Guide Expert Complet : Intégrer le JM-VG01U avec GPS-Track

Introduction : Les Fondamentaux IoT pour GPS Tracking

Ce guide vous donnera une compréhension complète du système de tracking GPS basé sur l'Internet des Objets (IoT). Nous couvrirons tous les aspects techniques nécessaires pour déployer un système fonctionnel, depuis le matériel jusqu'au logiciel.

Objectifs de ce guide :

- Comprendre les protocoles de communication GPS (GT06)
- Maîtriser l'installation et configuration du matériel
- Déployer la plateforme GPS-Track localement
- Diagnostiquer et résoudre les problèmes courants

PARTIE 1 : Architecture Technique Globale

1.1 Vue d'ensemble du système

Le système se compose de trois couches principales :

1. **Couche Matérielle (Hardware)** : Le tracker JM-VG01U
2. **Couche Communication** : Réseau GSM/GPRS + Protocole GT06
3. **Couche Application** : GPS-Track (plateforme Laravel)

```
[JM-VG01U] ──(GT06/TCP)──> [Réseau Mobile] ──> [GPS-Track Platform]
    ↓                                              ↓
[GPS + GSM]                                    [Base de Données]
```

1.2 Les technologies impliquées

GPS (Global Positioning System) :

- Système de satellites qui fournit la position (latitude, longitude, altitude)
- Temps d'acquisition initial : 1-3 minutes à froid
- Précision standard : 3-5 mètres

GPRS (General Packet Radio Service) :

- Extension du réseau GSM pour la transmission de données
- Permet l'envoi de données via TCP/IP
- Débit typique : 56-114 kbps

TCP (Transmission Control Protocol) :

- Protocole de transport fiable
- Garantit la livraison des paquets de données
- Utilisé par le protocole GT06

PARTIE 2 : Le Tracker JM-VG01U - Hardware et Fonctionnement

2.1 Spécifications techniques détaillées

Le JM-VG01U est un tracker avancé avec accéléromètre 3 axes et gyroscope 3 axes, capable de détecter 8 types de comportements de conduite inappropriés.

Composants intégrés :

- **Module GPS** : Récepteur multi-constellation (GPS, GLONASS, Galileo)
- **Module GSM/GPRS** : Communication cellulaire 2G/3G/4G
- **Accéléromètre 3 axes** : Détection des mouvements brusques
- **Gyroscope 3 axes** : Mesure de l'orientation et rotation
- **Microcontrôleur** : Traitement des données et communication

2.2 Système de navigation inertielle (INS)

Le système de navigation inertielle permet un accès stable à la localisation dans les zones de faible signal GPS comme les passages souterrains, centre-ville, tunnels, etc.

Principe de fonctionnement :

- En cas de perte du signal GPS, l'INS utilise les capteurs inertiels
- Calcule la position par rapport au dernier point GPS connu
- Maintient le tracking même dans les environnements difficiles

2.3 Analyse comportementale

Le tracker peut détecter 8 types de comportements :

1. **Accélération brusque** : > 2.5 m/s^2
2. **Freinage brusque** : < -2.5 m/s^2
3. **Virage serré à gauche** : > 0.35 g
4. **Virage serré à droite** : > 0.35 g
5. **Collision** : Impact > 4g
6. **Retournement** : Angle > 120°
7. **Remorquage** : Mouvement sans allumage
8. **Conduite fatiguée** : Analyse des patterns

2.4 Câblage et Installation

Schéma de câblage standard :

```
Rouge  (12-24V) ──> Batterie permanente (+)
Noir   (GND)    ──> Masse véhicule (-)
Jaune  (ACC)    ──> Contact/Allumage (+12V quand moteur ON)
Orange (IN1)    ──> Entrée digitale 1 (capteur optionnel)
Violet (OUT1)   ──> Sortie relais (coupure moteur)
```

Positions d'installation recommandées :

- À l'intérieur du véhicule (protection)

- Éviter les surfaces métalliques (interférence GPS)
- Accès facile pour la LED de statut
- Proche de la batterie pour l'alimentation

2.5 Indicateurs LED et diagnostics

LED	Couleur	État	Signification
PWR	Rouge	Fixe	Alimentation OK
GSM	Verte	Clignote 1x/sec	Connecté au réseau
GSM	Verte	Clignote 3x/sec	Recherche réseau
GPS	Bleue	Fixe	Signal GPS OK
GPS	Bleue	Clignote	Acquisition satellites

PARTIE 3 : Le Protocole GT06 - Communication Technique

3.1 Qu'est-ce que GT06 ?

Le GT06 est un protocole propriétaire développé par Concox/Jimi pour leurs trackers GPS. Il définit la structure des messages échangés entre le tracker et le serveur.

Caractéristiques du protocole :

- **Transport** : TCP/IP (port standard 5023)
- **Format** : Binaire (non ASCII)
- **Authentification** : Par IMEI du device
- **Keep-alive** : Heartbeat toutes les 30 secondes par défaut

3.2 Structure des trames GT06

En-tête de trame :

```
[Start Bits][Length][Protocol][Information Content][Serial][Error Check][Stop Bits]
    2 bytes   1 byte   1 byte      N bytes          2 bytes   2 bytes    2 bytes
```

Exemple de trame de position :

```
78 78 22 22 0F 0C 1D 02 33 05 C9 02 7A C8 18 0C 46 58 60 00 14 00 01 CC 00 28 7D 00 1F 71 0D 0A
```

3.3 Types de messages GT06

Code	Type	Description
0x01	Login	Authentification avec IMEI
0x12	Position	Données GPS + état du véhicule
0x13	Heartbeat	Message de maintien de connexion
0x16	Alarm	Alertes (SOS, vitesse, géofence)
0x80	Command	Réponse aux commandes serveur

3.4 Décodage des données de position

Structure des données GPS dans GT06 :

```
Date/Time    : 6 bytes (YY MM DD HH MM SS)
GPS Info     : 1 byte (satellites + fix status)
Latitude     : 4 bytes (format ±DDMM.MMMM)
Longitude    : 4 bytes (format ±DDDMM.MMMM)
Speed        : 1 byte (km/h)
Course       : 2 bytes (degrés 0-360)
MCC/MNC      : 2 bytes (code pays/opérateur)
LAC          : 2 bytes (zone de localisation)
Cell ID      : 3 bytes (identifiant cellule GSM)
```

PARTIE 4 : Configuration du JM-VG01U par SMS

4.1 Principe des commandes SMS

Le tracker peut être configuré à distance via SMS. Chaque commande suit une syntaxe précise et retourne une confirmation.

