Installation / Déploiement

Last modified: 11 mai 2025

Cette section décrit les étapes nécessaires pour installer et déployer l'application SUPMAP sur une machine locale. Elle couvre les prérequis, la configuration de l'environnement, et les instructions spécifiques pour chaque composant de l'application.

Cette page sera découpée en deux types principaux (en fonction de la nature de l'installation):

- Docker compose, Dockerfiles et Server side
- build et déploiement de l'application mobile

Le fichier docker compose fera l'objet d'une section à part entière.

> Tous les fichiers `.env` et le fichier `local.properties` viennent avec un exemple de configuration, dans les répertoires correspondants.

Serveur Web (frontend)

> type : Docker compose

Le serveur web est développé avec Vite et React. Il suffit de lancer le fichier docker compose pour démarrer le serveur.

Variables d'environnement

> ⚠ **Vérifiez la configutation de votre fichier .env avant de lancer le docker compose.**
>
> le fichier .env du serveur web doit se trouver à la racine du dossier web : `./web/` .

Liste des variables d'environnement :

```
VITE_GOOGLE_MAPS_API_KEY= <YOUR_GOOGLE_MAPS_API_KEY>
VITE_API_GRAPHHOPPER_URL= http:<IP of Server running the graphhopper>//:8
VITE_API_SERVER_URL=http://<IP of Server running the api>:3000/api # Même
VITE_GOOGLE_CLIENT_ID= <YOUR_GOOGLE_CLIENT_ID>
```

> ⚠ **Très important : le serveur nécessite l'utilisation de clés ssl pour se lancer. si les clés ssl sont manquantes à l'adresse `./web/sslcerts` (voir le readme directement dans le dossier), le serveur refusera catégoriquement de se lancer. Les clés ssl doivent être au format PEM et doivent être nommées web-server.key et web-server.crt.**
>
> *Vous pouvez générer des clés ssl auto-signées avec openssl, ou utiliser un certificat ssl valide. (au format "KEY,CRT")*

Serveur API (backend)

> ℹ type : Docker compose

Le serveur API est développé avec Node.js et Express. Il suffit de lancer le fichier docker compose pour démarrer le serveur.

Variables d'environnement

⚠ **Vérifiez la configutation de votre fichier .env avant de lancer le docker compose.** le fichier .env du serveur API doit se trouver à la racine du dossier api : `./api/` .

Liste des variables d'environnement :

```
MONGO_URI = "mongodb://<root_user>:<password>@supmap-db:27017/" # User et
JWT_SECRET = <YOUR_JWT_SECRET> # Secret utilisé pour signer les tokens JW
HTTP_PORT = 3080 # Port HTTP du serveur API, Il est recommandé de ne pas
HTTPS_PORT = 3000 # Port HTTPS du serveur API, Il est recommandé de ne pa
GOOGLE_WEB_CLIENT_ID= <YOUR_GOOGLE_WEB_CLIENT_ID>
GOOGLE_ANDROID_CLIENT_ID= <YOUR_GOOGLE_ANDROID_CLIENT_ID>
GRAPHHOPPER_URL = "http://graphhopper:8989" # Adresse du serveur Graphhop
```

⚠ **Très important : le serveur nécessite l'utilisation de clés ssl pour se lancer. si les clés ssl sont manquantes à l'adresse `./api/sslcerts` (voir le readme directement dans le dossier), le serveur refusera catégoriquement de se lancer. Les clés ssl doivent être au format PEM et doivent être nommées api-server.key et api-server.crt.**

Vous pouvez générer des clés ssl auto-signées avec openssl, ou utiliser un certificat ssl valide. (au format "KEY,CRT")

Serveur Graphhopper

ℹ type : Docker compose

Le serveur Graphhopper utilise directement l'image dockerhub oficielle de Graphhopper. Il suffit de lancer le fichier docker compose pour démarrer le serveur.

Nous effectuons tous les téléchargements et configurations nécessaires à son fonctionnement dans le docker compose.

Le conteneur graphhopper demande beaucoup de temps pour se build, il est donc important d'être patient et de manipuler ses données et volumes avec précaution.

Ce conteneur ne possède pas de variables d'environnement, simplement un fichier de configuration préconfiguré à l'adresse `./graphhopper/config/graphhopper-config.yml`. Il est fortement déconseillé de le modifier.

