Product Brief: dirCops

Nom de l'application : dirCops (suivi facturation / échéancier – Écloserie marine Gravelines)

Executive Summary

Le client (Ecloserie marine Gravelines / EMGI) gère aujourd'hui tout son suivi commercial dans Excel : facturation, paiements, échéanciers, encours clients, trésorerie. L'outil atteint ses limites : formules cassées, actualisation impossible, multiplication des feuilles. L'équipe commerciale ne s'y retrouve plus, les décisions s'appuient sur des données peu fiables et il est difficile d'expliquer un encours à un client. Une seule personne maîtrise le classeur. L'objectif est de remplacer cette partie par une application dédiée au **suivi** (sans gérer la création des factures) : suivi factures, paiements et échéanciers par client, avec les données métier (quantités, n° facture, indicateurs type encours, couverture, etc.) que les ERP du marché ne gèrent pas bien. L'application doit être **simple**, **facile à utiliser**, permettre de suivre clients, échéanciers et paiements sans effort, et de faire du **reporting rapidement**. Elle reste à coût maîtrisé (~20 k€), donc volontairement sobre et ciblée.

Core Vision

Problem Statement

L'équipe commerciale d'EMGI s'appuie sur des classeurs Excel partagés (SharePoint/Teams) pour le suivi des factures, des paiements et des échéanciers par client. Le système est devenu fragile (liens cassés, formules en erreur, message « impossible d'actualiser »), difficile à maintenir et concentré entre les mains d'une seule personne. Les utilisateurs ne s'y retrouvent plus.

Problem Impact

- **Erreurs de suivi** : données incohérentes ou obsolètes.
- **Temps perdu** : navigation entre onglets et feuilles, vérifications manuelles.
- **Décisions sur des données fausses** : risque pour la trésorerie et la relation client.
- **Difficulté à expliquer un encours à un client** : pas de vue fiable et partagée par toute l'équipe commerciale.

Why Existing Solutions Fall Short

Les ERP standards ne couvrent pas les spécificités métier du suivi tel qu'EMGI le pratique (indicateurs type échéance contrat, encours, couverture, écarts cumulés, suivi par type de produit / cycle, etc.). L'adaptation serait lourde ou inadaptée. Le besoin est ciblé : un **suivi** facturation / échéancier / paiement, pas un ERP complet.

Proposed Solution

Une application dédiée au **suivi facturation et échéancier de paiement** :

- Suivi des factures, paiements et échéanciers **par client**.

- Données essentielles : quantités, numéros de facture et de vente, montants, et indicateurs métier (encours, couverture, etc.) que les ERP ne gèrent pas « proprement ».
- Pas de gestion de la facture en tant que document (création/édition) : la facturation provient de **Pennylane** ; l'appli consolide et affiche le suivi de ce qui est déjà facturé et payé. Objectif : une seule source de vérité, utilisable par toute l'équipe commerciale.

Key Differentiators

- **Simplicité et facilité d'utilisation** : plus besoin qu'une seule personne sache « faire tourner le système ».
- **Suivi client, échéancier et paiement** : consultation et mise à jour faciles, sans se perdre dans des feuilles Excel.
- **Reporting rapide** : tableaux de bord et exports pour le commercial et la direction.
- **Métier-first** : modélisation des notions (encours, échéances, couverture, etc.) telles qu'EMGI les utilise, au lieu d'un ERP générique.
- **Coût maîtrisé / simplicité** : l'application ne coûte pas cher ; elle reste volontairement simple pour répondre au besoin sans sur-dimensionnement.

Target Users

Primary Users

Équipe commerciale — Utilisateurs principaux au quotidien. Ils consultent et mettent à jour le suivi des factures, des échéanciers et des paiements par client. L'objectif est qu'ils puissent le faire sans dépendre d'une seule personne maîtrisant un classeur Excel complexe : application simple, compréhensible par tous les commerciaux.

Secondary Users

Trésorerie et **Direction** — Bénéficiaires indirects. Ils profitent de données plus fiables et d'un reporting plus clair ; si le suivi commercial est mal fait, ils sont fortement impactés. Pas de conflit avec le commercial : le besoin est de simplifier le travail du commercial pour que trésorerie et direction aient une base de données fiable. Dans le cadre d'une application ciblée (~20 k€), aucun parcours ni persona dédié complexe pour ces rôles — un seul groupe utilisateur principal (commercial) est privilégié.

User Journey

Non détaillé dans ce brief ; l'accent est mis sur un usage simple et répétitif (consultation et mise à jour du suivi par client, reporting rapide) sans scénarios multiples.

Success Metrics

Succès utilisateur : l'équipe commerciale utilise l'application pour le suivi facturation / échéancier / paiement sans dépendre d'une seule personne ; consultation et reporting en quelques clics ; plus d'usage

d'Excel pour ce périmètre. **Contrainte de coût** : l'application reste peu chère et donc volontairement simple — le succès inclut de livrer une solution efficace dans ce cadre.

Succès business : une seule source de vérité pour trésorerie et direction ; moins d'erreurs de suivi et de décisions sur des données fausses ; temps gagné et possibilité d'expliquer un encours à un client de façon fiable.

Business Objectives

Adoption par l'équipe commerciale ; abandon du classeur Excel pour le suivi concerné ; données fiables pour la trésorerie et la direction.

Key Performance Indicators

- Utilisation effective par les commerciaux (suivi et reporting faits dans l'appli).
- Fin de la dépendance à une seule personne pour « faire tourner » le suivi.
- Réduction des erreurs / incohérences constatées par trésorerie et direction (qualitatif ou via feedback).

MVP Scope

Core Features

- **Suivi des factures** : liste par client, n° facture, n° vente, date, quantités, montants (ex. CA HT).
- **Suivi des paiements / échéancier** : ce qui est payé, ce qui reste dû (encours), échéances. Possibilité d'ajouter un **écheancier de paiement** à une facture : facture associée, nombre d'échéances, date de début d'écheancier.
- **Vue par client** : synthèse par client (factures, paiements, encours).
- **Données métier spécifiques** : au minimum encours, échéances ; couverture / indicateurs type Excel si possible dans la limite du budget.
- **Reporting simple** : exports ou vues synthétiques (tableaux de bord basiques) pour commercial, trésorerie, direction.
- **Une seule source de vérité** : données centralisées, plus de classeur Excel pour ce périmètre.

Out of Scope for MVP

- Création / édition des factures (document facture) — gérée par Pennylane.
- Gestion complète trésorerie (bancaire, prévisionnel détaillé).
- Parcours dédiés complexes pour trésorerie / direction (priorité au besoin commercial).
- Fonctionnalités « nice to have » qui feraient dépasser le cadre coût / simplicité.

MVP Success Criteria

- L'équipe commerciale utilise l'appli pour ce suivi et n'utilise plus Excel pour ce périmètre.
- Plus de dépendance à une seule personne ; trésorerie et direction s'appuient sur les mêmes données.

Future Vision

- Évolutions possibles après adoption : indicateurs métier supplémentaires, intégrations (ERP, compta), extensions vers d'autres processus — sans engagement dans le MVP.

Synthèse projet – Lecture client

Application : dirCops (suivi facturation / échéancier – Écloserie marine Gravelines)
Projet : Suivi facturation / échéancier / paiement — Écloserie marine
Date : 2026-02-10
Objectif : Permettre au client de relire et valider le périmètre en peu de temps.

---

1. En une page : objectif et livrable

- **Problème :** Aujourd'hui le suivi facturation, paiements et échéanciers se fait dans Excel. C'est fragile, peu lisible et dépendant d'une seule personne.
- **Solution :** Une application web **dirCops** dédiée au **suivi** : les factures sont émises dans **Pennylane** ; dans l'appli, l'équipe commerciale saisit et consulte les factures pour le suivi (encours, paiements, écheanciers) et dispose de tableaux de bord et d'exports.
- **Livré :** Application utilisable par l'équipe commerciale (liste **clients** — noms de clients réels, pas des champs Excel de type « lieu de départ » —, détail client avec factures/paiements/écheanciers, ajout/modification/suppression facture, ajout/suppression paiements, ajout/suppression écheanciers, tableau de bord (encours + synthèse + échéances à venir), **Top clients (Top 15)**, Quantités par client, **export encours CSV** et **export de l'ensemble des factures (CSV) en pièce jointe**). Une seule source de vérité, plus besoin d'Excel pour ce périmètre.
- **Accès :** authentification **Livinweb ModX** (identifiant / mot de passe), SSO en option.
- **Prochaines étapes :** Valider cette synthèse, répondre aux questions encore ouvertes (voir § 6), puis chiffrage et planning.

---

2. Glossaire (vocabulaire du projet)

Terme	Signification
Encours	Montant restant dû (somme des factures non encore soldées par les paiements).
Écheancier	Plan de paiement d'une facture : nombre d'échéances et date de début (ex. 3 échéances à partir du 01/02/2026).
Facture (suivi)	Saisie dans l'appli pour le suivi ; le document officiel (PDF) est émis dans Pennylane.
Pennylane	Logiciel de facturation utilisé pour éditer et émettre les factures ; l'appli ne fait pas la facturation, elle en fait le suivi.
Type produit	Alevins ou Larves (catégorie de produit pour le reporting).
Cycle	Période de référence (ex. 24-25, 25-26) pour filtrer les vues Top clients et reporting.
Top clients	Vue qui classe les clients par quantités et/ou CA (par type alevins/larves et optionnellement par cycle).
Client	Dans l'appli = nom du client (ex. AQUANARIA, PHILOSOFISH, AQN).

---

3. Ce qui est livré (écrans et fonctions)

- **Accueil / Liste des clients** : recherche par nom, accès au détail d'un client. Formulaire « Ajouter un client », bouton Supprimer par client (avec confirmation).
- **Fiche client** : encours, liste des factures (n°, date, type, cycle, quantités, montant, encours, état) avec boutons Modifier et Supprimer ; liste des paiements et des écheanciers avec bouton Supprimer. Formulaires : ajouter une facture (suivi), modifier une facture, ajouter un écheancier, enregistrer un paiement.
- **Tableau de bord** : encours par client (tri croissant), synthèse (encours total en k€, délai moyen de paiement en jours), échéances à venir (calculées à partir des écheanciers quand ils existent : date et montant par échéance ; sinon date de facture et encours). Liens : **export encours CSV** et **export de l'ensemble des factures (PJ)** (fichier téléchargeable contenant toutes les factures renseignées).
- **Top clients** : choix type (Alevins / Larves) et cycle ; tableau **Top 15** clients avec quantités, CA, pourcentages.
- **Quantités par client** : synthèse quantités facturées / livrées par client (filtre type optionnel).
- **Exports** : (1) CSV encours par client ; (2) CSV « ensemble des factures » (suivi-factures-YYYY-MM-DD.csv) utilisable en pièce jointe.

---

4. Règles métier et points d'alignement (résumé)

- **Encours** : encours facture = `montant_ht - somme(paiements)` ; si encours < 0, on affiche **0** (pas de crédit client affiché).

- **Paiement > montant facture** : autorisé ; encours affiché = **0**.
- **Unicité n° facture** : pas de contrainte dans le périmètre actuel (option future : unicité par client).
- **Clients en doublon** : pas de détection/fusion (acceptés tels quels).
- **Navigateurs** : Safari **non supporté** (option future) ; pas de recette Safari.

---

5. Ce qui n'est pas livré (hors périmètre)

- **Génération ou édition du document facture (PDF)** — gérée par Pennylane.
- **Gestion complète de la trésorerie** (bancaire, prévisionnel détaillé).
- **Parcours dédiés complexes** pour la trésorerie ou la direction (priorité au besoin commercial).
- **Import automatique depuis Pennylane** (pour l'instant saisie manuelle dans l'appli après émission dans Pennylane) — à discuter si besoin.
- **Crédit client** (si paiements > encours) : non géré/affiché dans le MVP (option future).

---

6. Questions restantes / à préciser

Les décisions déjà figées (accès, rôles, Top 15, Safari non supporté, règles métier clés) sont consolidées dans la doc projet.
Les points ci-dessous restent à préciser (selon besoin) :

- **Exports CSV** : liste exacte des colonnes attendues pour :
  - Export « Encours par client »
  - Export « Ensemble des factures (PJ) »
  - Formats (dates, montants, séparateur, compatibilité Excel)
- **Pennylane** : souhaitez-vous (plus tard) un import/synchronisation automatique des factures (et éventuellement des paiements), ou la saisie manuelle reste-t-elle la règle ?
- **Déploiement / hébergement** : où et par qui (déploiement, sauvegardes, accès) ?

Vos réponses permettront de finaliser le chiffrage et le planning.

---

6. Engagements réciproques (à valider)

- **Côté client :** Fourniture des accès nécessaires (hébergement si applicable), données de test si besoin, désignation d'un interlocuteur pour la recette et les validations, réponses aux questions ouvertes dans les délais convenus.
- **Côté livrant :** Livraison de l'application selon le périmètre décrit, déploiement (selon accord), recette selon le cahier des recettes fourni, et éventuellement formation et garantie (à préciser dans le devis).

Document à remettre au client pour relecture du projet. À adapter selon le contexte (nom du livrant, délais, détails du devis).

Product Requirements Document - dirCops

Application : dirCops (suivi facturation / échéancier – Écloserie marine Gravelines)
Author: Livinweb
Date: 2026-02-09

Success Criteria

User Success

L'équipe commerciale utilise l'application pour le suivi facturation / échéancier / paiement sans dépendre d'une seule personne ; consultation et reporting en quelques clics ; plus d'usage d'Excel pour ce périmètre. Application simple et efficace (~20 k€).

Business Success

Une seule source de vérité pour trésorerie et direction ; moins d'erreurs de suivi et de décisions sur des données fausses ; temps gagné et possibilité d'expliquer un encours à un client de façon fiable.

Technical Success

Application web simple, stable, facile à maintenir.

Measurable Outcomes

- Utilisation effective par les commerciaux (suivi et reporting faits dans l'appli).
- Fin de la dépendance à une seule personne pour « faire tourner » le suivi.
- Réduction des erreurs / incohérences constatées par trésorerie et direction (qualitatif ou via feedback).

Product Scope

MVP - Minimum Viable Product

Core Features:

- Suivi des factures : liste par client, n° facture, n° vente, date, quantités, montants (ex. CA HT).
- Suivi des paiements / échéancier : ce qui est payé, ce qui reste dû (encours), échéances.
- Vue par client : synthèse par client (factures, paiements, encours).
- Données métier spécifiques : au minimum encours, échéances ; couverture / indicateurs type Excel si possible dans la limite du budget.
- Reporting simple : exports ou vues synthétiques (tableaux de bord basiques) pour commercial, trésorerie, direction.
- Données centralisées, remplacement du classeur Excel pour ce périmètre.

In Scope (clarification):

- **Ajout manuel de factures pour le suivi** : la facturation (document facture, PDF, édition) provient de **Pennylane** ; dans l'appli, le commercial **ajoute à la main** les factures pour en assurer le suivi (n°

facture, n° vente, date, type produit, cycle, quantités, quantité livrée, montant HT).

