PRD v1 — KHADAMAT (Web + App Native) — SOURCE DE VÉRITÉ BACKEND

---

1. Vision & objectif

Khadamat est une marketplace marocaine de services à la demande reliant **clients** et **professionnels locaux**, avec une approche **WhatsApp-first**.

La plateforme est disponible sur :

- 🌐**Site web**
- ♾️**Application mobile native (iOS / Android)**

Objectifs MVP :

- Réservation rapide et sans attente passive
- Éviter le double-booking
- Responsabiliser clients et pros
- Préparer la monétisation PRO (Premium / Boost)

---

2. Périmètre géographique & services (MVP)

Villes

- Casablanca
- Rabat / Salé
- Marrakech

Catégories

- Plomberie
- Électricité
- Climatisation
- Serrurerie
- Ménage
- Peinture
- Bricolage
- Jardinage

---

3. Plateformes & notifications

- **Une seule API backend** pour Web + App
- Même logique métier partout

Notifications

- Web : notifications in-app
- App mobile : **push notifications**
- WhatsApp : actions critiques (confirmation, annulation)

---

4. Rôles & accès

Visiteur (sans compte)

- Peut naviguer (pages marketing, blog, profils PRO)
- Ne peut pas réserver

Client

- Inscription obligatoire
- Peut réserver, annuler, signaler
- Peut noter un PRO (🔲1–5 + texte)

Professionnel (PRO)

- Choisit **1 seule ville**
- Gère planning & réservations
- Peut être **non vérifié** ou **vérifié (KYC)**

Admin

- Interface sécurisée
- Validation KYC
- Gestion reports & sanctions

---

5. Authentification

Inscription

- Téléphone
- Email
- Mot de passe
- Nom / Prénom
- Rôle

Connexion

- Téléphone **ou** email + mot de passe

---

6. SEO & contenu

Blog

- Articles Khadamat (SEO Google + GPT / Gemini)
- Pages publiques indexables

---

7. Pages du site (indicatif)

Public

- /, /about, /faq, /blog, /pricing, /devenir-pro
- /services, /pros, /pro/[id]

Auth

- /auth/login
- /auth/signup

Client

- /client/bookings
- /client/profile
- /client/settings

Pro

- /pro/dashboard
- /pro/bookings
- /pro/schedule
- /pro/services
- /pro/subscription
- /pro/profile

---

8. Timezone

- DB : UTC
- API / UI : Africa/Casablanca
- Pas de DST

---

9. Disponibilités

- Créneaux fixes de 1h
- Horaires hebdomadaires
- Exceptions ponctuelles

---

10. Booking — Machine à états

Statuts

```
PENDING
CONFIRMED
DECLINED
CANCELLED_BY_CLIENT
CANCELLED_BY_CLIENT_LATE
CANCELLED_BY_PRO
CANCELLED_AUTO_FIRST_CONFIRMED
EXPIRED
COMPLETED
```

Règles clés

- Max **3 PENDING** par client (ville + catégorie)
- PENDING expire après 12h
- CONFIRMED crée un SlotLock
- **First-confirmed-wins** transactionnel

---

11. Annulations

Client

- PENDING → libre

CONFIRMED :

12h : neutre - <12h : pénalité - après début : interdit → Report

PRO

- PENDING → DECLINED (pas de pénalité)
- CONFIRMED → CANCELLED_BY_PRO (pénalité)

---

12. Completion

- PRO uniquement
- now ≥ timeSlot + 1h
- Ouvre fenêtre report 48h

---

13. Reports

- Créables à tout moment
- ≤48h après COMPLETED = priorité haute
- Traitement ADMIN

---

14. KYC PRO (CNDP)

Statuts

- NOT_SUBMITTED
- PENDING
- APPROVED
- REJECTED

Documents

- CIN recto
- CIN verso
- Selfie avec CIN

Règles

- PRO non vérifié : badge « Non vérifié »
- Pas d’abonnement ni boost sans KYC APPROVED
- Suppression physique des documents si REJECTED ou compte supprimé

