

Fondation Mérieux

Lab Book

LabBook v3.6

API LabBook

Échanger des données avec d'autres logiciels (dossiers, résultats)

V3.1

Octobre 2026

Table des matières

1. À qui s'adresse ce guide 3
2. Mettre en place l'accès 4
2.1 Créer l'utilisateur API 4
2.2 Déclarer le logiciel tiers (client OAuth) 5
2.3 Les droits (scopes) 5
2.4 Obtenir un jeton d'accès 6
3. Tester l'API avec Swagger 8
4. Les services en détail 9
4.1 Créer un dossier — POST /external/record/det/0 9
4.2 Lister les dossiers — POST /external/record/list/{id_pres} 10
4.3 Rechercher un patient — POST /external/patient/list 10
4.4 Lire les résultats — GET /external/result/{id_rec} 10
4.5 Envoyer des résultats — POST /external/result/{id_rec} 11
4.6 Référentiel et utilisateurs 12
Annexe A — Schéma de données de création d'un dossier 13
Annexe B — Historique du document 17

1. À qui s'adresse ce guide

Ce guide s'adresse à l'**informaticien** ou à l'**intégrateur** qui veut connecter LabBook à un autre logiciel : dossier patient informatisé (DPI), logiciel de facturation, application mobile, plateforme régionale…

Public	Développeurs, intégrateurs, responsables informatiques
Prérequis	Accès root pour créer le compte API et le client OAuth ; notions de REST/JSON
Norme	OpenAPI 3 (Swagger) ; authentification **OAuth 2.0**

Ce que permet l'API :

Besoin	Service
Créer une demande d'analyses depuis un DPI	`POST /external/record/det/0`
Retrouver un patient	`POST /external/patient/list`
Lister les dossiers (par prescripteur, statut, période…)	`POST /external/record/list/{id_pres}`
Récupérer les résultats d'un dossier	`GET /external/result/{id_rec}`
Envoyer des résultats (ex. depuis un middleware)	`POST /external/result/{id_rec}`
Consulter le référentiel des analyses	`POST /external/analysis/list`, `GET /external/analysis/det/{id_ana}`
Vérifier qu'un code d'analyse est libre	`GET /external/analysis/code/check/{code}`
Lister les utilisateurs	`POST /external/user/list`

À savoir — adresses :

- Documentation interactive (Swagger) : `http(s)://<serveur>/sigl/api`
- Racine des services : `http(s)://<serveur>/sigl/services`
- Exemple : `https://192.168.1.10/sigl/services/external/patient/list`

Attention : l'API donne accès à des **données de santé**.

- Utilisez le **HTTPS** (guide E2).
- Créez un compte dédié par logiciel tiers.
- Ne donnez que les droits (*scopes*) nécessaires.
- Toutes les opérations sont tracées dans le **journal d'audit** (guide D1, § 11.1).

2. Mettre en place l'accès

2.1 Créer l'utilisateur API

1. Connectez-vous à LabBook avec **root**.
2. Allez dans **Paramètres › Gestion des utilisateurs**, puis **Ajouter un utilisateur**.
3. Choisissez le rôle **API**.
4. Renseignez l'identifiant et un mot de passe robuste, puis enregistrez.

Exemple : identifiant `api_dpi`, pour le connecteur avec le DPI de l'hôpital.

