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`