Format général : `COMMANDE,param1,param2,...#`

4.2 Configuration réseau et APN (Burkina Faso)

Qu'est-ce qu'un APN ?
L'APN (Access Point Name) est l'adresse qui permet au tracker de se connecter au réseau de données de l'opérateur mobile.

Configuration pour Moov Africa Burkina Faso :

```
APN,internet#                    // Configuration Moov BF
```

Autres APN Burkina Faso :

```
APN,internet#                    // Moov Africa BF (recommandé)
APN,orange.bf#                   // Orange Burkina (si disponible)
APN,telecel.bf#                  // Télécel Burkina (si disponible)
```

Vérification de la configuration :

```
APN#                             // Voir APN actuel
PARAM#                           // Voir toute la config
CSQ#                             // Force du signal GSM
```

Important pour Burkina Faso :

- Vérifiez que votre forfait Moov inclut la data/internet mobile
- Le crédit doit être suffisant pour les transmissions GPRS
- En cas de problème, contactez le service client Moov : **4040**

4.3 Configuration serveur et test Burkina Faso

Configuration de base :

```
IP,1,192.168.1.100,5023#         // IP locale pour test
SERVER,1,tracker.mondomaine.fr,5023#  // Domaine pour production
```

Exemple complet pour Moov Africa BF :

```
APN,internet#                     // APN Moov Burkina
IP,1,192.168.1.100,5023#        // Votre serveur local
TIMER,60,120#                     // Économiser la data (1min/2min)
ADMIN,+22670123456#               // Numéro BF (+226 + numéro)
PARAM#                            // Vérifier la config
```

Spécificités Burkina Faso :

- Format numéro : **+226 + 8 chiffres** (ex: +22670123456)
- Coût GPRS : Optimiser avec `TIMER,60,180#` pour réduire la consommation
- Signal réseau variable selon zone géographique

4.4 Gestion des intervalles et conditions

Intervalles de position :

```
TIMER,30,30#                      // 30 sec en mouvement, 30 sec à l'arrêt
TIMER,10,60#                      // 10 sec en mouvement, 60 sec à l'arrêt
```

Conditions avancées :

```
ANGLE,30#                         // Envoyer si changement direction >30°
DISTANCE,200#                     // Envoyer si déplacement >200m
SPEED,10#                         // Envoyer si changement vitesse >10km/h
```

4.5 Configuration des alertes

Numéros d'urgence :

```
ADMIN,+33123456789#               // Ajouter admin principal
NOADMIN,+33123456789#             // Supprimer admin
ADMIN#                            // Voir liste admins
```

Types d'alertes :

```
SPEED,80#                         // Alerte si vitesse >80 km/h
SHOCK,1#                          // Activer alerte choc/collision
POWER,1#                          // Alerte coupure alimentation
LOW,11.5#                         // Alerte batterie <11.5V
```

4.6 Commandes de diagnostic

État du système :

```
STATUS#                          // État GPS, GSM, batterie
IMEI#                           // Numéro IMEI
VERSION#                        // Version firmware
BAT#                            // Niveau batterie
```

Position manuelle :

```
URL#                            // Recevoir lien Google Maps
G123456#                        // Position par mot de passe
```

PARTIE 5 : GPS-Track - Plateforme Laravel

5.1 Analyse de la plateforme GPS-Track

GPS-Track est une plateforme open source de gestion de dispositifs GPS développée avec Laravel 12, PHP 8.2 et MySQL 8. Cette solution complète de tracking GPS offre des performances robustes et une interface utilisateur intuitive.

Protocoles supportés :

- **Sinotrack** : Modèles ST-90X confirmés
- **Coban** : Modèle TK303G avec protocole GP103
- **Teltonika** : Via TCP avec protocole Teltonika
- **Concox/JimiLab** : Modèle JM-LL01 via protocole GT06
- **Queclink** : Modèle GV500MA
- **OsmAnd** : Via protocole HTTP
- **Autres** : iTriangle/Aquila, TKStar

5.2 Architecture technique Laravel

Stack technologique :

- **Backend** : Laravel 12 (framework PHP)
- **Base de données** : MySQL 8.0.12+ (avec support GIS)
- **Cache** : Redis (sessions, queues)
- **Frontend** : Blade templates + JavaScript
- **Cartes** : OpenStreetMap/Leaflet

Structure MVC (Model-View-Controller) :

```
app/
├── Models/           # Entités (Device, Trip, Position)
├── Controllers/      # Logique métier
├── Services/         # Services (Protocol handlers)
└── Console/          # Commandes artisan

resources/views/      # Templates Blade
public/              # Assets (JS, CSS, images)
database/migrations/  # Structure base de données
```

5.3 Gestion des protocoles

Service de protocole GT06 :
Le système utilise des services dédiés pour chaque protocole. Pour GT06 :

```
// Exemple de structure (simplifié)
class GT06ProtocolService
{
    public function decode($rawData)
    {
        // Décodage de la trame binaire GT06
        // Extraction GPS, état véhicule, etc.
    }

    public function sendCommand($device, $command)
    {
        // Envoi commandes au tracker
    }
}
```