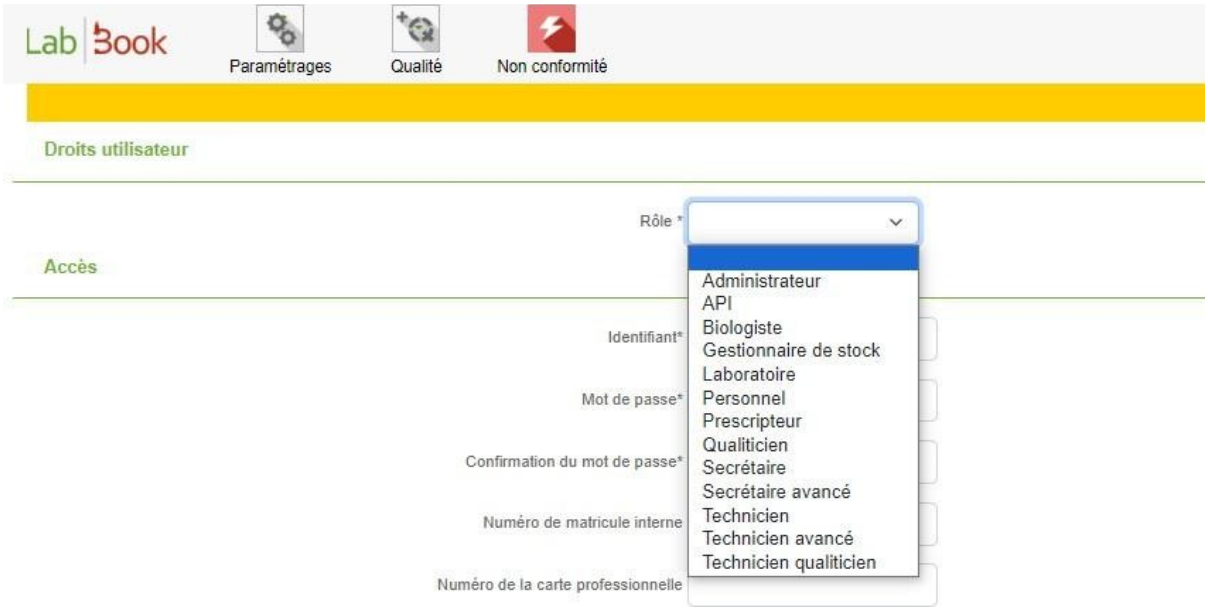

Figure 1 – Choix du rôle API à la création de l'utilisateur

Un utilisateur de rôle **API** qui se connecte à LabBook arrive directement sur la page **Swagger** : il n'a accès à aucun autre écran.

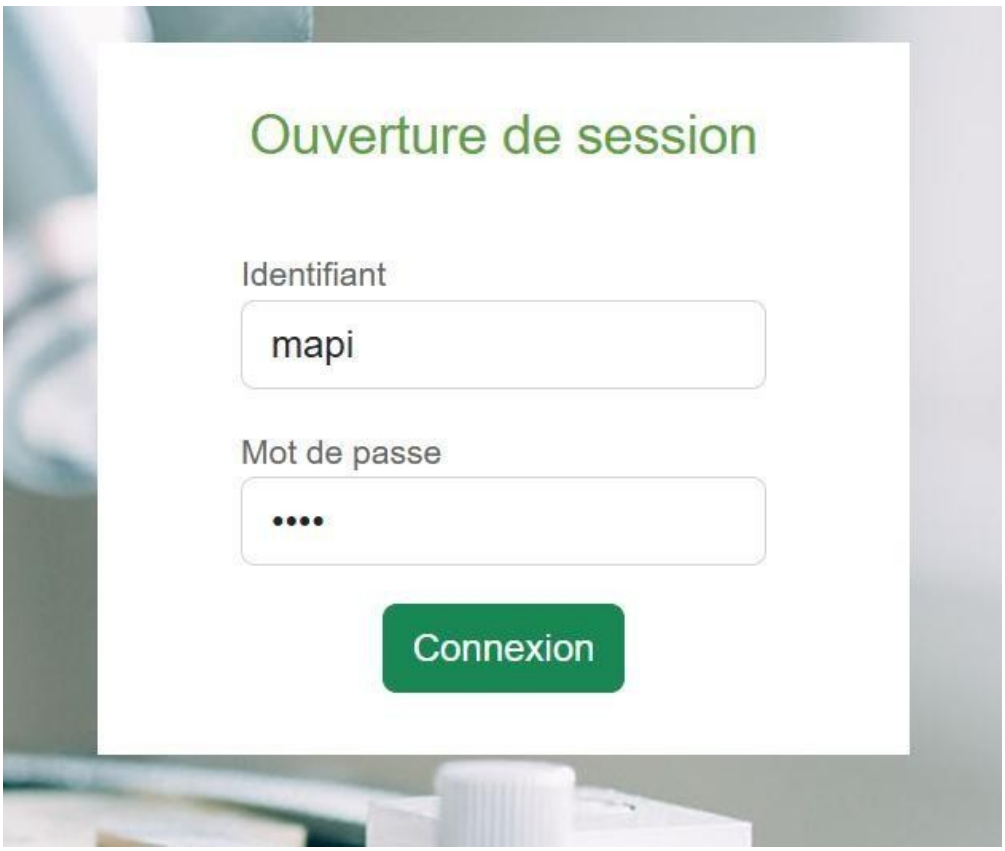

Figure 2 – Connexion avec l'utilisateur API

2.2 Déclarer le logiciel tiers (client OAuth)

Un logiciel qui appelle l'API sans passer par Swagger doit être déclaré comme **client OAuth**.

