

2025

TRAFICANDME : Documentation technique



Abdoul-Waris Konaté, Abdoul Fadel Biao,
Hamza Betyahiaoui, Jean-philippe Delon
Trafine

SOMMAIRE

Présentation du projet.....	2
Stack technique	3
Architecture de l'application	5
Description des composants	6
API Rest	10
Déploiement	11
Sécurité	14
Perspectives d'évolution	17
Conclusion	18

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
- **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	Pour une interface web moderne, rapide et réactive grâce à la compilation Vite.
Frontend Mobile	React Native	Développement multiplateforme natif pour Android et iOS avec un seul codebase.
Backend	Spring Boot (Java)	Framework robuste pour construire des API REST performantes et maintenables.
Base de données	PostgreSQL	SGBD relationnel fiable, puissant, open-source, adapté aux données structurées.
Authentification	JWT	Sécurisation des accès utilisateurs
Déploiement	Docker	Conteneurisation pour faciliter le déploiement, la scalabilité et la portabilité.
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

Pourquoi ce choix de stack ?

Notre objectif était de construire une application **moderne, évolutive, performante et maintenable**.

- Le **couplage React / React Native** nous permet de mutualiser une partie du code tout en ciblant plusieurs plateformes (web et mobile).

- **Spring Boot** a été choisi pour sa stabilité, sa scalabilité, et sa large adoption en entreprise.
- **JWT** permet une authentification simple à mettre en œuvre dans un environnement décentralisé comme une architecture microservices.
- **PostgreSQL** offre une grande fiabilité pour les données critiques (utilisateurs, signalements).
- Enfin, **Docker** simplifie considérablement le déploiement local et en production, rendant notre solution facilement portable.

Architecture de l'application

La structure de notre projet :

