
carbackend        latest    ad7c9a3781e5   2 minutes ago    556MB
simple-node-app   latest    17727c867212   43 minutes ago   124MB
nginx             latest    a830707172e8   2 weeks ago      192MB
mariadb           latest    9f3d79eba61e   2 months ago     328MB

C:\Users\USER\Documents\M1-GLSI\Semestre 2\IPDL\docker-fullstack-app\backend>S
```

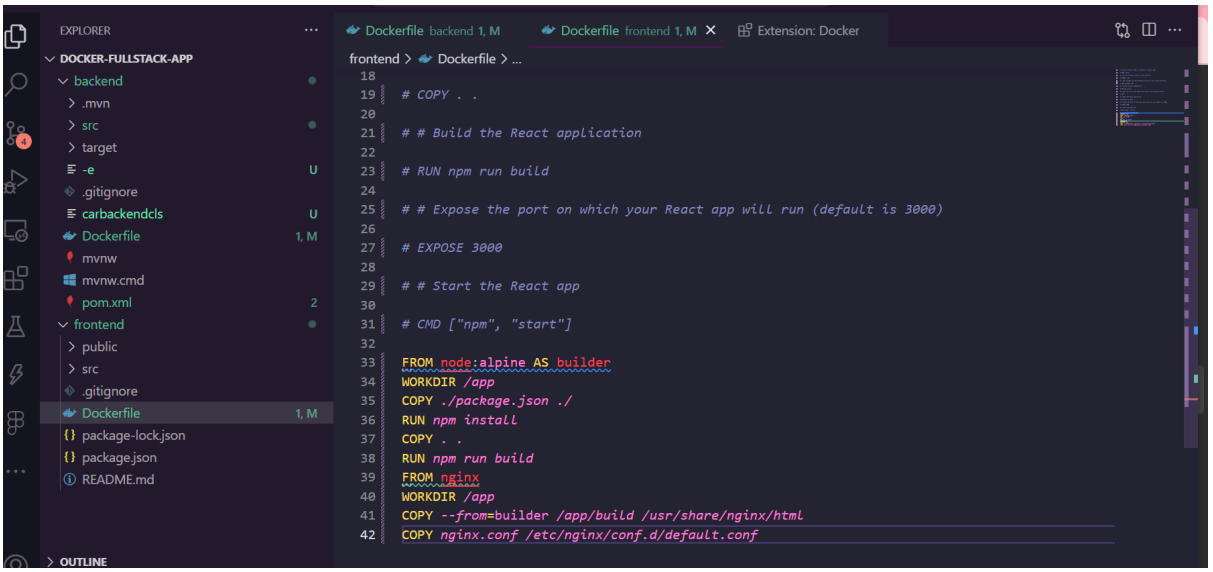
5.4 Exécution du backend
Maintenant que l'image du backend est prête, nous pouvons exécuter un conteneur à partir de cette image.
docker run -p 8080:8080 --name backend-car --net car-net -e MARIADB_HOST=cardb -e MARIADB_USER=root -e MARIADB_PASSWORD=root -e MYSQL_PORT=3306 carbackend

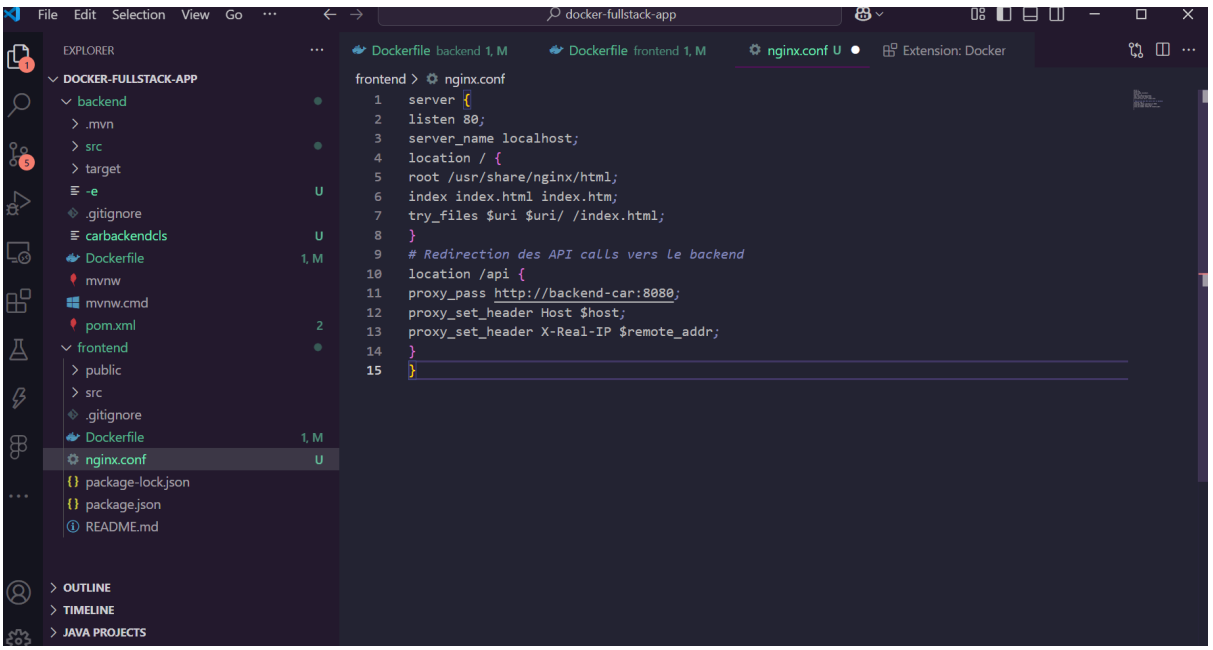
```
C:\Users\USER\Documents\M1-GLSI\Semestre 2\IPDL\docker-fullstack-app\backend>docker run -p 8080:8080 --name backend-car --net car-net -e MARIADB_HOST=cardb -e MARIADB_USER=
root -e MARIADB_PASSWORD=root -e MYSQL_PORT=3306 carbackend

