

UBCG Research — Concept technique pour le développeur

Oct 14, 2026 · @ubcg

1. Produit

UBCG Research est une marketplace exploitée par UBCG SARL (Marrakech) : on y achète des études et on y engage des chercheurs vérifiés, à l'heure ou au mois. UBCG encaisse tous les paiements, garde l'argent des missions sur un séquestre et reverse chaque mois aux chercheurs, commission déduite.

Acteur	Ce qu'il fait	Ce qu'il paie / reçoit
Visiteur	Parcourt le catalogue, les chercheurs, les missions ; achète une étude sans compte	Paie l'étude (lien signé 30 jours par e-mail)
Utilisateur	Compte gratuit sans transaction (téléchargements gratuits, liste)	—
Client	Compte avec au moins un achat, une mission publiée ou un chercheur engagé	Paie études, dépôts de séquestre, Premium client, options de visibilité
Chercheur	Publie des études, répond aux missions, déclare ses heures	Reçoit un reversement mensuel (brut – commission – retenue) ; peut payer Premium chercheur et visibilité
Équipe UBCG (admin)	Valide chercheurs et études, arbitre les litiges, pilote finance, offres, conformité	—

Un seul compte peut être client **et** chercheur : « Mon espace » bascule entre les deux modes (`session('space_mode')`).

Sources de revenus UBCG : commission sur les ventes d'études (défaut 30 %), commission sur les contrats (défaut 30 %), abonnements Premium (client 199 MAD/mois, chercheur 95 MAD/mois, –50 % étudiants), options de visibilité (positions #1–#3, broadcast), études UBCG vendues à 100 %.

Périmètre : Maroc et Union européenne, interface FR / EN / AR (arabe en RTL), montants en MAD, TVA 20 % par défaut.

2. Stack et architecture

Laravel 12 (PHP 8.2+), MariaDB 10.6+ en production, Blade côté serveur, sans build front-end. Chaque requête suit le chemin Route → Controller par fonctionnalité → Eloquent → Blade ; JSON uniquement pour les actions AJAX.

Couche	Dossier	Contenu
Routes	<code>routes/web.php</code> , <code>routes/admin.php</code> , <code>routes/console.php</code>	Site + Mon espace ; back-office /admin ; tâches planifiées
Middleware	<code>app/Http/Middleware</code>	<code>SetLocale</code> (langue), <code>SecurityHeaders</code> (CSP nonce, HSTS), <code>EnsurePermission</code> (alias perm), <code>EnsureResearcher</code> (alias researcher)
Contrôleurs	<code>app/Http/Controllers</code> (+ <code>Auth/</code> , <code>Space/</code> , <code>Admin/</code>)	Un contrôleur par fonctionnalité : Home, Study, Search, Researcher, Mission, Proposal, Contract, Checkout, File, Favorite, Review, Message, Space, Account, Privacy, Premium, Visibility, BecomeResearcher, Ai, Legal, Report, Consent
Validation	<code>app/Http/Requests</code>	<code>StudyForm</code> , <code>MissionRequestForm</code> ; ailleurs <code>\$request->validate()</code>
Modèles	<code>app/Models</code>	31 modèles Eloquent (§ 4)
Services	<code>app/Services</code>	Uniquement la logique métier réelle : <code>EscrowService</code> , <code>PayoutService</code> , <code>SearchService</code> , <code>ComplianceService</code> , <code>Ai/Translator</code> , <code>Payments/PaymentGateway</code>
Support	<code>app/Support</code>	<code>Pricing</code> (offres en %), <code>Ranking</code> (classement), <code>Levels</code> , <code>Money</code> , <code>Countries</code> , <code>Icons</code>
Notifications	<code>app/Notifications</code>	<code>PlatformNotice</code> (base + mail), <code>AnnouncementMail</code> (désinscription signée), <code>AdminLoginCode</code> (2FA)

Couche	Dossier	Contenu
Vues	<code>resources/views</code>	<code>layouts/</code> (app, space, legal, admin), <code>partials/</code> , pages publiques, <code>space/</code> , <code>admin/</code> , <code>legal/</code> (+ <code>en/</code> , <code>ar/</code>)
Traductions	<code>lang/</code>	<code>en.json</code> , <code>ar.json</code> (clés = texte français), <code>fr/ar/en/validation.php</code>
Front	<code>public/css</code> , <code>public/js</code>	<code>site.css</code> , <code>admin.css</code> , <code>app.css</code> ; <code>app.js</code> et <code>admin.js</code> en amélioration progressive