traficandme/

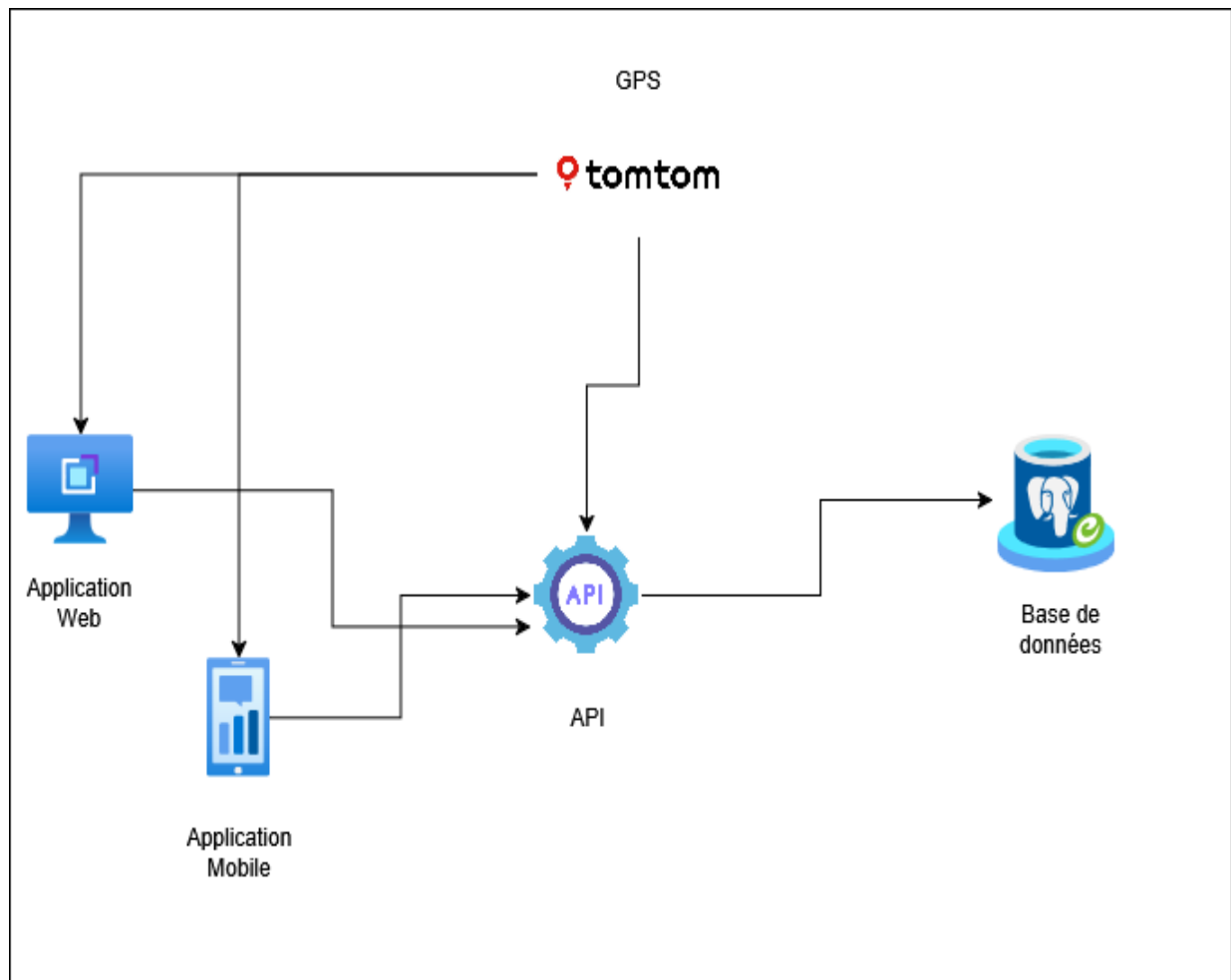
└─ api-traficandme/