  .   ____          _            __ _ _
 /\\ / ___'_ __ _ _(_)_ __  __ _ \ \ \ \
( ( )\___ | '_ | '_| | '_ \/ _` | \ \ \ \
 \\/  ___)| |_)| | | | | || (_| |  ) ) ) )
  '  |____| .__|_| |_|_| |_\__, | / / / /
 =========|_|==============|___/=/_/_/_/
 :: Spring Boot ::                (v3.1.2)

2025-05-05T10:58:00.894Z  INFO 1 --- [           main] com.dam.uasz.sbcar6.SbCarBackend        : Starting SbCarBackend v0.0.1-SNAPSHOT using Java 21 with PID 1 (/app/app.j
ar started by root in /app)
2025-05-05T10:58:00.899Z  INFO 1 --- [           main] com.dam.uasz.sbcar6.SbCarBackend        : No active profile set, falling back to 1 default profile: "default"
2025-05-05T10:58:02.831Z  INFO 1 --- [           main] .s.d.r.c.RepositoryConfigurationDelegate : Bootstrapping Spring Data JPA repositories in DEFAULT mode.
2025-05-05T10:58:02.929Z  INFO 1 --- [           main] .s.d.r.c.RepositoryConfigurationDelegate : Finished Spring Data repository scanning in 84 ms. Found 3 JPA repository
interfaces.
2025-05-05T10:58:04.678Z  INFO 1 --- [           main] o.s.b.w.embedded.tomcat.TomcatWebServer  : Tomcat initialized with port(s): 8080 (http)
2025-05-05T10:58:04.702Z  INFO 1 --- [           main] o.apache.catalina.core.StandardService   : Starting service [Tomcat]
2025-05-05T10:58:04.702Z  INFO 1 --- [           main] o.apache.catalina.core.StandardEngine    : Starting Servlet engine: [Apache Tomcat/10.1.11]
2025-05-05T10:58:04.985Z  INFO 1 --- [           main] o.a.c.c.C.[Tomcat].[localhost].[/]       : Initializing Spring embedded WebApplicationContext
2025-05-05T10:58:04.988Z  INFO 1 --- [           main] w.s.c.ServletWebServerApplicationContext : Root WebApplicationContext: initialization completed in 3813 ms
2025-05-05T10:58:05.416Z  INFO 1 --- [           main] o.hibernate.jpa.internal.util.LogHelper  : HHH000204: Processing PersistenceUnitInfo [name: default]
2025-05-05T10:58:05.524Z  INFO 1 --- [           main] org.hibernate.Version                    : HHH000412: Hibernate ORM core version 6.2.6.Final
2025-05-05T10:58:05.531Z  INFO 1 --- [           main] org.hibernate.cfg.Environment            : HHH000406: Using bytecode reflection optimizer
2025-05-05T10:58:06.391Z  INFO 1 --- [           main] o.h.b.i.BytecodeProviderInitiator        : HHH000021: Bytecode provider name : bytebuddy
2025-05-05T10:58:06.684Z  INFO 1 --- [           main] o.s.o.j.p.SpringPersistenceUnitInfo      : No LoadTimeWeaver setup: ignoring JPA class transformer
2025-05-05T10:58:06.710Z  INFO 1 --- [           main] com.zaxxer.hikari.HikariDataSource       : HikariPool-1 - Starting...
```

```
2025-05-05T11:13:20.932Z  INFO 1 --- [           main] j.LocalContainerEntityManagerFactoryBean : Initialized JPA EntityManagerFactory for persistence unit 'default'
2025-05-05T11:13:21.847Z  INFO 1 --- [           main] o.s.d.j.r.query.QueryEnhancerFactory     : Hibernate is in classpath; If applicable, HQL parser will be used.
2025-05-05T11:13:22.975Z  WARN 1 --- [           main] JpaBaseConfiguration$JpaWebConfiguration : spring.jpa.open-in-view is enabled by default. Therefore, database queries
 may be performed during view rendering. Explicitly configure spring.jpa.open-in-view to disable this warning
2025-05-05T11:13:23.440Z  INFO 1 --- [           main] o.s.s.web.DefaultSecurityFilterChain     : Will secure any request with [org.springframework.security.web.session.Dis
ableEncodeUrlFilter@57151b3a, org.springframework.security.web.context.request.async.WebAsyncManagerIntegrationFilter@26457986, org.springframework.security.web.context.Sec
urityContextHolderFilter@11df2829, org.springframework.security.web.header.HeaderWriterFilter@764fffa0, org.springframework.web.filter.CorsFilter@2dff7085, org.springframew
ork.security.web.authentication.logout.LogoutFilter@5ec6fede, org.springframework.security.web.savedrequest.RequestCacheAwareFilter@f245bdd, org.springframework.security.we
b.servletapi.SecurityContextHolderAwareRequestFilter@280fafd, org.springframework.security.web.authentication.AnonymousAuthenticationFilter@2faa55bb, org.springframework.se
curity.web.access.ExceptionTranslationFilter@5b733ef7, org.springframework.security.web.access.intercept.AuthorizationFilter@3a296107]
2025-05-05T11:13:24.619Z  INFO 1 --- [           main] o.s.b.w.embedded.tomcat.TomcatWebServer  : Tomcat started on port(s): 8080 (http) with context path ''
2025-05-05T11:13:24.744Z  INFO 1 --- [           main] com.dam.uasz.sbcar6.SbCarBackend         : Started SbCarBackend in 13.385 seconds (process running for 14.701)
2025-05-05T11:13:24.976Z  INFO 1 --- [           main] com.dam.uasz.sbcar6.SbCarBackend         : Ford Mustang
2025-05-05T11:13:24.976Z  INFO 1 --- [           main] com.dam.uasz.sbcar6.SbCarBackend         : Nissan Leaf
2025-05-05T11:13:24.976Z  INFO 1 --- [           main] com.dam.uasz.sbcar6.SbCarBackend         : Toyota Prius
```

5.5 Création de l'image pour le frontend React

Ensuite, nous allons créer un Dockerfile pour notre application frontend React.





Construction de l'image du frontend :

Naviguons dans le dossier du frontend et exécutez la commande suivante :

cd ../frontend

docker build -t carfrontend .

Vérifiez que l'image a été correctement créée :

docker images