5.4 Fonctionnalités de GPS-Track

Tracking en temps réel :

- Visualisation live sur carte
- Mise à jour automatique des positions
- Indicateurs de statut (en ligne/hors ligne)

Rapports et historiques :

- Historique complet des trajets
- Rapports de vitesse, distance, temps
- Analyses de comportement de conduite

Système d'alertes :

- Géofencing (entrée/sortie de zones)
- Alertes de vitesse, mouvement
- Notifications Telegram configurables

Gestion multi-utilisateurs :

- Comptes avec différents niveaux d'accès
- Attribution de devices par utilisateur
- Partage public de trajets (optionnel)

PARTIE 6 : Installation et Configuration

6.1 Prérequis système détaillés

Système d'exploitation :

- Ubuntu 20.04+ / Debian 11+ / CentOS 8+
- 2GB RAM minimum (4GB recommandé)
- 10GB espace disque libre

Logiciels requis :

```
# PHP 8.2 avec extensions
sudo apt update
sudo apt install php8.2 php8.2-{bcmath,bz2,intl,mbstring,opcache,pdo-mysql,pcntl,redis,sock

# MySQL 8.0
sudo apt install mysql-server-8.0

# Redis
sudo apt install redis-server

# Composer (gestionnaire PHP)
curl -sS https://getcomposer.org/installer | php
sudo mv composer.phar /usr/local/bin/composer
```

6.2 Installation pas à pas

Étape 1 : Clonage et dépendances

```
# Cloner le projet
git clone https://github.com/Zizouk22/GPS-Track.git
cd GPS-Track

# Installer les dépendances PHP
composer install --no-dev --optimize-autoloader

# Permissions sur les dossiers Laravel
sudo chown -R www-data:www-data storage bootstrap/cache
sudo chmod -R 755 storage bootstrap/cache
```

Étape 2 : Configuration environnement

```
# Copier le fichier d'environnement
cp .env.example .env

# Générer la clé de l'application
php artisan key:generate
```

Étape 3 : Configuration .env

```
# Application
APP_NAME="GPS-Track"
APP_ENV=production
APP_DEBUG=false
APP_URL=http://localhost:8000

# Base de données
```

```
DB_CONNECTION=mysql
DB_HOST=127.0.0.1
DB_PORT=3306
DB_DATABASE=gps_track
DB_USERNAME=gps_user
DB_PASSWORD=motdepasse_secure

# Redis
REDIS_HOST=127.0.0.1
REDIS_PASSWORD=null
REDIS_PORT=6379

# Serveurs de protocoles (ports d'écoute)
PROTOCOL_GT06_PORT=5023
PROTOCOL_TELTONIKA_PORT=5024
PROTOCOL_OSMAND_PORT=5055
```

Étape 4 : Base de données

```
# Créer la base de données
mysql -u root -p -e "CREATE DATABASE gps_track CHARACTER SET utf8mb4 COLLATE utf8mb4_unicod
mysql -u root -p -e "CREATE USER 'gps_user'@'localhost' IDENTIFIED BY 'motdepasse_secure';"
mysql -u root -p -e "GRANT ALL PRIVILEGES ON gps_track.* TO 'gps_user'@'localhost';"

# Exécuter les migrations
php artisan migrate

# Créer un utilisateur admin
php artisan user:create --email=admin@example.com --password=admin123 --admin
```

6.3 Configuration des services de protocole

Service systemd pour GT06 :

```
# Créer le fichier service
sudo nano /etc/systemd/system/gps-track-gt06.service
```

Contenu du fichier service :

```
[Unit]
Description=GPS-Track GT06 Protocol Service
After=mysql.service redis.service

[Service]
Type=simple
User=www-data
Group=www-data
WorkingDirectory=/var/www/GPS-Track
ExecStart=/usr/bin/php artisan protocol:gt06
Restart=always
RestartSec=3
```

```
[Install]
WantedBy=multi-user.target
```

Activation du service :

```
sudo systemctl daemon-reload
sudo systemctl enable gps-track-gt06
sudo systemctl start gps-track-gt06
sudo systemctl status gps-track-gt06
```

6.4 Configuration Nginx (optionnel)

```
server {
    listen 80;
    server_name gps.mondomaine.fr;
    root /var/www/GPS-Track/public;
    index index.php;

    location / {
        try_files $uri $uri/ /index.php?$query_string;
    }

    location ~ \.php$ {
        fastcgi_pass unix:/var/run/php/php8.2-fpm.sock;
        fastcgi_index index.php;
        fastcgi_param SCRIPT_FILENAME $realpath_root$fastcgi_script_name;
        include fastcgi_params;
    }
}
```

PARTIE 7 : Test et Validation du Système

7.1 Vérifications préliminaires

Test de la carte SIM :

1. Insérer dans un téléphone
2. Vérifier crédit et forfait data
3. Tester envoi/réception SMS
4. Vérifier l'APN opérateur

Test du tracker :

1. Alimentation : LED rouge fixe
2. GSM : LED verte clignotante après 1-2 min
3. GPS : LED bleue fixe en extérieur après 2-3 min

7.2 Configuration initiale du tracker

Séquence de configuration recommandée :

```
# 1. Réinitialiser (optionnel)
FACTORY#

# 2. Configurer APN (adapter selon opérateur)
APN,internet#

# 3. Configurer serveur (IP de votre machine)
IP,1,192.168.1.100,5023#

# 4. Configurer intervalles
TIMER,30,60#

# 5. Ajouter numéro admin
ADMIN,+33612345678#

# 6. Vérifier configuration
PARAM#
```

7.3 Tests de connectivité

Côté serveur (GPS-Track) :

```
# Vérifier que le service écoute sur le port
sudo netstat -tlnp | grep 5023

# Surveiller les logs en temps réel
tail -f storage/logs/laravel.log | grep GT06

# Test de connexion TCP basique
telnet localhost 5023
```

