DOCUMENT DE VISION

Plateforme de livraison multi-services

Vision, architecture et idées d'évolution

Une infrastructure de livraison ouverte, pensée API-first

Version 1.0 — Septembre 2026

Document de travail — confidentiel

Sommaire

1. La vision : la livraison comme service
2. Architecture d'ensemble
3. Le modèle de données
4. Les modules clés
5. Choix techniques recommandés
6. Idées pour aller plus loin
7. Feuille de route

1. La vision : la livraison comme service

L'idée centrale est de ne pas construire « la livraison de ma boutique et de mon restaurant », mais une **infrastructure de livraison**. Les projets existants (gestion de boutiques, entrepôts et ventes ; gestion de restaurant et réservation de tables) deviennent les deux premiers clients de cette plateforme, exactement au même titre que n'importe quel développeur externe. C'est le modèle qu'ont suivi Stripe pour le paiement, ou Onfleet et Shipday pour la livraison.

La règle d'or : la plateforme de livraison ne doit rien savoir de ce qu'est un « produit de boutique » ou un « plat de restaurant ». Elle manipule des **colis** qui vont d'un **point A** à un **point B**, avec des contraintes : délai, fragilité, température, paiement à la livraison. C'est cette abstraction qui la rend réutilisable pour une pharmacie, un pressing, un e-commerçant, un laboratoire d'analyses ou un service de colis entre particuliers.

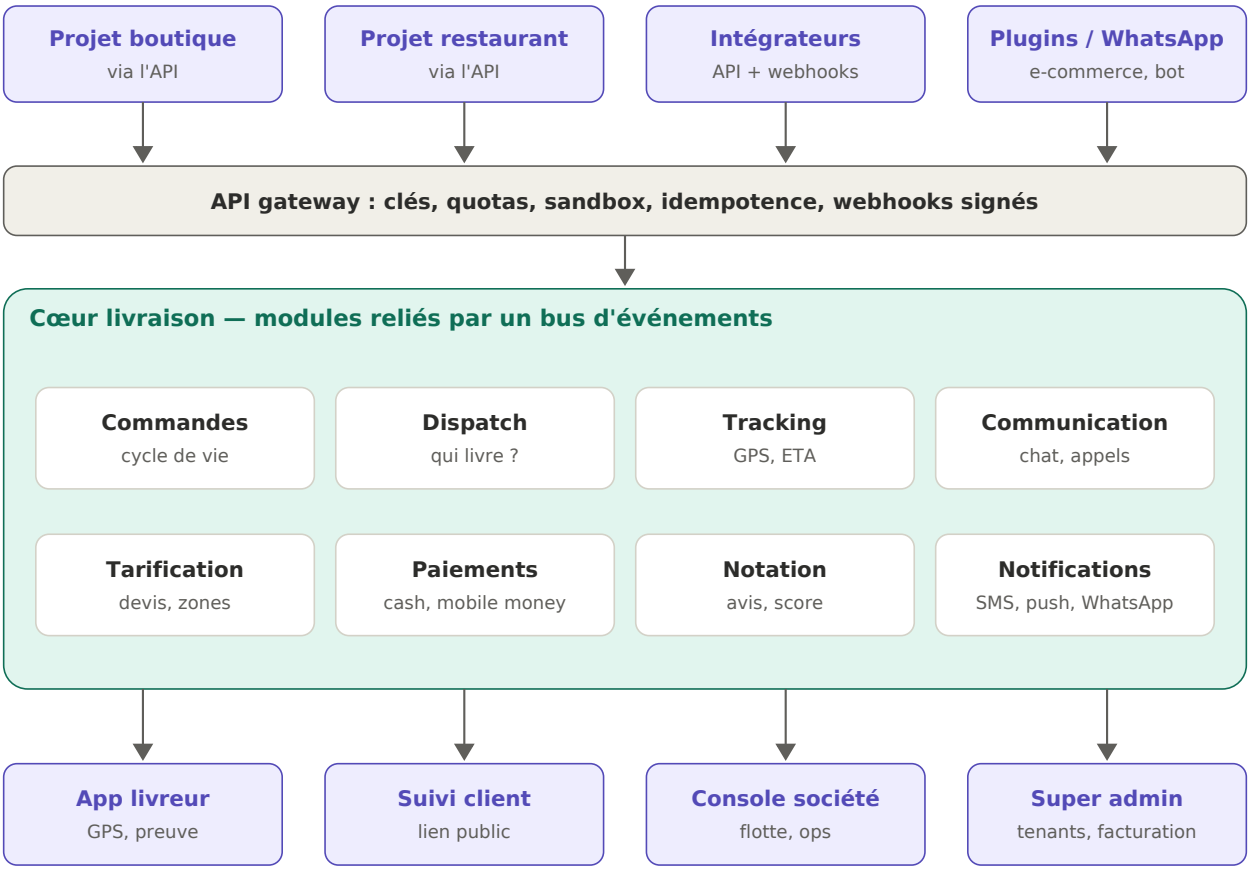
> **Principe du « dogfooding ».** Les projets boutique et restaurant appellent la même API publique que tout le monde, avec leur propre clé. Si ton propre système passe par la porte principale, l'API est forcément complète, testée et bien documentée, puisque tu en dépends toi-même.