```
C:\Users\USER\Documents\M1-GLSI\Semestre 2\IPDL\docker-fullstack-app\frontend>docker build -t carfrontend .
[+] Building 281.6s (16/16) FINISHED                                                                    docker:desktop-linux
 => [internal] load build definition from Dockerfile                                                                0.0s
 => => transferring dockerfile: 863B                                                                                0.0s
 => [internal] load metadata for docker.io/library/nginx:latest                                                     0.0s
 => [internal] load metadata for docker.io/library/node:alpine                                                      3.4s
 => [internal] load .dockerignore                                                                                   0.1s
 => => transferring context: 2B                                                                                     0.0s
 => [builder 1/6] FROM docker.io/library/node:alpine@sha256:86703151a18fcd06258e013073508c4afea8e19cd7ed451554221dd00aea83fc  33.8s
 => => resolve docker.io/library/node:alpine@sha256:86703151a18fcd06258e013073508c4afea8e19cd7ed451554221dd00aea83fc  0.2s
 => => sha256:86703151a18fcd06258e013073508c4afea8e19cd7ed451554221dd00aea83fc 6.41kB / 6.41kB                      0.0s
 => => sha256:0d468be7d2997dd2f6a3cda45e121a6b5140eb7ba3eba299a215030dbb0fb1ca 1.72kB / 1.72kB                      0.0s
 => => sha256:2b99bc550caad6f10cf0fd4ad72f86a14cc9818a05a66cc72d1997a4e8ee5c77 6.18kB / 6.18kB                      0.0s
 => => sha256:eff3549bb67a78cf95f3427c8a58e37085e210ac45ffd8f770cf633e4ebaa6b8 51.88MB / 51.88MB                   24.4s
 => => sha256:2c5da61da4d728e06869108119c05ea4b36003c7ea33431f8cb74375bcc9c2bf 447B / 447B                          0.7s
 => => sha256:201f41eb6ef73d0252a7abc37656dd7925a50ee95d67dd27de9420bb23071788 1.26MB / 1.26MB                      3.3s
 => => extracting sha256:eff3549bb67a78cf95f3427c8a58e37085e210ac45ffd8f770cf633e4ebaa6b8                           7.9s
 => => extracting sha256:201f41eb6ef73d0252a7abc37656dd7925a50ee95d67dd27de9420bb23071788                           0.1s
 => => extracting sha256:2c5da61da4d728e06869108119c05ea4b36003c7ea33431f8cb74375bcc9c2bf                           0.0s
 => [internal] load build context                                                                                   0.3s
 => => transferring context: 1.03MB                                                                                 0.2s
 => [stage-1 1/4] FROM docker.io/library/nginx:latest                                                               0.6s
 => [stage-1 2/4] WORKDIR /app                                                                                      0.2s
 => [builder 2/6] WORKDIR /app                                                                                      0.3s
 => [builder 3/6] COPY ./package.json ./                                                                            0.2s
 => [builder 4/6] RUN npm install                                                                                 198.4s
 => [builder 5/6] COPY . .                                                                                          0.1s
 => [builder 6/6] RUN npm run build                                                                                42.0s
 => [stage-1 3/4] COPY --from=builder /app/build /usr/share/nginx/html                                              0.1s
 => [stage-1 4/4] COPY nginx.conf /etc/nginx/conf.d/default.conf                                                    0.1s
 => exporting to image                                                                                              0.2s
 => => exporting layers                                                                                             0.1s
 => => writing image sha256:61b5921469812c67522539ee935d2acff998e0e7192e68df8017daea1c06ea0a                        0.0s
 => => naming to docker.io/library/carfrontend                                                                      0.0s

View build details: docker-desktop://dashboard/build/desktop-linux/desktop-linux/lctf2yxuadjnq9wilfxpcs8cn

What's next:
    View a summary of image vulnerabilities and recommendations → docker scout quickview
```

5.6 Exécution du frontend

Maintenant que l'image du frontend est prête, nous pouvons exécuter un conteneur à partir de cette image.

```
C:\Users\USER\Documents\M1-GLSI\Semestre 2\IPDL\docker-fullstack-app\frontend>docker run -d --name frontend-car -p 3001:80 --net car-net carfrontend
a842a0380d3e2703cfdd719f8bea97d8eb2f20a8b24061ef75bab5e997614bc1

C:\Users\USER\Documents\M1-GLSI\Semestre 2\IPDL\docker-fullstack-app\frontend>
```

5.7 Test de l'application complète

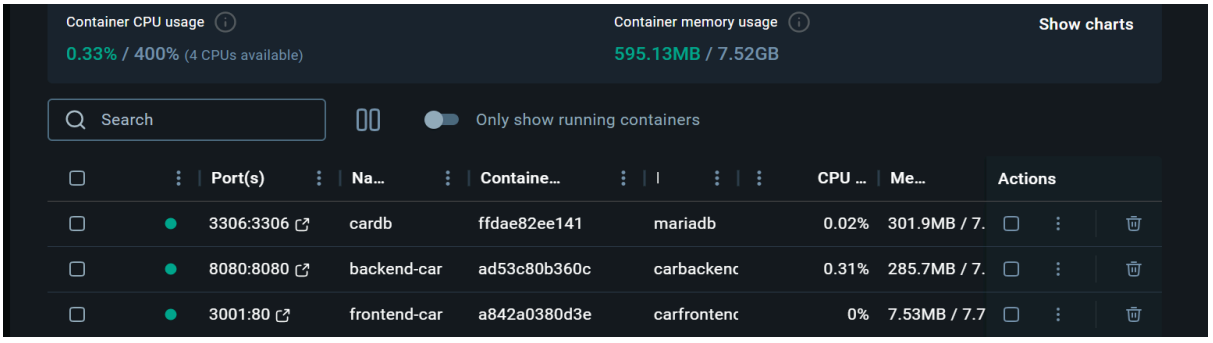
Une fois tous les conteneurs démarrés, nous pouvons tester notre application full-stack.

Test de l'application :

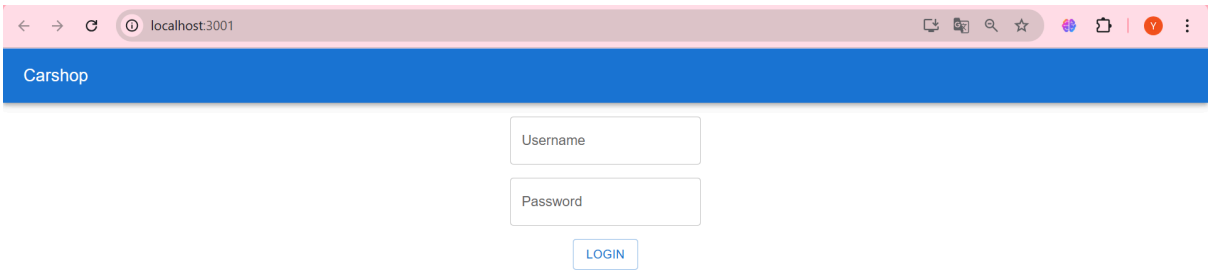
1. Vérifions que tous les conteneurs sont en cours d'exécution :

docker ps

