VOYAGE+

Guide des Webhooks

Système de notifications temps réel pour les agences Plan Entreprise

1. Vue d'ensemble

Le systeme de webhooks permet aux agences Plan Entreprise de recevoir automatiquement les evenements de leur compte Voyage+ (reservations, paiements, soldes wallet, offres) directement vers leur ERP, CRM ou logiciel comptable -- en temps reel, sans avoir besoin de poller l'API.

Comment ca marche en une phrase : Quand un evenement se produit chez Voyage+ (ex: une reservation est confirmee), Voyage+ envoie automatiquement une requete HTTP POST vers l'URL que l'agence a configuree, contenant les details de l'evenement.

2. Configuration (cote agence)

Ou configurer

Tableau de bord -> Parametres -> API & Integrations -> onglet **Webhooks**

Etapes

1. Cliquer sur "Configurer un webhook"
2. Remplir le formulaire : Description, URL HTTPS, Events a recevoir
3. Cliquer sur "Creer"
4. Copier le secret HMAC immediatement -- il ne sera plus jamais reaffiche

Le secret HMAC

Le secret est une chaine de 64 caracteres commencant par whsec_... Il est **different de votre cle API** (vp_live_...) et sert uniquement a verifier que les requetes recues viennent bien de Voyage+.

Important : si vous perdez le secret, vous devez supprimer et recreer la config webhook.

3. Les 24 evenements disponibles

Reservations (6 evenements)

Evenement	Declencheur	Description
booking.created	Nouvelle reservation	Statut PENDING, en attente de paiement
booking.confirmed	Paiement reussi	Reservation confirmee, sieges reserves
booking.cancelled	Paiement echoue	Reservation annulee, sieges liberes
booking.expired	Auto-annulation (5 min)	Reservation PENDING expiree
booking.updated	Modification reservation	Champs financiers, succursale, etc.
booking.deleted	Suppression reservation	Par l'agence ou l'admin

Voyages (4 evenements)

Evenement	Declencheur
trip.created	Nouveau voyage planifie
trip.status.ongoing	Depart atteint -- voyage en cours
trip.status.completed	Arrivee atteinte -- voyage termine
trip.modified	Date/heure de depart modifiees

Offres promotionnelles (4 evenements)

Evenement	Declencheur
offer.created	Nouvelle offre publiee
offer.activated	Offre planifiee activee (fenetre de validite)
offer.deactivated	Offre expiree (end_date depasse)
offer.deleted	Offre supprimee

Succursales (2 evenements)

Evenement	Declencheur
branch.created	Nouvelle succursale creee
branch.updated	Succursale modifiee (nom, adresse, limite retrait)

Wallet / Comptabilite (5 evenements)

Evenement	Declencheur
wallet.deposit.created	Depot credite au wallet
wallet.withdrawal.requested	Demande de retrait (en attente validation)
wallet.withdrawal.completed	Retrait valide par l'admin
wallet.transfer.created	Transfert inter-succursales effectue
wallet.commission_debit.created	Commission plateforme debitee (booking confirme)

Abonnements (3 evenements)

Evenement	Declencheur
subscription.activated	Abonnement active ou renouvele
subscription.expired	Abonnement expire -- agence desactivee
subscription.expiring_soon	Rappel expiration (J-3 ou J-1)

4. Format de la requete recue

Headers HTTP

```
POST /votre-url HTTP/1.1
Host: votre-erp.cd
Content-Type: application/json
X-VoyagePlus-Signature: sha256=a1b2c3d4e5f6...
```

```
X-VoyagePlus-Event: booking.confirmed
X-VoyagePlus-Delivery: 550e8400-e29b-41d4-a716-446655440000
User-Agent: VoyagePlus-Webhook/1.0
```

Body (exemple pour booking.confirmed)

```
{
"id": "550e8400-e29b-41d4-a716-446655440000",
"booking_reference": "VP-2026-0824-001",
"status": "CONFIRMED",
"is_paid": true,
"travel_date": "2026-08-25",
"passenger": {
"full_name": "Jean Dupont",
"phone": "+243800000000"
},
"route": {
"departure_city": "Kinshasa",
"arrival_city": "Lubumbashi"
},
"seats": ["1A", "1B"],
"ticket_price_cdf": "50000",
"net_agency_amount_cdf": "48500",
"branch_name": "Succursale Matadi"
}
```

5. Verification de la signature (obligatoire)

Chaque requete contient un header X-VoyagePlus-Signature avec une signature HMAC-SHA256 du body. L'agence doit verifier cette signature pour confirmer que la requete vient bien de Voyage+.