Accès : **Intégrations › Configuration OAuth Client** (rôle Administrateur).

Champ	Rôle	Exemple
Id client *	Identifiant du logiciel	`dpi-hopital`
Secret	Mot de passe du client (applications serveur)	chaîne aléatoire longue
Redirect URIs	Adresse(s) de retour, pour le flux *authorization code*	`https://dpi.hopital.local/callback`
Scope	Droits accordés (voir § 2.3)	`external/record external/result`
Grant types	Flux autorisés	`client_credentials` (serveur à serveur) ou `authorization_code`

2.3 Les droits (scopes)

Scope	Donne accès à
`external`	Tous les services `/external/*`
`external/analysis`	Référentiel des analyses

Scope	Donne accès à
`external/patient`	Patients
`external/record`	Dossiers (création, liste)
`external/result`	Résultats (lecture, écriture)
`external/user`	Utilisateurs

Exemple : un DPI qui crée des demandes et récupère les résultats n'a besoin que de `external/record external/result external/patient`.

2.4 Obtenir un jeton d'accès

LabBook délivre des jetons OAuth 2.0 valables **2 heures** (7 200 secondes). Pour obtenir un jeton, un logiciel s'adresse à deux adresses de LabBook :

- **Point d'autorisation** : `/sigl/services/oauth/authorize`. Il sert quand une **personne** est derrière l'écran (page Swagger, front-end LabBook) : LabBook affiche sa page de connexion, l'utilisateur s'identifie et autorise l'accès, puis LabBook remet un **code provisoire**.
- **Point de jeton** : `/sigl/services/oauth/token` (méthode POST). Il échange ce code provisoire contre un **jeton d'accès** ; un logiciel serveur s'y adresse aussi **directement**, avec son identifiant et son secret de client.
- **Flux pris en charge** : **authorization code avec PKCE obligatoire** (méthode S256 recommandée), utilisé par le front-end LabBook et par Swagger ; **client credentials**, pour les échanges de serveur à serveur (par exemple LabBook Connect). Le secret du client peut être envoyé en authentification Basic ou dans le corps de la requête.

Situation	Adresses utilisées	Comparaison
Une personne se connecte (Swagger, LabBook)	① Point d'autorisation → code provisoire ; ② Point de jeton → jeton	L'accueil vérifie l'identité et remet un ticket ; le guichet échange le ticket contre un badge
Un logiciel serveur appelle l'API (DPI, LabBook Connect)	Point de jeton uniquement, avec identifiant et secret du client	Le logiciel présente directement ses identifiants au guichet et reçoit le badge

> **Version minimale : LabBook 3.6.25.** Avant cette version, un logiciel tiers déclaré comme client OAuth ne pouvait jamais obtenir de jeton : son secret était comparé à celui du front-end (correction indiquée dans le Changelog).

Exemple — flux « client credentials » avec curl :

```
curl -X POST https://192.168.1.10/sigl/services/oauth/token \
  -d grant_type=client_credentials \
  -d client_id=dpi-hopital \
  -d client_secret=XXXXXXXX \
  -d "scope=external/record external/result"
```

Ce que dit chaque ligne de la commande :

Ligne	Signification
`curl -X POST …/sigl/services/oauth/token`	Envoie une demande au **point de jeton** du serveur LabBook
`-d grant_type=client_credentials`	« Je suis un logiciel, pas une personne » : flux serveur à serveur
`-d client_id=dpi-hopital`	Identifiant du logiciel, déclaré dans LabBook (§ 2.2)
`-d client_secret=XXXXXXXX`	Secret (mot de passe) du logiciel, saisi au § 2.2
`-d "scope=external/record external/result"`	Droits demandés : ici les dossiers et les résultats (§ 2.3)

LabBook répond par un objet JSON qui contient le jeton (`access_token`), son type (`Bearer`) et sa durée de validité en secondes (`expires_in`). Exemple de réponse :

```
{
  "access_token": "a1b2c3d4e5f6...",
  "token_type": "Bearer",
  "expires_in": 7200
}
```

Le logiciel joint ensuite ce jeton dans l'en-tête de **chaque** appel à l'API :

```
Authorization: Bearer <access_token>
```

Il n'y a **pas de jeton de rafraîchissement** (refresh token) : au bout de 2 heures, le jeton n'est plus accepté et le logiciel doit **demander un nouveau jeton** au point de jeton. Prévoyez ce renouvellement dans le connecteur, par exemple dès qu'un appel renvoie une erreur 401.

