

Paramètres et sécurité

Du réglage au mot de passe maître

Référence de la fenêtre des préférences, du magasin qui les tient, des cinq verrous qui gardent l'application, et de ce qu'il reste à faire quand le mot de passe maître est perdu.

Sommaire

Comprendre

1. Modèle mental	4
2. Le cycle d'un réglage	5
3. Où vit un réglage	6
4. Déclarer une option	7
5. La fenêtre	8
6. Les panneaux qui ne sont pas des listes	10

Régler

7. Le poste	12
8. La science	14
9. Le système	15
10. Profil transportable et politique d'établissement	16

Sécuriser

11. Cinq verrous, et l'ordre du démarrage	18
12. La licence	19
13. Chiffrer la base	20
14. Vivre avec le mot de passe maître	21
15. Perdre le mot de passe maître	22
16. Comptes, sessions et verrouillage	24
17. Le code de récupération de compte	25
18. Accès API externe	26
19. Journal d'audit et intégrité	27

Référence

20. Routes appelées	29
21. Clés de stockage	31
22. Dépannage	32
23. Doublons et dettes connus	33

Comprendre

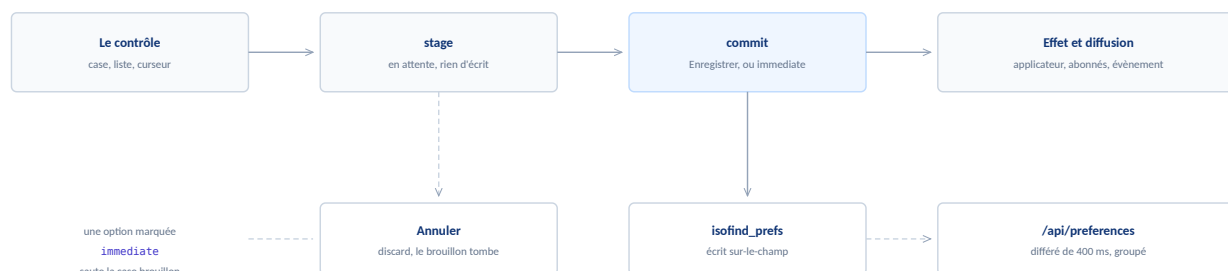
Ce qu'est un réglage, où il vit, quand il prend effet, et ce que la fenêtre en montre.

1. Modèle mental

Cinq objets, et le plus utile à distinguer est le premier des deux derniers : une valeur mise en attente n'est pas une valeur enregistrée. Toute la fenêtre travaille en différé, et la moitié des surprises vient de là.

Mot	Ce que c'est	Où il vit
Schéma	Le catalogue. Une option s'y déclare et nulle part ailleurs : le magasin en tire son défaut et ses contraintes, l'interface son libellé et son contrôle, l'applicateur son effet.	preferences_schema.js , en mémoire.
Magasin	Ce que chaque option vaut. Il tient trois tables : les valeurs enregistrées, les valeurs en attente, et celles que la politique a verrouillées.	preferences_store.js , adossé au stockage local et au compte.
Applicateur	Une table d'effets indexée par identifiant. Une option qui n'y figure pas n'a aucune conséquence immédiate : elle est lue au moment où elle sert.	preferences_applicator.js .
Brouillon	Une valeur mise en attente par stage . Elle s'affiche, elle compte dans le pied de la fenêtre, et elle ne touche encore à rien.	En mémoire, jusqu'à Enregistrer ou Annuler.
Enregistré	Ce que commit a écrit. Le stockage local est écrit tout de suite, le compte quatre cents millisecondes plus tard, et les abonnés sont prévenus.	isofind_prefs , puis /api/preferences .

Le trajet d'une valeur



Les deux temps sont séparés pour que la fenêtre puisse annuler. L'ancienne version écrivait directement dans l'objet des préférences, ce qui court-circuitait l'applicateur et obligeait à rappeler la propagation à la main après coup.

Trois idées qui expliquent le reste

1. Le schéma est la seule liste. Ajouter une option se fait par une entrée dans le catalogue. L'interface dérive les champs de la déclaration, sans gabarit par section, et l'applicateur ne connaît que les identifiants qui ont un effet. Une table tenue en parallèle du schéma se désynchronise au premier ajout, ce qui était exactement le défaut de la version précédente.

2. Une valeur n'est jamais recopiée chez son propriétaire. Quand un autre module tient déjà l'état, l'option le délègue par un [proxy](#) et n'en garde aucune copie ici. Deux copies du même état finissent toujours par diverger, et c'est l'interface qui affiche alors la fausse.

3. Ce qui est affiché est mesuré, pas déduit. Un chemin renseigné ne dit pas que le programme est là. Les sondes interrogent le poste, et la barre de cohérence ne signale que des couples réglage plus mesure : « X est activé et son programme est introuvable » est un fait, « vous devriez activer X » serait un avis.

2. Le cycle d'un réglage

Trois écritures possibles, et il faut savoir laquelle on emploie. `stage` met en attente, `commit` écrit et propage, `set` fait les deux d'affilée pour les appelants qui ne sont pas dans la fenêtre.

Appel	Ce qu'il fait
<code>stage(id, valeur)</code>	Compare à la valeur courante. Égale, le brouillon est retiré ; différente, il est posé. Rend faux si l'option est verrouillée par la politique.
<code>commit([ids])</code>	Écrit les brouillons, appelle le propriétaire pour les options déléguées, persiste, puis prévient les abonnés. Rend la liste de ce qui a effectivement changé.
<code>set(id, valeur)</code>	Un <code>stage</code> suivi d'un <code>commit</code> sur cette seule option.
<code>effective(id)</code>	Ce que l'interface doit afficher : le brouillon s'il existe, la valeur courante sinon.
<code>discard()</code>	Jette tous les brouillons. C'est le bouton Annuler.

La comparaison de `stage` porte sur la valeur *courante*, donc sur celle du propriétaire quand l'option est déléguée. Comparer à la copie locale laissait un réglage actif face à un défaut resté faux : décocher se lisait comme « rien n'a changé », le changement était effacé, et l'option ne pouvait plus revenir en arrière.

Un panneau qui écrit avec `set` est un panneau qu'on ne peut pas annuler. Le panneau Modules a commis cette faute : ses interrupteurs enregistraient sur-le-champ, le pied de la fenêtre annonçait « aucun changement en attente » alors qu'on venait d'en faire un, et Enregistrer n'avait rien à enregistrer. Un panneau personnalisé doit employer `stage` comme le reste de la fenêtre, et prévenir le pied par `refreshDirty`, que le circuit des lignes ordinaires appelle pour lui.

Les options immédiates

Une option marquée `immediate` ignore le découpage : activer puis fermer doit suffire. C'est le cas de la base communautaire, dont l'effet est d'ouvrir un accès et que personne n'ira chercher derrière un bouton Enregistrer.

La diffusion

Après écriture, le magasin prévient de deux façons : les abonnés inscrits par `subscribe`, et un évènement `preferenceChanged` sur la fenêtre. Le second sert aux modules qui ne connaissent pas le magasin, en particulier ceux qui peignent en JavaScript. Le thème émet en plus son propre évènement `themechange`, parce que les graphiques et les bacs à sable des extensions ne lisent pas le CSS et doivent être avertis.