Exemple en Python

```
import hashlib
import hmac

def verify_signature(raw_body, signature_header, secret):
expected = "sha256=" + hmac.new(
key=secret.encode(),
msg=raw_body,
digestmod=hashlib.sha256,
).hexdigest()
return hmac.compare_digest(expected, signature_header)
```

Exemple en PHP

```
function verifySignature($rawBody, $signatureHeader, $secret) {
$expected = 'sha256=' . hash_hmac('sha256', $rawBody, $secret);
return hash_equals($expected, $signatureHeader);
}
```

Exemple en JavaScript (Node.js)

```
const crypto = require('crypto');
```

```
function verifySignature(rawBody, signatureHeader, secret) {
const expected = 'sha256=' + crypto
.createHmac('sha256', secret)
.update(rawBody)
.digest('hex');
return crypto.timingSafeEqual(
Buffer.from(expected),
Buffer.from(signatureHeader)
);
}
```

Important : Toujours utiliser le **body brut** (non parse en JSON) pour calculer la signature. Si vous parsez le JSON puis re-serialisez, l'ordre des cles peut changer et la signature sera invalide.

6. Reponse attendue

L'agence doit repondre avec un **code HTTP 2xx** (200, 201, 204) pour confirmer la reception du webhook.

```
HTTP/1.1 200 OK
Content-Type: application/json

{"received": true}
```

Tout code HTTP en dehors de 2xx (400, 500, 502, 503, etc.) ou un timeout (> 15 secondes) declenche un retry automatique.

7. Retry automatique avec backoff exponentiel

Tentative	Delai avant retry	Statut
1	Immediat	PENDING -> RETRYING
2	+1 minute	RETRYING
3	+5 minutes	RETRYING
4 (derniere)	+30 minutes	RETRYING -> FAILED

Apres 3 tentatives echouees, la delivery est marquee FAILED et n'est plus retry automatiquement. Un filet de securite tourne toutes les 15 minutes pour rattraper les deliveries perdues (worker crash, etc.).

8. Idempotence (important)

Voyage+ peut envoyer le **memee evenement plusieurs fois** dans de rares cas (retry apres timeout, mais la requete d'origine avait bien ete recue). L'agence doit donc gerer l'idempotence.

Utiliser le header X-VoyagePlus-Delivery (UUID unique par delivery) comme cle de deduplication :

```
# Pseudo-code cote agence
delivery_id = request.headers["X-VoyagePlus-Delivery"]

# Verifier si on a deja traite cette delivery
if already_processed(delivery_id):
```

```
return HttpResponse(status=200) # Repondre 200 sans re-traiter

# Traiter l'evenement
process_event(json.loads(request.body))

# Marquer comme traite
mark_as_processed(delivery_id)

return HttpResponse(status=200)
```

9. Gestion cote service client Voyage+

Problemes courants et solutions

Probleme	Cause	Solution
Toutes les deliveries FAILED	URL incorrecte ou serveur down	Verifier l'URL, contacter l'agence
response_status_code = 0	Timeout (> 15s)	Serveur trop lent, optimiser le traitement
response_status_code = 403	Authentification requise	L'URL bloque les requetes sans auth
Signature invalide	Secret incorrect ou body re-serialise	Utiliser le body brut et le bon secret
Aucune delivery creee	Pas de config active ou pas souscrite	Verifier les events coches et is_active

10. Difference cle API vs secret webhook

Element	Cle API	Secret webhook
Prefixe	vp_live_...	whsec_...
Usage	Authentifier les requetes vers l'API	Signer les webhooks envoyes par Voyage+
Sens	Agence -> Voyage+	Voyage+ -> Agence
Stockage	Chez l'agence, header X-API-Key	Chez l'agence, verifier la signature
Par config	Une cle = une app	Un secret = une config webhook