> **Cycle du jeton.** 1) Le logiciel obtient un jeton avec la commande ci-dessus. 2) Il l'utilise pour tous ses appels pendant **2 heures**. 3) À l'expiration, LabBook répond **401** : le logiciel **relance la même commande** pour obtenir un nouveau jeton, puis reprend ses appels. Aucune prolongation automatique n'existe.

3. Tester l'API avec Swagger

1. Ouvrez `http(s)://<serveur>/sigl/api` et connectez-vous avec l'utilisateur API.
2. Cliquez sur **Authorize** (cadenas en haut à droite), cochez les scopes souhaités et validez. Swagger obtient un jeton par le flux *authorization code + PKCE*.
3. Dépliez un service, cliquez sur **Try it out**, complétez les paramètres ou le corps JSON, puis **Execute**.
4. Lisez la réponse : le code HTTP, puis le corps JSON.

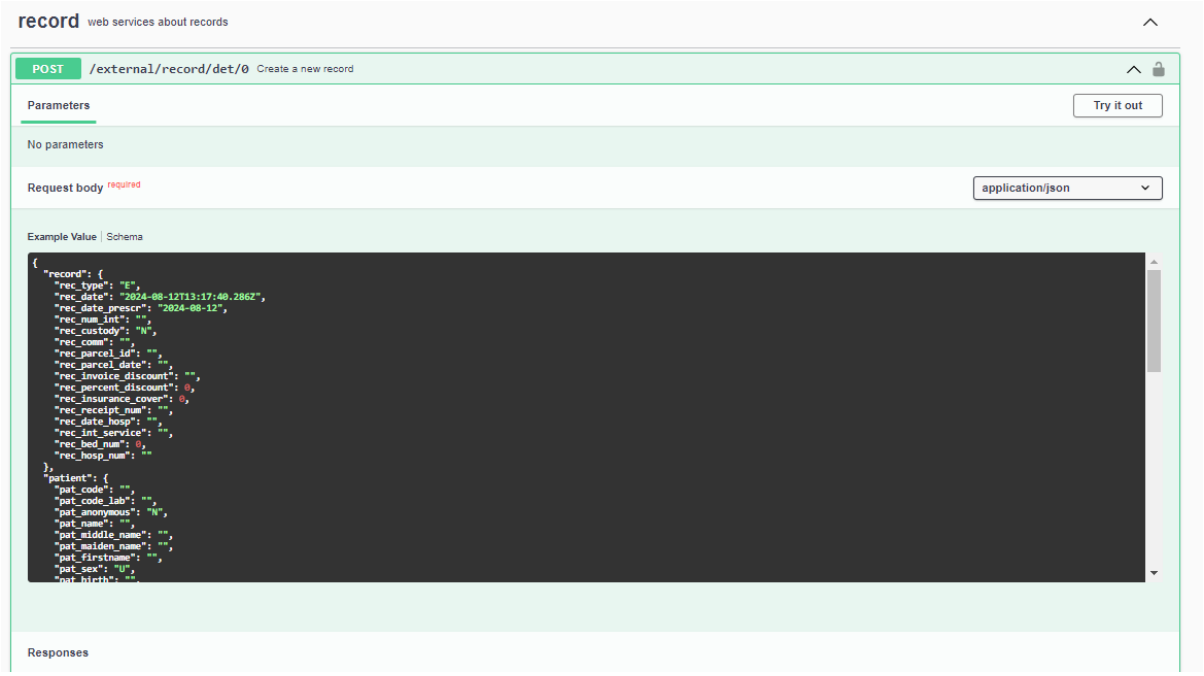

Figure 3 – Création d'un dossier dans Swagger (POST /external/record/det/0)

4. Les services en détail

4.1 Créer un dossier — `POST /external/record/det/0`

Crée une **demande d'analyses** complète en un seul appel : dossier, patient, prescripteur, analyses et prélèvements.

Recherche du patient et du prescripteur : LabBook réutilise les fiches existantes quand il les retrouve.