3. Où vit un réglage

Six endroits possibles, et le schéma dit lequel pour chaque option. Savoir lequel répond à deux questions pratiques : ce qui suit l'utilisateur d'un poste à l'autre, et ce qui reste quand le backend ne répond pas.

Emplacement	Ce qui y va	Conséquence
Stockage local	La table complète, sous <code>isofind_prefs</code> .	L'interface part sur ce qu'elle a, sans attendre le réseau.
Compte	Les mêmes valeurs, moins les verrouillées et les déléguées.	Les réglages suivent la personne d'un poste à l'autre.
Les deux	<code>language</code> et <code>theme</code> .	Lisibles avant l'authentification. La duplication est le but, pas un oubli.
Propriétaire délégué	Ce qu'un autre module tient déjà, par <code>proxy</code> .	Rien n'est recopié ici. La base communautaire est lue et écrite dans sa propre clé.
Politique du poste	<code>isofind_prefs_policy</code> , posée par l'installateur.	Verrouillé et non modifiable depuis l'application.
Côté Rust ou backend	La planification de sauvegarde, l'épinglage IGL, la langue des menus natifs, la réduction en zone de notification.	L'applicateur pose le réglage chez son vrai propriétaire.

Le poste, puis le compte

Au démarrage, le magasin lit le stockage local, applique la politique, synchronise la langue, migre si nécessaire, et démarre. La lecture du compte part en parallèle et corrige l'affichage quand le serveur répond. Attendre le réseau pour afficher une fenêtre reviendrait à faire dépendre l'apparence de l'application de la disponibilité du backend.

À la première connexion d'un compte, le serveur ne connaît rien de lui : ce que le poste avait est repris et remonté. C'est la migration, et elle n'a lieu qu'une fois, la ligne existant ensuite.

Backend arrêté, poste en air-gap sans service, session non authentifiée : la lecture échoue et on continue sur le stockage local. Des préférences qui cesseraient de fonctionner faute de réseau seraient pires que des préférences partagées.

Ce que le poste oublie quand un compte s'en va

`forgetLocal` remet chaque option à son défaut à la déconnexion et au changement d'utilisateur. Sans cela, le cache local reste celui du précédent, et le suivant travaille sous ses réglages entre le démarrage et la réponse du serveur.

La langue et le thème survivent : ils décrivent le poste autant que la personne, et les effacer ferait clignoter l'interface à chaque changement de compte.

Deux envois, deux raisons

L'écriture vers le compte est temporisée de quatre cents millisecondes : dix cases cochées d'affilée ne doivent pas produire dix écritures. La temporisation suppose seulement que la page vivra assez longtemps pour la voir expirer, ce qui est faux pour tout réglage dont l'effet est de changer de page. `vider` force l'envoi, et il est appelé sur `pagehide` et sur le passage en arrière-plan. La requête part avec `keepalive` , sans quoi un envoi parti à temps serait annulé par la redirection que le réglage vient de déclencher.

4. Déclarer une option

Une entrée du catalogue porte ce qu'elle vaut, ce qu'elle montre, et sous quelles conditions elle est disponible. Rien d'autre n'est à écrire : ni gabarit, ni feuille de style, ni branche dans l'applicateur si l'option n'a pas d'effet immédiat.

```
{
  id: 'data.backupInterval',          // identifiant, prefixe par domaine
  section: 'data',                    // ou elle s'affiche
  type: 'number',                      // boolean, number, enum, multi, text,
                                      // path, tags, action, info, custom
  control: 'number',                  // number, slider, select, segmented
  default: 24, min: 1, max: 168, step: 1,
  unit: { fr: 'heures', en: 'hours' },
  label: { fr: 'Intervalle', en: 'Interval' },
  description: { fr: '...', en: '...' },
  gate: { requires: 'data.backupEnabled' },
  badges: ['restart']
}
```

Champ	Rôle
<code>type</code>	Décide de la contrainte appliquée à la valeur et du contrôle rendu. <code>action</code> pose un bouton, <code>info</code> une mesure en lecture seule, <code>custom</code> monte un panneau entier.
<code>gate</code>	Condition de disponibilité, évaluée à l'affichage : <code>airgap: false</code> , <code>requires: 'autreOption'</code> , <code>capability: 'x'</code> , <code>comingSoon: true</code> , <code>present: 'igl'</code> .
<code>badges</code>	Mentions à côté du libellé : redémarrage, air-gap, pro, développeur, expérimental, bientôt.
<code>immediate</code>	L'option se commit dès qu'on la touche.
<code>proxy</code>	Lecture et écriture déléguées au module qui tient déjà l'état.
<code>choicesOf</code> , <code>source</code>	Liste construite à l'exécution : modèles d'un serveur, bases IGDB présentes, écrans détectés.

Une liste construite à l'exécution ne peut pas servir de garde. Un noyau apporté par une extension absente de cette session ferait rejeter un choix parfaitement valide au prochain démarrage. Pour un `enum`, la contrainte laisse donc passer toute chaîne quand la liste est dynamique. Pour un `multi`, ce garde-fou manquait et la conséquence était pire qu'un rejet visible : le filtre vidait le tableau en silence, aucune case ne restait cochée, et rien ne disait pourquoi.

Les identifiants hérités

Certains restent à plat, `language`, `theme`, `nexusGridSize`, parce que d'autres modules les lisent déjà par ce nom. Les nouveaux sont préfixés par domaine. C'est ce qui a permis de ne pas écrire de migration au premier démarrage de la nouvelle fenêtre.

Le mécanisme de migration existe cependant, indexé par version de départ, et le magasin l'exécute au chargement quand la version du poste est en retard. Une clé inconnue est conservée telle quelle : elle appartient sans doute à une extension absente de cette session, et l'écraser reviendrait à perdre sa configuration à chaque démarrage.

5. La fenêtre

Navigation latérale par groupes, recherche, badges de portée, et un pied qui dit ce qui attend d'être enregistré. Le rendu est générique : ajouter une option ne demande de toucher ni le fichier d'interface ni la feuille de style.

Élément	Ce qu'il fait
Groupes	Poste, Science, Système, Extensions. Une section vide ou fermée par sa condition n'apparaît pas.
Recherche	Porte sur les libellés et les descriptions, toutes sections confondues, et rend les lignes à plat avec le nom de leur section.
Barre de cohérence	En tête du contenu. Chaque item est un bouton qui mène à la section concernée.
Pied	Le nombre de changements en attente, leur détail au survol, Annuler et Enregistrer.
Fermeture	Avec des changements en attente, la fenêtre demande quoi en faire au lieu de les jeter.

Une option indisponible reste visible

Fermée par sa condition, elle s'affiche inerte, avec la raison. La masquer donnerait l'impression qu'elle n'existe pas, alors que la question utile est pourquoi elle est fermée. C'est le même raisonnement qui garde le canal de mise à jour affiché et grisé : l'application n'embarque aucun mécanisme de mise à jour automatique, et une option absente laisserait croire qu'il en existe un ailleurs.