```
C:\Users\USER\Documents\M1-GLSI\Semestre 2\IPDL\docker-fullstack-app\frontend>docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED             STATUS          PORTS                    NAMES
a842a0380d3e   carfrontend    "/docker-entrypoint.…"   56 seconds ago      Up 56 seconds   0.0.0.0:3001->80/tcp     frontend-car
ad53c80b360c   carbackend     "java -jar /app/app.…"   11 minutes ago      Up 11 minutes   0.0.0.0:8080->8080/tcp   backend-car
ffdae82ee141   mariadb        "docker-entrypoint.s…"   About an hour ago   Up 13 minutes   0.0.0.0:3306->3306/tcp   cardb

C:\Users\USER\Documents\M1-GLSI\Semestre 2\IPDL\docker-fullstack-app\frontend>
```

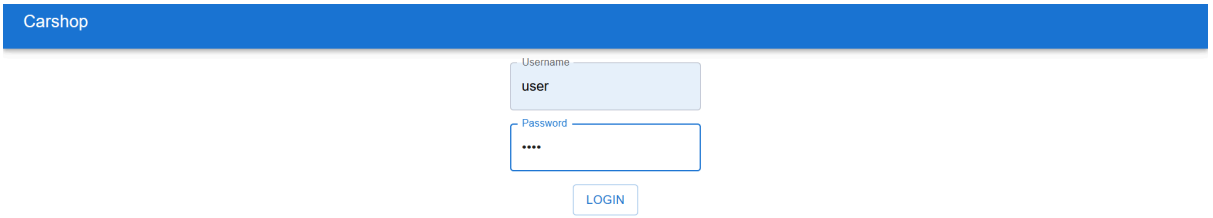