Règles d'architecture à respecter :

- Pas de service « Boot » ni de données globales injectées en JavaScript : chaque contrôleur charge ce dont sa page a besoin.
- Un service n'existe que s'il porte une règle métier partagée (séquestre, reversements, conformité). Le reste reste dans le contrôleur.
- Toute action fonctionne sans JavaScript (formulaire classique) ; le JS ajoute le glisser-déposer, les feuilles mobiles et l'AJAX.
- Réglages modifiables par l'admin dans la table `settings` via `Setting::get()` / `Setting::put()` (cache `settings.all`).
- Paiement derrière l'interface `PaymentGateway` : `FakeGateway` en démo, bloqué en production ; le prestataire agréé se branche ici.

3. Rôles et droits

Deux familles de comptes partagent la table `users` : les comptes de la plateforme (pas de `admin_role_id`) et l'équipe UBCG (`admin_role_id` renseigné). La catégorie d'un compte plateforme est calculée, jamais stockée (`User::accountCategory()`).

Rôle	Condition technique	Accès
Visiteur	Non connecté	Pages publiques, achat invité, signalement, pages légales
Utilisateur	Connecté, e-mail vérifié (<code>verified</code>), sans transaction	Mon espace : liste, messages, compte, confidentialité ; téléchargements gratuits

Rôle	Condition technique	Accès
Client	Au moins une commande, une mission publiée ou un contrat	+ achats, missions publiées, comparaison des propositions, séquestre, validation des heures, avis, Premium client
Chercheur	A un <code>researcher_profiles</code> (middleware <code>researcher</code>)	+ <code>/espace/chercheur</code> : profil, documents, études, propositions, heures, revenus ; proposer et publier exige <code>status = active</code>
Équipe UBCG	<code>admin_role_id</code> + code 2FA validé en session (<code>admin_2fa_passed</code>)	<code>/admin</code> , chaque page protégée par <code>perm:<droit></code>
Propriétaire	<code>is_owner = true</code> (créé par <code>OwnerSeeder</code>)	Rôle Admin, ne peut être ni retiré ni rétrogradé

Les 13 droits admin (`AdminRole::PERMISSIONS`) et les rôles créés par le seeder ; la matrice est modifiable dans Admin › Équipe & rôles, sauf pour Admin qui garde tout.

Droit	Pages	Admin	Sales Manager	Modérateur	Comptable
<code>overview</code>	Vue d'ensemble	✓	✓		✓
<code>users</code>	Utilisateurs	✓	✓	✓	
<code>researchers</code>	Chercheurs & études soumises, documents	✓	✓	✓	
<code>contracts</code>	Contrats & litiges, missions	✓	✓	✓	
<code>visibility</code>	Visibilité & options	✓	✓		
<code>offers</code>	Offres & messages	✓	✓		
<code>finance</code>	Facturation & reversements	✓	✓		✓
<code>commission</code>	Commission & fiscalité	✓			
<code>catalog</code>	Études UBCG, catégories, missions UBCG	✓	✓		
<code>modules</code>	Modules & langues, prix Premium, niveaux	✓			
<code>team</code>	Équipe & rôles	✓			
<code>audit</code>	Journal d'activité	✓	✓		
<code>compliance</code>	Conformité (DSA, RGPD, IA)	✓		✓	

Côté code : `User::hasPermission($p)` , une Gate par droit (Admin passe toujours via `Gate::before`), middleware `perm:` sur les groupes de `routes/admin.php` . Un non-membre qui ouvre `/admin` est renvoyé vers la connexion admin.

4. Modèle de données

43 tables, une migration par table

(`database/migrations/2026_10_14_0000NN_create_<table>_table.php`), numérotées dans l'ordre des clés étrangères : une table n'est créée qu'après celles qu'elle référence.

Testé sur MariaDB 10.11 (InnoDB, `utf8mb4_unicode_ci`) : 54 clés étrangères créées, seed complet. Les colonnes `created_at` / `updated_at` sont omises ci-dessous.