Les sondes

Une sonde interroge le poste et rend trois choses : si elle a pu mesurer, ce qu'elle a trouvé, et un libellé lisible. La distinction entre « mesuré et faux » et « pas mesuré » est ce qui empêche la barre de cohérence de crier sur un poste en bon état : une sonde qui n'a pas pu mesurer ne produit aucun avertissement, faute de quoi une commande manquante serait rapportée comme un réglage incohérent.

Sonde	Mesure
<code>igl</code>	Résout le binaire et l'interroge. Affiche la version et l'origine retenue, puisque plusieurs installations peuvent coexister.
<code>phreeqc</code>	Le chemin résolu de l'exécutable, épingleage compris.
<code>ollama</code>	Passe par le connecteur et non par un appel direct : c'est lui qui tient le transport et refuse toute adresse non locale.
<code>ollamaModelTools</code>	Si le modèle choisi déclare l'appel d'outils. Le nom du modèle voyage dans le nom de la sonde, sans quoi le cache rendrait la réponse du précédent.
<code>reportSigning</code>	Si un certificat de signature est installé.
<code>dbLocation</code> , <code>reportsDir</code>	Les emplacements, employés aussi par les boutons d'ouverture de dossier.

La barre de cohérence

Un réglage peut être valide et pourtant sans effet, parce qu'il attend quelque chose qui n'est pas là. Rien ne le disait : l'option restait cochée, et le silence se lisait comme un fonctionnement.

- Couplage PHREEQC activé, exécutable introuvable.
- Assistant activé, serveur de modèles injoignable, ou aucun modèle choisi.
- Modèle choisi ne déclarant pas l'appel d'outils, donc repli sur la lecture des réponses.
- Signature des rapports demandée, aucun certificat installé.

- Réglages IGL définis, aucune installation détectée.
- Sauvegardes planifiées sans intervalle.

6. Les panneaux qui ne sont pas des listes

Trois sections ne se rendent pas comme des champs : Modules, Sécurité et Diagnostic. Elles se déclarent en type `custom` et montent leur propre panneau, enregistré dans `IsoFindPrefsPanels`.

Modules

Les modules sont groupés par l'offre qui les a ouverts, et non alignés à plat : une liste de trente entrées par ordre alphabétique demande de savoir déjà ce qu'on cherche. Une offre ne fait groupe que si elle est entièrement ouverte, c'est-à-dire si tout ce qu'elle accorde l'est. Compter les modules en commun rangeait un module partagé sous une offre que le client n'avait pas achetée, au seul motif que celle-ci en partageait moins.

Un module appartenant à plusieurs offres ouvertes va sous la plus petite, donc la plus spécifique, et à égalité sous la première par ordre alphabétique, pour que l'affichage soit stable d'une ouverture à l'autre. Ce qu'aucune offre ouverte ne réclame est rangé sous le complément qui l'apporte ou, à défaut, sous « À la carte ».

Il n'y a pas d'interrupteur « désactiver le module »

Le backend décide au démarrage, sur la licence, et un réglage d'interface ne peut pas le contredire sans mentir. Ce que l'utilisateur veut réellement, c'est ne plus voir ce qu'il n'utilise pas : les interrupteurs masquent donc des pages dans la barre latérale, les routes restant servies.

La page ouverte n'est jamais masquée. Sinon l'entrée active disparaît sous les pieds de qui la consulte, et la barre ne dit plus où l'on est ; le réglage reprend effet dès qu'on la quitte.

Sécurité

Accès API externe, clients en attente, clients autorisés, et code de récupération de compte. La section porte un compteur dans la navigation, alimenté par le nombre de demandes en attente : c'est la seule section dont l'état réclame une action dans un délai. Elle est décrite en partie III.

Diagnostic

L'état des dépendances, mesuré par les mêmes sondes que la barre de cohérence. C'est la page à ouvrir avant d'écrire un rapport de panne : elle dit ce que le poste a trouvé, et non ce que les réglages prétendent.

Régler

Le tour des sections, section par section, avec ce que chaque réglage touche vraiment et ce qu'il ne touche pas.

7. Le poste

Trois sections décrivent la machine et la façon d'y travailler : Général, Apparence, Espace de travail. Ce sont les seules dont les valeurs sont aussi écrites en local, langue et thème devant être lisibles avant l'authentification.

Général

Réglage	Ce qu'il fait
<code>language</code>	Français ou anglais. L'effet écrit trois clés locales, applique les traductions, et prévient la couche Rust pour les menus natifs, ce qui demande un redémarrage.
<code>startupView</code>	La page ouverte au lancement.
<code>minimizeToTray</code>	Réduire dans la zone de notification. La valeur vit côté Rust, dans l'état d'interface.
<code>resetCloseChoice</code>	Une action, pas un réglage : redemander au prochain clic sur la croix ce qu'il faut faire de la fenêtre.
<code>autoSave</code> , <code>autoSaveInterval</code>	Enregistrement automatique et sa période.
<code>confirmDelete</code>	Confirmation avant suppression.
<code>dateFormat</code>	Format des dates affichées.

La langue ne remonte pas vers Tauri au démarrage. Elle n'a pas changé, les menus natifs la portent déjà, et la boîte de redémarrage s'ouvrirait donc à chaque lancement pour annoncer un changement qui n'avait pas eu lieu. Le même garde-fou vaut pour la réduction en zone de notification, que le Rust tient déjà dans son fichier d'état.

Apparence

Le thème n'est pas un booléen mais un identifiant de palette, posé en attribut sur la racine du document. Un attribut porte une valeur, une classe n'en porte pas, et c'est ce qui permet d'en ajouter autant qu'on veut sans toucher au code : une palette apportée par une extension se range correctement puisque le socle dit lui-même, par une variable, si elle est claire ou sombre.

Réglage	Ce qu'il fait
<code>theme</code>	Système, clair, sombre, ardoise. <code>auto</code> suit la préférence du système et se recalcule à chaque application.
<code>fontSize</code>	Posée sur la racine seulement. La poser aussi sur le corps appliquait l'échelle deux fois à toute taille exprimée en rem.
<code>appearance.uiScaleAuto</code>	Compenser la mise à l'échelle de Windows. Le facteur est calculé côté Rust, seul à connaître la taille physique du moniteur.
<code>uiZoom</code>	Le zoom du menu Affichage. Recopié dans le compteur de ce menu, sans quoi le premier zoom avant repartirait de cent.
<code>compactMode</code> , <code>animationsEnabled</code>	Deux classes sur le corps du document.
<code>monoFont</code>	Police à chasse fixe du carnet, d'IGL et des journaux. Vide pour celle du système.

Espace de travail

C'est la section la plus fournie, parce qu'elle décrit le shell : navigation par onglets ou par pages, pages masquées, barre d'état et ses indicateurs, mode du menu principal, comportement à la fermeture, restauration des panneaux et des fenêtres flottantes, rail, panneau bas élargi, détachement des panneaux et ses quatre réglages, noyau du carnet et coloration syntaxique.

Deux d'entre eux méritent une note. `workspace.navigat`ion redirige, parce que le choix porte sur la page servie à la racine et qu'aucun réglage d'interface ne peut le rendre en place ; il ne redirige jamais depuis un onglet, où la redirection remplacerait le contenu du cadre et non la fenêtre. `nav.hiddenPages` pose un attribut au lieu de retirer le nœud, parce que le système d'édition retire déjà ceux que la licence n'ouvre pas, et deux mécanismes qui suppriment le même élément ne savent plus lequel doit le rendre.