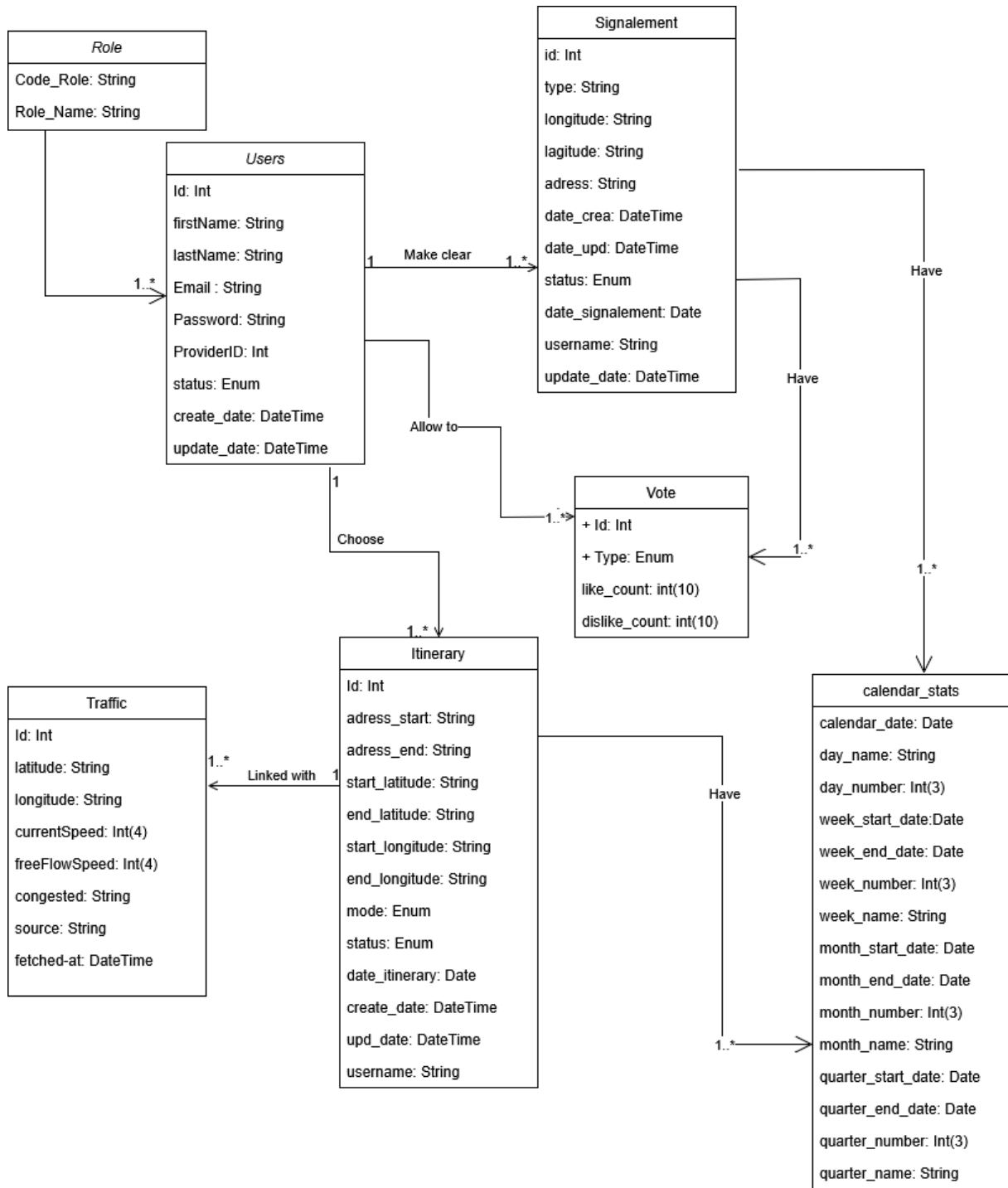
└─ web-traficandme/ # Application web pour utilisateurs et administrateurs