- **Type de produit et cycle** : alevins, larves ; cycles (ex. 24-25, 25-26) pour le reporting.
- **Quantité livrée** : suivi de la quantité déjà livrée par facture (QTE FACT / QTE LIVREE).
- **Top clients par catégorie** : vues « Top clients alevins », « Top clients larves » (quantités + CA), filtrable par cycle.
- **Quantités vendues par client** : synthèse par client des quantités facturées et livrées (avec répartition par type si besoin).
- **Écheancier de paiement** : pour une facture, le commercial peut ajouter un écheancier avec facture associée, nombre d'échéances et date de début d'écheancier.
- **Clients** : les libellés « client » désignent les **noms de clients** (ex. AQUANARIA, PHILOSOFISH, AQN). EMGI est un **lieu de départ**, pas un nom de client (cf. référentiel Excel EMGI_FTI_Suivi des Ventes).
- **Tableau de bord** : encours par client (tri croissant), synthèse (encours total en k€, délai moyen de paiement en jours), colonnes optionnelles accord / j out / situation (à alimenter si les données existent). Export encours CSV et **export de l'ensemble des factures en pièce jointe** (fichier téléchargeable).

Out of Scope for MVP:

- Génération ou édition du **document** facture (PDF, impression) — gérée par **Pennylane**.
- Gestion complète trésorerie (bancaire, prévisionnel détaillé).
- Parcours dédiés complexes pour trésorerie / direction (priorité au besoin commercial).
- Fonctionnalités « nice to have » qui feraient dépasser le cadre coût / simplicité.

MVP Success Criteria:

- L'équipe commerciale utilise l'appli pour ce suivi et n'utilise plus Excel pour ce périmètre.
- Plus de dépendance à une seule personne ; trésorerie et direction s'appuient sur les mêmes données.

Growth Features (Post-MVP)

Non défini à ce stade — focus sur l'adoption du MVP d'abord.

Vision (Future)

Évolutions possibles après adoption : indicateurs métier supplémentaires, intégrations (ERP, compta), extensions vers d'autres processus — sans engagement dans le MVP.

User Journeys

Journey 1: Commercial consulte le suivi d'un client

Opening Scene : Marie, commerciale, reçoit un appel client demandant l'état des factures et ce qui reste à payer. Actuellement, elle doit ouvrir Excel, naviguer entre plusieurs onglets, chercher le client, vérifier manuellement les paiements et calculer l'encours — avec risque d'erreur ou de données obsolètes.

Rising Action : Avec la nouvelle application, Marie ouvre l'appli, tape le nom du client, et voit immédiatement : liste des factures (n°, date, montant, quantités), état des paiements, encours calculé

automatiquement, échéances à venir.

Climax : Pendant l'appel, Marie peut expliquer précisément l'état, confirmer les paiements, donner l'encours exact, et rassurer le client avec des données fiables partagées par toute l'équipe.

Resolution : Plus besoin d'Excel pour ce suivi, plus de dépendance à une seule personne, données fiables et à jour, confiance restaurée avec le client.

Journey 2: Commercial met à jour un paiement

Opening Scene : Thomas apprend qu'un client a effectué un paiement. Actuellement, il doit ouvrir Excel, trouver la bonne feuille et ligne, mettre à jour manuellement avec risque de casser une formule.

Rising Action : Thomas ouvre l'appli, sélectionne le client, voit les factures en attente, clique sur "Enregistrer un paiement", saisit le montant et la date. L'encours se met à jour automatiquement.

Climax : En quelques clics, le paiement est enregistré et visible par toute l'équipe. Plus besoin de modifier Excel ou de dépendre d'une seule personne.

Resolution : Mise à jour rapide et simple, données à jour pour toute l'équipe, plus d'erreurs de formules Excel.

Journey 3: Commercial génère un reporting rapide

Opening Scene : Sophie doit préparer un rapport pour la direction sur les encours clients. Actuellement, elle doit ouvrir plusieurs onglets Excel, copier-coller des données, faire des calculs manuels avec risque d'erreurs.

Rising Action : Sophie ouvre l'appli, va dans "Reporting" ou "Tableaux de bord", voit une vue synthétique : encours par client, factures en attente, échéances à venir. Elle peut exporter ou faire une capture d'écran.

Climax : En quelques clics, elle a une vue claire et à jour pour la direction, sans manipulation Excel.

Resolution : Reporting rapide et fiable, données cohérentes avec trésorerie/direction, gain de temps.

Web App Specific Requirements

Project-Type Overview

Application web classique (Multi-Page Application) pour usage interne de l'équipe commerciale.

Technical Architecture Considerations

Architecture : Multi-Page Application (MPA) - approche traditionnelle adaptée à une application interne.

Browser Support : Navigateurs modernes (Chrome, Firefox, Edge) - pas de support nécessaire pour navigateurs anciens.

SEO Strategy : Non applicable - application interne, pas d'indexation par les moteurs de recherche nécessaire.

Real-time Updates : Non requis - rafraîchissement manuel suffit pour les besoins de suivi facturation/paiement.

Accessibility Level : Basique - conformité WCAG niveau A suffisante pour usage interne.

Browser Matrix

- Chrome (dernières 2 versions) : Support complet
- Firefox (dernières 2 versions) : Support complet
- Edge (dernières 2 versions) : Support complet
- Safari : Support basique si nécessaire
- Navigateurs anciens : Non supportés

Responsive Design

Interface responsive pour usage sur desktop principalement, avec support basique tablette si nécessaire. Pas de priorité mobile-first.

Performance Targets

- Temps de chargement initial : < 3 secondes
- Temps de réponse aux interactions : < 1 seconde

Implementation Considerations

- Stack technique simple et maintenable
- Architecture permettant évolution future si besoin

Functional Requirements

Client Management

- FR1: Commercial peut rechercher un client par nom
- FR2: Commercial peut voir la vue détaillée d'un client (factures, paiements, encours sur une seule page)
- FR2b: Commercial peut ajouter un client (depuis la liste clients)
- FR2c: Commercial peut supprimer un client (avec confirmation ; cascade factures, paiements, écheanciers)

Invoice Tracking

- FR3: Commercial peut voir la liste des factures d'un client avec détails (n° facture, n° vente, date, type produit, cycle, quantités, quantité livrée, montants CA HT, état payé/en attente)
- FR3b: Commercial peut ajouter manuellement une facture pour le suivi (saisie des données de suivi ; la facturation provient de Pennylane)
- FR3c: Commercial peut modifier une facture existante (correction de saisie)
- FR3d: Commercial peut supprimer une facture (avec confirmation ; cascade paiements et écheancier)

Payment Management

- FR4: Commercial peut enregistrer un paiement (montant, date) pour une facture
- FR4b: Commercial peut ajouter un écheancier de paiement à une facture (facture associée, nombre d’échéances, date de début d’écheancier)

- FR5: Commercial peut voir l'historique des paiements d'un client
- FR5b: Commercial peut voir l’écheancier associé à une facture (nombre d’échéances, date de début)
- FR5c: Commercial peut supprimer un paiement (avec confirmation)
- FR5d: Commercial peut supprimer un écheancier (avec confirmation)

Financial Calculations

- FR6: Le système calcule automatiquement l'encours (montant restant dû) par client
- FR7: Le système calcule automatiquement l'encours total
- FR8: Le système affiche les échéances à venir (dates ≥ aujourd’hui) : pour les factures avec écheancier, dates et montants sont calculés à partir de l’écheancier (date début + N mois, montant par échéance) ; pour les factures sans écheancier, date de facture et encours
- FR9: Le système met à jour automatiquement les calculs après enregistrement d'un paiement

Reporting & Analytics

- FR10: Commercial peut voir un tableau de bord synthétique (encours par client, factures en attente, échéances)
- FR10b: Commercial peut voir les top clients par type de produit (alevins, larves) avec quantités et CA, filtrable par cycle
- FR10c: Commercial peut voir la synthèse des quantités vendues par client (quantités facturées, quantités livrées), avec répartition par type si pertinent
- FR11: Commercial peut exporter les données

Data Management

- FR12: Le système maintient une seule source de vérité centralisée pour toutes les données de suivi
- FR13: Les données sont accessibles par toute l'équipe commerciale

Non-Functional Requirements

Security

- NFR1: Accès restreint à l'équipe commerciale autorisée uniquement

Architecture — dirCops

Application : dirCops — suivi facturation / échéancier / paiement
Mode : POC puis migration vers MODX en production

---

1. Contexte et périmètre

Exigences fonctionnelles : 13 FR en 6 catégories (Client Management, Invoice Tracking, Payment Management, Financial Calculations, Reporting & Analytics, Data Management). Application web MPA, calculs automatiques (encours, échéances), export CSV.

Exigences non fonctionnelles : Sécurité (accès restreint), Performance (< 3s chargement, < 1s interactions).

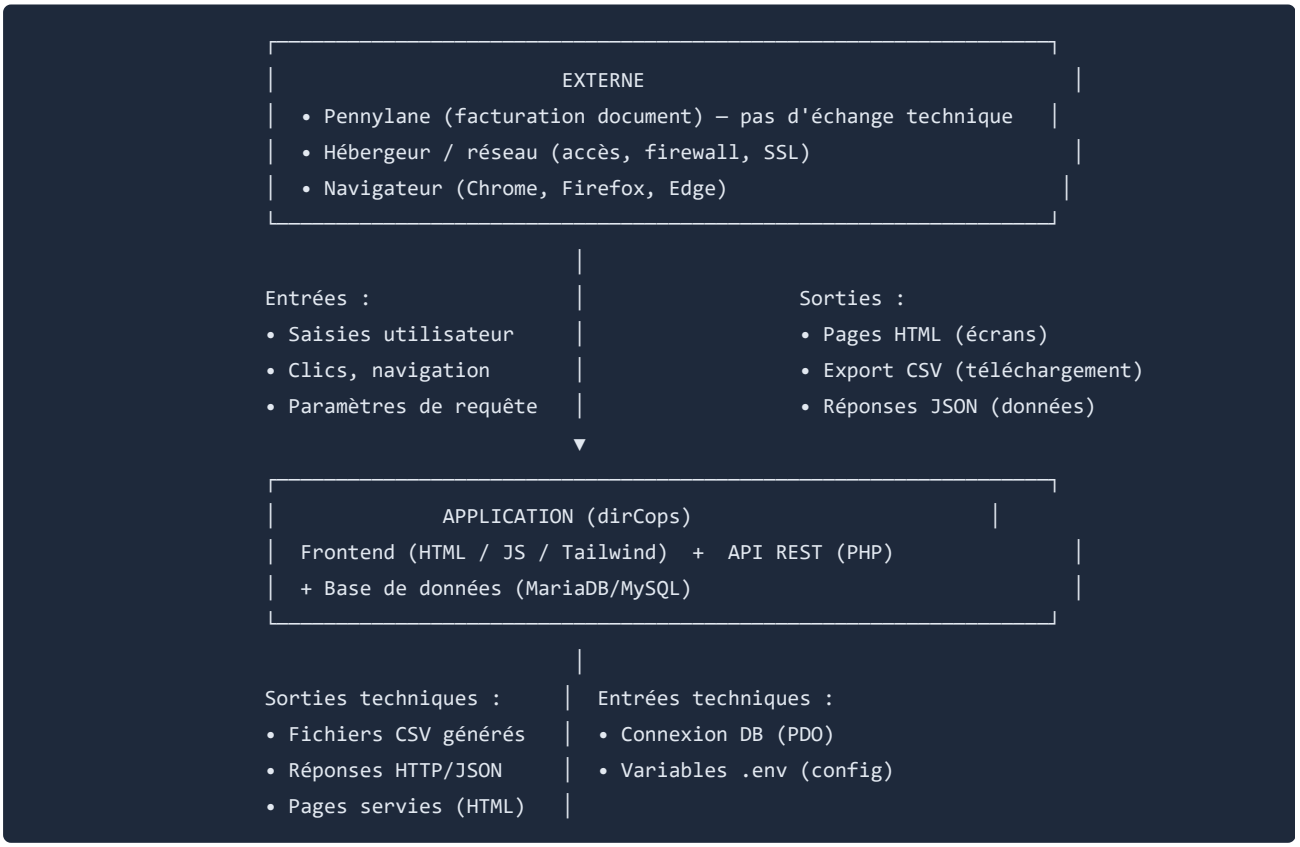
Contraintes : Facturation gérée dans Pennylane ; l'appli assure uniquement le suivi (saisie manuelle des factures). Budget restreint → architecture minimale. Stack : PHP (API REST) + JavaScript + Tailwind + HTML + MariaDB/MySQL. Application interne uniquement.

Périmètre utilisateurs : Un seul type traité : **commercial / admin** (accès complet). Les autres rôles (lecteur, stagiaire, etc.) sont en option.

---

2. Vue d'ensemble et architecture fonctionnelle

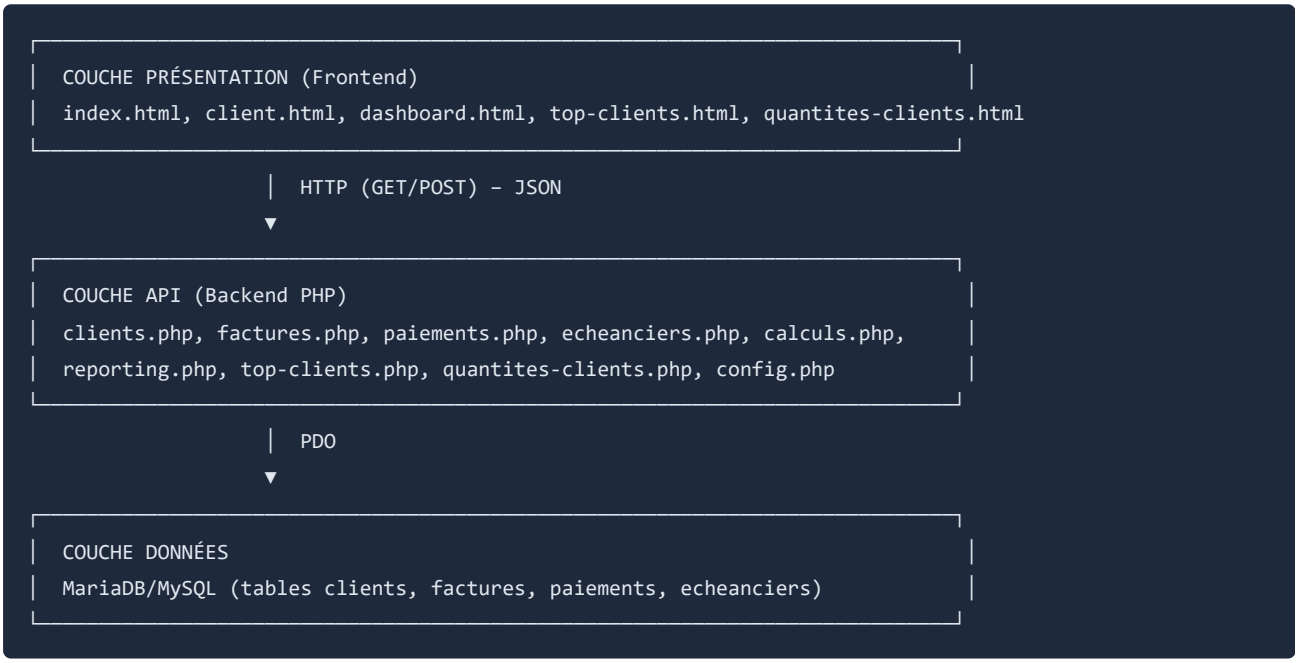
2.1 Schéma de contexte



2.2 Acteurs

Acteur	Rôle	Entrées fournies	Sorties reçues
Commercial (équipe EMGI)	Utilisateur principal	Recherche client, saisie facture/paiement/écheancier, filtres (type, cycle), demande d'export	Liste clients, fiche client, tableau de bord, Top clients, Quantités, fichier CSV
Trésorerie / Direction (si accès)	Consultation	Consultation écrans, export	Mêmes sorties en lecture
Système (automatique)	—	—	Calculs encours, état facture (payé/en attente), échéances

2.3 Vue par couches



2.4 Modules fonctionnels

Module	Responsabilité	Fichiers principaux
Config	Connexion DB, .env, helpers (sendJson, sendError)	`api/config.php`, `.env`
Clients	Liste, recherche, détail, création, suppression	`api/clients.php`, `index.html`, `client.html`, `js/client-detail.js`
Factures	Liste par client, ajout/modification/suppression (suivi)	`api/factures.php`, `client.html`, `js/facture-form.js`
Paiements	Liste, enregistrement, suppression	`api/paiements.php`, `client.html`, `js/paiements.js`
Écheanciers	Liste, création, suppression	`api/echeanciers.php`, `client.html`, `js/echeancier-form.js`
Calculs	Encours, échéances à venir	`api/calculs.php`, `api/reporting.php`, `dashboard.html`, `js/dashboard.js`
Reporting	Tableau de bord, export CSV (encours, factures PJ)	`api/reporting.php`, `dashboard.html`
Top clients	Agrégation type/cycle, quantités, CA	`api/top-clients.php`, `top-clients.html`, `js/top-clients.js`
Quantités par client	Synthèse quantités facturées/livrées	`api/quantites-clients.php`, `quantites-clients.html`, `js/quantites-clients.js`

Dépendances : Clients indépendant ; Factures → Clients ; Paiements → Factures ; Écheanciers → Factures ; Calculs → Clients, Factures, Paiements ; Reporting → Calculs ; Top clients, Quantités → Factures, Clients.