Côté tracker :

```
# Demander position actuelle
URL#

# Vérifier statut système
STATUS#

# Forcer envoi position
G123456#  # (si mot de passe = 123456)
```

7.4 Validation des données

Interface web GPS-Track :

1. Connexion admin : http://localhost:8000
2. Ajouter device avec IMEI du tracker
3. Vérifier statut "Online" dans la liste
4. Consulter la dernière position reçue
5. Vérifier l'historique des positions

Données à vérifier :

- **Position GPS** : Précision ±5 mètres
- **Vitesse** : Cohérence avec déplacement réel
- **Direction** : Orientation correcte (0-360°)
- **Timestamp** : Heure UTC correcte
- **Statut** : ACC ON/OFF selon allumage

PARTIE 8 : Diagnostic et Résolution de Problèmes

8.1 Problèmes de connectivité GPS

Symptômes : LED GPS bleue clignote ou éteinte
Causes possibles :

- Position en intérieur ou sous obstacle
- Antenne GPS endommagée
- Interférence électromagnétique

Solutions :

1. Déplacer en extérieur dégagé
2. Attendre 5-10 minutes pour acquisition
3. Vérifier position antenne (éloignée des masses métalliques)
4. Redémarrer : `RESTART#`

8.2 Problèmes de réseau GSM

Symptômes : LED GSM clignote rapidement ou éteinte
Diagnostic :

```
CSQ#        # Force du signal (>10 = bon)
STATUS#     # État général du système
```

Solutions courantes :

- Vérifier crédit/forfait SIM
- Tester SIM dans téléphone
- Changer d'antenne GSM
- Vérifier APN opérateur

8.3 Problèmes de communication serveur

Symptômes : Tracker online mais pas de données GPS-Track

Diagnostic serveur :

```
# Vérifier service GT06 actif
sudo systemctl status gps-track-gt06

# Vérifier port ouvert
```

```
sudo ss -tlnp | grep 5023

# Logs Laravel
tail -f storage/logs/laravel.log

# Connexions TCP actives
sudo netstat -an | grep 5023
```

Diagnostic tracker :

```
PARAM#      # Vérifier IP/port serveur
IP,1,VOTRE_IP,5023#  # Reconfigurer si nécessaire
```

8.4 Problèmes de données incorrectes

Position fixe ou erronée :

```
RESET#      # Réinitialiser GPS
FACTORY#    # Reset complet (perdra la config)
```

Données de mouvement incohérentes :

- Vérifier montage du tracker (orientation)
- Calibrer accéléromètre : `SENSOR,1#`
- Ajuster seuils : `MOVE,100#` (seuil mouvement)

8.5 Optimisation des performances

Réduction consommation data :

```
TIMER,60,300#       # 1 min en mouvement, 5 min arrêt
ANGLE,45#           # Position si changement >45°
DISTANCE,500#       # Position si déplacement >500m
```

Amélioration précision :

```
ACCURACY,50#        # Précision GPS requise <50m
FILTER,1#           # Filtre positions aberrantes
```

PARTIE 9 : Monitoring et Maintenance

9.1 Surveillance système

Métriques importantes :

- Uptime des services GT06
- Utilisation mémoire/CPU
- Espace disque base de données

- Nombre de devices connectés

Script de monitoring :

```
#!/bin/bash
# check_gps_track.sh
SERVICE_STATUS=$(systemctl is-active gps-track-gt06)
MYSQL_STATUS=$(systemctl is-active mysql)
REDIS_STATUS=$(systemctl is-active redis)

echo "GT06 Service: $SERVICE_STATUS"
echo "MySQL: $MYSQL_STATUS"
echo "Redis: $REDIS_STATUS"

# Vérifier port GT06
if netstat -tlnp | grep -q ":5023 "; then
    echo "GT06 Port: LISTENING"
else
    echo "GT06 Port: ERROR - Not listening"
fi
```

9.2 Sauvegarde et maintenance

Sauvegarde base de données :

```
# Sauvegarde quotidienne
mysqldump -u gps_user -p gps_track > backup_$(date +%Y%m%d).sql

# Automatisation via crontab
0 2 * * * /usr/bin/mysqldump -u gps_user -pmotdepasse gps_track > /backup/gps_track_$(date
```

Nettoyage périodique :

```
# Nettoyer anciennes positions (>90 jours)
php artisan position:clean --days=90

# Optimiser base de données
php artisan db:optimize

# Nettoyer cache Laravel
php artisan cache:clear
php artisan config:clear
```

PARTIE 10 : Sécurité et Bonnes Pratiques

10.1 Sécurisation du tracker

Changement mots de passe :

```
PASSWORD,123456,NOUVEAU_MDP#     # Changer mot de passe par défaut
ADMIN,OLD_ADMIN#                 # Supprimer anciens admins
```

Restrictions d'accès :

```
MONITOR,+33612345678#            # Seul ce numéro peut configurer
NOMONITOR,+33987654321#          # Retirer autorisation
```

10.2 Sécurisation serveur

Firewall iptables :

```
# Autoriser seulement le trafic GPS sur port 5023
sudo iptables -A INPUT -p tcp --dport 5023 -s 0.0.0.0/0 -j ACCEPT
sudo iptables -A INPUT -p tcp --dport 80 -j ACCEPT    # HTTP
sudo iptables -A INPUT -p tcp --dport 443 -j ACCEPT   # HTTPS
sudo iptables -A INPUT -p tcp --dport 22 -j ACCEPT    # SSH
sudo iptables -P INPUT DROP  # Bloquer le reste
```

HTTPS avec Let's Encrypt :

```
sudo apt install certbot python3-certbot-nginx
sudo certbot --nginx -d gps.mondomaine.fr
```