- **Patient** : d'abord par `pat_code`, puis par `pat_code_lab`, puis par la combinaison *nom + prénom + sexe + date de naissance (ou âge et unité)*. S'il ne trouve rien, il crée un nouveau patient.
- **Prescripteur** : par `prescr_code`, puis par *nom + prénom*.

Exemple de corps de requête (patiente externe, glycémie et créatininémie, prélèvement sanguin fait) :

```
{
  "record": {
    "rec_type": "E",
    "rec_date": "2026-10-23 08:15:00",
    "rec_date_prescr": "2026-10-22",
    "rec_num_int": "DPI-2026-004512",
    "rec_custody": "N",
    "rec_comm": "Bilan pré-opératoire"
  },
  "patient": {
    "pat_code_lab": "IPP-778812",
    "pat_name": "TEST",
    "pat_firstname": "Awa",
    "pat_sex": "F",
    "pat_birth": "1988-04-12",
    "pat_phone1": "770000000"
  },
  "prescriber": {
    "prescr_code": "PR10",
    "prescr_title": "DO",
    "prescr_name": "EXEMPLE",
    "prescr_firstname": "Moussa"
  },
  "ana_list": [
    { "ana_code": "B005", "ana_emer": "N", "ana_req": "Y", "ana_out": "N" },
    { "ana_code": "B004", "ana_emer": "N", "ana_req": "Y", "ana_out": "N" }
  ],
  "samp_list": [
    { "samp_ana": "B005", "samp_type": "BLD", "samp_status": "D",
      "samp_date": "2026-10-23 08:00:00", "samp_code": "HE000125" }
  ]
}
```

Réponse :

- `200` : le dossier est créé ; la réponse contient l'**identifiant du dossier**.

- `4xx` / `5xx` : erreur ; lisez le message (champ manquant, code d'analyse inconnu, format de date…).

4.2 Lister les dossiers — `POST /external/record/list/{id_pres}`

- `{id_pres}` : identifiant du prescripteur, ou `0` pour tous les prescripteurs.
- Filtres du corps : `num_rec`, `stat_work`, `stat_rec`, `lastname`, `firstname`, `code`, `date_beg`, `date_end`, `type_ana`, `code_pat`, `emer`.

Exemple : dossiers du 1er au 23 octobre 2026, tous prescripteurs :

```
POST /sigl/services/external/record/list/0
{ "date_beg": "2026-10-01", "date_end": "2026-10-23" }
```

4.3 Rechercher un patient — `POST /external/patient/list`

Filtres : `code`, `code_lab`, `lastname`, `firstname`.

Exemple : `{ "code_lab": "IPP-778812" }`

4.4 Lire les résultats — `GET /external/result/{id_rec}`

Renvoie les analyses, les variables et les résultats du dossier `{id_rec}`.

Code	Signification
`200`	Succès
`401`	Authentification absente ou invalide
`404`	Dossier introuvable ou non autorisé

