

Cahier des charges

Boutique en ligne interactive (Produits, Services, Abonnements, Chat)

20 août 2025

Client / Projet : Boutique en ligne interactive

Portée : MVP e-commerce + services + abonnements + chat temps réel

Technologies : Backend Python (FastAPI), Frontend HTML/CSS/JS, PostgreSQL, Stripe, WebSocket, Celery/Redis

Version du document : 1.0 (20 août 2025)

Table des matières

0.1	Contexte et objectifs	1
0.2	Public cible	1
0.3	Fonctionnalités	1
0.3.1	Catalogue produits & services	1
0.3.2	Paielements & abonnements	1
0.3.3	Prise de rendez-vous (services)	2
0.3.4	Chat & notifications	2
0.3.5	Analytics & recommandations	2
0.3.6	Collecte WhatsApp (optionnelle)	2
0.4	Exigences techniques	2
0.4.1	Stack applicative	2
0.4.2	Sécurité & conformité	2
0.5	Parcours utilisateur	3
0.6	Administration (back-office)	3
0.7	Contraintes et exigences de qualité	3
0.8	Planning prévisionnel (3 semaines)	3
0.9	Livrables attendus	4
0.10	Indicateurs de succès & critères d'acceptation	4
Annexe A	— Matrice d'exigences (extrait)	5

0.1 Contexte et objectifs

La boutique en ligne vise à offrir une expérience moderne et fluide permettant aux clients de découvrir et d'acheter des **produits**, de **réserver des services**, de **s'abonner** à des offres récurrentes, et d'**échanger en temps réel** avec le commerçant via un **chat** (style Alibaba). Les objectifs principaux sont :

- Proposer une **navigation simple et attractive** (catalogue produits et services).
- Permettre l'**achat** et les **abonnements** (paiements sécurisés).
- Offrir la **prise de rendez-vous** en ligne.
- Activer une **interaction instantanée** (chat + notification mobile).
- Mettre en avant automatiquement les **best-sellers** et les **combinaisons** de produits.
- **Optionnel** : collecter des **numéros WhatsApp** avec consentement.

0.2 Public cible

- **Clients particuliers** souhaitant commander des produits et/ou réserver des services.
- **Clients réguliers** intéressés par des **abonnements** pour bénéficier de tarifs/avantages.
- **Nouveaux prospects** attirés par une expérience claire, mobile-first et interactive.

0.3 Fonctionnalités

0.3.1 Catalogue produits & services

- Page d'accueil avec **produits phares** (best-sellers) et **combinaisons** « souvent achetés ensemble ».
- **Catalogue** filtrable (catégories, recherche) et **pages produit** détaillées (description, prix, photos, stock).
- **Catalogue services** avec durée, tarifs et éventuelles options.
- **Mise à jour des stocks** en temps réel côté administrateur.

0.3.2 Paiements & abonnements

- **Paiement ponctuel** (carte) via Stripe.
- **Abonnements** (mensuels, hebdomadaires, etc.) via Stripe Billing.
- **Webhooks** pour suivre les statuts (succès, échec, renouvellement, annulation).

0.3.3 Prise de rendez-vous (services)

- Réservation de créneaux (calendrier/slots), avec **règles de disponibilité**.
- **Confirmation** et **statuts** (en attente, confirmé, annulé).

0.3.4 Chat & notifications

- **Chat temps réel** client ↔ administrateur (WebSocket).
- **Notifications mobiles** pour l'administrateur à l'ouverture d'une conversation (ex. **bot Telegram**).
- Historisation des **conversations** (threads, messages).

0.3.5 Analytics & recommandations

- Section **best-sellers** et **souvent achetés ensemble** (*co-occurrences* par commande).
- **Tableau de bord** basique : CA 30 jours, top produits, activité RDV.

0.3.6 Collecte WhatsApp (optionnelle)

- **Opt-in explicite** + champ numéro ; lien `wa.me` vers WhatsApp Business.
- **Conformité RGPD** : consentement clair et révocable.

0.4 Exigences techniques

0.4.1 Stack applicative

- **Backend** : Python (FastAPI), **REST** + **WebSocket**.
- **Base de données** : PostgreSQL ; ORM **SQLAlchemy** ; migrations **Alembic**.
- **Tâches asynchrones** : **Celery** + **Redis** (recalcul best-sellers, envois, planifications).
- **Paielements** : **Stripe** (paiements et abonnements).
- **Frontend** : HTML5, CSS3 (ou **Tailwind**), JavaScript (vanilla/HTMX).
- **Notifications** : **Bot Telegram** (v1), option Web Push (PWA) ultérieure.
- **Déploiement** : Docker, reverse-proxy **Nginx**, HTTPS.