Les objectifs stratégiques

- **Mutualiser** : une seule base de livreurs, de sociétés et d'outils pour tous les métiers.
- **Ouvrir** : permettre à n'importe quel système tiers d'intégrer la livraison en quelques heures via l'API.
- **Fédérer** : accueillir plusieurs sociétés de livraison indépendantes, chacune avec sa flotte.
- **Rassurer** : traçabilité complète, suivi en direct, preuve de livraison et notation pour tous les acteurs.

2. Architecture d'ensemble

Les consommateurs (en violet, en haut) passent tous par la même passerelle d'API. Le cœur métier (en vert) est composé de modules indépendants qui communiquent par événements. Les applications (en bas) sont les interfaces des différents acteurs.



Couche	Rôle
Consommateurs	Projet boutique, projet restaurant, intégrateurs externes, plugins e-commerce, bot WhatsApp.
API gateway	Authentification par clé, quotas, séparation sandbox/production, idempotence, envoi des webhooks signés.
Cœur livraison	Commandes, dispatch, tracking, communication, tarification, paiements, notation, notifications.
Applications	App livreur, page de suivi client, console des sociétés de livraison, super administration.

3. Le modèle de données

C'est la partie la plus importante : une erreur ici coûte très cher plus tard. Voici les concepts à poser dès le départ.

L'Organisation (tenant)

C'est l'entité racine. Une organisation peut jouer deux rôles, séparément ou simultanément. Elle peut être **expéditrice** (une boutique, un restaurant, un e-commerçant qui a besoin de livrer) ou **transporteur** (une société de livraison qui possède une flotte). Un grand restaurant avec ses propres livreurs est les deux à la fois. Ce double rôle est ce qui rend la plateforme universelle.

Le Livreur

Il appartient à une flotte, ou il est indépendant et rattaché directement à la plateforme. Il possède un véhicule (moto, vélo, voiture, tricycle, camionnette) avec des capacités : poids maximal, volume, caisson isotherme. Il a un statut en temps réel : hors ligne, disponible, en course, en pause.

La Livraison et ses tâches

La livraison ne doit pas être un simple couple « départ/arrivée ». Elle se modélise comme un ensemble de **tâches** : un ou plusieurs retraits (*pickups*) et une ou plusieurs remises (*dropoffs*). Ce choix donne gratuitement la livraison multi-arrêts, le regroupement de commandes et les retours. Chaque livraison porte une référence externe (l'identifiant de commande chez le client de l'API), des métadonnées libres, des contraintes (fenêtre horaire, fragile, froid) et un mode de paiement (prépayé ou à encaisser).

Le cycle de vie

Le cycle de vie est une machine à états stricte. Chaque transition émet un **événement** ; tout le reste (notifications, webhooks, tracking, facturation, statistiques) réagit à ces événements, ce qui rend le système extensible sans toucher au cœur.