Comptes et organisation

N°	Table	Rôle	Colonnes clés
01	<code>admin_roles</code>	Rôles de l'équipe UBCG	<code>key</code> , <code>label</code> , <code>permissions</code> (JSON), <code>is_system</code>
02	<code>users</code>	Tous les comptes (plateforme + équipe)	<code>email</code> , <code>password</code> , <code>locale</code> , <code>organization</code> , <code>account_kind</code> , <code>client_plan</code> , <code>admin_role_id</code> , <code>is_owner</code> , <code>marketing_opt_in</code> , <code>terms_version</code> , <code>deleted_at</code>
03-10	<code>password_reset_tokens</code> , <code>sessions</code> , <code>cache</code> , <code>cache_locks</code> , <code>jobs</code> , <code>job_batches</code> , <code>failed_jobs</code> , <code>notifications</code>	Tables techniques Laravel ; <code>notifications</code> = boîte « Messages »	<code>notifications.data</code> (JSON), <code>read_at</code>
11	<code>settings</code>	Réglages éditables par l'admin	<code>key</code> , <code>value</code> (JSON)

Catalogue et chercheurs

N°	Table	Rôle	Colonnes clés
12	<code>categories</code>	Domaines (Politique, Marché, Culture, Gouvernance, Société, Économie)	<code>slug</code> , <code>name</code> (JSON FR/EN/AR), <code>sort</code>
13	<code>industries</code>	Secteurs clients	<code>slug</code> , <code>name</code> (JSON)
14	<code>researcher_profiles</code>	Profil chercheur d'un utilisateur (1-1)	<code>user_id</code> , <code>slug</code> , <code>kind</code> , <code>hourly_rate</code> , <code>monthly_rate</code> , <code>tax_status</code> , <code>tax_id</code> ¹ , <code>payout_account</code> ¹ , <code>dac7_tin</code> ¹ , <code>plan</code> , <code>commission_override</code> , <code>status</code> , <code>rating</code> , <code>reviews_count</code>
15	<code>category_researcher_profile</code>	Domaines d'un chercheur	pivot
16	<code>industry_researcher_profile</code>	Secteurs d'un chercheur	pivot
17	<code>researcher_documents</code>	Pièces justificatives (disque privé)	<code>kind</code> , <code>path</code> , <code>verified_at</code>
18	<code>studies</code>	Études (UBCG, chercheur, partenaire)	<code>source</code> , <code>researcher_profile_id</code> , <code>category_id</code> , <code>slug</code> , <code>title</code> , <code>summary</code> , <code>toc</code> , <code>translations</code> (JSON), <code>price</code> , <code>external_url</code> , <code>status</code> , <code>published_at</code> , <code>deleted_at</code>
19	<code>industry_study</code>	Secteurs visés par une étude	pivot
20	<code>study_licenses</code>	Prix par licence	<code>type</code> (solo, team, org), <code>price</code> , <code>active</code>

N°	Table	Rôle	Colonnes clés
21	<code>study_files</code>	PDF et annexes	<code>kind</code> , <code>language</code> , <code>path</code>

Ventes

N°	Table	Rôle	Colonnes clés
22	<code>offers</code>	Réductions en % (Premium, visibilité)	<code>percent</code> , <code>targets</code> (JSON), <code>starts_on</code> , <code>ends_on</code> , <code>active</code>
23	<code>orders</code>	Achat d'étude, compte ou invité	<code>number</code> , <code>user_id</code> ou <code>guest_email</code> , <code>study_id</code> , <code>study_license_id</code> , <code>amount</code> , <code>vat</code> , <code>commission_rate</code> , <code>status</code> , <code>paid_at</code>
24	<code>downloads</code>	Téléchargements d'études gratuites	<code>user_id</code> , <code>study_id</code>
25	<code>favorites</code>	« Ma liste »	<code>user_id</code> , <code>favoritable</code> (étude ou chercheur)