10.3 Chiffrement des communications

Important : Le protocole GT06 n'est pas chiffré par défaut. Pour une sécurité renforcée :

- Utiliser un VPN entre tracker et serveur
- Implémenter SSL/TLS sur les ports de protocole
- Chiffrer la base de données MySQL

PARTIE 11 : Extensions et Personnalisations

11.1 API REST GPS-Track

La plateforme fournit une API REST complète :

Endpoints principaux :

```
GET /api/devices           # Liste des trackers
GET /api/device/{id}/positions  # Positions d'un device
GET /api/trips           # Historique trajets
POST /api/device/{id}/command   # Envoyer commande
```

Exemple utilisation :

```
curl -H "Authorization: Bearer TOKEN" \
     http://localhost:8000/api/devices
```

11.2 Notifications Telegram

Configuration dans GPS-Track :

1. Créer bot Telegram (@BotFather)
2. Récupérer token bot
3. Configurer dans l'interface admin
4. Associer chat_id aux alertes

11.3 Intégrations possibles

Webhooks : Recevoir événements temps réel
MQTT : Publication positions IoT
WebSocket : Tracking live en JavaScript
Mobile App : APIs pour applications mobiles

PARTIE 12 : Cas d'Usage Avancés

12.1 Gestion de flotte

Configuration multi-véhicules :

- Groupes de vehicles par service
- Alertes spécifiques par type
- Rapports consolidés

Exemple configuration camion :

```
TIMER,20,120#       # 20s en route, 2min arrêt
SPEED,90#           # Alerte >90 km/h
IDLE,300#           # Alerte ralenti >5 min
FUEL,1#             # Monitoring carburant
```

12.2 Assurance télématique (UBI)

Le Concox JM-VG01U est un tracker GPS d'assurance pour applications UBI (Usage-Based Insurance).

Métriques UBI collectées :

- Distance parcourue
- Vitesses pratiquées
- Horaires de conduite
- Comportements de conduite (8 types)
- Zones fréquentées

12.3 Sécurité et antivol

Configuration antivol :

```
SHOCK,1#            # Alerte choc/vibration
POWER,1#            # Alerte coupure batterie
MOVE,100#           # Alerte mouvement >100m sans allumage
FENCE,IN,45.1234,2.5678,500#  # Géofence (lat,lon,rayon)
SOS,1#              # Activation bouton panique
```

Mode discret :

```
LED,0#              # Éteindre toutes les LED
CALL,0#             # Désactiver les appels automatiques
SMS,0#              # Mode silencieux SMS
```

PARTIE 13 : Tableaux de Référence Technique

13.1 Codes d'erreur GT06

Code	Description	Action
E001	GPS non acquis	Déplacer à l'extérieur, attendre
E002	Échec connexion serveur	Vérifier IP/port, réseau GSM
E003	APN incorrect	Reconfigurer APN opérateur
E004	Batterie faible	Vérifier alimentation véhicule
E005	Module GSM défaillant	Redémarrer ou remplacer SIM

13.2 Correspondance LED/États

État Système	PWR	GSM	GPS	Action
Démarrage	Rouge fixe	Éteinte	Éteinte	Normal, attendre
Recherche réseau	Rouge fixe	Vert clignotant rapide	Éteinte	Attendre signal GSM
Connecté réseau	Rouge fixe	Vert clignotant lent	Éteinte	Sortir pour GPS
Acquisition GPS	Rouge fixe	Vert clignotant lent	Bleu clignotant	Attendre fix GPS
Opérationnel	Rouge fixe	Vert clignotant lent	Bleu fixe	Système OK
Erreur GPS	Rouge fixe	Vert clignotant lent	Éteinte	Problème antenne/position
Erreur GSM	Rouge fixe	Éteinte	Bleu variable	Problème SIM/réseau

13.3 APN par opérateur (France)

Opérateur	APN	Utilisateur	Mot de passe	Commande SMS
Orange	orange.fr	orange	orange	`APN,orange.fr,orange,orange#`
SFR	sl2sfr	(vide)	(vide)	`APN,sl2sfr#`

Opérateur	APN	Utilisateur	Mot de passe	Commande SMS
Bouygues	mmsbouygtel.com	(vide)	(vide)	`APN,mmsbouygtel.com#`
Free Mobile	free	(vide)	(vide)	`APN,free#`
Sosh	internet	(vide)	(vide)	`APN,internet#`
Moov Africa BF	**internet**	**(vide)**	**(vide)**	**`APN,internet#`**

13.4 Ports protocoles GPS-Track

Protocole	Port par défaut	Variable .env	Usage
GT06	5023	PROTOCOL_GT06_PORT	Concox, Jimi
GP103	5001	PROTOCOL_GP103_PORT	Coban
Teltonika	5024	PROTOCOL_TELTONIKA_PORT	Teltonika
OsmAnd	5055	PROTOCOL_OSMAND_PORT	OsmAnd app
Queclink	2101	PROTOCOL_QUECLINK_PORT	Queclink

PARTIE 14 : Scripts d'Automatisation

14.1 Script de test de connectivité

```
#!/bin/bash
# test_tracker_connectivity.sh

TRACKER_IMEI="$1"
SERVER_IP="$2"
SERVER_PORT="5023"

if [ $# -lt 2 ]; then
    echo "Usage: $0 <IMEI> <SERVER_IP>"
    exit 1
fi

echo "=== Test de connectivité tracker $TRACKER_IMEI ==="

# Test 1: Port serveur ouvert
echo "1. Test port serveur..."
if timeout 5 bash -c "</dev/tcp/$SERVER_IP/$SERVER_PORT"; then
    echo "✓ Port $SERVER_PORT ouvert sur $SERVER_IP"
else
    echo "✗ Port $SERVER_PORT fermé ou inaccessible"
    exit 1
fi

# Test 2: Vérifier service GT06
echo "2. Test service GT06..."
if systemctl is-active --quiet gps-track-gt06; then
    echo "✓ Service GT06 actif"
```

```
else
    echo "✗ Service GT06 inactif"
    echo "Redémarrage du service..."
    sudo systemctl restart gps-track-gt06
fi

# Test 3: Connexions actives
echo "3. Connexions actives sur port $SERVER_PORT:"
netstat -an | grep ":$SERVER_PORT" | head -5

# Test 4: Derniers logs
echo "4. Derniers logs GT06:"
tail -10 storage/logs/laravel.log | grep GT06

echo "=== Test terminé ==="
```