Serveur Documentation

type : Docker compose

Le serveur de documentation (Peut-être la page sur laquelle vous vous trouvez) est développé avec Writerside (et délivré avec nginx). Il suffit de lancer le fichier docker compose pour démarrer le serveur.

Pas de variables d'environnement sur ce serveur

Application Mobile

type : Flutter

L'application mobile est développée avec Flutter.

Prérequis

- Flutter SDK installé sur votre machine
- Android Studio ou Visual studio code pour la compilation mobile.
- Un émulateur Android ou un appareil physique pour tester l'application

Pour installer Flutter, suivez les instructions sur le site officiel de Flutter : Guide d'installation de Flutter ↗

Variables d'environnement

Liste des variables d'environnement :

fichier .env : `./flutter/.env`

```
GOOGLE_MAPS_API_KEY= <YOUR_GOOGLE_MAPS_API_KEY>
SERVER_API_URL=https://<IP of Server running the api>:3000/api # Si le do
WEB_SOCKET_URL=wss://<IP of Server running the api>:3000 # Même raisonnem
GRAPHHOPPER_API_URL= http://<IP of Server running Graphhopper>:8989 # Mêm
GOOGLE_ANDROID_CLIENT_ID= <YOUR_GOOGLE_ANDROID_CLIENT_ID>
```

fichier local.properties (à ajouter aux clés déjà existantes) : `./flutter/android/local.properties`

```
GoogleMapsApiKey=<YOUR_GOOGLE_MAPS_API_KEY> # Même clé que dans le fichie
```

Veillez bien à ouvrir les différents ports sur votre pare-feu (3000, 3080, 8989) pour permettre la communication entre l'application mobile et le Serveur.

Compilation et installation de l'application mobile

1. Ouvrez un terminal à la racine du projet et naviguez jusqu'au répertoire de l'application mobile.

```
cd ./flutter/
```

2. Ensuite, exécutez la commande suivante pour installer les dépendances du projet (si elles ne sont pas déjà inclues):

```
flutter pub get
```

3. Vérifiez que le contenu de vos fichiers .env et local.properties (`./flutter/.env` et `./flutter/android/local.properties`) soient corrects. Voir la section précédente pour le détail sur les variables d'environnement.

4. Compilez l'application pour Android en utilisant la commande suivante :

```
flutter build apk --release
```

5. Une fois la compilation terminée, vous trouverez le fichier APK dans le répertoire `./flutter/build/app/outputs/flutter-apk/` . Vous pouvez l'installer sur un appareil Android en le transférant et en l'installant manuellement si vous avez un appareil physique à disposition, pensez à activer l'option d'installation par apk, si ce n'est pas déjà le cas.

6. Si vous préférez utiliser un émulateur Android, il est fortement conseillé d'utiliser Android Studio pour le lancer depuis ADB/AVD. Flutter/Android Studio vous proposera directement de lancer l'application sur vos appareils disponibles.

⚠ Cette méthode est plus difficile.

Docker compose

ℹ type : Docker compose

Le fichier docker compose est situé à la racine du projet. Il permet de lancer tous les services nécessaires au bon fonctionnement de l'application. Il est configuré pour utiliser les images officielles de MongoDB, Graphhopper, Node et Nginx. Il est également configuré pour utiliser les Dockerfiles du serveur API, du serveur Web et du Serveur Nginx pour la documentation.

📖 **Il est fortement recommandé de ne pas modifier le docker compose, sauf si vous savez ce que vous faites.**

Variables d'environnement

Le fichier Docker compose utilise uniquement les variables d'environnement pour mongodb. On retrouve donc les deux variables suivantes :

```
MONGO_ROOT_USER = <YOUR_MONGO_ROOT_USER>
MONGO_ROOT_PASSWORD = <YOUR_MONGO_ROOT_PASSWORD>
```

📖 **Veillez à bien faire correspondre l'utilisateur et le mot de passe de MongoDB avec ceux utilisés dans le serveur API. Sinon, l'api ne pourra pas démarrer.**

Une copie de cette page sera exportée en fichier PDF pour permettre une lecture hors ligne, nécessaire à la première utilisation.