---

15. Monétisation

Premium

- Sans engagement : 350 MAD / mois
- Engagement 12 mois : 300 MAD / mois
- −50 MAD le premier mois (nouveaux)

Avantages :

- Badge Premium
- Visibilité x2
- Reco semaine
- Dashboard stats
- Profil enrichi
- 3 catégories

Boost

- 200 MAD / semaine glissante
- Badge Sponsorisé
- Mise en avant ville + catégorie

- Non cumulable Premium

---

16. Ranking PRO

1. Disponibilité
2. Ville
3. Catégorie
4. Boost
5. Premium
6. Score qualité
7. Random léger

---

17. Notifications

- Confirmation / annulation → WhatsApp client
- Web → in-app
- App → push

---

18. Admin

- Validation / rejet KYC
- Gestion reports
- Ban / unban

---

19. Règles backend critiques

- Slots alignés à l'heure
- First-confirmed-wins en transaction
- SlotLock unique (proId + timeSlot)
- CRON nettoyage SlotLocks orphelins
- CRON suppression docs KYC rejetés (CNDP)
- **Privacy Shield :** Masquage strict des numéros de téléphone (Client & Pro) dans l'API tant que `BookingStatus !== CONFIRMED`.

SCHÉMA PRISMA (ALIGNÉ PRD v1)

```
generator client {
  provider = "prisma-client-js"
}

datasource db {
  provider = "postgresql"
```

```
  url      = env("DATABASE_URL")
}

enum Role {
  CLIENT
  PRO
  ADMIN
}

enum UserStatus {
  ACTIVE
  SUSPENDED
  BANNED
}

enum KycStatus {
  NOT_SUBMITTED
  PENDING
  APPROVED
  REJECTED
}

enum BookingStatus {
  PENDING
  CONFIRMED
  DECLINED
  CANCELLED_BY_CLIENT
  CANCELLED_BY_CLIENT_LATE
  CANCELLED_BY_PRO
  CANCELLED_AUTO_FIRST_CONFIRMED
  EXPIRED
  COMPLETED
}

enum BookingEventType {
  CREATED
  CONFIRMED
  DECLINED
  EXPIRED
  CANCELLED
  COMPLETED
  SLOTS_RELEASED
}

enum PenaltyType {
  CLIENT_CANCEL_LATE
  PRO_CANCEL_CONFIRMED
}

enum SubscriptionPlan {
  PREMIUM_MONTHLY_NO_COMMIT
```

```
  PREMIUM_ANNUAL_COMMIT
}

enum SubscriptionStatus {
  ACTIVE
  CANCELLED
  EXPIRED
}

enum BoostStatus {
  ACTIVE
  EXPIRED
}

enum EstimatedDuration {
  H1
  H2
  H3
  H4
  H8
}

enum ReportStatus {
  OPEN
  IN_REVIEW
  RESOLVED
  REJECTED
}

enum Platform {
  IOS
  ANDROID
  WEB
}

model User {
  id        String     @id @default(cuid())
  role      Role
  status    UserStatus @default(ACTIVE)

  phone     String     @unique
  email     String?
  password  String
  firstName String
  lastName  String

  // CLIENT penalty system (PRD)
  clientLateCancelCount30d Int      @default(0)
  clientSanctionTier       Int      @default(0) // 0 none, 1=48h, 2=7d, 3=1m,
4=ban
  bookingCooldownUntil     DateTime?
```

```
  clientPenaltyResetAt     DateTime?

  // Ban audit (global)
  bannedAt   DateTime?
  banReason  String?

  createdAt DateTime @default(now())
  updatedAt DateTime @updatedAt

  proProfile ProProfile?

  bookingsAsClient Booking[] @relation("BookingsAsClient")

  deviceTokens DeviceToken[]
  penaltyLogs  PenaltyLog[]
  reportsAsClient Report[] @relation("ReportsAsClient")
  reviewsAsClient Review[] @relation("ReviewsAsClient")
}

model ProProfile {
  userId String @id
  user   User   @relation(fields: [userId], references: [id], onDelete:
Cascade)

  // MVP: PRO choisit UNE ville
  cityId   String

  // WhatsApp-first
  whatsapp String

  // KYC
  kycStatus KycStatus @default(NOT_SUBMITTED)
  kycCinFrontUrl String?
  kycCinBackUrl  String?
  kycSelfieUrl   String?
  kycRejectionReason String?

  // Subscription flags (source of truth via rows ci-dessous)
  premiumActiveUntil DateTime?
  boostActiveUntil   DateTime?

  // PRO penalty counters (PRD)
  proCancelCount30d          Int @default(0)