14.2 Script de configuration batch tracker

```
#!/bin/bash
# configure_tracker.sh - Configuration automatique par SMS

TRACKER_PHONE="$1"
SERVER_IP="$2"
ADMIN_PHONE="$3"

if [ $# -lt 3 ]; then
    echo "Usage: $0 <PHONE_TRACKER> <SERVER_IP> <ADMIN_PHONE>"
    echo "Exemple: $0 +33123456789 192.168.1.100 +33987654321"
    exit 1
fi

echo "Configuration du tracker $TRACKER_PHONE"

# Fonction d'envoi SMS (nécessite gammu ou équivalent)
send_sms() {
    local phone="$1"
    local message="$2"
    echo "Envoi SMS à $phone: $message"
    # echo "$message" | gammu sendsms TEXT "$phone"
    # ou utiliser une API SMS
    echo "SMS simulé: $message"
    sleep 2
}

# Séquence de configuration
echo "1. Réinitialisation (optionnelle)"
# send_sms "$TRACKER_PHONE" "FACTORY#"
# sleep 30

echo "2. Configuration APN"
send_sms "$TRACKER_PHONE" "APN,internet#"

echo "3. Configuration serveur"
```

```
send_sms "$TRACKER_PHONE" "IP,1,$SERVER_IP,5023#"

echo "4. Intervalles de position"
send_sms "$TRACKER_PHONE" "TIMER,30,60#"

echo "5. Numéro admin"
send_sms "$TRACKER_PHONE" "ADMIN,$ADMIN_PHONE#"

echo "6. Alertes de base"
send_sms "$TRACKER_PHONE" "SPEED,80#"
send_sms "$TRACKER_PHONE" "POWER,1#"

echo "7. Vérification configuration"
send_sms "$TRACKER_PHONE" "PARAM#"

echo "Configuration terminée. Vérifiez les SMS de confirmation."
```

14.3 Script de monitoring système

```
#!/bin/bash
# monitor_gps_track.sh

LOG_FILE="/var/log/gps-track-monitor.log"
ALERT_EMAIL="admin@mondomaine.fr"

log_message() {
    echo "$(date '+%Y-%m-%d %H:%M:%S') - $1" | tee -a "$LOG_FILE"
}

check_service() {
    local service="$1"
    if systemctl is-active --quiet "$service"; then
        log_message "✓ $service actif"
        return 0
    else
        log_message "✗ $service inactif - redémarrage"
        systemctl restart "$service"
        return 1
    fi
}

check_port() {
    local port="$1"
    if ss -tlnp | grep -q ":$port "; then
        log_message "✓ Port $port ouvert"
        return 0
    else
        log_message "✗ Port $port fermé"
        return 1
    fi
}

check_disk_space() {
```

```
    local usage=$(df -h /var/lib/mysql | awk 'NR==2 {print $5}' | sed 's/%//')
    if [ "$usage" -gt 80 ]; then
        log_message "   Espace disque MySQL: ${usage}%"
        # Nettoyage automatique des anciennes positions
        cd /var/www/GPS-Track && php artisan position:clean --days=30
    else
        log_message "✓ Espace disque OK: ${usage}%"
    fi
}

check_connections() {
    local active_connections=$(ss -tn | grep ":5023" | wc -l)
    log_message "Connexions GT06 actives: $active_connections"
}

# Exécution des vérifications
log_message "=== Début monitoring GPS-Track ==="

check_service "gps-track-gt06"
check_service "mysql"
check_service "redis-server"
check_service "nginx"

check_port "5023"
check_port "80"

check_disk_space
check_connections

# Vérifier si des trackers sont connectés
cd /var/www/GPS-Track
online_devices=$(php artisan device:count --online)
log_message "Devices en ligne: $online_devices"

log_message "=== Fin monitoring ==="
```

PARTIE 15 : Optimisations de Performance

15.1 Optimisation base de données

Configuration MySQL pour GPS-Track :

```
-- /etc/mysql/mysql.conf.d/gps-track.cnf
[mysqld]
# Optimisations pour données GPS
innodb_buffer_pool_size = 1G
innodb_log_file_size = 256M
innodb_flush_log_at_trx_commit = 2
innodb_flush_method = O_DIRECT

# Index spatiaux pour coordonnées GPS
ft_min_word_len = 3
```

```
ft_boolean_syntax = ' +-><()~*:""&|'

# Connexions simultanées
max_connections = 200
thread_cache_size = 50
table_open_cache = 2000
```

Index de performance pour positions :

```
-- Optimisation table positions
CREATE INDEX idx_positions_device_datetime ON positions (device_id, created_at);
CREATE INDEX idx_positions_coords ON positions (latitude, longitude);
CREATE SPATIAL INDEX idx_positions_point ON positions (point);

-- Partitionnement par mois (pour gros volumes)
ALTER TABLE positions PARTITION BY RANGE (YEAR(created_at) * 100 + MONTH(created_at))
(
    PARTITION p202401 VALUES LESS THAN (202402),
    PARTITION p202402 VALUES LESS THAN (202403),
    -- etc...
);
```

