📘SPEC FONCTIONNELLE — Plateforme IA RGPD multi-tenant

---

1. Objectif du document

Ce document décrit le fonctionnement **fonctionnel et métier** de la plateforme IA RGPD, **indépendamment de l'implémentation technique**.

Il permet à : - un humain (fondateur, product owner, auditeur, développeur), - un futur client (avocat, comptable, médecin), - une IA de génération (ex. Claude Code),

➡️ de comprendre clairement : - **qui peut faire quoi**, - **dans quel périmètre**, - **comment les responsabilités sont séparées**, - **comment le backend et les futures interfaces UI s'articulent**.

☸️ **Important** : ce document **n'introduit aucun nouveau concept**. Il formalise uniquement ce qui **existe déjà dans le code et les EPICs**.

---

2. Vision métier globale

La plateforme est un **SaaS IA conforme RGPD**, destiné à des professionnels manipulant des **données sensibles** : - avocats, - comptables, - médecins, - professions réglementées.

Elle repose sur un **modèle multi-tenant strict** : - la plateforme est exploitée par un **éditeur** (toi), - chaque client est une **organisation indépendante** (cabinet, clinique, étude), - chaque organisation possède **ses propres utilisateurs et ses propres données**, - **aucune donnée ne circule entre organisations**.

---

3. Les 3 niveaux d'acteurs (concept clé)

Toute action dans le système est effectuée par un **acteur**, appartenant **à un seul niveau**.

Ces niveaux sont formalisés dans le code par le champ :

```
actorScope: "SYSTEM" | "PLATFORM" | "TENANT"
```

3.1 SYSTEM — le système lui-même

Définition métier

Le niveau **SYSTEM** représente la plateforme **en tant que logiciel**, sans utilisateur humain.

Quand SYSTEM agit ?

- installation initiale de la plateforme,
- création du **tout premier super-administrateur**,
- jobs automatiques (purge, rétention, maintenance),
- scripts CLI.

Caractéristiques

- pas de login,
- pas d'interface UI,
- pas de tenant,
- **traçabilité obligatoire** (audit).

Exemple :

> « Le système initialise la plateforme et crée le premier superadmin. »

3.2 PLATFORM — l'éditeur de la plateforme

Définition métier

Le niveau **PLATFORM** représente les **administrateurs de la plateforme SaaS** (ex. toi, ou ton équipe).

Ce que peut faire un acteur PLATFORM

- créer un nouveau client (**tenant**),
- créer un administrateur pour un client,
- consulter l'état global de la plateforme,
- accéder aux audits globaux,
- configurer des règles globales (à venir).

Ces utilisateurs **ne sont pas des clients**. Ils **administrent le SaaS lui-même**.

3.3 TENANT — un client de la plateforme

Définition métier

Un **TENANT** est une **organisation cliente** : - cabinet d'avocats, - cabinet comptable, - clinique, etc.

Chaque tenant : - est totalement isolé des autres, - possède ses propres utilisateurs, - possède ses propres données, - est juridiquement responsable de ses traitements.

Ce que peut faire un acteur TENANT

- gérer les utilisateurs internes,
- utiliser les outils IA,
- consulter ses audits,
- exercer les droits RGPD (export, effacement, consentement).

☸Un acteur TENANT **ne peut jamais** : - créer un autre tenant, - voir des données d'un autre tenant, - accéder aux fonctions PLATFORM.

---

4. Modèle métier : organisations et utilisateurs

4.1 Les tenants (clients)

Un **tenant** représente une organisation cliente.

Exemples : - Cabinet Dupont, - Clinique Saint-Jean, - Étude Martin & Associés.

Fonctionnellement, un tenant : - est créé par un acteur PLATFORM, - possède un identifiant interne (`tenantId`), - est référencé par un identifiant lisible (`tenantSlug`), - possède des paramètres propres (rétention, options futures).

☸**Côté code** : `TenantRepo`

4.2 Les utilisateurs de la plateforme (Platform Users)

Les **Platform Users** sont les comptes qui administrent la plateforme SaaS.

Exemples : - super-administrateur (toi), - support plateforme (plus tard), - auditeur interne (plus tard).

☸Ils **ne sont rattachés à aucun tenant**.

☸**Côté code** : `PlatformUserRepo`

4.3 Les utilisateurs d'un tenant (Tenant Users)

Les **Tenant Users** sont les utilisateurs internes à un client.

Exemples : - administrateur du cabinet, - avocat, - comptable, - médecin.

Ils sont toujours rattachés à : - un tenant, - un rôle (admin, user, etc.).

☸**Côté code** : `TenantUserRepo`

---

5. Traçabilité et audit (RGPD by design)

Toute action significative génère un **AuditEvent**.

```
AuditEvent = {
  eventName,
  actorScope,
```

```
  actorId?,
  tenantId?,
  targetId?,
  metadata?
}
```

Rôle métier de l’audit

- prouver la conformité RGPD,
- reconstruire l’historique des actions,
- démontrer l’absence d’accès illégitime,
- répondre à un audit CNIL ou client.

☸Les audits **ne contiennent jamais de données métier sensibles**.

---

6. Le bootstrap : pourquoi il existe

Problème métier

Au moment de l’installation : - il n’y a **aucun utilisateur**, - mais il faut un **premier super-administrateur**.

♆Impossible via une UI classique (personne n’est encore connecté).

---

7. Le CLI bootstrap (rôle exact)

Le **CLI bootstrap** est un **outil d’installation**, pas une fonctionnalité métier.

Ce qu’il fait

- initialise la base de données,
- crée le premier **Platform SuperAdmin**,
- marque la plateforme comme « bootstrapped »,
- génère un audit **SYSTEM**.

Ce qu’il ne fait pas

- pas de login,
- pas d’usage quotidien,
- pas accessible aux clients,
- pas exposé via API.

☸Il est : - autorisé uniquement en phase de bootstrap, - bloqué ensuite (**non-rejouable**).

---

8. Lien entre CLI, backend et futures UI

Action	Aujourd'hui	Demain (UI)
Créer le 1er superadmin	CLI (SYSTEM)	❌jamais
Créer un tenant	❌	UI PLATFORM
Créer un admin tenant	CLI / PLATFORM	UI PLATFORM
Gérer utilisateurs	❌	UI TENANT
Utiliser l'IA	❌	UI TENANT

☸️Le **CLI ne concurrence pas l'UI** : il couvre uniquement **ce qui est impossible autrement**.