---

3. Décisions techniques

3.1 Stack POC

- **Backend :** PHP (API REST)
- **Frontend :** JavaScript vanilla + Tailwind CSS + HTML (MPA)
- **Base de données :** MariaDB/MySQL
- **Authentification :** Sessions PHP simples ou auth basique ; pas de JWT/OAuth

3.2 Base de données

Tables : `clients`, `factures`, `paiements`, `echeanciers`
Relations : factures → client, paiements → facture, echeanciers → facture
Champs factures : n_facture, n_vente, date_facture, type_produit (alevins/larves), cycle, quantites, qte_livree, montant_ht

Champs échéanciers : facture_id, nb_echeances, date_debut_echeancier
État payé/en attente : dérivé des paiements

3.3 API REST

Endpoints : `/api/clients`, `/api/factures`, `/api/paiements`, `/api/echeanciers`, `/api/calculs`, `/api/reporting`, `/api/top-clients`, `/api/quantites-clients`
Format : JSON ; erreurs `{error: "message"}` avec codes HTTP 200, 400, 404, 500

3.4 Frontend

Pages : `index.html`, `client.html`, `dashboard.html`, `top-clients.html`, `quantites-clients.html`
State : JavaScript vanilla, variables globales ou localStorage
Pas de SPA : rechargement de page après certaines actions

3.5 Infra

Dev : `php -S` ou Apache/Nginx
Déploiement : Hébergement PHP + MariaDB ; `sql/schema.sql` une fois ; `.env` en production

---

4. Patterns et conventions

4.1 Nommage

Contexte	Convention	Exemple
Tables et colonnes	snake_case, pluriel tables	`clients`, `client_id`, `n_facture`
API (endpoints, params, JSON)	snake_case	`?client_id=1`, `{client_id: 1}`
Code PHP/JS	camelCase	`getClientData()`, `$clientId`
Fichiers	minuscules, tirets	`client-detail.js`

4.2 Formats API

- **Succès :** réponse directe `{...}` (pas de wrapper `data`)
- **Erreur :** `{error: "message"}` + code HTTP approprié
- **Dates :** ISO (`2026-02-09T10:00:00Z`)

4.3 Règles agents IA

- snake_case en DB et JSON API ; camelCase en PHP/JS
- Respecter la structure des endpoints et du projet
- Retourner les erreurs au format `{error: "message"}`
- **Anti-patterns :** pas de wrapper `{data: ...}` ; pas de mélange camelCase/snake_case dans le même contexte

---

5. Interfaces logicielles

5.1 I1 — Utilisateur ↔ Application

Points d'entrée : `index.html` (liste clients), `client.html?id=` (fiche client), `dashboard.html`, `top-clients.html`, `quantites-clients.html`, lien « Exporter CSV » → `api/reporting.php?format=csv`.

5.2 I2 — Frontend ↔ API

Protocole : GET (lecture), POST (création), PUT (modification), DELETE (suppression). Body JSON pour POST/PUT.
Erreurs : 4xx/5xx avec `{error: "message"}` ; front : `apiGet`, `apiPost`, etc. dans utils.js.

Appels principaux :

Appelant	Méthode	Endpoint	Rôle
index.html	GET/POST/DELETE	`/api/clients.php`	Liste, recherche, création, suppression client
client.html	GET	`/api/clients`, `/api/factures`, `/api/paiements`, `/api/calculs`, `/api/echeanciers`	Détail client
client.html	POST/PUT/DELETE	`/api/factures.php`, `/api/paiements.php`, `/api/echeanciers.php`	Création, modification, suppression
dashboard.html	GET	`/api/reporting.php`	Tableau de bord ; `format=csv` ou `format=factures_csv` pour export
top-clients.html	GET	`/api/top-clients.php`	Données Top clients (params type, cycle)
quantites-clients.html	GET	`/api/quantites-clients.php`	Quantités par client (param type)

5.3 I3 — API ↔ Base de données

Protocole : PDO, pilote pdo_mysql. Connexion dans `config.php` (getPdo()).
Opérations : SELECT, INSERT, UPDATE, DELETE ; requêtes préparées. Charset utf8mb4.

5.4 I4 — Export CSV

Endpoint : `api/reporting.php` avec `format=csv` (encours par client) ou `format=factures_csv` (ensemble factures PJ). En-têtes Content-Type text/csv, Content-Disposition attachment.

5.5 I5 — Pennylane (hors MVP)

Non réalisé en MVP. Facturation dans Pennylane ; saisie manuelle dans l'appli. Évolution possible : import ou sync factures/paiements.

5.6 Récapitulatif

Id	Interface	Sens	Protocole
I1	Utilisateur ↔ Application	Bidirectionnel	HTTP, navigateur, HTML/CSV
I2	Frontend ↔ API	Front ↔ API	HTTP, JSON
I3	API ↔ Base de données	API ↔ DB	PDO, MySQL/MariaDB
I4	Application → Utilisateur (export)	API → navigateur	HTTP, CSV
I5	Application ↔ Pennylane	— (hors MVP)	À définir

---

6. Structure projet et validation

6.1 Arborescence

```
dircops/
├── README.md
├── .env.example
├── .env
├── index.html
├── client.html
├── dashboard.html
├── top-clients.html
├── quantites-clients.html
├── api/
│   ├── config.php
│   ├── clients.php
│   ├── factures.php
│   ├── paiements.php
│   ├── calculs.php
│   ├── reporting.php
│   ├── top-clients.php
│   └── quantites-clients.php
├── js/
│   ├── utils.js
│   ├── client-detail.js
│   ├── paiements.js
│   ├── dashboard.js
│   ├── facture-form.js
│   ├── top-clients.js
│   └── quantites-clients.js
├── css/
└── sql/
    └── schema.sql
```

6.2 Flux de données

- Liste clients : `index.html` → `api/clients.php` → DB

- Détail client : `client.html` → API (clients, factures, paiements, calculs, écheanciers) → DB
- Dashboard : `dashboard.html` → `api/calculs.php`, `api/reporting.php` → DB
- Paiement : `client.html` + `js/paiements.js` → `api/paiements.php` → DB → recalcul API

6.3 Validation

Statut : READY FOR IMPLEMENTATION

Points forts : Stack simple ; séparation nette API / frontend / DB ; patterns explicites.
À préciser en implémentation : Schéma SQL détaillé, format export CSV.
Évolutions futures : Auth détaillée, migration MODX.

6.4 Priorité d'implémentation

1. `sql/schema.sql`
2. `api/config.php`
3. Endpoints de base : `clients.php`, `factures.php`, `paiements.php`, `echeanciers.php`

Document de référence pour toute question d'architecture. À mettre à jour en cas d'ajout d'écrans, d'endpoints ou d'intégrations externes.

Schéma relationnel simplifié et tables de données

Application : dirCops
Projet : Suivi facturation / échéancier / paiement (EMGI)
Date : 2026-02-09
Référence physique : `app/sql/schema.sql`

---

1. Schéma relationnel simplifié

Représentation des tables et des relations (clés primaires et étrangères). Les cardinalités sont indiquées (1:N, 0:1).



Relations :

- **clients** 1 —— N **factures** (un client a plusieurs factures ; une facture appartient à un client).
- **factures** 1 —— N **paiements** (une facture a plusieurs paiements ; un paiement est rattaché à une facture).
- **factures** 1 —— 0..1 **echeanciers** (une facture a au plus un écheancier ; un écheancier est rattaché à une facture).

Contraintes :

- Suppression en cascade : suppression d'un client → suppression de ses factures, puis de leurs paiements et écheanciers. Suppression d'une facture → suppression de ses paiements et de son écheancier.

---

2. Tables – Détail des champs

Format : pour chaque table, description courte puis pour chaque colonne : **Nom**, **Type**, **Description**, **Clé**.

Table `clients`

Table des clients (sociétés ou entités facturées). Une facture est toujours liée à un client.

Nom	Type	Description	Clé
id	int UNSIGNED	Identifiant unique auto incrémenté	Primaire
nom	varchar(255)	Nom du client (raison sociale ou libellé)	
created_at	datetime	Date et heure de création de l'enregistrement	

Table `factures`

Table des factures pour le suivi. La facturation (document, PDF) est gérée dans Pennylane ; ici on enregistre les données de suivi (n° facture, montant, type produit, cycle, quantités, etc.).

Nom	Type	Description	Clé
id	int UNSIGNED	Identifiant unique auto incrémenté	Primaire
client_id	int UNSIGNED	Référence du client concerné	Étrangère (→ clients.id), NOT NULL
n_facture	varchar(64)	Numéro de facture (ex. FA-001)	NOT NULL
n_vente	varchar(64)	Numéro de vente ou référence vente (optionnel)	
date_facture	date	Date de la facture	NOT NULL
type_produit	varchar(32)	Type de produit : alevins, larves (ou NULL)	
cycle	varchar(16)	Cycle de référence (ex. 24-25, 25-26)	
quantites	decimal(12,2)	Quantité facturée	
qte_livree	decimal(12,2)	Quantité déjà livrée	
montant_ht	decimal(12,2)	Montant HT de la facture (€)	NOT NULL
created_at	datetime	Date et heure de création de l'enregistrement dans l'appli	

Index : `client_id`, `type_produit`, `cycle`.

Table `paiements`

Table des paiements enregistrés pour une facture. L'encours d'une facture est calculé comme (montant_ht − somme des montants des paiements de cette facture).

Nom	Type	Description	Clé
id	int UNSIGNED	Identifiant unique auto incrémenté	Primaire
facture_id	int UNSIGNED	Référence de la facture concernée	Étrangère (→ factures.id), NOT NULL
montant	decimal(12,2)	Montant du paiement (€)	NOT NULL
date_paiement	date	Date du paiement	NOT NULL
created_at	datetime	Date et heure de création de l'enregistrement	

Index : `facture_id`, `date_paiement` (idx_paiements_date).

Contraintes de validation : `CHECK (montant > 0)`.

Table `echeanciers`

Table des échéanciers de paiement associés à une facture. Une facture peut avoir au plus un échéancier. Les dates d'échéance sont dérivées à l'affichage : date_debut_echeancier + 0 mois, +1 mois, +2 mois, etc. (échéances mensuelles). Le champ `montant_par_echeance` est calculé à la création (encours / nb_echeances) et stocké.

Nom	Type	Description	Clé
id	int UNSIGNED	Identifiant unique auto incrémenté	Primaire
facture_id	int UNSIGNED	Référence de la facture concernée	Étrangère (→ factures.id), NOT NULL, UNIQUE
nb_echeances	int UNSIGNED	Nombre d'échéances prévues	NOT NULL
date_debut_echeancier	date	Date de début de l'échéancier (1ère échéance)	NOT NULL
montant_par_echeance	decimal(12,2)	Montant par échéance (calculé à la création : encours / nb_echeances)	NOT NULL
created_at	datetime	Date et heure de création de l'enregistrement	

Index : `facture_id` (UNIQUE pour garantir un échéancier par facture).

Contraintes de validation : `CHECK (nb_echeances > 0)`, `CHECK (montant_par_echeance > 0)`.

---

3. Résumé des clés et contraintes

Table	Clé primaire	Clés étrangères	Contrainte cascade
clients	id	—	—
factures	id	client_id → clients	ON DELETE CASCADE (client supprimé)
paiements	id	facture_id → factures	ON DELETE CASCADE (facture supprimée)
echeanciers	id	facture_id → factures	ON DELETE CASCADE (facture supprimée)

3.1 Contraintes de validation (CHECK)

Table	Contrainte	Description
paiements	chk_paiements_montant	montant > 0
echeanciers	chk_echeanciers_nb_echeances	nb_echeances > 0
echeanciers	chk_echeanciers_montant	montant_par_echeance > 0

3.2 Index (référence app/sql/schema.sql)

Table	Index	Colonne(s)
factures	idx_factures_client_id	client_id
factures	idx_factures_type_produit	type_produit
factures	idx_factures_cycle	cycle
paiements	idx_paiements_facture_id	facture_id
paiements	idx_paiements_date	date_paiement
echeanciers	uk_echeanciers_facture_id (UNIQUE)	facture_id
echeanciers	idx_echeanciers_facture_id	facture_id

Note : Les statuts « payé » / « en attente » ne sont pas stockés ; ils sont calculés (encours facture = montant_ht − somme des paiements ; si encours ≤ 0 alors payé, sinon en attente).

Schéma des workflows métier

Application : dirCops
Projet : Suivi facturation / échéancier / paiement (EMGI)
Date : 2026-02-09
Objectif : Décrire de façon détaillée tous les workflows (états, transitions, acteurs, règles). En MVP il n'y a qu'un seul cycle de vie formalisé : celui de la facture (état dérivé de l'encours).

---

1. Vue d'ensemble

Dans ce projet, les **workflows sont volontairement simples** : pas de chaîne de validation (RH, IT), pas de statuts stockés en base pour les factures, pas de SLA ni relances. Le seul cycle de vie explicite est celui de la **facture** (état calculé : en attente / payée). Les autres entités (client, paiement, écheancier) n'ont pas de machine à états.

Workflow / entité	États ?	Stocké en base ?	Acteur	Détail ci-dessous
Facture (suivi)	Oui (2 états dérivés)	Non (calculé)	Commercial / système	§ 2
Paiement	Non	—	Commercial	Création directe, pas de statut
Écheancier	Non	—	Commercial	Création directe, pas de statut
Client	Non	—	—	Pas de cycle de vie métier

---

2. Workflow « Facture (état de suivi) »

L'état d'une facture est **dérivé** des paiements (non stocké en base). Il sert à l'affichage (Payé / En attente) et au filtrage (échéances à venir = factures en attente).

2.1 Entité concernée

- **Table :** `factures` (+ agrégation des `paiements`).
- **Champ état :** aucun. L'état est calculé : `encours_facture = montant_ht - SUM(paiements.montant)` ; si `encours_facture <= 0` → **Payée**, sinon → **En attente**.

2.2 États

Code état	Libellé	Condition (calcul)
`en_attente`	En attente	encours_facture > 0
`paye`	Payée	encours_facture ≤ 0

2.3 Transitions

Transition	Depuis	Vers	Déclencheur	Acteur	Règle métier
Paiement enregistré	En attente	Payée*	POST paiement (montant ≥ encours restant)	Commercial	Automatique après enregistrement du paiement. Pas de validation.