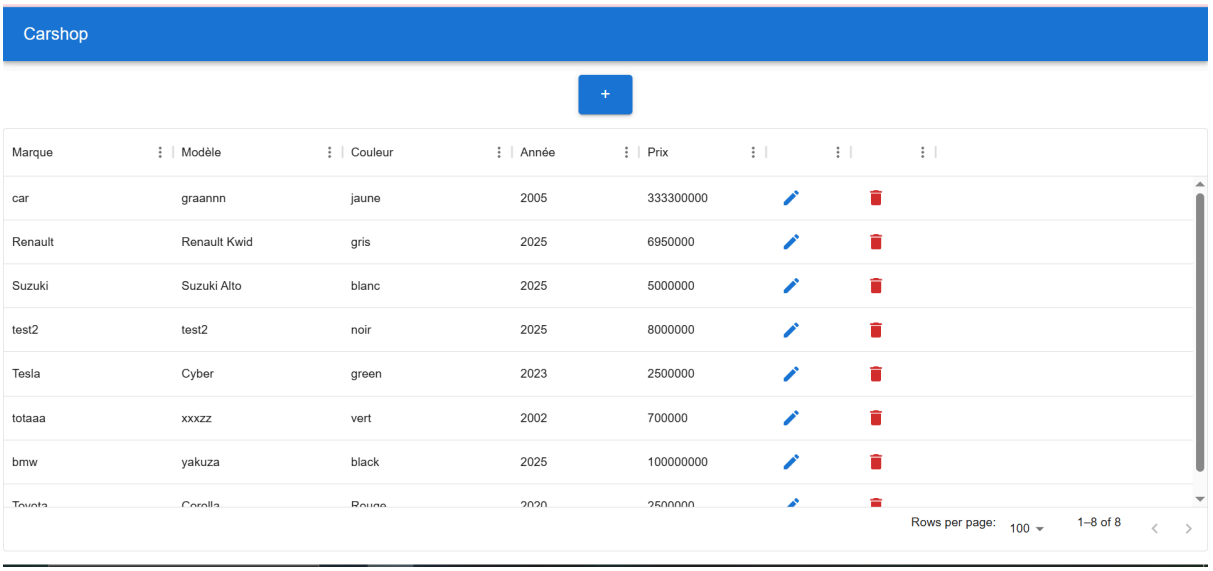
2. Ouvrons votre navigateur et accédons à http://localhost:3001



3. Nous devrions voir l'interface utilisateur de notre application de gestion de voitures

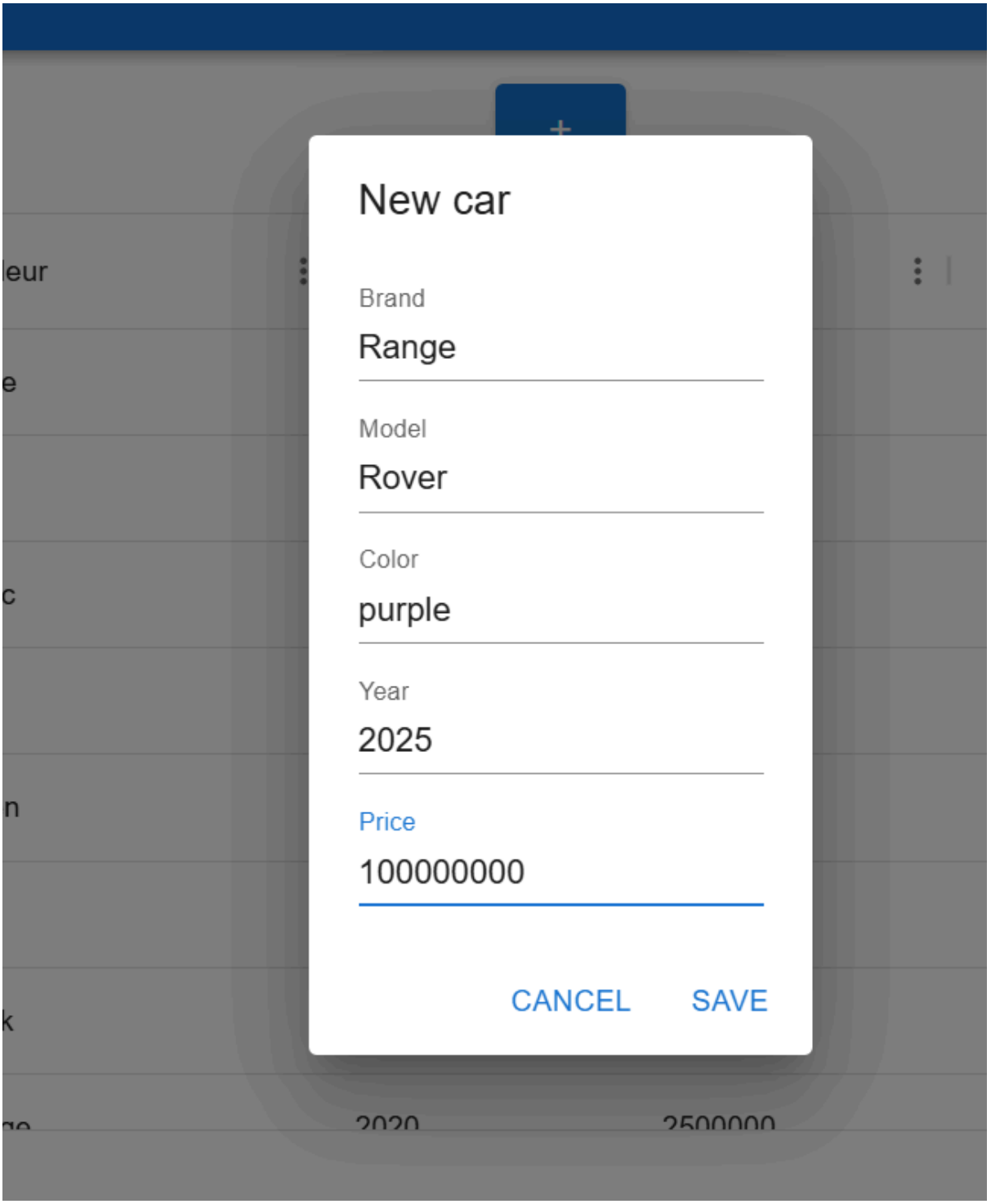
4. Essayez de nous connecter avec les identifiants user/user





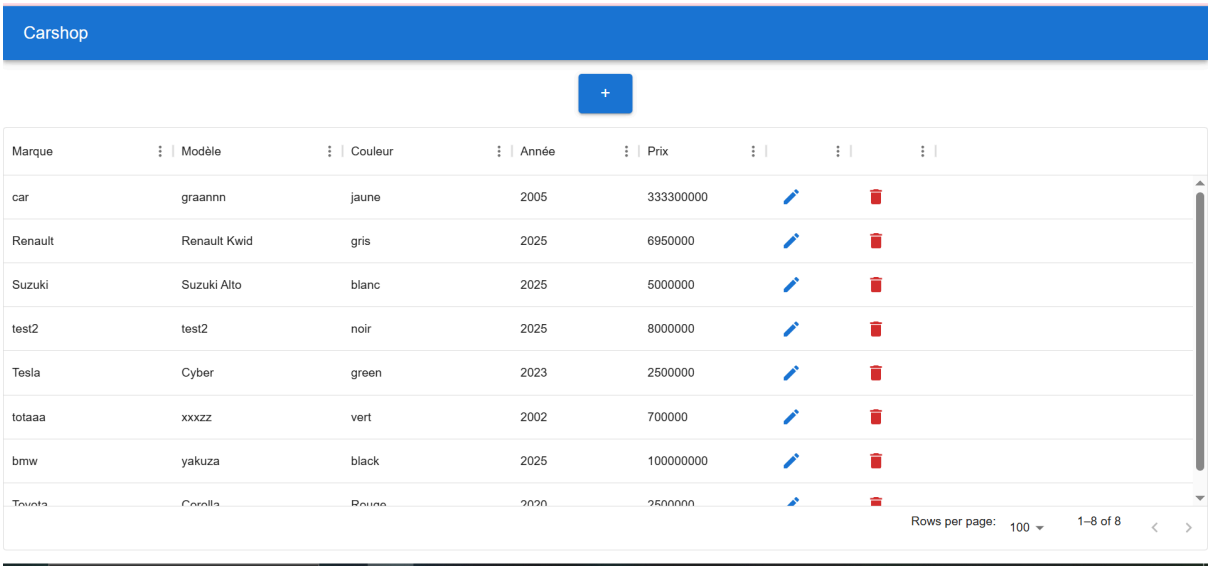
5. Testons les fonctionnalités CRUD (Création, Lecture, Mise à jour, Suppression) des voitures

- **CREATE**

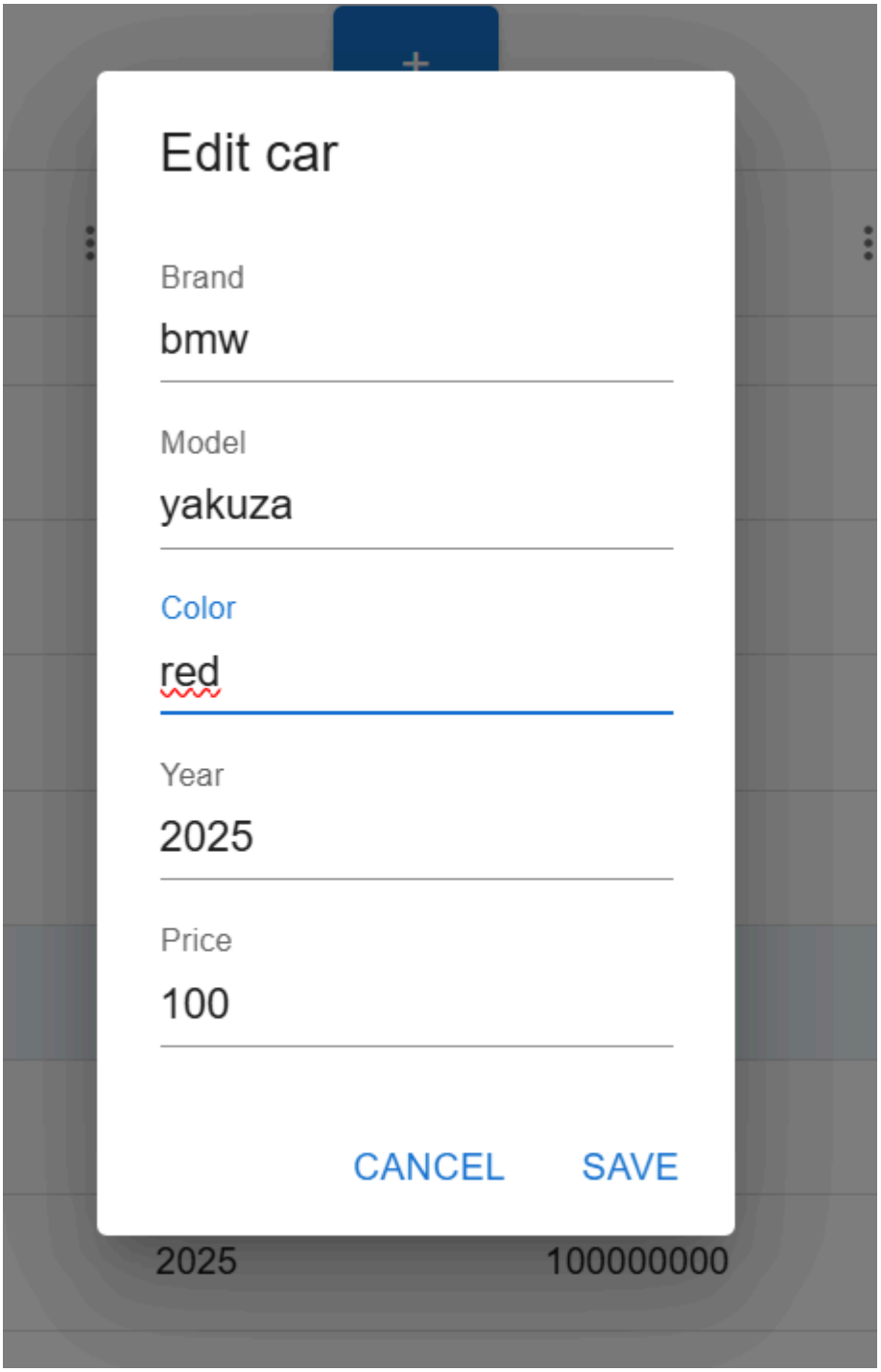


Range	Rover	purple	2025	100000000		

- **READ**

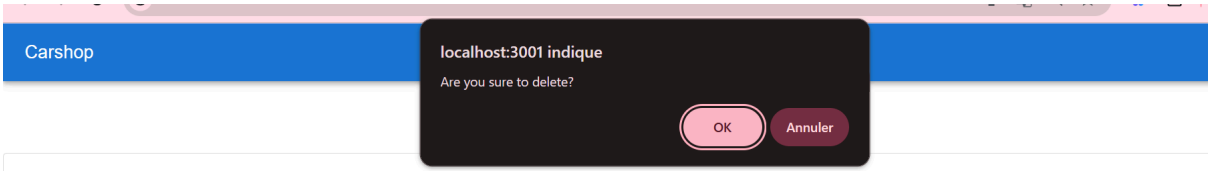

Carshop
+
Marque
Modèle
Couleur
Année
Prix
car graannn jaune 2005 333300000
Renault Renault Kwid gris 2025 6950000
Suzuki Suzuki Alto blanc 2025 5000000
test2 test2 noir 2025 8000000
Tesla Cyber green 2023 2500000
totaaa xxxzz vert 2002 700000
bmw yakuza black 2025 100000000
Rows per page: 100
1–8 of 8

- **UPDATE**



bmw	yakuza	red	2025	100		

- **DELETE**



Marque	Modèle	Couleur	Année	Prix
Renault	Renault Kwid	gris	2025	6950000
Suzuki	Suzuki Alto	blanc	2025	5000000
test2	test2	noir	2025	8000000

Marque	Modèle	Couleur	Année	Prix
car	graannn	jaune	2005	333300000
Renault	Renault Kwid	gris	2025	6950000
Suzuki	Suzuki Alto	blanc	2025	5000000
Tesla	Cyber	green	2023	2500000
totaaa	xxxzz	vert	2002	700000
bmw	yakuza	red	2025	100
Range	Rover	purple	2025	100000000

docker logs cardb

