2025

Abdoul-Waris Konaté, Abdoul Fadel Biaou, Hamza Belyahiaoui, Jean-philippe Delon

Trafine



TRAFICANDME : Documentation technique

SOMMAIRE

**Présentation du projet3**

Stack technique4

**Architecture du projet** 3

**Description des composants4**

API Rest5

Déploiement 6

**Sécurité** 7

**Perspectives d’évolution** 9

Présentation du projet

Traficandme est une application web et mobile de **navigation intelligente et participative**, conçue pour **améliorer l’expérience de conduite** en zones urbaines et périurbaines.  
Grâce à son interface moderne et communautaire, l'application permet de :

* Visualiser le trafic en temps réel
* Signaler les incidents (accidents, bouchons, dangers)
* Calculer des itinéraires optimisés
* Estimer le coût des trajets
* Recevoir des alertes en fonction de la position

Traficandme repose sur une architecture **microservices** moderne, combinant :

* **React + Vite** pour le frontend web
* **React Native** pour le mobile
* **Spring Boot** pour le backend
* **PostgreSQL** pour la base de données
* **OAuth2 + JWT** pour l’authentification
* **Docker** pour le déploiement
* **TomTom API** pour la cartographie et la navigation
* **GitHub** pour la gestion du code source et la collaboration

Compatible avec des intégrations tierces via une API REST, **Traficandme** vise aussi bien les particuliers que les collectivités ou entreprises souhaitant enrichir leurs services avec une solution de navigation collaborative.

Stack technique

Voici ci-dessous un tableau récapitulatif des technologies utilisées pour développer notre SaaS :

|  |  |  |
| --- | --- | --- |
| Composant | Technologie utilisée | Description |
| Frontend Web | React + Vite | Interface web rapide, moderne et responsive |
| Frontend Mobile | React Native | Application mobile native multiplateforme |
| Backend | Spring Boot (Java) | API REST performante et sécurisée |
| Base de données | PostgreSQL | Stockage relationnel des données utilisateurs et signalements |
| Authentification | OAuth2+JWT | Sécurisation des accès utilisateurs |
| Déploiement | Docker | Conteneurisation de tous les services |
| Cartographie/ GPS | TomTom SDK/API | Navigation GPS, affichage des cartes et gestion du trafic |
| Outil de gestion | Github | Hébergement du code source et collaboration en équipe |

Architecture de l’application

Les prérequis pour démarrer l’application :

* Node.js (v22 ou plus)
* Npm (v10 ou plus)
* MongoDB

La structure de notre projet :

traficandme/

├── api-traficandme/

├── web-traficandme/ # Application web pour utilisateurs et administrateurs

├── mobile-traficandme/ # Application mobile pour utilisateurs

* api-traficandme contient tous les services backend (authentification, signalements, itinéraires...)
* web-traficandme représente l’application web pour utilisateurs et l’interface administrateur
* mobile-traficandme est l’application mobile pour utilisateurs

Description des composants

L’architecture microservice de **Traficandme** repose sur plusieurs composants clés, chacun jouant un rôle spécifique dans l’écosystème global. Cette modularité favorise l’évolutivité, la maintenabilité et la robustesse de la plateforme.

* **Frontend Web**
* **Technologie :** React + Vite
* **Rôle :** Fournir une interface web responsive pour les utilisateurs et les administrateurs.
* **Fonctionnalités principales :**
  + Connexion et inscription sécurisées
  + Consultation du trafic en temps réel
  + Signalement d’incidents
  + Visualisation des itinéraires optimisés
  + Tableau de bord administrateur (gestion des utilisateurs et des signalements)
* **Frontend Mobile**
* **Technologie :** React Native
* **Rôle :** Permettre une expérience mobile fluide et native sur Android et iOS.
* **Fonctionnalités principales :**
* Navigation GPS avec affichage en temps réel
* Signalement rapide d’événements via géolocalisation
* Notifications push contextuelles
* Sauvegarde des trajets favoris
* Backend
* **Technologie :** Spring Boot (Java)
* **Rôle :** Fournir l’ensemble des services métiers via des endpoints REST.
* **Services gérés :**
* Authentification et gestion des rôles
* Gestion des utilisateurs
* Traitement des signalements
* Calcul d’itinéraires
* Envoi de notifications
* Intégration avec les services tiers (TomTom, OAuth, JWT)
* **Base de données**
* **Technologie :** PostgreSQL
* **Rôle :** Stocker les données de manière fiable et structurée.
* **Données gérées :**
* Informations utilisateurs
* Historique de signalements
* Logs d’événements
* Statistiques de trafic
* **Service TomTom**
* **Technologie :** SDK/API TomTom
* **Rôle :** Fournir les services de cartographie, de géolocalisation et de calcul d’itinéraires.
* **Fonctionnalités intégrées :**
  + Affichage de la carte
  + Suivi GPS en temps réel
  + Calcul dynamique de trajets
  + Données de trafic actualisées
* Système d’Authentification
* **Technologies :** OAuth2 + JWT
* **Rôle :** Sécuriser l’accès aux services et aux données utilisateur.
* **Fonctionnement :**
* OAuth2 assure la délégation d’identité avec des fournisseurs tiers
* JWT (JSON Web Token) est utilisé pour maintenir la session utilisateur de manière stateless
* Intégration avec Spring Security pour la gestion des autorisations
* Conteneurisation
* **Technologie :** Docker
* **Rôle :** Assurer le déploiement, l’isolation et la portabilité de chaque service.
* **Avantages :**
* Environnement unifié pour les développeurs
* Déploiement simplifié sur n’importe quelle machine
* Meilleure scalabilité et gestion des versions

API REST

L’API REST de **Traficandme** constitue le cœur de la communication entre les différents composants (frontend web, mobile et services tiers). Développée avec **Spring Boot**, elle suit les standards RESTful pour garantir une structure claire, maintenable et facilement extensible.

Pour avoir plus de détails sur l’API, consultez ce [lien](http://localhost:8080/swagger-ui/indeX.html).

Déploiement

* Prérequis
* Node.js (v22 ou supérieur)
* npm (v10 ou supérieur)
* Docker et Docker Compose
* java v17
* API de cartographie "Tomtom map"
* **Installation**

**1. Cloner le dépôt**

git clone https://github.com/trafine/traficandme.git

cd traficandme

**2. Installer les Dépendances**

Installer les dépendances pour chaque service :

* Front-End

cd web-traficandme

./mvnw spring-boot:run

* Mobile

cd ../mobile-traficandme

npm install

* API

cd ../api-traficandme

npm install

**4. Initialiser la Base de Données**

Initialiser la base de données avec les données initiales :

Processus automatique après le lancement du projet

**5. Démarrer les Services**

Démarrer chaque service dans un terminal séparé :

* API

cd api-traficandme

./mvnw spring-boot:run

* Front-End

cd web-traficandme

npm run dev

* Mobile (développement)

cd mobile-traficandme

npx expo start

* Via Docker

# Construire les images Docker

docker-compose build

# Démarrer les services

docker-compose up -d

**Utilisation de l'Application**

Pour utiliser l'application, suivez ces étapes :

1. Assurez-vous que MongoDB est en cours d'exécution sur votre système
2. Ouvrez un terminal et naviguez vers le répertoire racine du projet
3. Démarrez tous les services dans des terminaux séparés en utilisant les commandes ci-dessus
4. Accédez à l’application :
   * Front-End Web : <http://localhost:5173>
   * Documentation API : <http://localhost:8080/swagger-ui/indeX.html>
   * API : <http://localhost:8080/api>

Sécurité

Perspective d’évolution

Conclusion