8. La science

Quatre sections : Analyse et calcul, Nexus, IGL, Rapports et exports. Ce sont des valeurs par défaut, appliquées à l'affichage et aux calculs sauf réglage propre à une étude.

Analyse et calcul

Réglage	Ce qu'il fait
<code>defaultPrecision</code>	Décimales affichées.
<code>defaultUnits</code>	Unité par défaut des compositions.
<code>analysis.deltaNotation</code>	Rendu typographique ou ASCII des deltas.
<code>analysis.defaultStandard</code>	Standard proposé quand un import n'en nomme pas.
<code>analysis.relativeMode</code>	Travailler en écarts plutôt que contre un standard absolu.
<code>showUncertainty</code> , <code>analysis.propagation</code>	Affichage des incertitudes et méthode de propagation.

Nexus

Minimap, magnétisme et pas de la grille pour l'éditeur de graphe ; tolérance, nombre maximum de candidats et graine pour le moteur. La graine à zéro laisse le moteur décider, ce qui n'est pas la même chose que zéro comme valeur de graine : c'est le réglage à fixer quand on veut deux exécutions identiques.

IGL

La section porte un état plutôt qu'un titre seul : IsoFind s'adapte à l'installation présente sur le poste, et la première chose à savoir est laquelle a été retenue.

Réglage	Ce qu'il fait
<code>igl.binaryPath</code> , <code>igl.home</code>	Épinglage. Les deux partent ensemble vers la couche Rust, qui les traite comme un couple : le chemin choisi ne sert à rien tant qu'elle ne le connaît pas, puisque c'est elle qui résout le binaire.
<code>igl.profile</code>	Core, identification, étendu.
<code>igl.engine</code>	Moteur Rust ou moteur Nexus.
<code>igl.databases</code>	Bases IGDB retenues, liste construite à partir de celles qui sont présentes.
<code>igl.phreeqcEnabled</code> , <code>igl.phreeqcPath</code>	Couplage PHREEQC et chemin de l'exécutable. La sonde mesure le second, la barre de cohérence signale l'incohérence entre les deux.
<code>igl.keepIntermediate</code>	Garder le graphe intermédiaire à côté du modèle.
<code>igl.formatOnSave</code> , <code>igl.logEngineCalls</code>	Formatage à l'enregistrement, journalisation des appels au moteur.

Rapports et exports

Langue et format numérique des rapports, image de marque du laboratoire, dossier de sortie, et signature des exports ISOE. Ce dernier réglage est le seul de la section à toucher la sécurité : il demande qu'un certificat de laboratoire soit installé, et la barre de cohérence le dit lorsqu'il est activé sans certificat.

9. Le système

Sept sections, dont trois montent leur propre panneau. Les quatre autres sont des listes de champs ordinaires : Assistant IA, Extensions, Données et sauvegarde, Réseau et Avancé.

Assistant IA

Désactivé par défaut. Hôte, modèle, fenêtre de contexte, outils autorisés, rendu mathématique, conservation des échanges, plafond de tours d'un agent, délai par tour. Les échanges conservés sont écrits dans la base chiffrée du poste, jamais en clair.

Deux réglages sont des garde-fous plutôt que des préférences. Le plafond de tours arrête un agent qui tourne en rond, et il est figé au lancement de la session. Le délai par tour vaut trois minutes par défaut, parce qu'un modèle de quatorze milliards de paramètres sur un portable sans carte graphique peut mettre une minute, et que couper trop tôt fait échouer un travail qui allait aboutir.

Extensions

Réglage	Ce qu'il fait
<code>plugins.directory</code>	Dossier des extensions.
<code>plugins.requireSignature</code>	Exiger une signature valide. Activé par défaut.
<code>plugins.allowUnsigned</code>	Autoriser les extensions non signées. Réservé au développement, et fermé tant que le mode développeur n'est pas actif.
<code>plugins.autoUpdateRegistry</code>	Mise à jour automatique du registre. Fermé en air-gap.

Données et sauvegarde

L'emplacement de la base est affiché et non modifiable. Le déplacer demanderait que le backend rouvre ailleurs au démarrage, donc touche la résolution de chemin et la chaîne SQLCipher : un champ modifiable ici promettait un déplacement que rien n'effectue, et une erreur s'y paierait en base introuvable. Un bouton ouvre le dossier, par la seule ouverture autorisée, celle qui confie le chemin au gestionnaire de fichiers du système sans donner aucun moyen d'exécuter quoi que ce soit.

Les sauvegardes planifiées ont trois réglages, et ils partent ensemble vers le backend : celui-ci tient un seul document, et n'en envoyer qu'un écraserait les deux autres.

Réseau

Le mode réseau est gravé par l'installateur et affiché en lecture seule. La base communautaire est une option immédiate, déléguée à sa propre clé de stockage : sans cette délégation, la case cochant une copie que personne ne consultait, le réglage semblait ne pas se retenir, et l'accès restait fermé. Le proxy relit l'objet avant de le modifier, la clé portant d'autres réglages communautaires.

Avancé

Mode développeur, niveau de journalisation, écriture des journaux sur disque, accélération WebGL, fils de calcul, vidage du cache local, et la réinitialisation générale. Le mode développeur n'ouvre rien par lui-même : il déverrouille les réglages marqués développeur ailleurs dans la fenêtre, le plus notable étant l'autorisation des extensions non signées.

10. Profil transportable et politique d'établissement

Deux mécanismes voisins et de sens opposé : l'un permet à un utilisateur d'emporter ses réglages, l'autre permet à un exploitant de lui en interdire.

Le profil

Déployer un second poste de laboratoire à l'identique se fait par un fichier JSON, exporté depuis la section Données. Il porte un format, une version de schéma, une date et les valeurs.

```
{
  "format": "isofind-preferences",
  "schemaVersion": 2,
  "exported": "2026-08-20T09:14:00.000Z",
  "values": { "theme": "isofind-sombre", "defaultPrecision": 4, ... }
}
```

À l'import, un fichier dont le format ne correspond pas est refusé, et un fichier dont la version de schéma est plus récente que celle du poste aussi : accepter une déclaration qu'on ne sait pas lire reviendrait à poser des valeurs dont on ignore la contrainte. Les valeurs importées sont mises en attente et non écrites, ce qui laisse les relire avant d'enregistrer.

La politique

Les clés verrouillées sont exclues de l'export : elles appartiennent au poste et non à l'utilisateur. Elles sont posées par l'installateur, dans une clé de stockage distincte, et ne sont pas modifiables depuis l'application. Sur un poste sensible, l'exploitant ne doit pas pouvoir rouvrir l'accès API ni quitter l'air-gap parce que quelqu'un a coché une case.

Une option verrouillée s'affiche, inerte, avec sa raison. Le magasin refuse toute mise en attente la concernant, et la réinitialisation générale la laisse en place.