11. Exemple complet d'integration (Python/Flask)

```
from flask import Flask, request, jsonify
import hashlib, hmac, json

app = Flask(__name__)
WEBHOOK_SECRET = "whsec_votre_secret"

@app.route("/webhooks/voyageplus", methods=["POST"])
def receive_webhook():
# 1. Recuperer le body brut
raw_body = request.get_data()

# 2. Verifier la signature
signature = request.headers.get("X-VoyagePlus-Signature", "")
expected = "sha256=" + hmac.new(
```

```
WEBHOOK_SECRET.encode(), raw_body, hashlib.sha256
).hexdigest()

if not hmac.compare_digest(expected, signature):
return jsonify({"error": "Invalid signature"}), 401

# 3. Parser le payload
payload = json.loads(raw_body)
event_type = request.headers.get("X-VoyagePlus-Event", "")
delivery_id = request.headers.get("X-VoyagePlus-Delivery", "")

# 4. Gerer l'idempotence
# if already_processed(delivery_id):
# return jsonify({"received": True}), 200

# 5. Router vers le handler approprie
if event_type == "booking.confirmed":
handle_booking_confirmed(payload)
elif event_type == "booking.cancelled":
handle_booking_cancelled(payload)
elif event_type == "wallet.deposit.created":
handle_deposit(payload)

# 6. Repondre 200
return jsonify({"received": True}), 200


def handle_booking_confirmed(payload):
"""Creer une facture dans l'ERP."""
print(f"Reservation confirmee: {payload['booking_reference']}")
print(f" Passager: {payload['passenger']['full_name']}")
print(f" Trajet: {payload['route']['departure_city']}"
f" -> {payload['route']['arrival_city']}")
print(f" Montant net: {payload['net_agency_amount_cdf']} CDF")
# Creer la facture dans Odoo/Sage/...
```

12. Limites et bonnes pratiques

Limites techniques

Parametre	Valeur
Timeout HTTP	15 secondes
Taille max du body	~50 KB
Retries max	3 (backoff : 1min, 5min, 30min)
URL requise	HTTPS obligatoire
Plan requis	Entreprise (69 USD/mois)

Bonnes pratiques pour l'agence

- Repondre rapidement : traiter le webhook en asynchrone (file d'attente) et repondre 200 immediatement

- Gerer l'idempotence : utiliser X-VoyagePlus-Delivery comme cle de deduplication

- Verifier la signature : ne jamais traiter un webhook sans verifier la signature HMAC

- Logger les erreurs : si le traitement echoue, logger et repondre 500 pour declencher un retry

- Surveiller les failures : si une URL ne repond plus, suspendre le webhook

- Utiliser HTTPS : les webhooks ne sont envoyes qu'en HTTPS pour la securite

13. FAQ

Q: Les webhooks remplacent-ils l'API publique ?

R: Non, ils sont complementaires. L'API publique permet de lire des donnees a la demande (pull). Les webhooks poussent les evenements en temps reel (push). Utilisez les deux : webhooks pour les mises a jour temps reel, API pour les requetes a la demande.

Q: Que se passe-t-il si mon serveur est en maintenance ?

R: Les webhooks seront retry 3 fois (1min, 5min, 30min), puis marques FAILED. Vous pouvez aussi suspendre le webhook depuis le tableau de bord pendant la maintenance.

Q: Puis-je avoir plusieurs webhooks ?

R: Oui, une agence peut configurer plusieurs webhooks (ex: un pour Odoo production, un pour un script de sauvegarde). Chacun a son propre secret et sa propre liste d'evenements.

Q: Les webhooks consomment-ils le quota API (120 req/min) ?

R: Non, les webhooks sont sortants (Voyage+ -> agence) et ne consomment pas le quota API. Le quota s'applique uniquement aux requetes entrantes (agence -> Voyage+).

Q: Comment tester mon webhook en local ?

R: Utilisez un service comme ngrok (ngrok http 5000) pour exposer votre serveur local en HTTPS, puis configurez l'URL ngrok dans Voyage+.