*Une facture peut rester « En attente » après un paiement partiel (plusieurs paiements possibles).

2.4 Diagramme d'états (machine à états simplifiée)



2.5 Règles complémentaires

- **Transition retour possible** : la suppression d'un paiement (DELETE) fait recalculer l'encours ; une facture « Payée » peut redevenir « En attente » si le dernier paiement est supprimé.
- **Pas de validation métier** : la saisie d'un paiement est directe (pas d'approbation RH/IT).
- **Pas de SLA, relances, escalades** : hors périmètre MVP.
- **Historisation** : les dates de création sont dans `factures.created_at` et `paiements.created_at` ; pas d'audit log « qui a changé quel statut ».

---

3. Entités sans workflow (création directe)

3.1 Paiement

- **Création** : formulaire « Enregistrer un paiement » → POST `/api/paiements.php` (facture_id, montant, date_paiement). Acteur : Commercial.

- **Suppression** : bouton Supprimer → confirmation → DELETE `/api/paiements.php?id=`. Acteur : Commercial. L'encours de la facture est recalculé à la volée après suppression.
- **Pas d'états** : pas de statut « brouillon », « validé », etc. Enregistrement direct.

3.2 Écheancier

- **Création** : formulaire « Ajouter un écheancier » → POST `/api/echeanciers.php` (facture_id, nb_echeances, date_debut_echeancier). Acteur : Commercial.
- **Suppression** : bouton Supprimer → confirmation → DELETE `/api/echeanciers.php?id=`. Acteur : Commercial.
- **Pas d'états** : une facture a ou n'a pas d'écheancier ; pas de statut « en cours », « terminé » sur l'écheancier en MVP.
- **Évolution possible** (post-MVP) : statut par échéance (à venir / payée / partielle) si le client le demande.

3.3 Client

- **Création** : formulaire « Ajouter un client » (accueil) → POST `/api/clients.php` (nom). Acteur : Commercial.
- **Suppression** : bouton Supprimer → confirmation → DELETE `/api/clients.php?id=`. En cascade : factures, paiements, écheanciers du client. Vérification applicative possible (refus si factures existantes selon règle métier). Acteur : Commercial.
- Pas de cycle de vie métier (pas de statut actif/archivé en MVP).

---

4. Suppressions (actions destructives)

Entité	Action	Déclencheur	Confirmation	Effet
Client	Suppression	Bouton Supprimer (liste clients)	Oui (avec avertissement cascade)	DELETE clients.php?id= ; cascade factures → paiements, écheanciers. Vérification possible côté API (refus si factures).
Facture	Suppression	Bouton Supprimer (liste factures, fiche client)	Oui	DELETE factures.php?id= ; cascade paiements, écheancier.
Paiement	Suppression	Bouton Supprimer (liste paiements, fiche client)	Oui	DELETE paiements.php?id= ; encours recalculé à la volée.
Écheancier	Suppression	Bouton Supprimer (liste écheanciers, fiche client)	Oui	DELETE echeanciers.php?id= ; facture inchangée.

Aucun workflow de validation : la confirmation utilisateur (dialog) suffit ; pas d'audit log des suppressions en MVP.

---

5. Ce qui n'est pas dans le périmètre workflow (MVP)

- **Validation en plusieurs étapes** (ex. brouillon → nouveau → en attente lettre → en traitement IT → terminé) : non.
- **Statuts RH / IT distincts** : non.
- **SLA, relances, escalades** : non.
- **Audit log des changements d'état** : non.
- **Champ `global_status` / `hr_status` / `it_status` en base** : non (seul l'état dérivé « en attente / payée » pour la facture, calculé à la volée).

---

6. Synthèse pour le chiffrage

Élément	Présent en MVP ?	Détail
États facture (en attente / payée)	Oui	Dérivés, non stockés ; diagramme § 2.4.
Transitions avec acteur	Oui	1 transition : enregistrement paiement → passage à Payée si encours ≤ 0.
Workflow de validation (approbation)	Non	Saisie directe.
Statuts stockés en base	Non	Uniquement calcul à l'affichage.
SLA / relances / escalades	Non	Hors scope.
Historisation / audit log	Non	Uniquement created_at.

Les workflows sont donc **bien détaillés en schéma** pour ce qui existe (facture) ; le reste du projet repose sur des créations directes sans machine à états.

Schéma de chiffrage & grille de complexité — dirCops (EMGI)

Application : Suivi FTI EMGI
Projet : Suivi facturation / échéancier / paiement (EMGI)
Date : 2026-02-09
Public : Client (ampleur du travail), équipes techniques (chiffrage, lots, complexité)

> *Ce document **fusionne** et **remplace** :*
>
> - `schema-chiffrage-projet.md`
> - `grille-complexite-chiffrage.md`
>
> *Objectif : cadrer le périmètre et fournir un référentiel unique pour structurer le backlog et chiffrer.*

---

1) Schéma fonctionnel par modules (WBS / lots chiffrables)

Objectif : découper le projet en lots livrables et chiffrables.
Livrable : backlog structuré par module + dépendances.

Modules identifiés (périmètre aligné)

Module	Fonctionnalités	Règles métier principales	Dépendances	Priorité
Config / Fondations	Base de données, config (`.env` , PDO), mode mock vs DB	Mock et DB doivent rester cohérents (mêmes règles métier)	—	P0
Clients	Liste, détail client (en-tête + encours), ajout, suppression	Encours = somme factures − somme paiements (via Calculs)	Calculs	P0
Factures	Liste par client, ajout, modification, suppression	Facturation source Pennylane (ici : **suivi**) ; pas de contrainte d'unicité `n_facture` en MVP	Clients	P0
Paiements	Liste par client, enregistrement, suppression	montant > 0 ; recalcul encours après ajout/suppression	Factures, Calculs	P0
Écheanciers	Liste, création, suppression ; affichage par client/facture	1 écheancier max par facture ; échéances dérivées (ex. mensuelles)	Factures	P0
Calculs financiers	Encours par client, encours total, échéances à venir	Encours client = Σ montant_ht − Σ paiements ; état facture dérivé (payée / en attente)	Factures, Paiements	P0
Reporting	Tableau de bord, export encours CSV, export factures CSV, Top clients, Quantités par client	Agrégations par client/type/cycle ; exports compatibles Excel (UTF-8, séparateur `;`)	Calculs, Factures	P0
Accès (optionnel selon hébergement)	Login Livinweb ModX (identifiant/mot de passe), SSO en option	Accès restreint ; pas de RBAC multi-profils en MVP (option post-MVP)	Config	P1

Lots recommandés (périmètre actuel)

- **Lot 1 — Fondations & données** : schéma DB, config, mode mock, CRUD clients, liste + détail client.
- **Lot 2 — Factures & paiements** : APIs factures/paiements, recalcul encours, intégration page client.
- **Lot 3 — Écheanciers** : API écheanciers, formulaire + affichage, règle 1 écheancier / facture.
- **Lot 4 — Reporting** : dashboard, exports CSV, Top clients, Quantités clients.
- **Lot 5 — Accès (si requis)** : intégration login ModX / restriction d'accès (et SSO si demandé).

---

2) Schéma des rôles et permissions (matrice RBAC)

Objectif : éviter l'oubli fréquent du chiffrage (droits, écrans, actions par profil).
Livrable : matrice rôles × ressources/actions.

Contexte projet (aligné) :

- **MVP** : un seul profil (**Commercial / Admin**) avec tous les droits.
- **Option** : ajouter un profil **Lecture seule** (Trésorerie / Direction).

Matrice RBAC (cible)

Ressource / Action	Commercial / Admin	Lecteur (Tréso / Direction)
Clients	CRUD	R
Factures	CRUD	R
Paiements	CRD	R
Écheanciers	CRD	R
Calculs / Encours	R	R
Reporting / Export	R + Export	R + Export

Notes :

- **Pas de** délégation/héritage en MVP.
- **Pas de** multi-tenant (une seule entité EMGI).
- **Audit log** (qui a fait quoi) : hors MVP.

Références : `app/ROLES-PERMISSIONS-RBAC.md` , `app/ACTEURS-UTILISATEURS.md` .

---

3) Schéma de données (référence)

Objectif : chiffrer le back (tables, règles, index).
Livrable : schéma relationnel + description des champs.

Entités (simplifié) :

- **clients** (1) → (N) **factures**
- **factures** (1) → (N) **paiements**
- **factures** (0..1) → (1) **echeanciers**

Règles métier données (résumé) :

- Client : nom obligatoire ; suppression en cascade des factures (et donc paiements, écheanciers).
- Paiement : lié à une facture ; montant > 0 ; somme paiements vs montant facture = règle à préciser (bloquer/alerter).
- Écheancier : au plus un par facture ; dates d'échéance dérivées (règle : mensuelle, etc.).

Document détaillé : `schema-relationnel-et-tables-donnees.md` .
Schéma SQL : `app/sql/schema.sql` (+ `app/sql/migration-*.sql`).

---

4) Schéma des workflows métier (états et transitions)

Objectif : chiffrer validations, statuts, automatisations (léger sur ce projet).
Livrable : diagramme d'états + règles.

Résumé (aligné) :

- **Facture** : 2 états **dérivés** (En attente / Payée), **non stockés** ; Payée si encours ≤ 0 après paiement(s).
- **Paiement, écheancier, client** : pas de workflow (création directe).
- **Hors MVP** : SLA/relances/escalades, validations multi-étapes, audit log.

Document détaillé : `schema-workflows.md`.

---

5) Schéma des intégrations et flux (endpoints + points complexes)

Objectif : lister les APIs et points techniques complexes pour un chiffrage réaliste.
Livrable : liste des endpoints + points complexes.

Endpoints (API REST)

Méthode	Endpoint	Utilité
GET	`/api/clients.php?id=`	Détail d'un client (page client)
GET	`/api/clients.php`	Liste des clients
POST	`/api/clients.php`	Création d'un client
DELETE	`/api/clients.php?id=`	Suppression d'un client (cascade)
GET	`/api/factures.php?client_id=`	Liste des factures d'un client (avec encours/état)
POST	`/api/factures.php`	Ajout d'une facture
PUT	`/api/factures.php`	Modification d'une facture
DELETE	`/api/factures.php?id=`	Suppression d'une facture (cascade)
GET	`/api/paiements.php?client_id=`	Liste des paiements d'un client
POST	`/api/paiements.php`	Enregistrement d'un paiement
DELETE	`/api/paiements.php?id=`	Suppression d'un paiement
GET	`/api/echeanciers.php?client_id=` ou `?facture_id=`	Liste des écheanciers (client ou facture)
POST	`/api/echeanciers.php`	Création d'un écheancier
DELETE	`/api/echeanciers.php?id=`	Suppression d'un écheancier
GET	`/api/calculs.php?client_id=`	Encours d'un client
GET	`/api/calculs.php`	Encours global + échéances à venir (dashboard)
GET	`/api/reporting.php?format=csv`	Export encours CSV
GET	`/api/reporting.php?format=factures_csv`	Export factures CSV
GET	`/api/top-clients.php?type=&cycle=`	Top clients (type/cycle)
GET	`/api/quantites-clients.php?type=`	Quantités facturées/livrées par client

Points complexes à chiffrer spécifiquement (alignés)

Point	Description	Complexité
Encours + état facture	Encours = montant_ht − Σ paiements ; état dérivé ; cohérence après chaque mutation paiement	Moyenne
Échéances à venir	Dériver dates/montants à partir de l'écheancier ; filtrer date ≥ aujourd'hui ; tri	Faible → Moyenne
Export CSV (Excel)	UTF-8 (BOM si nécessaire), séparateur `;`, échappement, formats dates/nombres	Faible
Agrégations Top clients	Agrégations type_produit/cycle ; totaux et pourcentages	Moyenne
Mode mock vs DB	Même comportement métier entre JSON et PDO ; éviter divergence	À ne pas surchiffrer (discipline)

---

6) Schéma d'architecture applicative (rappel)

Objectif : visualiser les blocs du système pour répartir le travail.
Livrable : diagramme composants.

Résumé :

- **Frontend** : HTML + JS vanilla + Tailwind (MPA : une page = un écran)
- **API** : PHP, un script par ressource (`clients` , `factures` , `paiements` , `echeanciers` , `calculs` , `reporting` , `top-clients` , `quantites-clients`)
- **Données** : soit mock (JSON), soit DB (PDO)

Document étendu : `architecture-generale-fonctionnelle-et-interfaces.md` et `architecture.md` .

---

7) Grille de complexité (référentiel — version épurée)

Usage : barème d'estimation (ordre de grandeur) pour les lots/modules ci-dessus.

7.1 Écrans / Pages

Type d'écran	Complexité	Estimation (ordre de grandeur)	Détail
CRUD simple	Faible	1–2 j	Liste + formulaire, sans calculs complexes
Avec agrégations/calculs	Moyenne	2–3 j	Plusieurs sources + calculs (fiche client, dashboard)
Reporting avancé	Élevée	3–4 j	Filtres, agrégations, export (Top clients)
Auth (si applicable)	Faible	0,5–1 j	Page login + session (si intégration existante)

7.2 API / Backend

Type d'API	Complexité	Estimation (ordre de grandeur)	Détail
CRUD simple (GET/POST)	Faible	~0,5 j	Une table, validation simple
Calculs / agrégations	Moyenne	1–2 j	Logique métier, agrégations
Règles métier avancées	Élevée	2–3 j	Contraintes multiples, transactions
Export CSV/JSON	Faible	0,5–1 j	Génération fichier, encodage, téléchargement

7.3 Tests & qualité (si chiffrés)

Type	Estimation (ordre de grandeur)	Détail
Tests manuels par écran	0,25–0,5 j	Cas nominaux + limites
Tests API par endpoint	~0,25 j	Postman/curl + erreurs
Recette complète (cahier)	2–3 j	Exécution critères + corrections mineures

7.4 Estimation indicative par module (ordre de grandeur)

> *Cette estimation est un **point de départ** (à ajuster selon les décisions finales et le niveau d'exigence). Elle **n'inclut pas** les options hors périmètre listées en section 8.*

Module	Écrans	API	Tests	Total indicatif
Config / Fondations	—	1 j	0,25 j	1,25 j
Accès (login ModX / restriction) *(optionnel)*	1 j	0,5 j	0,25 j	1,75 j
Clients	1,5 j	0,5 j	0,5 j	2,5 j
Factures	2 j	1,5 j	0,5 j	4 j
Paiements	1,5 j	1 j	0,5 j	3 j
Écheanciers	1,5 j	1,5 j	0,5 j	3,5 j
Calculs financiers	—	2 j	0,5 j	2,5 j
Reporting / Dashboard + exports CSV	2,5 j	1,5 j	0,5 j	4,5 j
Top clients	2 j	1 j	0,5 j	3,5 j
Quantités clients	1,5 j	1 j	0,5 j	3 j
Intégration / Recette	—	—	2 j	2 j
Documentation technique	—	—	0,5 j	0,5 j

Notes :

- Total indicatif **avec** accès ModX : ~32 j (ordre de grandeur).
- Total indicatif **sans** accès ModX : ~30 j (ordre de grandeur).

---

8) Alignement périmètre (ce qu'on garde / ce qu'on retire)

Cette section est la consolidation des décisions de périmètre pour éviter de chiffrer du hors-scope.