└─ 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

Voici ci-dessous la structure en schéma de Traficandme :





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, 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** : JWT
- **Rôle** : Sécuriser l'accès aux services et aux données utilisateur.
- **Fonctionnement** :
 - 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, suivez les consignes de la rubrique déploiement puis consultez ce [lien](#).

Déploiement

- Prérequis

Avant de démarrer l'installation de Traficandme, assurez-vous que les outils suivants sont installés sur votre machine :

- **Node.js** (v22 ou supérieur)
- **npm** (v10 ou supérieur)
- **Java** (version 17)
- **Docker & Docker Compose**
- **TomTom API Key** (nécessaire pour les services de cartographie)
- **MongoDB** (doit être installé et lancé localement)

- **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-trafficandme
```

```
npm install
```

3. Initialiser la Base de Données

Initialiser la base de données avec les données initiales :

Processus automatique après le lancement du projet

4. Démarrer les Services

Ouvrir un terminal pour chaque service et exécuter les commandes suivantes :

- API

```
cd api-trafficandme
```

```
./mvnw spring-boot:run
```

- Front-End

```
cd web-trafficandme
```

```
npm run dev
```

- Mobile (développement)

```
cd mobile-trafficandme
```

```
npx expo start
```

5. Déploiement avec Docker

Pour lancer tous les services en mode conteneurisé :

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 :

Une fois tous les services démarrés, les interfaces sont accessibles via les adresses suivantes :

- **Frontend Web** : <http://localhost:5173>
- **Documentation API** (Swagger) : <http://localhost:8080/swagger-ui/index.html>
- **API Backend** : <http://localhost:8080/api>

Sécurité

La sécurité est au cœur de l'architecture de Traficandme. Pour garantir la confidentialité, l'intégrité et l'authenticité des échanges, nous avons mis en place plusieurs mécanismes solides de protection des utilisateurs et des services.

Authentification via JWT

L'authentification dans *Traficandme* repose actuellement sur l'utilisation de **JSON Web Tokens (JWT)**. Cette méthode permet de sécuriser les échanges entre le client et l'API, tout en maintenant une architecture sans état (*stateless*).

Fonctionnement :

1. **Connexion de l'utilisateur** : L'utilisateur saisit ses identifiants (email et mot de passe) via l'interface de connexion.
2. **Vérification par le backend** : Le backend valide les identifiants à l'aide des données stockées en base.
3. **Génération d'un JWT** : En cas de succès, un token JWT est généré, signé et renvoyé au client.
4. **Utilisation du token** : Ce token est stocké côté client et transmis dans le header `Authorization: Bearer <token>` pour chaque requête vers une route protégée.
5. **Contrôle du token** : Le backend vérifie la signature et les droits associés au JWT avant d'autoriser l'accès.

Avantages :

- Authentification rapide et sécurisée
- Architecture sans session côté serveur
- Facilité d'intégration sur mobile et web
- Réduction des échanges inutiles avec la base de données

Gestion des rôles et des autorisations

Les **rôles** des utilisateurs (utilisateur standard, administrateur...) sont codés dans les **claims** du token JWT.

Le backend Spring Boot utilise ces informations pour **restreindre l'accès** à certaines routes avec des annotations comme `@PreAuthorize`.

Sécurité de l'API

- **JWT obligatoire** sur toutes les routes privées
- **Contrôle d'accès par rôle**
- **Validation des entrées utilisateur**
- **CORS configuré finement** pour éviter les accès non autorisés
- **Swagger (documentation API)** accessible uniquement en mode développement

Sécurité des données

La sécurité des utilisateurs et de leurs informations personnelles est au cœur de l'architecture de *Traficandme*. Plusieurs mécanismes sont mis en place pour assurer la protection des données tout au long de leur cycle de vie :

- **Chiffrement des mots de passe** : Tous les mots de passe sont hashés de manière sécurisée avant d'être stockés en base de données.
- **Communication chiffrée** : Toutes les communications entre le client et le serveur sont sécurisées via HTTPS (protocole TLS), garantissant la confidentialité et l'intégrité des échanges.
- **Réponses épurées** : Aucune donnée sensible (comme les mots de passe, tokens, ou informations critiques) n'est transmise dans les réponses JSON.
- **Logs sécurisés** : Les journaux applicatifs sont configurés pour éviter toute fuite d'informations personnelles ou d'identifiants sensibles.

Sécurité Docker

- Services isolés par **conteneurs indépendants**
- **Secrets et variables d'environnement** stockés dans des fichiers `.env` (jamais versionnés)
- **Ports restreints** et exposés uniquement si nécessaire

Bonnes pratiques

- Rotation des tokens et gestion de l'expiration
- Mise à jour régulière des bibliothèques et dépendances
- Analyse de sécurité statique et dynamique
- Prévention des attaques XSS, CSRF, Injection, etc. via Spring Security et bonnes pratiques frontend

Perspective d'évolution

Le projet **Traficandme** est conçu pour évoluer et s'adapter aux besoins des utilisateurs. L'équipe prévoit d'ajouter plusieurs correctifs et nouvelles fonctionnalités afin d'améliorer l'expérience globale :

Correctifs prévus :

- Intégration de nouvelles langues (espagnol, italien, portugais) pour élargir l'accessibilité à un public international.
- Amélioration de l'interface graphique afin d'offrir une navigation plus intuitive et agréable.

Fonctionnalités à venir :

- Authentification via des comptes Google (OAuth2) et Apple pour faciliter la connexion.
- Intégration de la reconnaissance faciale pour renforcer la sécurité et offrir une méthode de connexion moderne.
- Affichage d'historiques de trajets personnalisés.
- Système de gamification (points, badges, classements) pour encourager l'utilisation participative de l'application.
- Partage d'itinéraires entre utilisateurs.
- Intégration d'un assistant vocal pour une interaction sans les mains.
- Ajout d'un système de covoiturage

Ces évolutions ont pour objectif de rendre l'application plus complète, conviviale et compétitive face aux solutions de navigation existantes.

Conclusion

En conclusion, **Traficandme** est un projet d'équipe ambitieux et prometteur, porté par une volonté commune de proposer une solution innovante et utile à la mobilité urbaine. Grâce à une collaboration efficace entre les membres du groupe, nous avons su mettre en œuvre une architecture technique moderne et performante, combinant des technologies récentes telles que **React**, **Spring Boot**, **PostgreSQL**, **Docker** et **TomTom**.

L'ensemble du système repose sur une approche **microservices**, garantissant une meilleure évolutivité, une maintenance simplifiée et une robustesse accrue. L'authentification sécurisée **JWT** illustre notre attention portée à la sécurité des utilisateurs. Le projet a également été pensé pour être accessible sur le web et sur mobile, afin de répondre aux besoins d'un large public.

Notre équipe a su relever les défis techniques et organisationnels en mettant en place une gestion rigoureuse du code avec **GitHub**, un déploiement conteneurisé, et des interfaces conviviales. Nous avons aussi conçu notre application pour être facilement étendue grâce à une API REST bien structurée.

Avec des perspectives d'évolution claires — ajout de langues, nouvelles méthodes de connexion, améliorations UI/UX — **Traficandme** est une base solide pour un produit pérenne. Ce projet a renforcé notre esprit d'équipe, notre sens de l'initiative, et notre capacité à concevoir une solution complète répondant à des problématiques concrètes du quotidien.