Missions, contrats et argent

N°	Table	Rôle	Colonnes clés
26	<code>mission_requests</code>	Besoin publié par un client (ou par UBCG)	<code>user_id</code> , <code>is_ubcg</code> , <code>category_id</code> , <code>title</code> , <code>description</code> , <code>type</code> (hourly, monthly), <code>budget</code> , <code>deadline</code> , <code>status</code> , <code>deleted_at</code>
27	<code>mission_request_files</code>	Pièces jointes de la mission	<code>path</code> , <code>original_name</code>
28	<code>contracts</code>	Engagement client ↔ chercheur	<code>client_id</code> , <code>researcher_profile_id</code> , <code>mission_request_id</code> , <code>type</code> , <code>rate</code> , <code>weekly_limit</code> , <code>months</code> , <code>start_date</code> , <code>commission_rate</code> , <code>status</code> , <code>dispute_reason</code>

N°	Table	Rôle	Colonnes clés
29	proposals	Réponse d'un chercheur à une mission	mission_request_id, researcher_profile_id, client_id, rate, message, status, contract_id
30	escrow_deposits	Versements sur le séquestre	contract_id, amount, method, status, payment_reference
31	time_logs	Heures déclarées	contract_id, worked_on, hours, status, decided_at
32	statements	Relevé mensuel d'un chercheur	month (AAAA-MM), gross, commission, withholding, carried_in, payout, status ; unique (chercheur, mois)
33	reviews	Avis client sur un chercheur	rating, comment, contract_id OU order_id, status

Visibilité et abonnements

N°	Table	Rôle	Colonnes clés
34	placements	Position sponsorisée #1-#3	kind, position, category_id, city, placeable (chercheur ou mission), starts_on, ends_on, price
35	subscriptions	Abonnements Premium	plan, student, price, offer_id, starts_on, ends_on, status
36	boost_orders	Ventes d'options (broadcast, positions, Premium)	kind, label, amount, offer_id
37	announcements	Messages de l'admin à une audience	audience, channels (JSON), offer_id, cta, reach_platform, reach_email
38	announcement_reads	Lu / non lu par compte	pivot + read_at

Conformité et traçabilité

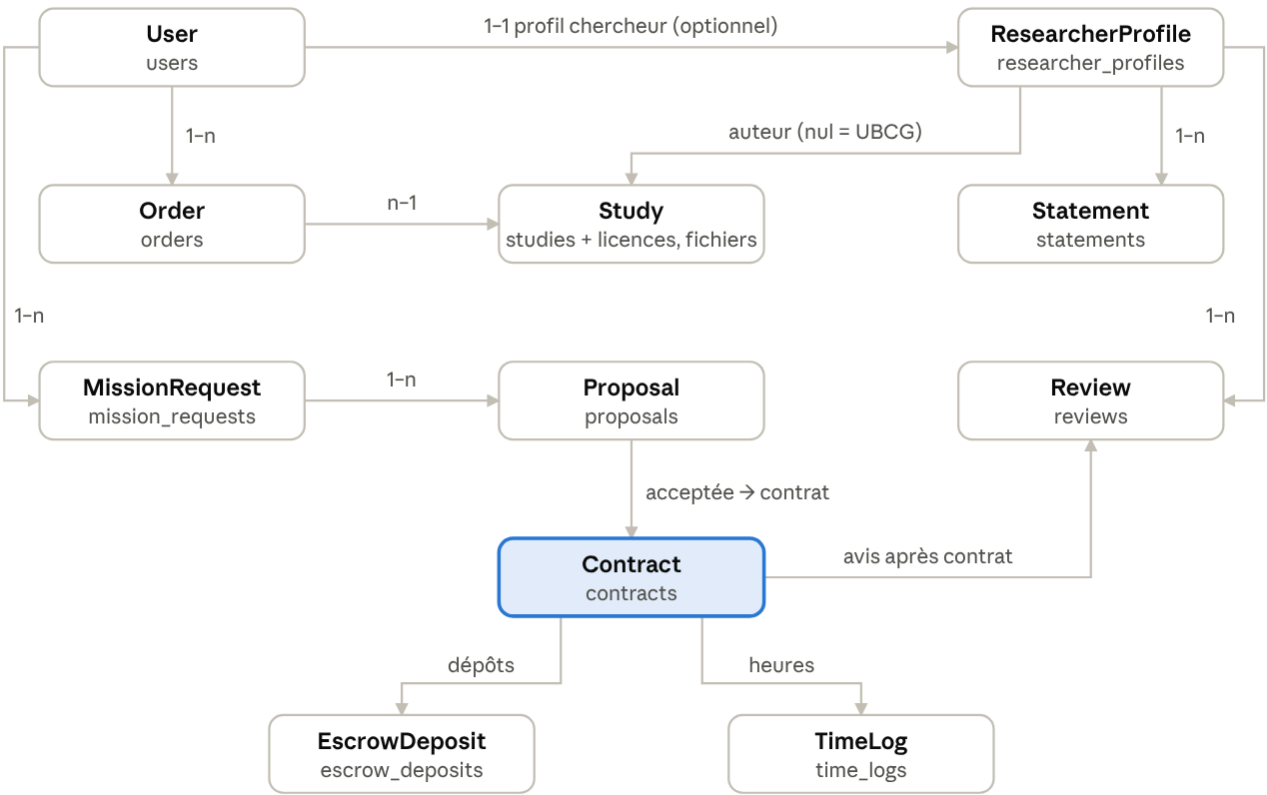
N°	Table	Rôle	Colonnes clés
39	consent_records	Preuve des consentements (RGPD art. 7)	purpose, version, granted, subject_hash, ip_hash
40	data_requests	Demandes des personnes (accès, effacement...)	type, status, due_at (1 mois)
41	content_reports	Signalements DSA	reason, reportable (polymorphe), decision, decided_by, status
42	ai_usage_logs	Usage de l'IA, sans contenu	feature, provider, input_chars
43	audit_logs	Journal des actions sensibles	action, subject (polymorphe), meta (JSON), ip_hash