  proConsecutiveCancelCount  Int @default(0)

  createdAt DateTime @default(now())
  updatedAt DateTime @updatedAt

  // Relations
  services ProService[]
  bookings Booking[]
```

```
  weeklyAvailability WeeklyAvailability[]
  availabilityExceptions AvailabilityException[]
  slotLocks SlotLock[]
  subscriptions ProSubscription[]
  boosts ProBoost[]
  reportsAsPro Report[] @relation("ReportsAsPro")
  reviewsAsPro Review[] @relation("ReviewsAsPro")
}

model ProService {
  id        String @id @default(cuid())
  proUserId String
  categoryId String
  isActive  Boolean @default(true)

  // Affichage prix (fourchette ou fixe) — info profil, le prix final est
négocié sur WhatsApp
  pricingType String? // "RANGE" | "FIXED" (texte simple MVP)
  minPriceMad Int?
  maxPriceMad Int?
  fixedPriceMad Int?

  createdAt DateTime @default(now())
  updatedAt DateTime @updatedAt

  pro ProProfile @relation(fields: [proUserId], references: [userId],
onDelete: Cascade)

  @@unique([proUserId, categoryId])
  @@index([categoryId])
}

model Booking {
  id        String @id @default(cuid())
  status    BookingStatus
  timeSlot  DateTime

  cityId     String
  categoryId String

  clientId String
  proId    String

  expiresAt   DateTime
  cancelledAt DateTime? // uniquement CANCELLED*
  completedAt DateTime?
  confirmedAt DateTime?

  // durée choisie à la confirmation (H1/H2/H3/H4/H8)
  estimatedDuration EstimatedDuration? @default(H1)
```

```
  // raison annulation (obligatoire quand actor=PRO sur CONFIRMED)
  cancelReason String?

  createdAt DateTime @default(now())
  updatedAt DateTime @updatedAt

  client User      @relation("BookingsAsClient", fields: [clientId],
references: [id])
  pro    ProProfile @relation(fields: [proId], references: [userId])

  // Relations
  slotLock SlotLock?
  events BookingEvent[]
  reports Report[]
  review  Review?

  @@index([clientId, cityId, categoryId, timeSlot])
  @@index([proId, timeSlot])
}

model BookingEvent {
  id        String @id @default(cuid())
  bookingId String
  type      BookingEventType

  actorUserId String? // null si SYSTEM
  actorRole   Role?

  metadata Json?

  createdAt DateTime @default(now())

  booking Booking @relation(fields: [bookingId], references: [id], onDelete:
Cascade)

  @@index([bookingId, createdAt])
}

model SlotLock {
  id        String @id @default(cuid())
  proUserId String
  bookingId String @unique
  timeSlot  DateTime

  createdAt DateTime @default(now())

  pro     ProProfile @relation(fields: [proUserId], references: [userId],
onDelete: Cascade)
  booking Booking    @relation(fields: [bookingId], references: [id],
onDelete: Cascade)
```

```
  @@unique([proUserId, timeSlot])
  @@index([createdAt])
}

model WeeklyAvailability {
  id        String @id @default(cuid())
  proUserId String

  // 0=Dimanche … 6=Samedi (standard JS)
  dayOfWeek Int

  // minutes depuis 00:00 (ex: 9h00 = 540)
  startMin Int
  endMin   Int

  isActive Boolean @default(true)

  createdAt DateTime @default(now())
  updatedAt DateTime @updatedAt

  pro ProProfile @relation(fields: [proUserId], references: [userId],
onDelete: Cascade)

  @@unique([proUserId, dayOfWeek])
}

model AvailabilityException {
  id        String @id @default(cuid())
  proUserId String

  startAt DateTime
  endAt   DateTime

  // raisons: "AUTO_BLOCK_BOOKING", "FULL_DAY_BOOKING", "MANUAL_BLOCK", etc.
  reason String

  // lien optionnel vers booking quand auto-block
  bookingId String?

  createdAt DateTime @default(now())