À retirer / hors périmètre (MVP)

- Recherche client par nom (FR1) **retirée**
- Contrainte d'unicité `n_facture` **non prévue**
- Cibles de performance (NFR) « < 3 s / < 1 s » **non contractualisées ici**
- Infra/CI-CD/monitoring **hors doc de chiffrage MVP**
- Workflows/statuts avancés (multi-étapes, SLA, relances, escalades, audit log) **hors MVP**
- Multi-tenant, délégation/héritage **hors MVP**

Fonctionnalités à conserver (périmètre à chiffrer)

Domaine	Fonctionnalité	Fichiers / API concernés
Clients	Liste, détail, ajout, suppression	`app/api/clients.php`, `app/index.html`, `app/client.html`, `app/js/client-detail.js`
Factures	Liste par client, ajout, modification, suppression	`app/api/factures.php`, `app/client.html`, `app/js/facture-form.js`, `app/js/client-detail.js`
Paiements	Liste par client, enregistrement, suppression	`app/api/paiements.php`, `app/client.html`, `app/js/paiements.js`
Écheanciers	Liste, création, suppression ; répartition encours	`app/api/echeanciers.php`, `app/client.html`, `app/js/echeancier-form.js`, `app/js/client-detail.js`
Calculs	Encours par client, encours total, échéances à venir	`app/api/calculs.php`, `app/dashboard.html`, `app/js/dashboard.js`
Reporting	Tableau de bord, export encours CSV, export factures CSV	`app/api/reporting.php`, `app/dashboard.html`
Meilleurs clients	Top clients par type (alevins/larves), cycle	`app/api/top-clients.php`, `app/top-clients.html`, `app/js/top-clients.js`
Quantités	Quantités facturées / livrées par client, filtre type	`app/api/quantites-clients.php`, `app/quantites-clients.html`, `app/js/quantites-clients.js`
Accès	Login Livinweb ModX (identifiant / mot de passe), SSO en option	Côté hébergement / MODX
Rôles	Un seul rôle : Commercial / Admin (tous droits) ; Lecteur en option	`app/ROLES-PERMISSIONS-RBAC.md`, `app/ACTEURS-UTILISATEURS.md`

Règles métier à clarifier

Application : dirCops
Projet : Suivi facturation / échéancier / paiement (EMGI)
Date : 2026-02-09
Usage : Liste des ambiguïtés métier à trancher avant développement ou chiffrage final

---

Règles déjà clarifiées ✅

Règle	Décision	Impact développement
Authentification	Login MODX Livinweb (identifiant/mot de passe)	Écran login + session PHP
Rôles	Un seul rôle en MVP (multi-rôles en option)	Pas de gestion de permissions
Import Pennylane	EN OPTION (MVP = saisie manuelle)	Aucun en MVP
Modification/suppression	Autorisé (facture, paiement, écheancier, client) avec confirmation	CRUD complet
Suppression client	Cascade (factures → paiements/écheanciers)	Confirmation + message avertissement

---

Règles à clarifier avec le client

1. Factures

1.1 Unicité du numéro de facture ⚠️ PRIORITÉ HAUTE

Question : Un client peut-il avoir deux factures avec le même numéro de facture ?

Options :

- **A) Unicité globale** : un n° facture ne peut exister qu'une seule fois dans toute la base
  - Contrainte DB : `UNIQUE(n_facture)`
  - Validation API : vérifier unicité avant INSERT
- **B) Unicité par client** : un client ne peut pas avoir deux factures avec le même n°, mais deux clients différents peuvent
  - Contrainte DB : `UNIQUE(client_id, n_facture)`
  - Validation API : vérifier unicité par client

Recommandation technique : Option B (unicité par client) — plus souple et évite blocages si deux clients ont un n° similaire

Impact :

- Option A : contrainte globale simple

- Option B : contrainte composite, requête légèrement plus complexe

Décision client : *À trancher*

1.2 Champs facture obligatoires ou optionnels ⚠️ PRIORITÉ MOYENNE

Question : Tous les champs facture sont-ils requis ou certains peuvent être laissés vides ?

Champs actuels :

Champ	Obligatoire actuel	À confirmer
n_facture	Oui (NOT NULL)	✅ Confirmé
n_vente	Non (NULL)	À confirmer
date_facture	Oui (NOT NULL)	✅ Confirmé
type_produit	Non (NULL)	**À clarifier** : si vide, impact reporting Top clients ?
cycle	Non (NULL)	**À clarifier** : si vide, impact reporting par cycle ?
quantites	Non (NULL)	**À clarifier** : si vide, impact reporting Quantités clients ?
qte_livree	Non (NULL)	**À clarifier** : si vide, OK ou bloquer ?
montant_ht	Oui (NOT NULL)	✅ Confirmé

Recommandation :

- Si un champ est **utilisé dans un reporting obligatoire**, le rendre **obligatoire** à la saisie (validation front + back)
- Exemple : si "Top clients par type" est essentiel, alors `type_produit` devrait être obligatoire

Décision client : *À trancher (valider si type_produit, cycle, quantites obligatoires ou pas)*

1.3 TVA et devise ⚠️ PRIORITÉ BASSE

Question : Faut-il ajouter des champs TVA et devise ?

Contexte : Actuellement seul le montant HT est stocké. Pas de gestion de TVA, TTC, ou devise.

Options :

- **MVP actuel** : montant HT uniquement (€ implicite)
- **Option évoluée** : ajouter champs `tva` (%), `montant_ttc`, `devise` (EUR, USD, etc.)

Recommandation : Garder MVP simple (HT uniquement, € implicite) sauf besoin explicite client

Décision client : *À trancher*

---

2. Paiements

2.1 Paiement supérieur au montant de la facture ⚠️ PRIORITÉ HAUTE

Question : Que se passe-t-il si un utilisateur enregistre un paiement > montant HT de la facture ?

Options :

- **A) Autoriser et encours = 0** : le paiement est enregistré, l'encours de la facture = 0 (pas de crédit client)
  - Avantage : souplesse (erreur client, arrondi, geste commercial)
  - Inconvénient : "trop-perçu" non suivi
- **B) Bloquer** : validation API refuse le paiement si somme des paiements (+ nouveau paiement) > montant HT
  - Avantage : cohérence stricte
  - Inconvénient : blocage si arrondi ou erreur de saisie facture

Recommandation technique : Option A (autoriser, encours = 0) avec avertissement visuel dans l'interface ("Paiement supérieur au montant, facture soldée")

Décision client : *À trancher*

2.2 Date de paiement dans le futur ⚠️ PRIORITÉ MOYENNE

Question : Peut-on enregistrer un paiement avec une date future (paiement "prévu") ?

Options :

- **A) Autoriser** : date libre (passé ou futur)
  - Utile pour anticiper des paiements prévus
  - Risque : confusion entre "reçu" et "prévu"
- **B) Bloquer date future** : validation API refuse si date > aujourd'hui
  - Plus strict : paiement = réellement reçu

Recommandation : Option B (bloquer date future) — si besoin de "paiement prévu", utiliser les échéanciers

Décision client : *À trancher*

---

3. Échéanciers

3.1 Fréquence des échéances ⚠️ PRIORITÉ MOYENNE

Question : Les échéances sont-elles toujours **mensuelles** ou faut-il permettre d'autres fréquences ?

Options :

- **A) Mensuel uniquement** (actuel) : date_debut + N mois
  - Simple à implémenter
  - Couvre la majorité des cas

- **B) Fréquence paramétrable** : mensuel, trimestriel, annuel, personnalisé
  - Plus flexible
  - +0,5-1 jour de développement (champ `frequence`, calcul dates adapté)

Recommandation : Option A (mensuel) pour MVP, option B en post-MVP si besoin réel

Décision client : *À trancher (mensuel suffit-il ?)*

3.2 Lien paiement ↔ échéance ⚠️ PRIORITÉ BASSE

Question : Faut-il lier un paiement à une échéance spécifique (suivi par échéance : à venir / payée / partielle / en retard) ?

Options :

- **A) Pas de lien** (actuel) : les paiements sont rattachés à la facture globalement, l'encours diminue
  - Simple : pas de table de liaison
- **B) Avec lien** : chaque paiement peut être affecté à une échéance
  - Plus précis : suivi échéance par échéance
  - +1-2 jours de développement (table liaison, interface)

Recommandation : Option A pour MVP (suivi global facture suffit), option B en post-MVP si demandé

Décision client : *À trancher*

---

4. Clients

4.1 Clients en doublon ⚠️ PRIORITÉ MOYENNE

Question : Comment gérer les clients avec des noms similaires (doublons potentiels) ?

Exemples :

- "AQUANARIA" vs "Aquanaria" vs "AQUANARIA SAS"
- "AQN" vs "AQN (Aqua Nord)"

Options :

- **A) Pas de vérification** : l'utilisateur peut créer "AQUANARIA" et "Aquanaria" comme deux clients distincts
  - Risque : doublons non détectés
- **B) Vérification suggérée** : lors de l'ajout, si un client avec un nom proche existe, afficher avertissement "Un client similaire existe : [Nom]"
  - +0,5 jour de développement (recherche floue, UI avertissement)
- **C) Contrainte stricte** : unicité du nom (insensible à la casse)
  - Contrainte DB : `UNIQUE(LOWER(nom))`

Recommandation : Option B (avertissement) — souple mais évite doublons accidentels

Décision client : *À trancher*

---

5. Exports

5.1 Format et encodage des exports CSV ⚠️ PRIORITÉ HAUTE

Question : Quel séparateur, encodage et format pour les exports CSV ?

Options :

Paramètre	Option A (Excel FR)	Option B (Standard)
Séparateur	`;` (point-virgule)	`,` (virgule)
Encodage	UTF-8 **avec BOM**	UTF-8 sans BOM
Décimale	`,` (virgule)	`.` (point)
Format date	`DD/MM/YYYY`	`YYYY-MM-DD`

Recommandation : Option A (Excel FR) si le client utilise Excel sur Windows en français
Recommandation alternative : Option B si interopérabilité avec outils internationaux

Décision client : *À trancher (test avec un export pour valider)*

5.2 Contenu de l'export "Encours par client" ⚠️ PRIORITÉ MOYENNE

Question : Quelles colonnes exactement dans l'export CSV "encours par client" ?

Colonnes actuelles (proposées) :

- Nom client
- Encours (€)
- (Optionnel) Accord
- (Optionnel) J out
- (Optionnel) Situation

À clarifier :

- Inclure aussi : nombre de factures en attente ? montant total facturé ? total payé ?
- Format montants : avec symbole € ou sans ?

Décision client : *À trancher (liste exacte des colonnes souhaitées)*

5.3 Export "Ensemble des factures (PJ)" ⚠️ PRIORITÉ MOYENNE

Question : Quelles colonnes et quel format pour l'export factures (suivi-factures-YYYY-MM-DD.csv) ?

Colonnes proposées (à valider) :

- Client

- N° facture
- N° vente
- Date facture
- Type produit
- Cycle
- Quantités facturées
- Quantités livrées
- Montant HT
- Total payé
- Encours
- État (Payé/En attente)

Colonnes supplémentaires possibles :

- EXO (exercice comptable, dérivé de la date facture)
- Mois de facturation
- Délai de paiement (date facture → date dernier paiement)

Décision client : *À trancher (colonnes exactes + format)*

---

6. Reporting et indicateurs

6.1 Top clients : limite ou pas ? ⚠️ PRIORITÉ BASSE

Question : Faut-il limiter l'affichage aux "Top 10" ou "Top 15" clients, ou afficher tous les clients selon le filtre ?

Options :

- **A) Limite (Top 10/15)** : affichage des N meilleurs clients seulement
  - Plus lisible si beaucoup de clients
- **B) Pas de limite** : afficher tous les clients du filtre, triés par quantités/CA décroissant
  - Plus complet, filtrage côté client (Excel) si besoin

Recommandation : Option B (pas de limite) pour MVP — le client peut exporter et filtrer dans Excel si besoin

Décision client : *À trancher*

6.2 Indicateurs "type Excel" (couverture, % GTT) ⚠️ PRIORITÉ MOYENNE

Question : Les indicateurs suivants sont-ils requis en MVP ou en post-MVP ?

Indicateur	Calcul	MVP ou post-MVP ?
Couverture livraison	(Qté livrée / Qté facturée) × 100	*À trancher*
% GTT (écart livraison)	Dérivé de couverture	*À trancher*
Délai moyen de paiement	(Somme des délais par facture payée) / Nb factures payées	*À trancher*

Recommandation : Post-MVP (priorité au suivi encours + paiements en MVP)

Décision client : *À trancher*

---

7. Tableau de bord

7.1 Colonnes optionnelles (Accord, J out, Situation) ⚠️ PRIORITÉ BASSE

Question : Les colonnes "Accord", "J out", "Situation" doivent-elles être affichées sur le dashboard ?

Contexte : Ces colonnes sont mentionnées dans les captures Excel "COMEX", mais les données ne sont pas dans la base actuelle.

Options :

- **A) Afficher avec valeurs vides ("—")** : colonnes présentes mais non alimentées
  - Prépare l'évolution future
  - Risque : confusion utilisateur ("pourquoi vide ?")
- **B) Masquer pour MVP** : ne pas afficher ces colonnes
  - Plus propre
  - À ajouter en post-MVP si les données sont disponibles

Recommandation : Option B (masquer) pour MVP

Décision client : *À trancher*

Récapitulatif des priorités

Priorité	Règle à clarifier	Impact chiffrage
🔴 HAUTE	1.1 Unicité n° facture	Contrainte DB + validation
🔴 HAUTE	2.1 Paiement > montant facture	Logique métier
🔴 HAUTE	5.1 Format export CSV	Développement export
🟠 MOYENNE	1.2 Champs facture obligatoires	Validation front/back
🟠 MOYENNE	2.2 Date paiement future	Validation
🟠 MOYENNE	3.1 Fréquence échéances	Calcul dates
🟠 MOYENNE	4.1 Clients en doublon	Vérification suggérée
🟠 MOYENNE	5.2 Colonnes export encours	Contenu export
🟠 MOYENNE	5.3 Colonnes export factures PJ	Contenu export
🟠 MOYENNE	6.2 Indicateurs type Excel	Scope MVP
🟢 BASSE	1.3 TVA et devise	Champs supplémentaires
🟢 BASSE	3.2 Lien paiement ↔ échéance	Table liaison
🟢 BASSE	6.1 Top clients limite	Affichage
🟢 BASSE	7.1 Colonnes optionnelles dashboard	Affichage

Recommandations pour l'atelier client

Objectif : Trancher les règles 🔴 HAUTE et 🟠 MOYENNE avant chiffrage final

Méthode :

1. Présenter chaque règle avec les options A/B/C
2. Expliquer l'impact technique et métier
3. Faire choisir le client (décision actée dans le compte-rendu)
4. Mettre à jour ce document avec les décisions

Durée estimée : 1h d'atelier

Document à compléter au fur et à mesure des décisions client. Référence pour développement et tests.