¹ Colonnes chiffrées par Laravel (encrypted cast) : illisibles en base sans APP_KEY . Ne jamais changer APP_KEY en production sans re-chiffrer.

5. Relations

Tout part de users : un utilisateur peut avoir un profil chercheur (1-1), passer des commandes, publier des missions et signer des contrats comme client. Le profil chercheur porte études, propositions, contrats, relevés et avis. Le schéma ci-dessous montre le cœur ; le tableau donne toutes les relations Eloquent.

Le contrat relie client et chercheur et porte l'argent des missions



modèle de données · cœur du produit, 11 tables

Les heures validées (`time_logs`) et les ventes d'études (`orders`) alimentent le relevé mensuel (`statements`) calculé par `PayoutService` ; il n'y a pas de clé étrangère directe entre eux.

Modèle	Appartient à (belongsTo)	Possède (hasMany / hasOne)	Plusieurs-à-plusieurs / polymorphe
User	adminRole	researcherProfile (1-1), orders , downloads , favorites , missionRequests , clientContracts , subscriptions , reviews , consentRecords	notifications (Notifiable)

Modèle	Appartient à (belongsTo)	Possède (hasMany / hasOne)	Plusieurs-à-plusieurs / polymorphe
ResearcherProfile	user	documents , studies , contracts , proposals , statements , reviews	domains (categories), industries ; placements (morph)
Study	researcher (nul = UBCG ou partenaire), category	licenses , files , orders	industries ; cible de favorites et content_reports
Order	user (nul = invité), study	—	—
MissionRequest	client (nul + is_ubcg = mission UBCG), category , industry	proposals , files	placements (morph)
Proposal	missionRequest , researcher , client , contract (une fois acceptée)	—	—
Contract	researcher , client , missionRequest (nul = engagement direct)	deposits , timeLogs , proposals	—
EscrowDeposit , TimeLog	contract	—	—
Statement	researcher	—	—
Review	researcher , user (+ contract_id ou order_id)	—	met à jour rating et reviews_count du profil

Modèle	Appartient à (belongsTo)	Possède (hasMany / hasOne)	Plusieurs-à-plusieurs / polymorphe
Placement	—	—	placeable = ResearcherProfile ou MissionRequest
Favorite	user	—	favoritable = Study ou ResearcherProfile
Announcement	offer	—	readers (users via announcement_reads)
ContentReport	reporter	—	reportable = Study, ResearcherProfile, MissionRequest ou Review
AuditLog	user	—	subject (n'importe quel modèle)
AdminRole	—	users	—

