Cahier des charges – Plateforme d'alerte communautaire "SafeCongo"

---

1. Contexte et objectifs

1.1 Contexte

En République Démocratique du Congo, certaines régions sont touchées par des conflits, des violences et des catastrophes naturelles. Les citoyens manquent d'outils fiables pour signaler rapidement ces incidents et alerter les autorités.

1.2 Objectifs

- Offrir aux citoyens un moyen simple et sécurisé de signaler incidents et dangers en temps réel
- Fournir aux autorités un tableau de bord complet pour gérer et suivre ces incidents
- Renforcer la sécurité publique et la réactivité des services de l'État
- Collecter des données fiables pour la planification et la prévention

---

2. Périmètre du projet

2.1 Public cible

- **Citoyens** : signaler des incidents et recevoir des alertes
- **Modérateurs / autorités locales** : valider, gérer et suivre les incidents
- **Administrateurs** : superviser l'ensemble de la plateforme

2.2 Fonctionnalités principales

Pour les citoyens :

- Création de compte et authentification
- Signalement d'incidents (texte, photo, géolocalisation)
- Visualisation des incidents sur une carte interactive
- Réception d'alertes locales par SMS, email ou notification push

Pour les modérateurs / autorités :

- Validation et gestion des signalements
- Assignation des incidents à des services compétents
- Visualisation des incidents par type, région et date
- Génération de rapports statistiques

Pour les administrateurs :

- Gestion des utilisateurs et rôles (citoyen, modérateur, admin)
- Modération des contenus
- Configuration des zones et types d'alertes

---

3. Architecture technique

3.1 Backend

- Framework : Symfony 6+
- ORM : Doctrine
- API : REST/GraphQL via API Platform
- Authentification : JWT (JSON Web Token)
- Notification asynchrone : Messenger + RabbitMQ
- Stockage fichiers : VichUploaderBundle pour photos

3.2 Frontend

- Dashboard administrateur : Twig + Bootstrap / Tailwind
- Carte interactive : Leaflet.js ou Mapbox
- Version mobile (optionnelle) : React Native ou Flutter consommant l'API

3.3 Base de données

- PostgreSQL (support géospatial) ou MySQL

3.4 Sécurité

- Authentification JWT avec rôles
- Validation côté serveur (Symfony Validator)
- Protection CSRF/XSS, chiffrement des données sensibles

3.5 Déploiement

- Docker pour environnement local et production
- Hébergement cloud (AWS, GCP ou Azure)
- Backups réguliers et monitoring

---

4. Contraintes et exigences

- **Performance** : traitement des alertes en temps réel, support de milliers d'utilisateurs simultanés
- **Sécurité** : chiffrement des données, protection contre attaques
- **Scalabilité** : possibilité d'ajouter de nouvelles fonctionnalités (IA, analytics)
- **Accessibilité** : interface multilingue (français + langues locales)
- **Compatibilité** : navigateurs web modernes et mobile

---

5. Livrables

1. Application web Symfony complète (API + dashboard)
2. Base de données et scripts de migration Doctrine
3. Documentation technique et utilisateur
4. Déploiement sur serveur de test et production
5. Prototype fonctionnel pour démonstration

---

6. Planning prévisionnel

Étape	Durée estimée	Description
Analyse & conception	1-2 semaines	Cahier des charges, architecture, maquettes
Développement Backend	4-6 semaines	API, modèles Doctrine, authentification
Développement Frontend	3-4 semaines	Dashboard, carte interactive, notifications
Tests & corrections	2 semaines	Unit tests, intégration, sécurité
Déploiement & formation	1 semaine	Serveur, Docker, formation utilisateurs

---

7. Budget indicatif

- Développement Symfony : selon tarification locale / freelance
- Hébergement et notifications : dépend de l'échelle et du cloud choisi
- Maintenance annuelle : mise à jour sécurité + support

---

8. Suivi et maintenance

- Tableau de bord d'administration pour monitorer incidents et utilisateurs
- Logs d'audit pour suivi et reporting
- Support technique et mises à jour régulières