Décisions client — Chiffrage MVP

Application : dirCops (facturation / échéancier / paiement)

Décisions actées

Décision	Périmètre MVP	Impact
Authentification	Login MODX Livinweb (id/mot de passe) — écran login, session, protection API/pages, déconnexion	~1-2 jours
Pennylane	Saisie manuelle uniquement (suivi comme Excel). Facturation document reste dans Pennylane. Import = option post-MVP	0 jour
Rôles	Un seul rôle : tous les utilisateurs ont les mêmes droits (CRUD complet). Pas de distinction Commercial/Lecteur	0 jour

Questions à trancher

Question	Impact estimé
Export : format CSV (champs, encodage, séparateur)	0,5-1 j
Hébergement (où, déploiement, sauvegardes)	0,5-2 j
Migration MODX : confirmée, quand ?	Architecture
Indicateurs Excel (couverture, % GTT) : MVP ou après ?	1-2 j si MVP
Top clients : limite (10/15) ou pas ?	Minime
Safari requis ?	~0,25 j
Accessibilité WCAG A vs AA	0,5-1 j si AA
Échéances mensuelles ou personnalisables ?	+0,5 j si personnalisable
Suivi paiement ↔ échéance ?	+1 j si requis

Règles métier (recommandations internes)

- **Paiement > montant facture** : autoriser ; encours = 0 (pas de crédit client)
- **Unicité n° facture** : par client (UNIQUE client_id, n_facture)
- **Suppression** : cascade avec message de confirmation
- **Export CSV** : séparateur `;`, UTF-8 BOM (Excel FR)
- **Échéances** : mensuelles (date_debut + N mois)

Référence pour le chiffrage. Mise à jour au fil des décisions.

Logique paiements et écheanciers - Spécifications techniques

Application : dirCops
Date : 2026-02-09
Objectif : Définir la logique métier et technique pour les paiements et écheanciers, inspirée des bonnes pratiques ERP tout en restant simple.

---

1. Principes généraux

1.1 Source de vérité

- **Facture** : montant HT fixe (ne change pas après création).
- **Paiements** : liste des paiements enregistrés (un paiement = un enregistrement).
- **Encours** : calculé à la volée = `montant_ht - SUM(paiements.montant)`.
- **État facture** : dérivé de l'encours (`encours <= 0` → Payée, sinon → En attente).

1.2 Règles métier

1. **Un paiement est toujours positif** : `montant > 0`.
2. **Un paiement est lié à une facture** : `facture_id` obligatoire.
3. **L'encours ne peut pas être négatif** : si `SUM(paiements) > montant_ht`, l'encours = 0 (pas de crédit client géré).
4. **Un écheancier = un plan de paiement** : une facture peut avoir au plus un écheancier.
5. **Montant par échéance** : calculé au moment de la création de l'écheancier = `encours_actuel / nb_echeances`.

---

2. Calcul de l'encours

2.1 Formule standardisée

```
encours_facture = GREATEST(0, montant_ht - COALESCE(SUM(paiements.montant), 0))
```

Explication :

- `GREATEST(0, ...)` : garantit que l'encours ne soit jamais négatif.
- `COALESCE(..., 0)` : si aucun paiement, somme = 0.
- Arrondi à 2 décimales pour éviter les problèmes de précision.

2.2 Encours par client

```
encours_client = SUM(encours_facture) pour toutes les factures du client
```

2.3 Cas limites gérés

Cas	Comportement
Paiement supérieur au montant HT	Encours = 0 (pas de crédit)
Plusieurs paiements partiels	Encours = montant_ht - somme des paiements
Aucun paiement	Encours = montant_ht
Arrondis décimaux	Utilisation de ROUND() pour éviter les écarts

---

3. Écheancier de paiement

3.1 Structure

- **Une facture = un écheancier maximum** (contrainte d'unicité).
- **Champs** : `facture_id`, `nb_echeances`, `date_debut_echeancier`, `montant_par_echeance` (calculé et stocké).

3.2 Calcul du montant par échéance

Au moment de la création de l'écheancier :

```
montant_par_echeance = ROUND(encours_actuel / nb_echeances, 2)
```

Exemple :

- Facture : 1000 € HT
- Paiements déjà reçus : 200 €
- Encours actuel : 800 €
- Nombre d'échéances : 3
- Montant par échéance : 800 / 3 = 266.67 €

3.3 Dates d'échéances (calculées, non stockées)

Les dates sont calculées à l'affichage :

- Échéance 1 : `date_debut_echeancier`
- Échéance 2 : `date_debut_echeancier + 1 mois`
- Échéance 3 : `date_debut_echeancier + 2 mois`
- etc.

Fréquence : mensuelle (toujours).

3.4 Règles de création

1. **Un écheancier ne peut être créé que si** :
   - La facture existe.
   - La facture n'a pas déjà d'écheancier.
   - La facture a un encours > 0 (sinon pas besoin d'écheancier).
2. **Le montant par échéance est calculé au moment de la création** :
   - Basé sur l'encours actuel (montant_ht - paiements déjà reçus).
   - Stocké dans `echeanciers.montant_par_echeance` pour éviter de recalculer.

---

4. Base de données

4.1 Contraintes

Table	Contrainte	Détail
`paiements`	`montant > 0`	Un paiement doit être positif
`paiements`	`facture_id` FK	Référence facture obligatoire
`echeanciers`	`facture_id` UNIQUE	Une facture = un écheancier max
`echeanciers`	`nb_echeances > 0`	Au moins une échéance
`echeanciers`	`montant_par_echeance > 0`	Montant positif

4.2 Index recommandés

```
CREATE INDEX idx_paiements_facture_id ON paiements(facture_id);
CREATE INDEX idx_paiements_date ON paiements(date_paiement);
CREATE UNIQUE INDEX idx_echeanciers_facture_id ON echeanciers(facture_id);
```

4.3 Champs ajoutés (amélioration)

Table `echeanciers` :

- `montant_par_echeance DECIMAL(12,2)` : montant calculé au moment de la création.

---

5. Logique applicative

5.1 Enregistrement d'un paiement

Avant insertion :

1. Vérifier que la facture existe.
2. Vérifier que `montant > 0`.

3. Vérifier que `date_paiement` est valide.

Après insertion :

- L'encours est recalculé automatiquement lors des prochaines requêtes.
- Pas de mise à jour explicite nécessaire (calcul à la volée).

5.2 Création d'un échéancier

Avant insertion :

1. Vérifier que la facture existe.
2. Vérifier que la facture n'a pas déjà d'échéancier.
3. Calculer l'encours actuel de la facture.
4. Vérifier que `encours > 0`.
5. Calculer `montant_par_echeance = ROUND(encours / nb_echeances, 2)`.

Insertion :

```
INSERT INTO echeanciers (facture_id, nb_echeances, date_debut_echeancier, montant_par_echeance)
VALUES (?, ?, ?, ?)
```

5.3 Calcul de l'encours (fonction réutilisable)

En SQL :

```
SELECT
    f.id,
    f.montant_ht,
    COALESCE(SUM(p.montant), 0) AS total_paye,
    GREATEST(0, f.montant_ht - COALESCE(SUM(p.montant), 0)) AS encours_facture,
    CASE
        WHEN GREATEST(0, f.montant_ht - COALESCE(SUM(p.montant), 0)) <= 0 THEN 'paye'
        ELSE 'en_attente'
    END AS etat
FROM factures f
LEFT JOIN paiements p ON p.facture_id = f.id
WHERE f.id = ?
GROUP BY f.id, f.montant_ht
```

En PHP (mock) :

```php
function calculerEncoursFacture($factureId, $factures, $paiements) {
    $f = array_filter($factures, fn($x) => $x['id'] == $factureId)[0] ?? null;
    if (!$f) return null;

    $totalPaye = array_sum(array_map(
        fn($p) => (float)$p['montant'],
        array_filter($paiements, fn($p) => $p['facture_id'] == $factureId)
    ));

    $encours = max(0, (float)$f['montant_ht'] - $totalPaye);
    return [
        'montant_ht' => (float)$f['montant_ht'],
        'total_paye' => $totalPaye,
        'encours_facture' => $encours,
        'etat' => $encours <= 0 ? 'paye' : 'en_attente'
    ];
}
```

---

6. Cohérence des données

6.1 Vérifications à faire

Vérification	Quand	Action si échec
Facture existe	Avant paiement/écheancier	Erreur 404
Montant paiement > 0	Avant insertion paiement	Erreur 400
Encours > 0	Avant création écheancier	Erreur 400 "Facture déjà soldée"
Pas d'écheancier existant	Avant création écheancier	Erreur 400 "Écheancier déjà existant"

6.2 Normalisation des montants

- **Stockage** : `DECIMAL(12,2)` (2 décimales).
- **Calculs** : utiliser `ROUND()` pour éviter les écarts.
- **Comparaisons** : utiliser `<= 0.01` pour considérer comme nul (tolérance d'arrondi).

---

7. Échéances à venir (tableau de bord)

Les « échéances à venir » affichées sur le tableau de bord sont calculées à partir des **écheanciers** lorsque la facture en possède un, sinon à partir de la date de facture.

7.1 Règles de calcul

- **Facture avec écheancier :** pour chaque échéance prévue (i = 0 à nb_echeances − 1), on crée une ligne avec :
  - **date_echeance** = `date_debut_echeancier + i mois`
  - **montant_echeance** = `montant_par_echeance` (stocké dans l'écheancier)

- Seules les échéances dont la date est ≥ **aujourd'hui** sont conservées (échéances à venir).

- **Facture sans écheancier :** une seule ligne avec **date_echeance** = date de facture, **montant_echeance** = encours de la facture (si date_facture ≥ aujourd'hui).

7.2 Tri et affichage

- Les lignes sont triées par **date_echeance** croissante.
- L'affichage peut indiquer le numéro d'échéance (ex. « Éch. 1/3 ») pour les factures avec écheancier.

---

8. Exemples de requêtes SQL

8.1 Liste des factures avec encours

```sql
SELECT
    f.*,
    COALESCE(SUM(p.montant), 0) AS total_paye,
    GREATEST(0, f.montant_ht - COALESCE(SUM(p.montant), 0)) AS encours_facture,
    CASE
        WHEN GREATEST(0, f.montant_ht - COALESCE(SUM(p.montant), 0)) <= 0 THEN 'paye'
        ELSE 'en_attente'
    END AS etat
FROM factures f
LEFT JOIN paiements p ON p.facture_id = f.id
WHERE f.client_id = ?
GROUP BY f.id
ORDER BY f.date_facture DESC
```

8.2 Encours par client

```sql
SELECT
    c.id AS client_id,
    c.nom,
    COALESCE(SUM(GREATEST(0, f.montant_ht - COALESCE(sp.total_paye, 0))), 0) AS encours
FROM clients c
LEFT JOIN factures f ON f.client_id = c.id
LEFT JOIN (
    SELECT facture_id, SUM(montant) AS total_paye
    FROM paiements
    GROUP BY facture_id
) sp ON sp.facture_id = f.id
GROUP BY c.id, c.nom
HAVING encours > 0
ORDER BY encours ASC
```

---

9. Évolutions possibles (post-MVP)

- **Rapprochement paiements/échéances** : lier un paiement à une échéance spécifique.

- **Statut par échéance** : à venir / payée / partielle / en retard.
- **Gestion des crédits** : permettre encours négatif (avoir client).
- **Modification/suppression paiements** : avec historique.
- **Échéances non mensuelles** : trimestrielles, annuelles, etc.

Document de référence pour l'implémentation. À mettre à jour si la logique métier évolue.

Cahier des recettes – Livraison application dirCops

Application : dirCops
Projet : Suivi facturation / échéancier / paiement — Écloserie marine (remplacement du fichier Excel)
Version : 1.0
Date : 2026-02-10

Usage : cocher chaque critère après test de recette et indiquer **O (OK)** / **N (Non conforme)** / **NA (Non applicable)** + commentaire si nécessaire.

---

0. Périmètre et règles d'alignement (références)

Écrans/pages (MVP)

- **Clients** : `app/index.html`
- **Fiche client** : `app/client.html?id=...`
- **Tableau de bord** : `app/dashboard.html`
- **Meilleurs clients** : `app/top-clients.html` (**Top 15**)
- **Quantités** : `app/quantites-clients.html`

Référence : `app/LISTING-ECRANS-PAGES.md`.

Accès / rôles

- L'authentification est **Livinweb ModX** (identifiant/mot de passe). Le contrôle d'accès est **hors scope applicatif** si géré par l'environnement de déploiement.
- Rôle MVP : **Commercial / Admin** (lecture/écriture). Autres rôles éventuels = options.

Références : `app/README.md`, `app/ROLES-PERMISSIONS-RBAC.md`, `app/ACTEURS-UTILISATEURS.md`.

Règles métier clés (à respecter en recette)

- **Encours facture** = `montant_ht - somme(paiements)` ; si encours < 0, afficher **0** (pas de crédit client affiché).
- **Paiement > montant facture** : autorisé ; encours affiché = **0**.
- **Unicité** `n_facture` : pas de contrainte en périmètre actuel.
- **Clients en doublon** : non gérés (pas de détection/fusion).
- **Navigateurs** : **Safari non supporté** (ne pas exiger de recette Safari).

Références : `app/REGLES-METIER.md`, `app/LOGIQUE-PAIEMENTS-ECHEANCIERS.md`, `app/QUESTIONS-A-TRANCHER.md`.

Modes de test (au choix)

- **Mode simulation** : données depuis `app/mock/*.json` (pas de MySQL).
- **Mode base** : MySQL/MariaDB + `MOCK_MODE=0` + schéma `app/sql/schema.sql`.

Référence : `app/README.md`.

---

1. Gestion clients

#	Critère de recette	O/N/NA	Commentaires
1.1	Depuis l'accueil **Clients**, la liste des clients s'affiche.		
1.2	La recherche par nom de client filtre correctement la liste (saisie partielle).		
1.3	Depuis l'accueil, on peut **ajouter un client** ; le nouveau client apparaît dans la liste.		
1.4	Depuis l'accueil, on peut **supprimer un client** ; une confirmation est demandée ; la liste est mise à jour (et les données associées sont gérées selon la règle d'implémentation).		
1.5	Les **doublons de clients** (même nom) sont acceptés et affichés tels quels (pas de blocage).		

---

2. Fiche client (affichage)

#	Critère de recette	O/N/NA	Commentaires
2.1	En cliquant sur un client, la **fiche client** s'ouvre avec le bon `id` et le bon nom.		
2.2	La fiche client affiche : **nom**, **encours client**, **liste des factures**, **liste des paiements**, et la zone **échéancier** par facture (si présent).		
2.3	Après une action (ajout/modif/suppression facture/paiement/échéancier), la fiche se met à jour et reste cohérente (montants/états).		

---

3. Factures (suivi)

#	Critère de recette	O/N/NA	Commentaires
3.1	Pour un client, la liste des factures affiche au minimum : **n° facture**, **n° vente**, **date facture**, **type produit**, **cycle**, **quantités**, **montant HT**, **total payé**, **encours**, **état** (Payée / En attente).		
3.2	Le formulaire **Ajouter une facture (suivi)** est accessible et compréhensible.		
3.3	L'ajout d'une facture avec les champs obligatoires (ex. n° facture, date, montant HT) enregistre la facture et l'affiche dans la liste.		
3.4	L'ajout/modification d'une facture avec type (alevins/larves), cycle, quantités et quantité livrée enregistre et ré-affiche correctement ces champs (cas "quantité livrée vide" géré sans erreur).		
3.5	On peut **modifier** une facture : le formulaire se remplit ; après validation, les données sont mises à jour et l'encours/état se recalculent.		

4. Paiements

#	Critère de recette	O/N/NA	Commentaires
4.1	Sur la fiche client, la **liste des paiements** s'affiche (facture concernée, date, montant).		
4.2	Le formulaire **Enregistrer un paiement** propose uniquement des factures **en attente** (encours > 0).		
4.3	La saisie d'un montant et d'une date enregistre le paiement et met à jour l'encours/état de la facture (sans incohérence).		
4.4	Un paiement supérieur au montant restant dû est **accepté** ; l'**encours affiché** de la facture (et du client) reste à **0** (pas de crédit affiché).		