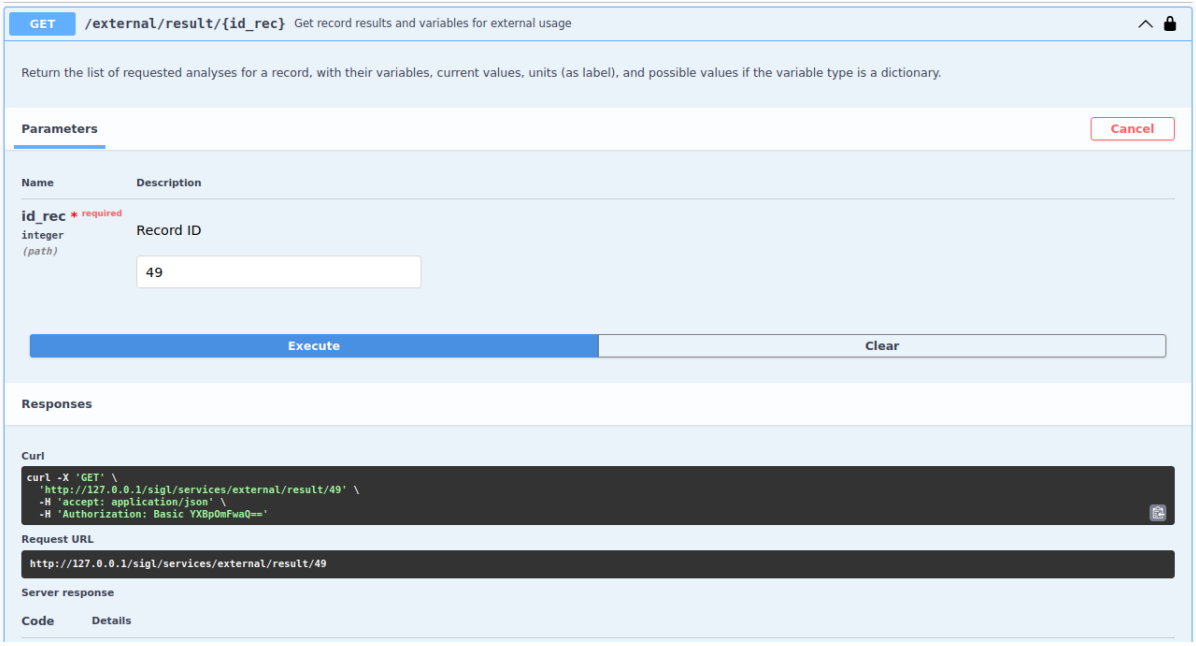

Figure 4 – Lecture des résultats d'un dossier

4.5 Envoyer des résultats — `POST /external/result/{id_rec}`

Met à jour la valeur d'un ou plusieurs résultats. Chaque résultat est désigné par son identifiant `id_res`, obtenu avec `GET /external/result/{id_rec}`.

Exemple :

```
POST /sigl/services/external/result/1542
{ "list_results": [
    { "id_res": 88231, "value": "5.4" },
    { "id_res": 88232, "value": "72" }
] }
```

Code	Signification
`200`	Tous les résultats ont été mis à jour
`207`	Succès partiel : certains résultats n'ont pas pu être mis à jour
`400`	Paramètres manquants ou invalides
`401`	Authentification absente ou invalide
`500`	Erreur interne

> **À savoir** : les résultats envoyés par l'API sont **enregistrés**, mais **pas validés**. La validation technique et la validation biologique restent faites dans LabBook par le personnel habilité.

4.6 Référentiel et utilisateurs

Service	Corps / paramètres	Usage
`POST /external/analysis/list`	`status`, `name`, `type_ana`, `type_prod`	Lister les analyses (ex. analyses actives d'une famille)
`GET /external/analysis/det/{id_ana}`	identifiant de l'analyse	Détail d'une analyse et de ses variables
`GET /external/analysis/code/check/{code}`	code	Vérifier qu'un code n'est pas déjà utilisé
`POST /external/user/list`	`status`, `login`, `lastname`, `firstname`	Lister les utilisateurs

Annexe A — Schéma de données de création d'un dossier

Tous les champs sont de type texte (`string`) sauf mention contraire. Les dates suivent le format indiqué. Les valeurs `Y` / `N` signifient oui / non.

Objet `record` (dossier)

Champ	Type	Défaut	Valeurs / format	Description
`rec_type`	string	`E`	`E`, `I`	Type de dossier : E = externe, I = hospitalisé (interne)
`rec_date`	string		`AAAA-MM-JJ HH:MM:SS`	Date et heure de réception
`rec_date_prescr`	string		`AAAA-MM-JJ`	Date de prescription
`rec_num_int`	string			Numéro de dossier interne au laboratoire
`rec_custody`	string	`N`	`Y`, `N`	Demande de garde (O/N)
`rec_comm`	string			Commentaire / renseignements cliniques
`rec_parcel_id`	string			Identifiant du colis (envoi d'échantillons)
`rec_parcel_date`	string		`AAAA-MM-JJ HH:MM:SS`	Date et heure de réception du colis
`rec_invoice_discount`	string		`S`, `E`, `O`	Remise : S = personnel, E = exonération, O = autre
`rec_percent_discount`	number	`0`		Pourcentage de remise
`rec_insurance_cover`	number	`0`		Pourcentage de prise en charge (assurance)
`rec_receipt_num`	string			Numéro de quittance
`rec_date_hosp`	string		`AAAA-MM-JJ`	Date d'admission (hospitalisé)
`rec_int_service`	string			Service demandeur
`rec_bed_num`	integer	`0`		Numéro de lit
`rec_hosp_num`	string			Numéro d'identification hospitalière