```
C:\Users\USER\Documents\M1-GLSI\Semestre 2\IPDL\docker-fullstack-app\frontend>docker logs cardb
2025-05-05 10:15:26+00:00 [Note] [Entrypoint]: Entrypoint script for MariaDB Server 1:11.7.2+maria~ubu2404 started.
2025-05-05 10:15:27+00:00 [Warn] [Entrypoint]: /sys/fs/cgroup/name=systemd:/docker/ffdae82ee141c0ba7b72bb09a9d8fc374f96b775a251e7b91f8bc79d14c898e6
14:misc:/docker/ffdae82ee141c0ba7b72bb09a9d8fc374f96b775a251e7b91f8bc79d14c898e6
13:rdma:/docker/ffdae82ee141c0ba7b72bb09a9d8fc374f96b775a251e7b91f8bc79d14c898e6
12:pids:/docker/ffdae82ee141c0ba7b72bb09a9d8fc374f96b775a251e7b91f8bc79d14c898e6
11:hugetlb:/docker/ffdae82ee141c0ba7b72bb09a9d8fc374f96b775a251e7b91f8bc79d14c898e6
10:net_prio:/docker/ffdae82ee141c0ba7b72bb09a9d8fc374f96b775a251e7b91f8bc79d14c898e6
9:perf_event:/docker/ffdae82ee141c0ba7b72bb09a9d8fc374f96b775a251e7b91f8bc79d14c898e6
8:net_cls:/docker/ffdae82ee141c0ba7b72bb09a9d8fc374f96b775a251e7b91f8bc79d14c898e6
7:freezer:/docker/ffdae82ee141c0ba7b72bb09a9d8fc374f96b775a251e7b91f8bc79d14c898e6
6:devices:/docker/ffdae82ee141c0ba7b72bb09a9d8fc374f96b775a251e7b91f8bc79d14c898e6
5:memory:/docker/ffdae82ee141c0ba7b72bb09a9d8fc374f96b775a251e7b91f8bc79d14c898e6
4:blkio:/docker/ffdae82ee141c0ba7b72bb09a9d8fc374f96b775a251e7b91f8bc79d14c898e6
3:cpuacct:/docker/ffdae82ee141c0ba7b72bb09a9d8fc374f96b775a251e7b91f8bc79d14c898e6
2:cpu:/docker/ffdae82ee141c0ba7b72bb09a9d8fc374f96b775a251e7b91f8bc79d14c898e6
1:cpuset:/docker/ffdae82ee141c0ba7b72bb09a9d8fc374f96b775a251e7b91f8bc79d14c898e6
0::/docker/ffdae82ee141c0ba7b72bb09a9d8fc374f96b775a251e7b91f8bc79d14c898e6/memory.pressure not writable, functionality unavailable to MariaDB
2025-05-05 10:15:27+00:00 [Note] [Entrypoint]: Switching to dedicated user 'mysql'
2025-05-05 10:15:27+00:00 [Note] [Entrypoint]: Entrypoint script for MariaDB Server 1:11.7.2+maria~ubu2404 started.
2025-05-05 10:15:28+00:00 [Note] [Entrypoint]: Initializing database files
2025-05-05 10:15:32+00:00 [Note] [Entrypoint]: Database files initialized
2025-05-05 10:15:32+00:00 [Note] [Entrypoint]: Starting temporary server
2025-05-05 10:15:32+00:00 [Note] [Entrypoint]: Waiting for server startup
2025-05-05 10:15:33 0 [Note] Starting MariaDB 11.7.2-MariaDB-ubu2404 source revision 80067a69feaeb5df30abb1bfaf7d4e713ccbf027 server_uid KjyTHYnzkWxRQBEIh5uvt04ggeY= as pro
cess 97
2025-05-05 10:15:33 0 [Note] InnoDB: Compressed tables use zlib 1.3
2025-05-05 10:15:33 0 [Note] InnoDB: Using transactional memory
2025-05-05 10:15:33 0 [Note] InnoDB: Number of transaction pools: 1
2025-05-05 10:15:33 0 [Note] InnoDB: Using crc32 + pclmulqdq instructions
2025-05-05 10:15:33 0 [Note] mariadbd: O_TMPFILE is not supported on /tmp (disabling future attempts)
2025-05-05 10:15:33 0 [Note] InnoDB: Using liburing
2025-05-05 10:15:33 0 [Note] InnoDB: Initializing buffer pool, total size = 128.000MiB, chunk size = 2.000MiB
2025-05-05 10:15:33 0 [Note] InnoDB: Completed initialization of buffer pool
2025-05-05 10:15:33 0 [Note] InnoDB: File system buffers for log disabled (block size=4096 bytes)
```

docker logs backend-car