15.2 Optimisation Laravel

Configuration cache Redis :

```
// config/database.php - Configuration Redis optimisée
'redis' => [
    'client' => 'predis',
    'default' => [
        'host' => '127.0.0.1',
        'port' => 6379,
        'database' => 0,
        'options' => [
            'serialization' => 'php',
            'compression' => 'gzip',
        ],
    ],
    'cache' => [
        'host' => '127.0.0.1',
        'port' => 6379,
        'database' => 1,
    ],
],
```

Optimisation des queues :

```
# Configuration queues pour traitement asynchrone
php artisan queue:work --queue=positions,alerts,high --timeout=60 --daemon

# Service systemd pour queue worker
```

```
sudo systemctl enable gps-track-queue
sudo systemctl start gps-track-queue
```

15.3 Gestion mémoire et ressources

Configuration PHP optimisée :

```
; /etc/php/8.2/fpm/pool.d/gps-track.conf
[gps-track]
user = www-data
group = www-data

listen = /run/php/php8.2-fpm-gps-track.sock
listen.owner = www-data
listen.group = www-data

pm = dynamic
pm.max_children = 20
pm.start_servers = 5
pm.min_spare_servers = 5
pm.max_spare_servers = 10
pm.max_requests = 1000

; Optimisations mémoire
php_admin_value[memory_limit] = 256M
php_admin_value[max_execution_time] = 300
php_admin_value[post_max_size] = 50M
php_admin_value[upload_max_filesize] = 50M
```

PARTIE 16 : Troubleshooting Avancé

16.1 Analyse des trames GT06

Outil de debug des trames :

```
#!/bin/bash
# debug_gt06_frames.sh

# Capturer le trafic sur port 5023
sudo tcpdump -i any -X port 5023 > gt06_capture.txt &
TCPDUMP_PID=$!

echo "Capture démarrée (PID: $TCPDUMP_PID)"
echo "Demandez au tracker d'envoyer une position..."
echo "Appuyez sur Entrée pour arrêter la capture"
read

sudo kill $TCPDUMP_PID

# Analyse basique des trames
```

```
echo "=== Analyse des trames capturées ==="
grep -A 10 "78 78" gt06_capture.txt | head -20
```

Décodeur hexadécimal simple :

```
#!/usr/bin/env python3
# decode_gt06.py - Décodeur basique GT06

import struct
import datetime

def decode_gt06_position(hex_data):
    """Décode une trame GT06 de position basique"""
    try:
        # Conversion hex vers bytes
        data = bytes.fromhex(hex_data.replace(' ', ''))

        # Vérification en-tête (78 78)
        if data[:2] != b'\x78\x78':
            print("En-tête GT06 invalide")
            return

        # Extraction des composants
        length = data[2]
        protocol = data[3]

        if protocol == 0x22:  # Position data
            # Date/Time (6 bytes)
            dt_data = data[4:10]
            year = 2000 + dt_data[0]
            month = dt_data[1]
            day = dt_data[2]
            hour = dt_data[3]
            minute = dt_data[4]
            second = dt_data[5]

            # GPS Info
            gps_info = data[10]
            satellites = (gps_info >> 4) & 0x0F

            # Coordonnées (8 bytes)
            lat_raw = struct.unpack('>I', data[11:15])[0]
            lon_raw = struct.unpack('>I', data[15:19])[0]

            latitude = lat_raw / 1800000.0
            longitude = lon_raw / 1800000.0

            # Vitesse
            speed = data[19]

            # Affichage
            print(f"Date: {year:04d}-{month:02d}-{day:02d} {hour:02d}:{minute:02d}:{second:
            print(f"Satellites: {satellites}")
            print(f"Position: {latitude:.6f}, {longitude:.6f}")
```

```
            print(f"Vitesse: {speed} km/h")

    except Exception as e:
        print(f"Erreur décodage: {e}")

# Exemple d'utilisation
if __name__ == "__main__":
    # Trame exemple (à remplacer par capture réelle)
    sample_frame = "78 78 22 22 0F 0C 1D 02 33 05 C9 02 7A C8 18 0C 46 58 60 00 14 00 01 CC
    decode_gt06_position(sample_frame)
```

16.2 Diagnostics réseau avancés

Test de latence GPS-Track :

```
#!/bin/bash
# test_latency.sh

SERVER="localhost"
PORT="5023"

echo "=== Test latence réseau GPS-Track ==="

# Test ping basique
echo "1. Test ping serveur:"
ping -c 5 $SERVER

# Test ouverture port TCP
echo -e "\n2. Test connexion TCP port $PORT:"
time timeout 5 bash -c "</dev/tcp/$SERVER/$PORT" && echo "Connexion réussie" || echo "Échec

# Test charge serveur
echo -e "\n3. Charge système:"
uptime
free -h
df -h /var/lib/mysql

# Connexions actives GT06
echo -e "\n4. Connexions GT06 actives:"
ss -tn sport = :5023 | wc -l

# Processus Laravel
echo -e "\n5. Processus Laravel actifs:"
ps aux | grep "artisan\|php-fpm" | wc -l
```

16.3 Résolution problèmes spécifiques

Problème : Positions dupliquées

```
-- Nettoyer positions dupliquées
DELETE p1 FROM positions p1
INNER JOIN positions p2
```

```
WHERE p1.id > p2.id
  AND p1.device_id = p2.device_id
  AND p1.latitude = p2.latitude
  AND p1.longitude = p2.longitude
  AND ABS(TIMESTAMPDIFF(SECOND, p1.created_at, p2.created_at)) < 5;
```

Problème : Mémoire PHP saturée

```
# Surveiller utilisation mémoire
watch 'ps aux | grep php | awk "{sum+=\$6} END {print \"Mémoire PHP totale: \" sum/1024 \" M
```

```
# Redémarrer PHP-FPM si nécessaire
sudo systemctl reload php8.2-fpm
```