Sécuriser

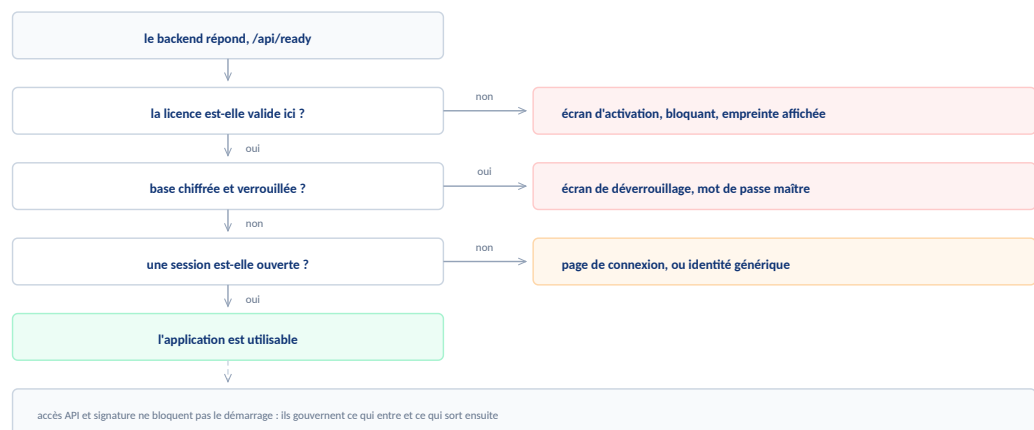
Les cinq verrous, ce que chacun garde, et ce qu'il reste à faire quand le mot de passe maître est perdu.

11. Cinq verrous, et l'ordre du démarrage

Ils sont indépendants et se posent dans un ordre qui n'est pas négociable : chacun garde une chose différente, et franchir l'un ne dit rien des autres. Les confondre est la première cause d'appels au support, parce que trois d'entre eux se présentent comme un écran bloquant qui demande quelque chose à taper.

Verrou	Ce qu'il garde	Ce qui l'ouvre
Licence	Le droit d'exécuter cette édition sur cette machine, et les modules ouverts.	Un fichier de licence signé, lié à l'empreinte de la machine.
Chiffrement de la base	Les données au repos : le fichier de base est illisible sans la clé.	Le mot de passe maître, qui dérive la clé SQLCipher.
Compte et session	Qui agit, et sous quel nom les écritures sont consignées.	Un mot de passe de compte, distinct du précédent.
Accès API externe	Ce qu'un programme tiers peut demander au backend local.	Un mode choisi, et l'autorisation explicite de chaque client.
Signature	L'origine et l'intégrité de ce qui sort : exports ISO, rapports, extensions.	Un certificat de laboratoire, délivré par la PKI.

L'ordre au lancement



Les deux premiers écrans sont bloquants au sens strict : pas de croix, pas de fermeture au clic extérieur, pas d'échappement. Seule une activation réussie retire le premier, seul un déverrouillage réussi retire le second.

Le z-index de la barre de titre est une contrainte de sûreté. Depuis le passage à une fenêtre sans décorations natives, un écran bloquant qui recouvrirait la barre laisserait la fenêtre sans aucun moyen d'être réduite ni fermée. L'écran de déverrouillage monte très haut dans l'empilement, et la barre monte plus haut encore, en se réduisant aux seuls boutons de fenêtre tant qu'un écran bloquant est affiché.

12. La licence

Une licence est un fichier signé, lié à l'empreinte de la machine, qui accorde une édition et une liste de modules. Le contrôle est fermé par défaut : à défaut de réponse claire, l'application ne démarre pas.

L'écran d'activation

Il s'affiche au démarrage quand le backend signale un blocage, et il porte trois choses : l'empreinte de la machine, avec un bouton pour la copier et un autre qui prépare le courriel de demande ; une zone de dépôt qui accepte un fichier `.license` ou `.json` ; et la raison du blocage, traduite en une phrase.

Raison	Ce qu'elle veut dire
<code>no_valid_license</code>	Aucune licence installée, ou elle ne vaut pas pour cette machine.
<code>license_check_error</code> , <code>license_module_unavailable</code>	Le module de vérification n'a pas pu répondre. Ce n'est pas un refus, c'est une absence de réponse, et elle bloque de la même façon.
<code>edition_mismatch</code> , <code>tier_not_for_vertical</code> , <code>vertical_tier_on_isofind</code> , <code>missing_edition_field</code>	La licence ne correspond pas au binaire installé.

Hors blocage, le menu Licence donne l'état, l'empreinte, l'activation et la désactivation, la liste des modules et des certificats, et les demandes d'extension. L'empreinte se lit par deux routes équivalentes, la française étant tentée d'abord et l'anglaise servant de repli pour les déploiements internationaux.

Ce que la licence ouvre, et ce qu'elle ne protège pas

La licence accorde des capacités : une édition, des modules, un nombre d'utilisateurs. Elle ne chiffre rien et ne protège aucune donnée. Une base non chiffrée reste lisible par qui a le fichier, licence ou pas, et c'est le rôle du verrou suivant.

13. Chiffrer la base

Le chiffrement se demande une fois, depuis le menu Protection, et il transforme la base en place. Le mot de passe saisi ce jour-là dérive la clé : il n'est stocké nulle part, et c'est ce qui rend la section suivante nécessaire.

L'activation, pas à pas

Étape	
1	Menu Protection, puis Activer. Le niveau est figé sur le chiffrement réel : le backend n'accepte plus rien d'autre.
2	Une case, cochée par défaut, propose de garder une sauvegarde en clair pendant sept jours. Lire la section 15 avant de la décocher.
3	Le mot de passe maître, deux fois. Huit caractères au minimum, contrôlé côté interface.
4	Une confirmation bloquante, qui répète ce que fait l'opération et dit s'il y aura une sauvegarde ou non.
5	La base est chiffrée, et douze mots s'affichent. Ils ne seront plus jamais affichés.
6	La fenêtre des douze mots ne se ferme pas tant que la case « J'ai noté ces 12 mots » n'est pas cochée. Il n'y a volontairement ni croix ni bouton Annuler : forcer la lecture est l'objectif.

La fenêtre propose de copier la clé dans le presse-papiers et de la télécharger en fichier texte. Le fichier porte un en-tête, la date d'émission, le logiciel, le type de clé, et un avertissement en toutes lettres. La grille des mots est affichée en police à chasse fixe, pour éviter toute confusion entre I et 1, entre O et 0.

Sans SQLCipher, rien ne se passe. Le backend répond alors par un refus explicite que l'interface traduit, plutôt que d'activer une protection de façade. Une base annoncée chiffrée et qui ne le serait pas est pire qu'une base annoncée en clair.

Ce qui est chiffré, et ce qui ne l'est pas

Objet	État
La base : échantillons, analyses, méthodes, campagnes, historiques, échanges de l'assistant	Chiffré par SQLCipher, clé dérivée du mot de passe maître.
Le magasin de certificats ISOF	Chiffré à part, par sa propre clé, dans son propre dossier.
Les préférences, les workflows, les extensions installées	En clair, dans le stockage local du poste.
Les fichiers exportés : ISOF, CSV, rapports	En clair sur le disque, là où on les a écrits. Un ISOF est signé, ce qui n'est pas la même chose que chiffré.
Les journaux d'audit	Dans la base, donc chiffrés avec elle.