  pro     ProProfile @relation(fields: [proUserId], references: [userId],
onDelete: Cascade)
  booking Booking?   @relation(fields: [bookingId], references: [id],
onDelete: Cascade)

  @@index([proUserId, startAt])
  @@index([bookingId])
}

model PenaltyLog {
```

```
  id        String @id @default(cuid())
  userId    String
  type      PenaltyType
  bookingId String?

  createdAt DateTime @default(now())

  user    User     @relation(fields: [userId], references: [id], onDelete:
Cascade)
  booking Booking? @relation(fields: [bookingId], references: [id], onDelete:
SetNull)

  @@index([userId, createdAt])
  @@index([type, createdAt])
}

model Report {
  id        String @id @default(cuid())
  bookingId String

  clientId String
  proId    String

  title   String
  details String

  // Priorité calculable au runtime, mais on stocke le bool MVP
  isWithinDisputeWindow Boolean @default(false) // true si <=48h apres
COMPLETED

  status ReportStatus @default(OPEN)

  // pièces jointes (URLs cloud)
  attachments Json? // ex: ["https://...", "https://..."]

  createdAt DateTime @default(now())
  updatedAt DateTime @updatedAt

  booking Booking   @relation(fields: [bookingId], references: [id],
onDelete: Cascade)
  client  User     @relation("ReportsAsClient", fields: [clientId],
references: [id], onDelete: Cascade)
  pro     ProProfile @relation("ReportsAsPro", fields: [proId], references:
[userId], onDelete: Cascade)

  @@index([bookingId])
  @@index([proId, createdAt])
}

model Review {
  id        String @id @default(cuid())
```

```
  bookingId String @unique

  clientId String
  proId    String

  rating Int // 1..5
  comment String?

  createdAt DateTime @default(now())

  booking Booking   @relation(fields: [bookingId], references: [id],
onDelete: Cascade)
  client  User     @relation("ReviewsAsClient", fields: [clientId],
references: [id], onDelete: Cascade)
  pro     ProProfile @relation("ReviewsAsPro", fields: [proId], references:
[userId], onDelete: Cascade)

  @@index([proId, createdAt])
}

model DeviceToken {
  id       String @id @default(cuid())
  userId   String
  platform Platform
  token    String

  revokedAt DateTime?
  createdAt DateTime @default(now())

  user User @relation(fields: [userId], references: [id], onDelete: Cascade)

  @@unique([platform, token])
  @@index([userId])
}

model ProSubscription {
  id        String @id @default(cuid())
  proUserId String

  plan   SubscriptionPlan
  status SubscriptionStatus @default(ACTIVE)

  // Stripe
  stripeCustomerId String?
  stripeSubscriptionId String?

  // engagement annuel (plan PREMIUM_ANNUAL_COMMIT)
  commitmentStartsAt DateTime?
  commitmentEndsAt   DateTime?

  // pricing snapshot
```

```
  priceMad Int
  introDiscountMad Int? // -50 MAD premier mois

  startedAt DateTime @default(now())
  endedAt   DateTime?

  pro ProProfile @relation(fields: [proUserId], references: [userId],
onDelete: Cascade)

  @@index([proUserId, status])
}

model ProBoost {
  id        String @id @default(cuid())
  proUserId String

  // ciblage
  cityId     String
  categoryId String

  status BoostStatus @default(ACTIVE)

  // semaine glissante
  startsAt DateTime
  endsAt   DateTime

  // Stripe
  stripePaymentIntentId String?

  // pricing snapshot
  priceMad Int @default(200)

  createdAt DateTime @default(now())

  pro ProProfile @relation(fields: [proUserId], references: [userId],
onDelete: Cascade)

  @@index([cityId, categoryId, status, startsAt])
  @@index([proUserId, endsAt])
}
```

FIN — PRD v1 KHADAMAT