5. Échéancier de paiement

#	Critère de recette	O/N/NA	Commentaires
5.1	Depuis la fiche client, on peut **ajouter un échéancier** à une facture (facture, nb échéances, date de début).		
5.2	Après ajout, l'échéancier apparaît pour la facture concernée (nb échéances, date de début, montant par échéance si affiché).		
5.3	Une facture ne peut avoir **qu'un seul échéancier** ; une tentative de doublon est empêchée (message ou blocage).		
5.4	Une facture **soldée** (encours ≤ 0) ne permet pas de créer un échéancier (blocage + message).		

6. Calculs financiers (encours, états, échéances)

#	Critère de recette	O/N/NA	Commentaires
6.1	L'encours **par facture** correspond à `montant_ht - somme(paiements)` et est plafonné à **0** si négatif.		
6.2	L'encours **par client** correspond à la somme des encours factures du client (après plafonnement à 0).		
6.3	L'état facture est **Payée** si encours ≤ 0, **En attente** sinon, et se met à jour après ajout/suppression de paiement.		
6.4	Le bloc **échéances à venir** (tableau de bord) liste des échéances cohérentes : avec échéancier (date début + N mois, montant par échéance), sans échéancier (date facture + encours).		

7. Tableau de bord / Top clients / Quantités (pages analytiques)

#	Critère de recette	O/N/NA	Commentaires
7.1	Le **tableau de bord** affiche : encours par client, synthèse (dont encours total), échéances à venir, et les liens d'exports.		
7.2	La page **Meilleurs clients** permet de choisir le type (Alevins/Larves) et optionnellement un cycle, et affiche un **Top 15** cohérent (totaux/% cohérents).		
7.3	La page **Quantités** affiche des quantités par client et le filtre type (Alevins/Larves) restreint correctement les lignes.		
7.4	Les liens vers la **fiche client** depuis les pages (dashboard/top/quantités) ouvrent le bon client.		

8. Exports CSV

> *Référence :* `app/EXPORT-CSV-SPEC.md` *(colonnes “retenues” + options à trancher).*

#	Critère de recette	O/N/NA	Commentaires
8.1	Export **Encours par client** : téléchargement d’un fichier `reporting-encours-YYYY-MM-DD.csv` ; séparateur cohérent (ex. `;`), encodage OK (accents), au minimum colonnes **Nom client** et **Encours**.		
8.2	Export **Ensemble des factures** : téléchargement d’un fichier `suivi-factures-YYYY-MM-DD.csv` ; en-têtes/colonnes conformes à la spécification retenue (client, n° facture, dates, type, cycle, quantités, montant, total payé, encours, état).		
8.3	Les formats **dates** (jj/mm/aaaa ou ISO selon décision) et **montants** (virgule/points, symbole € ou non) sont cohérents et documentés.		

---

9. Navigation, erreurs, non-fonctionnel

#	Critère de recette	O/N/NA	Commentaires
9.1	Depuis chaque page, l’en-tête propose la navigation : **Clients**, **Tableau de bord**, **Meilleurs clients**, **Quantités**.		
9.2	Les montants sont affichés en **euros** et les dates dans un format cohérent sur l’ensemble des pages.		
9.3	En cas d’erreur (champ obligatoire, API indisponible, erreur serveur), un message **compréhensible** s’affiche (sans écran blanc).		
9.4	Temps de réponse acceptable en conditions normales (ordre de grandeur : affichage initial < 3s, actions usuelles ~< 1s) et accès restreint aux personnes autorisées (selon déploiement ModX).		

Cahier des recettes – Livraison application dirCops

Application : dirCops
Projet : Suivi facturation / échéancier / paiement (EMGI)
Version : 1.0
Date : 2026-02-09

Usage : Cocher chaque ligne après test de recette. Indiquer O (OK) / N (Non conforme) / NA (non applicable) si besoin.

---

1. Gestion clients (FR1, FR2)

#	Critère de recette	O/N
1.1	Depuis l'accueil, la liste des clients s'affiche.	
1.2	La recherche par nom de client filtre bien la liste (saisie partielle).	
1.3	En cliquant sur un client, la page détail client s'ouvre avec le bon client.	
1.4	La page détail affiche : nom du client, encours, liste des factures, liste des paiements.	
1.5	Depuis l'accueil, on peut ajouter un client (formulaire « Ajouter un client ») ; le nouveau client apparaît dans la liste.	
1.6	Depuis l'accueil, on peut supprimer un client (icône/bouton supprimer) ; une confirmation est demandée ; la suppression supprime aussi factures, paiements et échéanciers associés (ou un message d'erreur si factures existantes selon règle métier).	

2. Factures (FR3, FR3b)

#	Critère de recette	O/N
2.1	Pour un client, la liste des factures affiche : n° facture, n° vente, date, type produit, cycle, quantité facturée, quantité livrée, montant HT, encours facture, état (Payé / En attente).	
2.2	Depuis la page détail client, le formulaire « Ajouter une facture (suivi) » est accessible et compréhensible.	
2.3	L'ajout d'une facture avec n° facture, date, montant HT obligatoires enregistre bien la facture et la fait apparaître dans la liste du client.	
2.4	L'ajout d'une facture avec type (alevins/larves), cycle, quantités et quantité livrée enregistre correctement ces champs et les affiche dans la liste.	
2.5	Une facture sans quantité livrée affiche « - » ou 0 selon le cas, sans erreur.	
2.6	On peut modifier une facture existante (bouton/icône Modifier) : le formulaire se remplit avec les données de la facture ; après envoi, les données sont mises à jour.	
2.7	On peut supprimer une facture (bouton/icône Supprimer) ; une confirmation est demandée ; la facture et ses paiements/échéancier associés sont supprimés.	

3. Écheancier de paiement (FR4b, FR5b)

#	Critère de recette	O/N
3.0.1	Depuis la page détail client, on peut ajouter un écheancier à une facture (facture associée, nombre d'échéances, date de début d'écheancier).	
3.0.2	Après ajout, l'écheancier apparaît pour la facture concernée (nombre d'échéances, date de début).	
3.0.3	Une facture ne peut avoir qu'un seul écheancier (ou pas d'écheancier).	
3.0.4	On peut supprimer un écheancier (bouton/icône Supprimer) ; une confirmation est demandée ; l'écheancier disparaît de la liste.	

4. Paiements (FR4, FR5)

#	Critère de recette	O/N
3.1	Sur la page client, la liste des paiements du client s'affiche (facture concernée, date, montant).	
3.2	Le formulaire « Enregistrer un paiement » propose uniquement les factures en attente (encours > 0).	
3.3	La saisie d'un montant et d'une date puis la soumission enregistre le paiement et met à jour l'encours de la facture.	
3.4	Après enregistrement d'un paiement, la page se recharge et les montants (encours client, encours par facture) sont cohérents.	
3.5	On peut supprimer un paiement (bouton/icône Supprimer) ; une confirmation est demandée ; le paiement disparaît et l'encours de la facture se met à jour.	

5. Calculs financiers (FR6, FR7, FR8, FR9)

#	Critère de recette	O/N
4.1	L'encours affiché par client correspond à la somme des montants HT des factures moins la somme des paiements de ce client.	
4.2	Le tableau de bord affiche un encours total cohérent avec la somme des encours par client.	
4.3	Les échéances à venir listent les échéances prévues (date ≥ aujourd'hui) : pour les factures avec écheancier, les dates et montants sont calculés à partir de l'écheancier (date début + N mois, montant par échéance) ; pour les factures sans écheancier, la date de facture et l'encours sont affichés. Colonnes : client, n° facture (éventuellement numéro d'échéance), date échéance, montant.	
4.4	Après ajout d'un paiement, l'encours du client et l'état des factures (Payé / En attente) se mettent à jour sans re-saisie.	

6. Reporting et tableaux de bord (FR10, FR10b, FR10c, FR11)

#	Critère de recette	O/N
5.1	Le tableau de bord affiche : encours par client (tri croissant), synthèse (encours total en k€, délai moyen de paiement en j), échéances à venir, liens export encours CSV et export ensemble des factures (PJ).	
5.1b	Les clients affichés sont des noms de clients (pas des lieux de départ ; EMGI = lieu de départ, pas un client).	
5.2	La page « Top clients » permet de choisir le type (Alevins / Larves) et optionnellement le cycle (24-25, 25-26).	
5.3	La page « Top clients » affiche pour le type et cycle choisis : client, quantités, % qté, CA HT, % CA, lien vers fiche client. Les totaux sont cohérents.	
5.4	La page « Quantités » affiche par client : quantité facturée, quantité livrée, lien vers fiche client.	
5.5	Le filtre par type (Alevins / Larves) sur la page « Quantités » restreint correctement les lignes affichées.	
5.6	L'export encours CSV (reporting?format=csv) télécharge un fichier avec encours par client (séparateur cohérent, caractères accentués corrects).	
5.7	L'export « ensemble des factures (PJ) » (reporting?format=factures_csv) télécharge un CSV calqué sur l'onglet Factures Excel (EMGI_FTI_Suivi des Ventes) : en-têtes EXO, Mois, CLIENT, PAYS, DEPART DEM, NEG., TYPE, CYCLE, produc, PAI, DATE FACTUR, N° DE VENTE, N° FACTURE, QTE FACT, QTE LIVREE, % GT, CA FACTURE HT, PU, Total payé, Encours ; colonnes renseignées selon les données appli, format français (dates jj/mm/aaaa, nombres virgule/espaces).	

7. Navigation et ergonomie

#	Critère de recette	O/N
6.1	Depuis chaque page, l'en-tête affiche le nom de l'application (dirCops) et les liens vers Clients, Tableau de bord, Top clients, Quantités.	
6.2	Les montants sont affichés en euros (format français). Les dates sont au format français.	
6.3	En cas d'erreur (ex. champ obligatoire manquant, API indisponible), un message d'erreur compréhensible s'affiche.	

8. Non fonctionnel (NFR1, performance)

#	Critère de recette	O/N
7.1	L'application est accessible uniquement aux personnes autorisées (contrôle d'accès selon environnement de déploiement).	
7.2	Le chargement initial de la liste des clients ou du tableau de bord reste acceptable (< 3 s dans des conditions normales).	
7.3	Les actions (recherche, clic, enregistrement) répondent en moins d'environ 1 s.	

---

9. Synthèse

Section	Nombre de critères	Validés (O)	Non conformes (N)	NA
1. Gestion clients	6			
2. Factures	7			
3. Écheancier	4			
4. Paiements	5			
5. Calculs financiers	4			
6. Reporting	9			
7. Navigation	3			
8. Non fonctionnel	3			
Total	**41**			

Livraison acceptée si : tous les critères bloquants sont à O (ou NA documenté). Les critères N doivent être listés et traités (correctif ou décision d'acceptation sous réserve).

Document généré à partir du PRD et des fonctionnalités livrées. À compléter par le client ou la maîtrise d'ouvrage lors de la recette.

Rapport d’aptitude à l’implémentation — dirCops

Date: 2026-02-09
Projet / application : dirCops (suivi facturation / échéancier / paiement — EMGI)

1) Inventaire des documents (Step 1)

Product brief : `product-brief-bmad-2026-02-09.md`
PRD : `prd.md`
Architecture : `architecture.md`
Listing des écrans/pages (spéc UI) : `listing-complet-ecrans-pages.md`
Logique paiements/écheanciers : `logique-paiements-echeanciers.md`
Cahier de recette : `cahier-recettes-livraison.md`
Décisions client (chiffrage) : `decisions-client-chiffrage.md`
Chiffrage (schéma + grille) : `chiffrage-schema-et-grille.md`
Schéma relationnel & tables : `schema-relationnel-et-tables-donnees.md`
Workflows métier : `schema-workflows.md`
Règles métier à clarifier : `général et regles-metier-a-clarifier.md`
Questions en suspens : `questions-en-suspens.md`
Epics & Stories : non trouvé
UX design (wireframes/maquettes) : non trouvé (le listing écran est une spéc UI textuelle, sans maquettes)

Évaluation réalisée sur l’ensemble des documents disponibles ci-dessus.

2) Analyse du PRD

2.1 Exigences fonctionnelles

Le PRD liste :

- **13 FR “principales”** (FR1 → FR13)
- **11 sous-FR** (CRUD + reporting) : FR2b–FR2c, FR3b–FR3d, FR4b, FR5b–FR5d, FR10b–FR10c

Total exigences fonctionnelles (PRD) : 24 items (13 + 11)

2.2 Exigences non fonctionnelles

NFR1: Accès restreint à l'équipe commerciale autorisée uniquement

Performance (web app) : Temps de chargement initial < 3 s, temps de réponse aux interactions < 1 s
Accessibilité : WCAG A (usage interne)
Navigateurs : Chrome / Firefox / Edge (dernières 2 versions). Safari non prioritaire (support best-effort si besoin).
Responsive : desktop prioritaire, support tablette basique

Total NFRs : 1 explicite (sécurité) + contraintes performance/accessibilité/navigateurs.

2.3 Contraintes / périmètre

- Contraintes : Budget ~20 k€, application interne, MPA, stack simple et maintenable.
- Hors périmètre MVP : création/édition de factures (gérée par Pennylane), trésorerie complète, parcours dédiés trésorerie/direction.
- Critères de succès mesurables : adoption par les commerciaux, fin de la dépendance à une seule personne, réduction des erreurs/incohérences.

2.4 Évaluation de complétude

Le PRD est globalement complet pour cadrer un MVP : exigences regroupées en 6 catégories, contraintes et hors-scope explicités, user journeys utiles. Les ambiguïtés restantes sont davantage des **décisions de règles métier** (exports, unicités, fréquences, etc.) que des trous majeurs de périmètre.

3) Couverture “exigences → architecture”

Epics & Stories : non trouvé. La validation de couverture via epics n’est donc pas possible.

En revanche, l’architecture fournit un mapping “exigences → composants” (modules + pages + endpoints). À noter : la logique métier paiements/écheanciers est également détaillée dans `logique-paiements-echeanciers.md`, ce qui renforce la readiness sur cette partie.

Couverture (niveau FR principales) : 13/13 couvertes (100 %) via les modules décrits.

Couverture (sous-FR CRUD/reporting du PRD) : couverte “par conception” (endpoints CRUD et écrans prévus), mais nécessite de **valider quelques règles** (ex. contraintes d’unicité, champs obligatoires, formats export) avant finalisation.

4) Alignement UX / écrans (Step 4)

Statut : pas de wireframes/maquettes. En revanche, un document de spécification UI textuelle existe : `listing-complet-ecrans-pages.md`.

Alignement : globalement cohérent avec le PRD et l’architecture (liste clients, fiche client, dashboard, top clients, quantités). Le listing ajoute une **page login** et des états UI (vides/erreur/chargement) utiles.

Authentification : une décision “Login MODX Livinweb (id/mot de passe)” est mentionnée dans `decisions-client-chiffrage.md`. Il reste à préciser l’implémentation (intégration MODX, protection des pages/API, logout, environnements).

5) Qualité des epics / stories (Step 5)

Document Epics & Stories : non trouvé → revue non applicable.

6) Synthèse, risques et recommandations

6.1 Statut global de readiness

READY (avec décisions à cadrer) — L’implémentation peut démarrer sur la base PRD + Architecture + specs UI + logique paiements/écheanciers. Les documents manquants (epics/stories, maquettes) ne

bloquent pas un POC, mais certaines **décisions** doivent être prises rapidement pour éviter des retours arrière.