14. Vivre avec le mot de passe maître

Une fois la base chiffrée, le mot de passe est demandé à chaque démarrage, avant tout le reste. Trois opérations le concernent, et deux d'entre elles l'exigent.

Déverrouiller

L'écran s'affiche automatiquement quand le statut dit que la base est protégée, réellement chiffrée, et que la clé n'est pas déjà en mémoire. Il est bloquant, sans bouton de fermeture : l'utilisateur peut saisir le mot de passe ou fermer l'application. Entrée valide, la page se recharge après un court délai, pour que les composants qui avaient manqué leurs requêtes avant le déverrouillage se réinitialisent proprement. Entrée fausse, le champ se vide et reprend le focus.

Changer

Menu Protection, Paramètres. Trois champs : l'actuel, le nouveau, sa confirmation. Le nouveau mot de passe doit faire huit caractères au minimum. Changer le mot de passe ne change pas la clé de récupération : les douze mots notés à l'activation restent valides.

Régénérer la clé de récupération

Le bouton n'apparaît que lorsque la protection est au niveau chiffré, le magasin de clés n'existant que dans ce cas. La régénération demande d'abord une confirmation explicite, puis le mot de passe maître. L'ancienne clé est invalidée et la nouvelle s'affiche dans la même fenêtre qu'à l'activation.

La régénération exige le mot de passe maître. Elle sert à remplacer une clé compromise, perdue ou partagée avec quelqu'un qui n'aurait pas dû la voir. Elle ne sert pas à sortir d'un oubli : le jour où le mot de passe manque, ce bouton ne répond plus. C'est pour cette raison que la confirmation prévient que l'ancienne clé cessera de fonctionner, avant même de demander quoi que ce soit.

Désactiver

Le mot de passe actuel est demandé, la base est déchiffrée en place, et elle redevient lisible par qui possède le fichier. C'est une opération légitime, par exemple pour transférer une installation, mais elle n'a pas de retour arrière automatique : il faut réactiver, ce qui produit une nouvelle clé de récupération.

15. Perdre le mot de passe maître

Le mot de passe n'est stocké nulle part : il dérive la clé de chiffrement, et personne, ni le laboratoire, ni l'éditeur, ne peut le retrouver ni le contourner. Cette section dit ce qui reste possible, dans quel ordre l'essayer, et à partir de quel moment la réponse est qu'il n'y a plus rien à faire.

Ce qu'il faut vérifier en premier



Les quatre questions se posent dans cet ordre parce qu'elles vont de la moins destructrice à la plus coûteuse. Aucune ne dépend de l'éditeur, et aucune ne peut être remplacée par un appel au support.

La clé de douze mots

Elle est produite à l'activation, affichée une seule fois, et l'interface la présente comme le seul moyen de retrouver l'accès en cas d'oubli. Elle se régénère depuis les paramètres de protection, à condition de connaître le mot de passe.

L'usage de la clé n'est pas câblé dans l'interface livrée.

L'écran de déverrouillage n'accepte qu'un mot de passe : aucun champ, aucun lien, aucune autre fenêtre n'accepte les douze mots. Dans les fichiers d'interface examinés, la clé est produite, affichée, exportée et régénérée, et rien ne la consomme.

Deux conséquences pratiques. Le détenteur de la clé ne peut pas s'en servir seul depuis l'application, et la promesse écrite dans la fenêtre d'activation, « le seul moyen de récupérer l'accès », n'est pas encore tenue par un chemin visible. Tant que ce chemin n'existe pas, la clé doit être traitée comme une pièce à conserver dont l'usage passera par une procédure outillée, et la sauvegarde de sept jours reste le seul recours réellement disponible depuis le poste.

La fenêtre de sept jours

À l'activation, la case « conserver une sauvegarde en clair » est cochée par défaut, et la rétention est fixée à sept jours. Pendant cette période, une copie non chiffrée de la base d'avant l'activation existe encore sur le poste. C'est le seul filet réellement disponible depuis l'application, et il a trois limites qu'il faut énoncer :

- Elle date de l'activation. Tout ce qui a été saisi depuis n'y est pas.
- Elle disparaît au bout de sept jours, sans rappel.
- Décocher la case supprime ce recours entièrement, et la confirmation le dit en toutes lettres : aucune sauvegarde, la perte du mot de passe vaut perte totale des données.

Les sauvegardes planifiées ne remplacent pas la clé

Une sauvegarde produite après le chiffrement est chiffrée avec la même clé. La restaurer rend un fichier tout aussi illisible. Elles protègent d'une corruption, d'une suppression et d'une panne de disque, pas d'un mot de passe perdu.

Ce qui survit dans tous les cas

Objet	Où il vit
Préférences et profil	Stockage local du poste et compte, jamais dans la base chiffrée.
Workflows, points d'arrêt, alertes	Stockage local.
Extensions installées et leurs réglages	Stockage local et dossier des extensions.
Exports ISOF, CSV, rapports PDF	Là où ils ont été écrits. Un ISOF porte sa provenance et sa signature, donc il se réimporte en gardant sa valeur de preuve.
Fichiers de campagne et de méthode sur disque	Dans le dossier de campagne.
Bases IGDB et modèles IGL	Sur disque, hors de la base.
Certificat de laboratoire	Dans son propre magasin, chiffré par sa propre clé.

Reconstruire

Quand la base est perdue, la reconstruction part de ce qui a été exporté. L'ordre qui coûte le moins de travail :

	Étape
1	Mettre le fichier chiffré de côté plutôt que de le supprimer. Un moyen de le lire peut apparaître ; une suppression, non.
2	Repartir d'une base neuve, et la chiffrer immédiatement avec un nouveau mot de passe. La clé de récupération produite ce jour-là se note avant toute autre chose.
3	Réimporter les ISOF, qui ramènent les échantillons avec leur provenance et leur signature.
4	Réimporter les CSV et les fichiers de méthode pour ce que les ISOF ne couvrent pas.
5	Reprendre les descripteurs de campagne du disque, qui font foi pour la provenance.
6	Consigner la perte. Les journaux d'audit étaient dans la base : la nouvelle installation commence avec un journal vide, et le dire vaut mieux que de laisser croire à une continuité.

Ce qu'il faut faire avant, une fois pour toutes

- Imprimer la clé de douze mots et la ranger hors ligne, coffre ou classeur verrouillé. Le fichier texte proposé au téléchargement est une commodité de transport, pas un lieu de conservation.
- Noter qui la détient, et où. Une clé dont personne ne sait où elle est équivaut à une clé perdue.
- Ne pas décocher la sauvegarde de sept jours sans une raison écrite.
- Vérifier le déverrouillage avant de fermer la session le jour de l'activation, plutôt que le lendemain matin.
- Exporter en ISOF régulièrement : c'est la seule copie qui reste lisible sans la clé et qui garde sa valeur de preuve.
- Régénérer la clé immédiatement si elle a pu être vue par quelqu'un d'autre, tant que le mot de passe est encore connu.