```
C:\Users\USER\Documents\M1-GLSI\Semestre 2\IPDL\docker-fullstack-app\frontend>docker logs backend-car

  .   ____          _            __ _ _
 /\\ / ___'_ __ _ _(_)_ __  __ _ \ \ \ \
( ( )\___ | '_ | '_| | '_ \/ _` | \ \ \ \
 \\/  ___)| |_)| | | | | || (_| |  ) ) ) )
  '  |____| .__|_| |_|_| |_\__, | / / / /
 =========|_|==============|___/=/_/_/_/
 :: Spring Boot ::                (v3.1.2)

2025-05-05T11:13:12.438Z  INFO 1 --- [           main] com.dam.uasz.sbcar6.SbCarBackend         : Starting SbCarBackend v0.0.1-SNAPSHOT using Java 21 with PID 1 (/app/app.jar started by root in /app)
2025-05-05T11:13:12.452Z  INFO 1 --- [           main] com.dam.uasz.sbcar6.SbCarBackend         : No active profile set, falling back to 1 default profile: "default"
2025-05-05T11:13:14.213Z  INFO 1 --- [           main] .s.d.r.c.RepositoryConfigurationDelegate : Bootstrapping Spring Data JPA repositories in DEFAULT mode.
2025-05-05T11:13:14.335Z  INFO 1 --- [           main] .s.d.r.c.RepositoryConfigurationDelegate : Finished Spring Data repository scanning in 102 ms. Found 3 JPA repository interfaces.
2025-05-05T11:13:16.972Z  INFO 1 --- [           main] o.s.b.w.embedded.tomcat.TomcatWebServer  : Tomcat initialized with port(s): 8080 (http)
2025-05-05T11:13:17.007Z  INFO 1 --- [           main] o.apache.catalina.core.StandardService   : Starting service [Tomcat]
2025-05-05T11:13:17.008Z  INFO 1 --- [           main] o.apache.catalina.core.StandardEngine    : Starting Servlet engine: [Apache Tomcat/10.1.11]
2025-05-05T11:13:17.137Z  INFO 1 --- [           main] o.a.c.c.C.[Tomcat].[localhost].[/]       : Initializing Spring embedded WebApplicationContext
2025-05-05T11:13:17.143Z  INFO 1 --- [           main] w.s.c.ServletWebServerApplicationContext : Root WebApplicationContext: initialization completed in 4499 ms
2025-05-05T11:13:17.707Z  INFO 1 --- [           main] o.hibernate.jpa.internal.util.LogHelper  : HHH000204: Processing PersistenceUnitInfo [name: default]
2025-05-05T11:13:17.794Z  INFO 1 --- [           main] org.hibernate.Version                    : HHH000412: Hibernate ORM core version 6.2.6.Final
2025-05-05T11:13:17.800Z  INFO 1 --- [           main] org.hibernate.cfg.Environment            : HHH000406: Using bytecode reflection optimizer
2025-05-05T11:13:18.060Z  INFO 1 --- [           main] o.h.b.i.BytecodeProviderInitiator        : HHH000021: Bytecode provider name : bytebuddy
2025-05-05T11:13:18.376Z  INFO 1 --- [           main] o.s.o.j.p.SpringPersistenceUnitInfo      : No LoadTimeWeaver setup: ignoring JPA class transformer
2025-05-05T11:13:18.433Z  INFO 1 --- [           main] com.zaxxer.hikari.HikariDataSource       : HikariPool-1 - Starting...
2025-05-05T11:13:18.810Z  INFO 1 --- [           main] com.zaxxer.hikari.pool.HikariPool        : HikariPool-1 - Added connection org.mariadb.jdbc.Connection@2237bada
2025-05-05T11:13:18.814Z  INFO 1 --- [           main] com.zaxxer.hikari.HikariDataSource       : HikariPool-1 - Start completed.
2025-05-05T11:13:19.435Z  INFO 1 --- [           main] o.h.b.i.BytecodeProviderInitiator        : HHH000021: Bytecode provider name : bytebuddy
2025-05-05T11:13:20.496Z  INFO 1 --- [           main] o.h.e.t.j.p.i.JtaPlatformInitiator       : HHH000490: Using JtaPlatform implementation: [org.hibernate.engine.transaction.jta.platform.internal.N
rm]
2025-05-05T11:13:20.673Z  WARN 1 --- [           main] o.m.jdbc.message.server.ErrorPacket      : Error: 4091-42S02: Unknown SEQUENCE: 'cardb.car_seq'
2025-05-05T11:13:20.675Z  WARN 1 --- [           main] o.h.t.s.i.ExceptionHandlerLoggedImpl     : GenerationTarget encountered exception accepting command : Error executing DDL "drop sequence car_seq"
[(conn=4) Unknown SEQUENCE: 'cardb.car_seq']

org.hibernate.tool.schema.spi.CommandAcceptanceException: Error executing DDL "drop sequence car_seq" via JDBC [(conn=4) Unknown SEQUENCE: 'cardb.car_seq']
        at org.hibernate.tool.schema.internal.exec.GenerationTargetToDatabase.accept(GenerationTargetToDatabase.java:92) ~[hibernate-core-6.2.6.Final.jar!/:6.2.6.Final]
```

docker logs frontend-car

```
C:\Users\USER\Documents\M1-GLSI\Semestre 2\IPDL\docker-fullstack-app\frontend>docker logs frontend-car
/docker-entrypoint.sh: /docker-entrypoint.d/ is not empty, will attempt to perform configuration
/docker-entrypoint.sh: Looking for shell scripts in /docker-entrypoint.d/
/docker-entrypoint.sh: Launching /docker-entrypoint.d/10-listen-on-ipv6-by-default.sh
10-listen-on-ipv6-by-default.sh: info: Getting the checksum of /etc/nginx/conf.d/default.conf
10-listen-on-ipv6-by-default.sh: info: /etc/nginx/conf.d/default.conf differs from the packaged version
/docker-entrypoint.sh: Sourcing /docker-entrypoint.d/15-local-resolvers.envsh
/docker-entrypoint.sh: Launching /docker-entrypoint.d/20-envsubst-on-templates.sh
/docker-entrypoint.sh: Launching /docker-entrypoint.d/30-tune-worker-processes.sh
/docker-entrypoint.sh: Configuration complete; ready for start up
2025/05/05 11:23:45 [notice] 1#1: using the "epoll" event method
2025/05/05 11:23:45 [notice] 1#1: nginx/1.27.5
2025/05/05 11:23:45 [notice] 1#1: built by gcc 12.2.0 (Debian 12.2.0-14)
2025/05/05 11:23:45 [notice] 1#1: OS: Linux 5.15.167.4-microsoft-standard-WSL2
2025/05/05 11:23:45 [notice] 1#1: getrlimit(RLIMIT_NOFILE): 1048576:1048576
2025/05/05 11:23:45 [notice] 1#1: start worker processes
2025/05/05 11:23:45 [notice] 1#1: start worker process 28
2025/05/05 11:23:45 [notice] 1#1: start worker process 29
2025/05/05 11:23:45 [notice] 1#1: start worker process 30
2025/05/05 11:23:45 [notice] 1#1: start worker process 31
172.18.0.1 - - [05/May/2025:11:25:55 +0000] "GET / HTTP/1.1" 200 1015 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/135.0.0.0 Safari/537.36" "-"
172.18.0.1 - - [05/May/2025:11:25:55 +0000] "GET /static/css/main.3f2f11dc.css HTTP/1.1" 200 234629 "http://localhost:3001/" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chr
ome/135.0.0.0 Safari/537.36" "-"
172.18.0.1 - - [05/May/2025:11:25:57 +0000] "GET /static/js/main.d03ea594.js HTTP/1.1" 200 736503 "http://localhost:3001/" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrom
e/135.0.0.0 Safari/537.36" "-"
172.18.0.1 - - [05/May/2025:11:25:57 +0000] "GET /favicon.ico HTTP/1.1" 200 3870 "http://localhost:3001/" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/135.0.0.0 Safar
i/537.36" "-"
172.18.0.1 - - [05/May/2025:11:25:59 +0000] "GET /manifest.json HTTP/1.1" 200 517 "http://localhost:3001/" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/135.0.0.0 Safa
ri/537.36" "-"
172.18.0.1 - - [05/May/2025:11:25:59 +0000] "GET /logo192.png HTTP/1.1" 200 5347 "http://localhost:3001/" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/135.0.0.0 Safar
i/537.36" "-"
```