Statut	Signification
created	Livraison créée, en attente de traitement
searching	Recherche d'un livreur en cours
assigned	Un livreur a accepté la course
to_pickup / at_pickup	En route vers le point de retrait / arrivé au retrait
picked_up	Colis récupéré
to_dropoff / at_dropoff	En route vers le client / arrivé chez le client
delivered	Livrée, preuve validée
failed / returned / cancelled	Échec, retour à l'expéditeur, annulation

4. Les modules clés

4.1 API publique et webhooks

Chaque organisation dispose de clés *sandbox* et *production*. Un intégrateur crée une livraison, reçoit un devis, puis est notifié par webhook à chaque changement de statut. Trois détails distinguent une API professionnelle : les **clés d'idempotence** (un double clic ne crée pas deux livraisons), des webhooks **signés** (HMAC) et **rejoués automatiquement** en cas d'échec, et une **documentation interactive** (OpenAPI) accompagnée de SDK JavaScript, PHP et Python. Un endpoint de devis permet d'afficher le prix de livraison dans le panier avant la commande.

4.2 Moteur de dispatch

C'est le cerveau de la plateforme. Chaque société choisit son mode : **manuel** (l'opérateur attribue), **au plus proche**, **diffusion** (l'offre part à plusieurs livreurs, le premier qui accepte la prend) ou **score** (distance, note, charge actuelle, véhicule, équité). Pour la restauration, une subtilité fait toute la différence : le projet restaurant transmet le temps de préparation estimé, et le dispatch s'arrange pour que le livreur arrive quand le plat sort de cuisine. Ni livreur qui attend, ni plat qui refroidit.

4.3 Tracking en temps réel

L'app livreur envoie sa position toutes les 3 à 5 secondes via WebSocket (ou MQTT, plus économe en batterie et en données). La position courante vit dans Redis avec un index géographique ; l'historique va dans PostgreSQL/PostGIS ou TimescaleDB. Le client reçoit un **lien de suivi public** (sans compte) où il voit le livreur se déplacer et l'heure d'arrivée se mettre à jour. Le **geofencing** automatise les statuts : à 100 m du restaurant, le livreur passe en « arrivé au retrait » sans rien toucher. L'app fonctionne aussi **hors ligne** et se synchronise au retour du réseau.

4.4 Communication

Trois canaux, tous rattachés à une livraison : client ↔ livreur, client ↔ société (support), livreur ↔ société (dispatch). Le chat s'ouvre à l'assignation et se ferme quelques heures après la livraison. Les appels passent par des **numéros masqués** pour protéger les deux parties. Des **messages rapides** pré-écrits (« Je suis devant », « Je ne trouve pas l'adresse ») permettent au livreur de communiquer sans taper en conduisant.

4.5 Preuve de livraison

Selon la valeur du colis : code OTP donné par le client, photo, signature ou position GPS au moment de la validation. C'est ce qui règle l'immense majorité des litiges.

4.6 Notation

Bidirectionnelle : le client note le livreur ; le livreur note le client et le point de retrait (utile pour repérer les restaurants qui font toujours attendre). Le score public est **pondéré et récent**, écarte les notes injustes quand la faute vient d'un autre acteur, et s'appuie sur des tags (« poli », « rapide », « colis abîmé ») plus exploitables qu'une simple étoile.

4.7 Tarification et paiements

Le moteur de prix combine zones, distance réelle par la route, poids, horaire et éventuellement une majoration en forte demande. Le **paiement à la livraison** s'appuie sur un portefeuille livreur : le livreur encaisse, la plateforme sait ce qu'il doit reverser et la réconciliation est automatique. Le **mobile money** (MTN MoMo, Moov Money, Wave selon les pays) et le versement des gains aux livreurs complètent le dispositif ; la commission de la plateforme se prélève ici.

5. Choix techniques recommandés

> **Conseil principal :** ne pas commencer en microservices. Construire un **monolithe modulaire** bien découpé (un module par bloc de l'architecture) qui communique par événements internes. Le tracking et le dispatch seront extraits en services séparés quand la charge l'exigera.