16. Comptes, sessions et verrouillage

Un mot de passe de compte n'est pas le mot de passe maître. Le premier dit qui agit, le second rend la base lisible. Sur une installation mono-utilisateur, le premier peut ne pas exister du tout.

Les deux modes

Le mode d'installation est mono-utilisateur ou multi-utilisateur, et il est lu au démarrage. En mono-utilisateur, il n'y a ni vérification de session, ni menu de gestion des comptes, et l'identité est générique. En multi-utilisateur, la session est vérifiée périodiquement et son expiration renvoie vers la page de connexion.

Le menu utilisateur n'accorde pas ses fonctions sur le seul mode déclaré, parce que celui-ci reste figé après création d'un utilisateur. Il vérifie l'existence d'une session valide, qui est le seul indicateur fiable qu'on travaille sous une identité réelle, et il lit les permissions dans le niveau de licence plutôt que dans le mode.

Le verrouillage de session

Il est offert dès qu'un mot de passe est défini en base, indépendamment du mode. Le menu interroge le backend au moment du clic plutôt que de se fier à ce qu'il avait lu au chargement : la valeur peut ne pas être encore posée au premier clic après ouverture, et un utilisateur qui a défini un mot de passe doit toujours pouvoir verrouiller.

Opération	Ce qu'elle fait
Verrouiller la session	Consigne l'évènement et renvoie vers l'écran de verrouillage. La base reste déverrouillée.
Changer le mot de passe de compte	Ancien, nouveau, confirmation. Sans effet sur le chiffrement.
Sessions actives	Liste, avec la possibilité de fermer une session à distance.
Gestion des utilisateurs	Création, activation, suppression, rôles. Réservée aux administrateurs, et à une licence qui l'accorde.
Déconnexion	Ferme la session. Le poste oublie alors le cache de préférences du compte, langue et thème exceptés.

17. Le code de récupération de compte

Il existe un second objet appelé « récupération », et il n'a rien à voir avec le premier. Le confondre avec les douze mots est facile, et coûteux le jour où l'on croit détenir un recours qu'on n'a pas.

	Clé de récupération	Code de récupération
Forme	Douze mots.	Un code alphanumérique.
Concerne	Le chiffrement de la base.	Le compte administrateur.
Produite par	Le menu Protection, à l'activation.	La section Sécurité des préférences.
Disponible	Dès que la base est chiffrée.	En mode multi-utilisateur seulement.
Rend	Les données lisibles, en principe.	L'accès à un compte dont le mot de passe est perdu.
Ne rend pas	L'accès à un compte.	Les données d'une base chiffrée. Il ne remplace jamais le mot de passe maître.

La section Sécurité affiche l'état du code, propose de le générer, et l'affiche alors une seule fois dans une fenêtre dédiée, avec un bouton de copie. Générer un nouveau code invalide le précédent.

18. Accès API externe

Le backend écoute en local. La question n'est donc pas s'il est joignable, mais qui a le droit de lui parler : un autre programme du poste, un script, un instrument. Trois modes, et un seul est ouvert.

Mode	Ce qu'il autorise
disabled	Rien. Seule l'application parle au backend. C'est le défaut.
local_only	Les programmes du poste, sans autorisation individuelle.
authorized_clients	Les seuls clients explicitement autorisés, un par un.

Le choix du mode fait apparaître un avertissement décrivant ce qu'il ouvre, et il n'est appliqué qu'après un clic explicite : changer de mode n'est pas un réglage d'affichage. Le passage par la commande Rust est tenté d'abord, l'appel HTTP servant de repli.

Sous le mode par clients autorisés, deux listes s'affichent : les demandes en attente et les clients déjà autorisés. Le compteur de la section, dans la navigation, porte le nombre de demandes en attente : c'est le seul réglage de la fenêtre dont l'état réclame une décision dans un délai.

Air-gap et politique

Sur un poste en air-gap, les options réseau sont fermées et affichées comme telles. Quand la politique du poste verrouille le mode d'accès API, l'exploitant ne peut pas le rouvrir depuis l'application : c'est précisément le cas d'usage pour lequel la politique existe.

19. Journal d'audit et intégrité

Ce qui tourne sans témoin doit laisser une trace. Le journal consigne les événements qui engagent la responsabilité, et il vit dans la base, donc sous la même protection qu'elle.

Ce qui est consigné

Famille	Évènements
Système	Démarrage, arrêt, erreurs d'API.
Session	Connexion réussie, connexion refusée, expiration, verrouillage.
Données	Création, modification et suppression d'échantillon, mise à jour d'isotopes, import, export, analyse, correspondance.
ISOF	Signature, vérification, export, import.
Intégrité	Rupture détectée, scellement, tentative d'altération, rescèlement manuel.
Protection	Activation, désactivation, changement de mot de passe, verrouillage et déverrouillage de la base.
Sauvegarde	Création, restauration.
Journal	Purge, vérification.

Consulter

La fenêtre d'audit filtre par date, par événement et par gravité, cherche en texte libre, et pagine côté client sur un cache de cinquante lignes par page. Trois boutons complètent : rafraîchir, exporter, purger. La purge est elle-même un événement consigné, ce qui est le minimum attendu d'un journal.

La vérification du journal contrôle la chaîne d'intégrité et rapporte les ruptures. Une rupture n'est pas nécessairement une malveillance, une restauration de sauvegarde en produit une, mais elle doit être expliquée plutôt que rescellée sans commentaire.

Le journal est dans la base. Une base perdue emporte son journal. Pour une installation qui doit prouver la continuité de ses enregistrements, l'export périodique du journal est à traiter comme une sauvegarde à part entière, et non comme une commodité de consultation.

Référence

Les routes, les clés de stockage, ce qui se répare seul, et les doublons connus qu'il vaut mieux avoir vus avant d'ouvrir un fichier.

20. Routes appelées

Ce que l'interface demande au backend, groupé par sujet. Utile pour lire une trace réseau, et pour savoir laquelle des deux familles de routes de protection répond.

Préférences

Route	Emploi
GET /api/preferences	Réglages du compte. Une réponse sans document déclenche la remontée de ce que le poste avait.
PUT /api/preferences	Écriture, temporisée puis groupée, avec survie de la requête à la fermeture de la page.
POST /api/system/backup-schedule	Planification des sauvegardes, les trois réglages ensemble.

Protection de la base

Route	Emploi
GET /api/protection/status	État : protégée, chiffrée, déverrouillée, niveau, dernier contrôle.
POST /api/protection/enable	Activation. Rend les douze mots de la clé de récupération.
POST /api/protection/change-password	Changement du mot de passe maître.
POST /api/protection/disable	Désactivation, mot de passe exigé.
POST /api/database-protection/unlock	Déverrouillage au démarrage.
POST /api/database-protection/regenerate-recovery	Nouvelle clé de récupération, mot de passe exigé.
GET /api/protection/logs , /logs/export , /logs/clear , /logs/verify	Journal d'audit : lecture filtrée, export, purge, vérification.
GET /api/protection/code-info	Licence, version, identifiant d'installation.

Deux familles coexistent, `/api/protection/*` et `/api/database-protection/*`. Elles ne sont pas interchangeables : le déverrouillage et la régénération vivent dans la seconde, tout le reste dans la première.