Objet `patient`

Champ	Type	Défaut	Valeurs / format	Description
`pat_code`	string			Code patient LabBook ; s'il est renseigné, LabBook recherche ce patient en priorité
`pat_code_lab`	string			Code patient interne au laboratoire (recherche après pat_code)
`pat_anonymous`	string	`N`	`Y`, `N`	Patient anonyme (nom et sexe non enregistrés)
`pat_name`	string			Nom
`pat_middle_name`	string			Deuxième prénom
`pat_maiden_name`	string			Nom de jeune fille
`pat_firstname`	string			Prénom
`pat_sex`	string	`U`	`M`, `F`, `U`	Sexe : M, F, U = inconnu
`pat_birth`	string		`AAAA-MM-JJ`	Date de naissance
`pat_date_approx`	string	`N`	`Y`, `N`	Date de naissance approximative
`pat_age`	integer	`0`		Âge
`pat_age_unit`	string	`Y`	`D`, `W`, `M`, `Y`	Unité de l'âge : D = jours, W = semaines, M = mois, Y = années
`pat_address`	string			Adresse
`pat_pbox`	string			Boîte postale
`pat_area`	string			Quartier / secteur
`pat_zipcode`	string			Code postal
`pat_city`	string			Ville
`pat_phone1`	string			Téléphone 1
`pat_phone2`	string			Téléphone 2
`pat_job`	string			Profession
`pat_nationality`	string			Code nationalité
`pat_resident`	string	`Y`	`Y`, `N`	Résident
`pat_blood_group`	string		`A`, `B`, `AB`, `O`	Groupe sanguin
`pat_blood_rhesus`	string		`+`, `-`	Rhésus

Objet `prescriber` (prescripteur)

Champ	Type	Défaut	Valeurs / format	Description
`prescr_code`	string			Code prescripteur ; s'il est renseigné, LabBook recherche ce prescripteur
`prescr_title`	string		`MA`, `SI`, `MI`, `DO`, `PR`	Titre : MA = Madame, SI = Monsieur, MI = Mademoiselle, DO = Docteur, PR = Professeur
`prescr_name`	string			Nom
`prescr_firstname`	string			Prénom
`prescr_initial`	string			Initiales
`prescr_workplace`	string			Lieu de travail
`prescr_service`	string			Service
`prescr_address`	string			Adresse
`prescr_zipcode`	string			Code postal
`prescr_city`	string			Ville
`prescr_spe`	string		liste	Spécialité (34 codes : GEN = généraliste, PEDI = pédiatre, GYNE = gynécologue, INF = infectiologue…)
`prescr_phone`	string			Téléphone fixe
`prescr_mobile`	string			Téléphone mobile
`prescr_fax`	string			Fax
`prescr_email`	string			Email

Tableau `ana_list` (analyses demandées)

Champ	Type	Défaut	Valeurs / format	Description
`ana_code`	string			Code de l'analyse dans le référentiel
`ana_loinc`	string			Code LOINC (alternative au code)
`ana_emer`	string	`N`	`Y`, `N`	Analyse urgente
`ana_req`	string	`Y`	`Y`, `N`	Analyse demandée
`ana_out`	string	`N`	`Y`, `N`	Analyse sous-traitée

Tableau `samp_list` (prélèvements)

Champ	Type	Défaut	Valeurs / format	Description
`samp_ana`	string			Code de l'analyse concernée
`samp_date`	string		`AAAA-MM-JJ HH:MM:SS`	Date et heure du prélèvement
`samp_date_receipt`	string		`AAAA-MM-JJ HH:MM:SS`	Date et heure de réception de l'échantillon
`samp_type`	string		liste	Nature : BLD = sang, URN = urine, STL = selles, SPUT = crachat, CSF = LCR… (21 codes)
`samp_status`	string	`T`	`D`, `T`, `P`	Statut : D = fait, T = à faire, P = apporté
`samp_code`	string			Code du prélèvement (tube)
`samp_name`	string			Nom du préleveur
`samp_comm`	string			Commentaire

Annexe B — Historique du document

Version	Date	LabBook	Modifications
3.1	10/2026	3.6.28	§ 2.4 corrigé : durée des jetons 2 heures (et non 1 heure), rôle des points d'autorisation et de jeton expliqué, PKCE obligatoire, envoi du secret client, absence de refresh token, version minimale 3.6.25 pour les clients tiers
3.0	10/2026	3.6.28	Refonte : authentification OAuth 2.0 (clients, scopes, jetons), exemples JSON et curl, schémas en tableaux texte (et non plus en images), codes de réponse, règles de recherche patient/prescripteur
2	01/2026	3.6	Version précédente