DOSSIER TECHNIQUE : OEDIKA

Table des matières

1. Introduction

Oedika est une PWA développée dans le cadre d'un TP sur Docker. Elle permet aux utilisateurs de prendre des photos avec leur caméra, de les stocker, et d'interagir avec une carte interactive. L'application est divisée en deux parties :

- Le frontend est développé avec VueJS et TypeScript.
- Le backend est développé avec NodeJS et Express, et utilise une base de données SQLite pour le stockage des photos.

L'application est entièrement conteneurisée avec Docker, ce qui facilite son déploiement et son exécution sur n'importe quelle machine.

2. Installation

Prérequis

- **Docker**
- **Docker Compose**
- **Git**

Étapes d'installation

1. **Cloner le dépôt** :

```
git clone https://github.com/ClementDaguenet/M1-MDS-Docker-Oedika.git
```

```
cd M1-MDS-Docker-Oedika
```

2. **Lancer l'application avec Docker Compose** :

```
docker compose up --build
```

3. **Accéder à l'application** :
    - Le frontend est accessible en ouvrant un navigateur et en accédant à http://localhost:8080.
    - Le backend avec une API disponible à http://localhost:3000.
4. **Arrêter l'application** :

```
docker compose down
```

3. Fonctionnalités

Accueil

- Un menu burger qui permet d'accéder aux différentes sections de l'application.
- Une barre de navigation : qui facilite la navigation entre les pages.

Caméra

- L'utilisateur peut activer sa caméra après avoir donné son accord.
- Les photos prises par la caméra déclenchent une notification système.
- Les photos sont stockées dans une base de données SQLite.
- Les photos précédentes peuvent être visualisées dans un carrousel.

Cartes

- Une carte interactive centrée sur la position actuelle de l'utilisateur.
- L'utilisateur peut rechercher une adresse et naviguer vers celle-ci.

4. Documentation Technique

Choix technologiques

Frontend

- **VueJS** : Framework JavaScript moderne et réactif pour créer des interfaces utilisateur.
- **TypeScript** : Ajoute une typage statique pour une meilleure maintenabilité du code.
- **PWA** : L'application est une Progressive Web App, permettant une expérience utilisateur fluide et une installation hors ligne.

Backend

- **NodeJS** : Environnement d'exécution JavaScript pour le serveur.
- **Express** : Framework pour créer des APIs RESTful.
- **SQLite** : Base de données légère et portable pour stocker les photos.

Docker

- **Conteneurisation** : Utilisation de Docker pour simplifier le déploiement et l'exécution de l'application.
- **Docker Compose** : Pour orchestrer les conteneurs frontend et backend.

Architecture de l'application

Configuration Docker

Frontend

- L'image est basée sur nginx pour servir les fichiers statiques.
- **Il est e**xposé sur le port 8080 de l'hôte.

Backend

- L'image est basée sur node:18.
- Il est exposé sur le port 3000 de l'hôte.
- Utilisation d'un volume Docker pour persister la base de données SQLite.

Réseau

- Les conteneurs communiquent via un réseau Docker personnalisé (app-network).

5. Conclusion

Oedika est une application moderne et facile à déployer grâce à Docker. Elle combine des fonctionnalités interactives (caméra, carte) avec une architecture robuste (VueJS, NodeJS, SQLite). La conteneurisation avec Docker garantit une exécution fiable et portable sur n'importe quelle machine.

Ce dossier technique fournit une vue d'ensemble complète de l'application, de son installation, de ses fonctionnalités et de ses choix technologiques. Pour toute question ou suggestion, n'hésitez pas à contacter l'auteur.