Comptes et sécurité

Route	Emploi
GET /api/system/mode	Mode de développement, et si un mot de passe est défini en base.
GET /api/installation/mode	Mono ou multi-utilisateur, niveau de licence, fonctions accordées.
GET /api/auth/check	Session courante. Seul indicateur fiable d'une identité réelle.
POST /api/auth/lock , /api/auth/logout	Verrouillage et déconnexion.
POST /api/auth/change-password	Mot de passe du compte.
GET /api/auth/sessions , DELETE /api/auth/sessions/{id}	Sessions actives, fermeture à distance.
GET /api/auth/recovery-code-status , POST /api/auth/generate-recovery-code	Code de récupération de compte.
GET /api/users/list , POST /api/users/create , /toggle , DELETE	Gestion des utilisateurs.

Route	Emploi
GET et POST /api/local/api-access-config	Mode d'accès API et clients autorisés.

Licence

Route	Emploi
GET /api/licence/status	État de la licence installée.
POST /api/licence/activate , /deactivate	Activation par contenu de fichier, désactivation.
GET /api/licence/machine-fingerprint	Empreinte de la machine. Repli sur /api/license/* .
GET /api/ready	Porte l'état du verrou de licence au démarrage.
GET /api/edition/manifest	Édition, nom affiché, modules du binaire.

21. Clés de stockage

Clé	Contenu
<code>isofind_prefs</code>	La table des préférences, hors options déléguées.
<code>isofind_prefs_version</code>	Version de schéma du poste, pour les migrations.
<code>isofind_prefs_policy</code>	Politique de l'établissement, posée par l'installateur. Verrouille et impose.
<code>isofind_language</code> , <code>language</code>	Langue, dupliquée pour les modules qui la lisent sans passer par le magasin.
<code>isofind_community_settings</code>	Réglages communautaires, dont l'autorisation d'accès.
<code>isofind_zoom</code>	Facteur de zoom, partagé avec le menu Affichage.
<code>password_set</code>	Miroir local du fait qu'un mot de passe de compte existe, pour les appels synchrones du menu.

Aucune de ces clés ne porte de secret. Le mot de passe maître n'est écrit nulle part, la clé de récupération non plus, et le miroir ci-dessus ne dit que l'existence d'un mot de passe, jamais sa valeur.

22. Dépannage

Symptôme	Cause probable	Où regarder
Un réglage ne se retient pas	Option déléguée dont le propriétaire n'a pas été branché, ou panneau qui écrit sans passer par la mise en attente.	La déclaration de l'option : un proxy présent, et le panneau qui emploie stage .
Le pied dit « aucun changement » alors qu'on vient d'en faire un	Un panneau personnalisé qui a écrit directement.	Section 2. Le panneau doit prévenir le pied après avoir mis en attente.
Une case cochée revient décochée au démarrage	Contrainte de type appliquée contre une liste construite à l'exécution.	Le champ <code>choices0f</code> de l'option. Sans lui, le filtre vide le tableau en silence.
Un réglage semble sans effet	Il attend un programme qui n'est pas là.	La barre de cohérence, et la section Diagnostic.
La barre de cohérence signale quelque chose sur un poste sain	Une sonde qui n'a pas pu mesurer, et non un réglage incohérent.	Le libellé « non mesuré ». Une mesure absente ne doit produire aucun avertissement.
Les préférences d'un autre utilisateur s'appliquent	Le cache local du compte précédent n'a pas été effacé.	L'appel d'oubli local à la déconnexion. Langue et thème survivent volontairement.
Le canal de mise à jour est grisé	Il n'existe aucun mécanisme de mise à jour automatique dans l'application.	Rien à corriger : l'option est affichée pour dire ce qui manque.
L'écran de déverrouillage ne s'affiche pas alors que la base est chiffrée	Le statut lu au démarrage n'a pas la forme attendue, ou la clé est déjà en mémoire.	Section 23, premier point.
La fenêtre reste sans bouton de fermeture	Un écran bloquant recouvre la barre de titre.	L'empilement : la barre passe au-dessus et se réduit aux boutons de fenêtre.
Un bouton de la fenêtre ne fait rien	La commande native correspondante n'existe pas dans la couche Rust.	Section 23, dernier point.

23. Doublons et dettes connus

Ce qui suit n'est pas une liste de reproches mais une carte : plusieurs de ces fichiers font la même chose deux fois, et savoir lequel répond évite de corriger celui qui ne tourne pas.

Deux lectures du statut de protection

Le contrôle au démarrage lit le statut dans un champ, et la fenêtre d'état le lit dans un autre. Si le backend n'en sert qu'un seul, l'un des deux chemins voit une réponse vide et se tait. Un écran de déverrouillage qui ne s'affiche pas sur une base chiffrée se diagnostique là en premier.

Deux interfaces de protection

L'ancienne, encore présente, ajoute une icône de cadenas dans la barre des paramètres, propose plusieurs niveaux de protection, et active sans jamais produire de clé de récupération. Elle porte encore l'ancien nom du produit et des emojis dans son code. Le menu Protection actuel fige le niveau sur le chiffrement réel et impose la clé. Laisser les deux accessibles permet d'activer une protection sans clé de récupération, ce qui est exactement le scénario que la section 15 décrit comme sans issue.

Deux écrans de déverrouillage

Le menu Protection en monte un, et le module de jeton de sécurité en monte un autre, avec son propre identifiant et un empilement très élevé. Les deux demandent la même chose. C'est le second qui a motivé la contrainte d'empilement de la barre de titre, puisqu'il la recouvrait.

Deux fenêtres de préférences

L'ancienne, à onglets, contient encore l'onglet Sécurité complet, code de récupération de compte compris. La nouvelle rend ses sections depuis le schéma et délègue la sécurité à un panneau qui renvoie vers cet onglet. Tant que le renvoi existe, l'ancienne fenêtre reste nécessaire, et les deux doivent rester d'accord sur ce qu'elles affichent.

Deux gestionnaires d'utilisateur

Une interface adaptative historique construit un menu utilisateur en injectant du HTML et son propre CSS, avec des emojis comme icônes de rôle, et laisse deux fonctions en attente d'implémentation. Le menu utilisateur actuel couvre le même terrain sur le système de design commun. Le premier suppose en outre que le mode déclaré suffit à décider des permissions, ce que le second a explicitement cessé de faire.

Des réglages qui ne font rien

Certaines options appellent des commandes natives qui n'existent pas dans la couche Rust : réduction en zone de notification, redemande du choix de fermeture, mode réseau lu par commande. L'appel échoue en silence, et le réglage paraît accepté. Les identifiants concernés sont connus et la fenêtre continue de les afficher ; tant qu'ils ne sont pas branchés ou retirés, ils occupent la place d'un réglage qui fonctionnerait.

Ce qui manque le plus, par ordre de conséquence

Un chemin qui consomme la clé de douze mots. Sans lui, la promesse faite à l'activation n'est pas tenue et le seul recours réel dure sept jours.

Un rappel avant l'expiration de la sauvegarde en clair. Elle disparaît sans que rien ne le signale, alors que c'est le dernier filet.

Un retrait de l'ancienne interface de protection, qui permet aujourd'hui d'activer un chiffrement sans clé de récupération.