Problème : Base de données corrompue

```
# Vérification intégrité MySQL
sudo mysqlcheck -u root -p --check --all-databases
```

```
# Réparation si nécessaire
sudo mysqlcheck -u root -p --auto-repair --all-databases
```

```
# Optimisation tables
sudo mysqlcheck -u root -p --optimize gps_track
```

PARTIE 17 : Conclusion et Ressources

17.1 Récapitulatif des points clés

Configuration minimale fonctionnelle :

1. **Hardware** : JM-VG01U + SIM avec data
2. **Software** : GPS-Track sur serveur local
3. **Configuration tracker** : APN + IP serveur + Timer
4. **Validation** : LEDs OK + données dans GPS-Track

Commandes SMS essentielles :

```
APN,internet#                    # Configuration réseau
IP,1,192.168.1.100,5023#         # Serveur local
TIMER,30,60#                     # Fréquence positions
ADMIN,+33123456789#              # Numéro administrateur
PARAM#                           # Vérification config
```

Vérifications système :

```
# Services actifs
sudo systemctl status gps-track-gt06
sudo systemctl status mysql
```

```
# Port ouvert
sudo ss -tlnp | grep 5023

# Logs temps réel
tail -f storage/logs/laravel.log | grep GT06
```

17.2 Évolutions possibles

Phase 1 - Test local :

- Installation GPS-Track locale
- Configuration tracker basique
- Validation des données GPS

Phase 2 - Production :

- Serveur dédié avec nom de domaine
- HTTPS/SSL
- Sauvegarde automatisée
- Monitoring avancé

Phase 3 - Extensions :

- API personnalisée
- Notifications Telegram
- Rapports automatisés
- Multi-tenant

17.3 Ressources et documentation

GPS-Track Project :

- GitHub : https://github.com/Zizouk22/GPS-Track
- Demo : https://tracker-demo.lito.com.es/

Protocoles GPS :

- GT06 Protocol Specification
- Laravel 12 Documentation
- MySQL 8 GIS Functions

Outils utiles :

- Wireshark : Analyse trafic réseau
- MySQL Workbench : Gestion base de données
- Postman : Test API REST

17.4 Support et communauté

En cas de problème :

1. Vérifiez les logs Laravel : `storage/logs/laravel.log`
2. Consultez la documentation GPS-Track
3. Testez la configuration SMS du tracker
4. Vérifiez l'état des services système

Contribution open source :

- Reporter les bugs sur GitHub
- Proposer des améliorations
- Partager configurations spécifiques
- Documenter nouveaux protocoles

PARTIE 18 : Annexes Techniques

18.1 Référence complète commandes SMS

Commande	Syntaxe	Description	Exemple
APN	`APN,nom[,user,pass]#`	Configuration APN	`APN,internet#`
IP	`IP,index,adresse,port#`	Serveur TCP	`IP,1,192.168.1.10,5023#`
TIMER	`TIMER,mouvement,arrêt#`	Intervalles	`TIMER,30,120#`
ADMIN	`ADMIN,numéro#`	Ajouter admin	`ADMIN,+33123456789#`
SPEED	`SPEED,limite#`	Alerte vitesse	`SPEED,90#`
FENCE	`FENCE,type,lat,lon,rayon#`	Géofence	`FENCE,IN,45.123,2.456,500#`
PASSWORD	`PASSWORD,ancien,nouveau#`	Mot de passe	`PASSWORD,123456,monmdp#`
RESTART	`RESTART#`	Redémarrage	`RESTART#`
FACTORY	`FACTORY#`	Reset usine	`FACTORY#`
PARAM	`PARAM#`	Voir config	`PARAM#`

18.2 Structure base de données GPS-Track

Table devices :

```
CREATE TABLE devices (
  id bigint PRIMARY KEY AUTO_INCREMENT,
  name varchar(255),
  imei varchar(15) UNIQUE,
  protocol varchar(50),
  port int,
  status enum('online','offline'),
  last_connection timestamp,
  created_at timestamp,
  updated_at timestamp
);
```

Table positions :

```
CREATE TABLE positions (
  id bigint PRIMARY KEY AUTO_INCREMENT,
  device_id bigint,
  latitude decimal(10,8),
  longitude decimal(11,8),
```

```
  altitude decimal(8,2),
  speed decimal(6,2),
  course decimal(5,2),
  accuracy decimal(6,2),
  valid boolean,
  created_at timestamp,
  INDEX idx_device_time (device_id, created_at),
  SPATIAL INDEX idx_coords (latitude, longitude)
);
```

18.3 Configuration Docker (optionnelle)

```
# Dockerfile pour GPS-Track
FROM php:8.2-fpm-alpine

# Installation extensions PHP
RUN apk add --no-cache \
    mysql-client \
    redis \
    && docker-php-ext-install \
    pdo_mysql \
    sockets \
    bcmath

# Application GPS-Track
WORKDIR /var/www/html
COPY . .
RUN composer install --no-dev --optimize-autoloader

EXPOSE 5023
CMD ["php", "artisan", "serve", "--host=0.0.0.0"]


# docker-compose.yml
version: '3.8'
services:
  gps-track:
    build: .
    ports:
      - "8000:8000"
      - "5023:5023"
    depends_on:
      - mysql
      - redis

  mysql:
    image: mysql:8.0
    environment:
      MYSQL_ROOT_PASSWORD: rootpass
      MYSQL_DATABASE: gps_track
    volumes:
      - mysql_data:/var/lib/mysql
```

```
  redis:
    image: redis:alpine

volumes:
  mysql_data:
```

Ce guide vous donne maintenant tous les éléments techniques pour déployer un système de tracking GPS fonctionnel avec le JM-VG01U et GPS-Track. Commencez par l'installation locale, validez le fonctionnement, puis étendez selon vos besoins spécifiques.