Besoin	Technologie proposée	Pourquoi
Base de données	PostgreSQL + PostGIS	Requêtes géographiques natives, fiabilité transactionnelle
Positions en direct, cache	Redis (index GEO)	« Livreurs à moins de 2 km » en quelques millisecondes
Historique GPS	TimescaleDB	Séries temporelles volumineuses, compression
Bus d'événements	RabbitMQ ou NATS (Kafka plus tard)	Découplage des modules, rejeu
Temps réel	WebSocket / MQTT	Suivi en direct, faible consommation
Itinéraires et ETA	OSRM ou Valhalla + OpenStreetMap	Beaucoup moins cher que Google Maps à grande échelle
Applications mobiles	React Native ou Flutter	Android et iOS d'un seul code, priorité Android
Optimisation de tournées	Google OR-Tools	Résolution du problème de tournées de véhicules
Documentation API	OpenAPI + portail développeur	Intégration rapide par les tiers

6. Idées pour aller plus loin

L'adresse intelligente

Dans beaucoup de villes, l'adresse officielle ne suffit pas : on livre « derrière la pharmacie, portail bleu ». La plateforme peut **apprendre les adresses** : au premier passage, le livreur enregistre le point GPS exact de la porte, une photo et une note vocale. Le client reçoit un **code d'adresse court** réutilisable qu'il donne à n'importe quel commerçant. Chaque livraison améliore la carte : un avantage concurrentiel difficile à copier.

Le réseau de flottes partagé

Quand une société est débordée, ses commandes basculent automatiquement vers une flotte partenaire, avec partage de revenu. La plateforme devient une **bourse de capacité de livraison** et les petites sociétés deviennent collectivement aussi puissantes qu'un grand acteur.

La commande par WhatsApp

Un bot où un commerçant sans système informatique écrit « Livraison de tel quartier vers tel quartier, 2 colis » : la course est créée et le lien de suivi est renvoyé au client sur WhatsApp. Cela ouvre tout le marché informel, qui n'utilisera jamais une API.

Le positionnement prédictif

L'historique révèle les pics de demande par quartier et par heure. Une carte de chaleur suggère aux livreurs disponibles de se positionner *avant* la demande, ce qui réduit les délais sans recruter.

L'optimisation de tournées B2B

Pour réapprovisionner des dizaines de boutiques depuis l'entrepôt, on passe du dispatch course par course à l'optimisation de tournées. Une fonctionnalité à forte valeur pour les distributeurs.

Sécurité et bien-être des livreurs

Bouton SOS partageant la position en direct, détection de chute ou d'accident, suivi des heures de conduite contre l'épuisement. Rare sur le marché, et très fidélisant.

Le score de confiance comme actif financier

L'historique d'un livreur (régularité, revenus, notes, fiabilité en espèces) peut servir, avec des partenaires de microfinance, à **financer sa moto** ou lui avancer de l'argent. La plateforme devient un levier d'inclusion financière, argument fort auprès des investisseurs.

Points relais et livraison en chaîne

Des boutiques partenaires servent de points de retrait quand le client est absent. Avec une logique de hubs, un livreur local amène au hub, un camion fait l'inter-villes, un autre livreur fait le dernier kilomètre, le tout suivi comme une seule livraison.

La marque blanche

Une société de livraison utilise la plateforme sous son propre logo : app livreur, page de suivi, couleurs. Elle paie un abonnement ; la plateforme devient le fournisseur technique invisible de tout un secteur.

7. Feuille de route

Phase	Contenu	Objectif
Phase 1 — Socle (MVP)	Organisations et clés API, livraisons et machine à états, dispatch manuel et « au plus proche », app livreur avec GPS et preuve de livraison, page de suivi client, notation simple, webhooks, intégration des projets boutique et restaurant.	Livrer de vraies commandes de bout en bout
Phase 2 — Opérations	Chat et appels masqués, moteur de tarification, paiement à la livraison et mobile money, portefeuille livreur, console société complète, synchronisation avec le temps de préparation.	Rendre les sociétés autonomes et monétiser
Phase 3 — Écosystème	Portail développeur, SDK, plugins e-commerce, bot WhatsApp, marque blanche, réseau de flottes partagé.	Ouvrir la plateforme aux tiers
Phase 4 — Intelligence	Adresses intelligentes, positionnement prédictif, optimisation de tournées, ETA par apprentissage, score de confiance et partenariats financiers.	Se différencier durablement

Les priorités de la phase 4 peuvent être réordonnées selon les retours des premiers clients : c'est l'usage réel qui dira quelle idée crée le plus de valeur.