0.4.2 Sécurité & conformité

- **HTTPS** partout, **JWT** pour l'authentification API.
- Bonnes pratiques OWASP (CSRF selon besoin, XSS, injection SQL via ORM).

- **RGPD/PIPEDA** : minimisation des données, consentements explicites (WhatsApp/email), droit à l'oubli.
- **Stripe** conforme PCI-DSS pour les paiements.

0.5 Parcours utilisateur

1. **Découverte** : accueil, best-sellers, combos, recherche.
2. **Sélection** : ajout au panier *ou* choix d'un service *ou* choix d'un abonnement.
3. **Paiement** : redirection vers Stripe Checkout (achat ou souscription).
4. **Confirmation** : page + e-mail (selon configuration v1).
5. **Support** : chat temps réel, notif admin instantanée côté mobile.
6. **Suivi** : espace client (commandes, abonnements, rendez-vous).

0.6 Administration (back-office)

- **Produits/Services** : création, édition, activation/désactivation, images.
- **Stocks** : mise à jour rapide (édition en masse si besoin).
- **Commandes/Abonnements** : consultation, statuts, exports.
- **Rendez-vous** : vue calendrier, confirmations, annulations.
- **Chat** : liste des conversations, réponses, historique.
- **Analytics** : best-sellers, combos, indicateurs (CA, top 5).
- **Données clients** : export des opt-ins WhatsApp/email (conforme RGPD).

0.7 Contraintes et exigences de qualité

- **Performance** : LCP < 2,5 s sur la page d'accueil (objectif).
- **Accessibilité** : responsive (mobile-first), contrastes, navigation clavier.
- **Fiabilité** : gestion des erreurs de paiement, des conflits de créneaux.
- **Évolutivité** : architecture modulaire, séparation claire front/back.

0.8 Planning prévisionnel (3 semaines)

Semaine	Objectifs et livrables
S1	Conception (schéma BDD, endpoints, wireframes), Catalogue & Stock (listing produits, fiche produit, admin stock).
S2	Paiements & Abonnements (Stripe Checkout + webhooks), Prise de RDV (créneaux, réservation, statuts).

0.9 Livrables attendus

- **MVP opérationnel** déployé (API + Front + BDD + paiements + RDV + chat).
- **Documentation technique** : API (OpenAPI), installation, déploiement.
- **Guide administrateur** : gestion catalogue, stocks, RDV, exports.
- **Cahier de tests** : parcours achat, abonnement, RDV, chat, webhooks.

0.10 Indicateurs de succès & critères d'acceptation

Indicateurs de succès

- Un client peut **acheter**, **s'abonner** et **réserver** sans blocage.
- L'administrateur **reçoit une notification** dès qu'un chat démarre.
- Les **best-sellers** et **combos** s'affichent automatiquement et se mettent à jour.
- Respect des **exigences de sécurité** et **RGPD**.

Critères d'acceptation (exemples)

- **Catalogue** : affichage liste + détail, mise à jour stock visible immédiatement.
- **Paieement** : paiement carte validé, reçu Stripe, statut commande mis à jour via webhook.
- **Abonnement** : création d'un abonnement, renouvellement simulé, annulation prise en compte.
- **RDV** : réservation d'un créneau disponible, confirmation affichée, conflit évité.
- **Chat** : message client → notification admin sur téléphone en moins de 10 secondes (via bot).
- **Analytics** : bloc best-sellers/combos rempli à partir des commandes des 30 derniers jours (tâche planifiée).

Annexe A — Matrice d'exigences (extrait)

ID	Exigence	Priorité
F-01	L'utilisateur peut consulter le catalogue produits et services.	Haute
F-02	Paieement ponctuel via Stripe Checkout.	Haute
F-03	Souscription à des abonnements (Stripe Billing).	Haute
F-04	Réservation de créneaux avec confirmation.	Haute
F-05	Chat temps réel + notification admin mobile.	Haute
F-06	Section best-sellers et combos mise à jour automatiquement.	Moyenne
N-01	Sécurité (HTTPS, JWT, RGPD, PCI via Stripe).	Haute
N-02	Performance (LCP < 2,5 s sur home).	Moyenne
A-01	Back-office simple (gestion produits, stocks, RDV).	Haute

Document généré pour cadrer le développement *from scratch* (backend Python FastAPI, frontend HTML/CSS/JS).