6.2 Points critiques à trancher (avant “finir” le MVP)

1. **Exports CSV** : séparateur/encodage/format date + colonnes exactes. Le cahier de recette attend un export “factures PJ” **calqué sur l’onglet Excel** (en-têtes/format FR) → à valider très tôt avec un export réel.
2. **Hébergement / déploiement / sauvegardes** : à cadrer (impact planning et “definition of done”).
3. **Règles métier structurantes** (cf. `général et regles-metier-a-clarifier.md`) :
   - Unicité n° facture (globale vs par client)
   - Champs facture obligatoires (impact reporting)
   - Paiement > montant facture (autoriser vs bloquer)
   - Date de paiement future (autoriser vs bloquer)
   - Fréquence des échéances (mensuel uniquement vs paramétrable)
4. **Authentification MODX** : déjà actée “côté chiffrage”, mais à détailler techniquement (mécanisme exact, endpoints, protections, environnements).

6.3 Prochaines étapes recommandées

1. **Atelier de décision (≈ 1h)** pour trancher les items 🔴 / 🟠 de `général et regles-metier-a-clarifier.md` + valider le format CSV (test sur Excel Windows FR si cible).
2. **Démarrer l’implémentation socle** en suivant l’architecture (schéma SQL, config, endpoints CRUD, calcul encours).
3. **Formaliser un backlog minimal** (epics/stories) pour sécuriser le planning et la recette (même si “light”).
4. **Produire 4 wireframes rapides** (login, liste clients, fiche client, dashboard) pour sécuriser l’ergonomie et les états UI.

6.4 Note finale

Les exigences sont suffisamment cadrées pour démarrer. La couverture “exigences → architecture” est bonne (13/13 FR principales), et les sous-FR (CRUD/reporting) sont anticipées dans l’architecture + les specs d’écrans. Pour éviter les retours arrière, prioriser les décisions liées à l’authentification et aux exports.

Listing complet des écrans et pages

Application : dirCops
Projet : Suivi facturation / échéancier / paiement (EMGI)
Date : 2026-02-09
Usage : Liste exhaustive de toutes les pages/écrans pour le chiffrage et la conception

---

Vue d'ensemble

Type d'application : Multi-Page Application (MPA)
Architecture : HTML + JS vanilla + Tailwind CSS + API REST PHP
Nombre total de pages MVP complet : 7 pages principales + 1 page login + états variés

---

1. Authentification

1.1 Page de login

Fichier : `login.html`
Route : `/login.html` ou `/` (si non authentifié)
Objectif : Authentification utilisateur via MODX Livinweb

Contenu :

- Formulaire login (identifiant + mot de passe)
- Bouton "Se connecter"
- Message d'erreur si échec
- Redirection vers `/index.html` si succès

États :

- État nominal : formulaire vide
- État en cours : bouton disabled, loader
- État erreur : message "Identifiant ou mot de passe incorrect"
- État succès : redirection automatique

Interactions :

- POST `/api/auth.php` (login)
- Stockage session PHP
- Redirection si déjà authentifié

Wireframe/Maquette : Page simple centrée, logo EMGI, champs texte, bouton

---

2. Pages principales (après authentification)

2.1 Liste des clients (accueil)

Fichier : `index.html`
Route : `/` ou `/index.html`
Objectif : Liste et recherche des clients

Contenu :

- Barre de navigation (menu principal + bouton "Déconnexion")
- Champ de recherche (nom client)
- Liste des clients (nom, bouton "Voir détail")
- Bouton "Ajouter un client" (formulaire modal ou nouvelle page)
- Bouton "Supprimer" (confirmation)

États :

- État nominal : liste de tous les clients
- État recherche : liste filtrée
- État vide : message "Aucun client trouvé" + bouton "Ajouter un client"
- État chargement : skeleton ou loader
- État erreur : message "Erreur de chargement" + bouton "Réessayer"

Interactions :

- GET `/api/clients.php` (liste)
- GET `/api/clients.php?q=nom` (recherche)
- POST `/api/clients.php` (ajout client)
- DELETE `/api/clients.php?id=` (suppression)
- Clic client → navigation vers `/client.html?id=`

Formulaire "Ajouter un client" (modal ou section) :

- Champ : Nom du client (obligatoire)
- Bouton "Enregistrer" + "Annuler"
- Validation : nom non vide

2.2 Fiche client (détail)

Fichier : `client.html`
Route : `/client.html?id={client_id}`
Objectif : Vue complète d'un client (factures, paiements, écheanciers, encours)

Contenu (page longue, plusieurs sections) :

En-tête client

- Nom du client
- Encours total (calculé dynamique)
- Bouton "Retour à la liste"

Section "Factures"

- Tableau des factures :
  - Colonnes : N° facture, N° vente, Date, Type produit, Cycle, Quantités, Qté livrée, Montant HT, Encours, État (Payé/En attente)
  - Boutons par ligne : "Modifier", "Supprimer"
- Bouton "Ajouter une facture" → formulaire

Formulaire "Ajouter/Modifier une facture" :

- Champs :
  - N° facture (obligatoire)
  - N° vente (optionnel)
  - Date facture (obligatoire, date picker)
  - Type produit (select : alevins, larves, vide)
  - Cycle (texte libre, ex: 24-25, 25-26)
  - Quantités facturées (nombre décimal)
  - Quantité livrée (nombre décimal)
  - Montant HT (obligatoire, nombre décimal, €)
- Boutons : "Enregistrer", "Annuler"
- Validation : champs obligatoires, montant > 0

Section "Paiements"

- Tableau des paiements :
  - Colonnes : Date paiement, Montant, Facture associée, Bouton "Supprimer"
- Bouton "Enregistrer un paiement" → formulaire

Formulaire "Enregistrer un paiement" :

- Champs :
  - Facture (select : liste des factures du client avec n° et encours)
  - Montant (obligatoire, nombre décimal, €)
  - Date paiement (obligatoire, date picker)
- Boutons : "Enregistrer", "Annuler"
- Validation : facture sélectionnée, montant > 0, date valide

Section "Écheanciers"

- Tableau des écheanciers :
  - Colonnes : Facture associée, Nb échéances, Date début, Montant par échéance, Bouton "Supprimer"
  - (Optionnel) Détail échéances (dates calculées + montants)
- Bouton "Ajouter un écheancier" → formulaire

Formulaire "Ajouter un écheancier" :

- Champs :
  - Facture (select : factures avec encours > 0 et sans écheancier)
  - Nombre d'échéances (obligatoire, entier > 0)
  - Date de début (obligatoire, date picker)
- Montant par échéance : calculé et affiché (encours / nb échéances)
- Boutons : "Enregistrer", "Annuler"
- Validation : facture éligible, nb échéances > 0, date valide

États :

- État nominal : sections remplies avec données
- État vide (aucune facture) : message "Aucune facture pour ce client" + bouton "Ajouter une facture"
- État vide (aucun paiement) : message "Aucun paiement enregistré"
- État vide (aucun écheancier) : message "Aucun écheancier"
- État chargement : skeleton
- État erreur : message d'erreur + bouton "Réessayer"
- État après action (ajout/modification/suppression) : rechargement de la page ou mise à jour dynamique + message succès

Interactions :

- GET `/api/clients.php?id=` (détail client)
- GET `/api/factures.php?client_id=` (factures + encours)
- GET `/api/paiements.php?client_id=` (paiements)
- GET `/api/echeanciers.php?client_id=` (écheanciers)
- GET `/api/calculs.php?client_id=` (encours client)
- POST/PUT/DELETE `/api/factures.php`
- POST/DELETE `/api/paiements.php`
- POST/DELETE `/api/echeanciers.php`

2.3 Tableau de bord

Fichier : `dashboard.html`
Route : `/dashboard.html`

Objectif : Vue synthétique des encours et échéances

Contenu :

Indicateurs en haut (cards)

- Encours total (k€)
- Délai moyen de paiement (jours) — optionnel MVP
- Nombre de factures en attente

Tableau "Encours par client"

- Colonnes : Client, Encours (€), (Optionnel : Accord, J out, Situation)
- Tri : par encours croissant (ou boutons tri)
- Filtres : (optionnel) afficher uniquement encours > 0

Tableau "Échéances à venir"

- Colonnes : Client, N° facture, Date échéance, Montant échéance
- Filtre : date ≥ aujourd'hui
- Tri : par date croissante

Boutons d'export

- "Exporter encours CSV" → téléchargement `encours-clients-YYYY-MM-DD.csv`
- "Exporter l'ensemble des factures (PJ)" → téléchargement `suivi-factures-YYYY-MM-DD.csv`

États :

- État nominal : données affichées
- État vide (aucun encours) : message "Aucun encours en attente"
- État vide (aucune échéance) : message "Aucune échéance à venir"
- État chargement : skeleton
- État erreur : message d'erreur

Interactions :

- GET `/api/reporting.php` (données tableau de bord)
- GET `/api/reporting.php?format=csv` (export encours)
- GET `/api/reporting.php?format=factures_csv` (export factures PJ)

2.4 Top clients

Fichier : `top-clients.html`
Route : `/top-clients.html`
Objectif : Classement des clients par type de produit et cycle

Contenu :

Filtres en haut

- Type de produit (select : Tous, Alevins, Larves)
- Cycle (select : Tous, 24-25, 25-26, etc.)
- Bouton "Filtrer" ou filtrage automatique

Tableau "Top clients"

- Colonnes : Client, Type, Cycle, Quantités facturées, CA HT (€), % Quantités, % CA
- Tri : par quantités ou CA décroissant
- (Optionnel) Limite Top 10/15 ou affichage complet selon décision client

États :

- État nominal : tableau avec données
- État après filtre : tableau mis à jour
- État vide (aucun résultat) : message "Aucun client pour ces critères"
- État chargement : skeleton
- État erreur : message d'erreur

Interactions :

- GET `/api/top-clients.php?type=&cycle=` (données filtrées)

2.5 Quantités par client

Fichier : `quantites-clients.html`
Route : `/quantites-clients.html`
Objectif : Synthèse des quantités facturées et livrées par client

Contenu :

Filtre en haut

- Type de produit (select : Tous, Alevins, Larves)
- Bouton "Filtrer" ou filtrage automatique

Tableau "Quantités par client"

- Colonnes : Client, Type (si filtre "Tous"), Quantités facturées, Quantités livrées, Écart (qté facturée - qté livrée), % Livré
- Tri : par client ou quantités

États :

- État nominal : tableau avec données
- État après filtre : tableau mis à jour
- État vide (aucun résultat) : message "Aucune donnée pour ces critères"
- État chargement : skeleton

- État erreur : message d'erreur

Interactions :

- GET `/api/quantites-clients.php?type=` (données filtrées)

---

3. Navigation / Menu principal

Présent sur toutes les pages après authentification :

Menu horizontal (barre de navigation) :

- Logo / Titre : "dirCops"
- Liens de navigation :
  - "Clients" → `/index.html`
  - "Tableau de bord" → `/dashboard.html`
  - "Top clients" → `/top-clients.html`
  - "Quantités" → `/quantites-clients.html`
- Bouton "Déconnexion" (à droite) → POST `/api/auth.php?logout=1` → redirection `/login.html`

---

4. Pages d'erreur / états spéciaux

4.1 Page 404 (non trouvé)

Fichier : `404.html` (optionnel, ou gestion navigateur)
Contenu : Message "Page non trouvée" + lien "Retour à l'accueil"

4.2 Page 403 (non autorisé)

Cas : Tentative d'accès à une page sans authentification
Comportement : Redirection automatique vers `/login.html`

4.3 Page erreur serveur (500)

Comportement : Message "Erreur serveur, veuillez réessayer plus tard" + bouton "Retour"

---

5. Modals / Composants réutilisables

5.1 Modal de confirmation (suppression)

Déclencheurs : Suppression client, facture, paiement, écheancier

Contenu :

- Titre : "Confirmer la suppression"
- Message : détail de ce qui sera supprimé (+ cascade si applicable)

    - Ex : "Supprimer le client [Nom] ? Cela supprimera aussi toutes ses factures, paiements et écheanciers."
- Boutons : "Confirmer" (rouge/danger), "Annuler"

5.2 Toast / Message de succès

Déclencheurs : Action réussie (ajout, modification, suppression)

Contenu :

- Message court : "Client ajouté avec succès", "Paiement enregistré", etc.
- Fermeture automatique après 3-5 secondes ou bouton "×"
- Couleur : vert (succès), rouge (erreur), orange (avertissement)

5.3 Loader / Skeleton

Déclencheurs : Chargement de données (GET API)

Contenu :

- Spinner ou skeleton screen (simulation structure du contenu)
- Texte : "Chargement..." (optionnel)

---

6. Récapitulatif des pages

#	Page	Fichier	Route	Authentification requise
1	Login	`login.html`	`/login.html`	Non
2	Liste clients	`index.html`	`/` ou `/index.html`	Oui
3	Fiche client	`client.html`	`/client.html?id=`	Oui
4	Tableau de bord	`dashboard.html`	`/dashboard.html`	Oui
5	Top clients	`top-clients.html`	`/top-clients.html`	Oui
6	Quantités clients	`quantites-clients.html`	`/quantites-clients.html`	Oui
7	Page 404	`404.html`	(erreur)	Non

Total MVP complet : 7 pages + navigation + modals

---

7. Variantes par package (selon options-reduction-perimetre-chiffrage.md)

Package A – MVP minimal

Pages incluses :

- Login (1)
- Liste clients (2)

- Fiche client (3) — version simplifiée : factures (lecture seule), paiements (ajout uniquement), pas d'écheanciers
- Tableau de bord (4) — version simplifiée : encours par client, export CSV encours uniquement

Pages exclues :

- Top clients (5)
- Quantités clients (6)

Total : 4 pages + navigation

Package B – MVP sans écheancier ni pages reporting

Pages incluses :

- Login (1)
- Liste clients (2) — avec ajout/suppression client
- Fiche client (3) — factures (ajout/modification/suppression), paiements (ajout/suppression), pas d'écheanciers
- Tableau de bord (4) — complet avec export

Pages exclues :

- Top clients (5)
- Quantités clients (6)

Total : 4 pages + navigation

Package C – MVP complet

Pages incluses : Toutes (1-6) + navigation + modals

Total : 7 pages + navigation

---

8. Wireframes / Maquettes recommandés

Priorité haute (à créer pour validation client) :

1. Page login
2. Liste clients (états : vide, rempli, recherche)
3. Fiche client (sections factures, paiements, écheanciers)
4. Tableau de bord (encours, échéances)

Priorité moyenne : 5. Top clients (filtres + tableau) 6. Quantités clients

Priorité basse : 7. Pages d'erreur (404, 500) 8. Modals (confirmation, succès)

Document de référence pour la conception et le développement. À compléter avec wireframes/maquettes si disponibles.

Questions en suspens

Application : dirCops
Projet : Suivi facturation / échéancier / paiement (EMGI)
Date : 2026-02-09

Récapitulatif par priorité

Priorité	Thème	Question courte
Élevée	Pennylane	Import/sync Pennylane prévu ou saisie manuelle uniquement ?
Moyenne	Export	Format et contenu des exports (CSV/JSON, champs) ?
Moyenne	Couverture / GTT	Indicateurs « type Excel » et % GTT dans le MVP ou après ?
Moyenne	Top clients	Limite (Top 10/15) ou pas de limite ?
Basse	Accessibilité	Niveau WCAG (A vs AA) ?
Basse	Écheancier	Échéances toujours mensuelles ? Suivi par échéance requis ?
— (résolu)	Rôles	Un seul rôle (commercial/admin) en MVP.
— (résolu)	Safari	Pas requis, pas supporté (option).
— (résolu)	Correction données	Modification / suppression : implémenté (facture, paiement, écheancier, client).

Document à mettre à jour au fur et à mesure des réponses du prospect ou des décisions techniques.