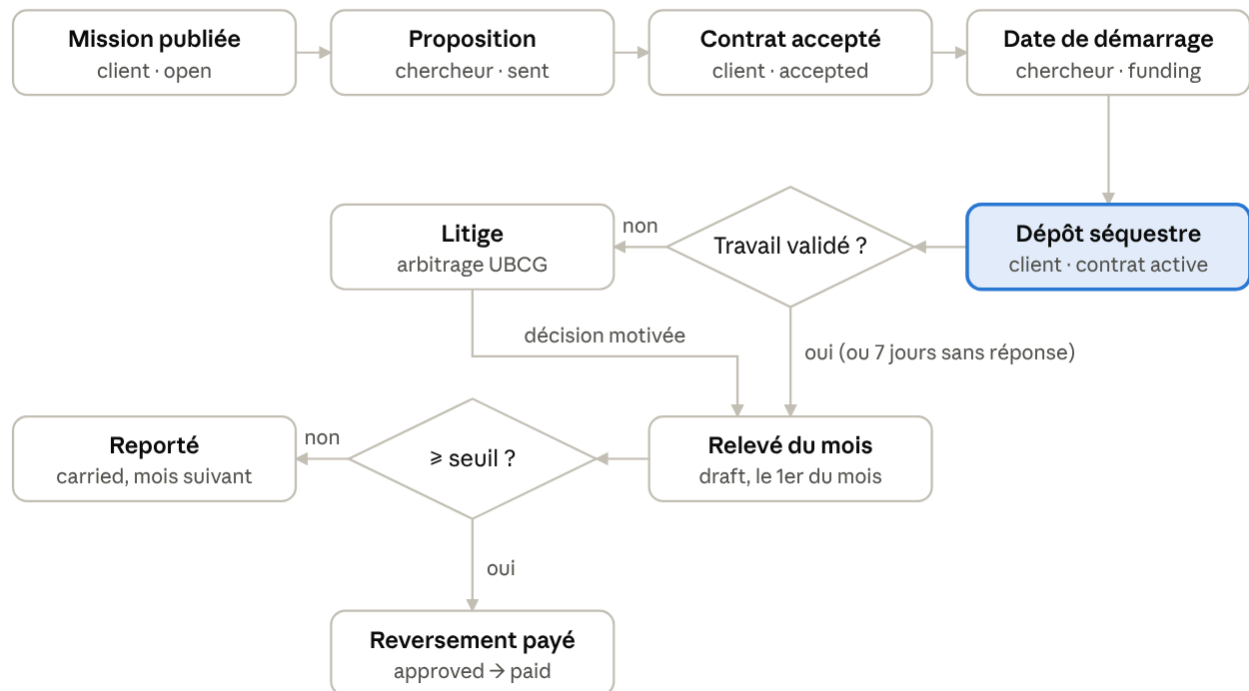
Règles de suppression (clés étrangères) :

- **RESTRICT** — une catégorie utilisée par une étude ou une mission, une étude déjà vendue (`orders.study_id`), un profil lié à un contrat : la suppression est refusée par la base.
- **CASCADE** — les enfants qui n'ont pas de sens seuls : fichiers, licences, documents, dépôts, heures, propositions, pivots.
- **SET NULL** — l'historique reste quand l'auteur disparaît : commandes, contrats, journaux, signalements, offres.
- `users`, `studies` et `mission_requests` utilisent `SoftDeletes` : rien n'est supprimé physiquement. L'effacement RGPD anonymise le compte puis le passe en `deleted_at` ; les factures restent (10 ans). Ne jamais faire de `forceDelete()` sur un utilisateur ou un profil chercheur : les relevés (`statements`) partiraient en cascade.

6. Flux métier

Le flux central est celui des missions : l'argent du client entre sur le séquestre avant le travail et n'en sort qu'une fois le travail validé, via le relevé mensuel. Chaque étape cite la route et le code qui l'exécute.

L'argent du client reste sur le séquestre jusqu'au travail validé



flux mission · de la publication au reversement

Un litige ne bloque que les heures contestées ; après arbitrage, elles entrent dans le relevé du mois comme les autres.

6.1 Mission → contrat → séquestre → reversement

- 1. Le client publie une mission** — `POST /missions` (`MissionController@store` , `MissionRequestForm`) → `mission_requests.status = open` . Option : visibilité payante (`POST /missions/{id}/visibilite` → `placements + boost_orders`).
- 2. Les chercheurs proposent** — `POST /espace/chercheur/missions/{id}/proposer` (`ProposalController@store`), réservé aux profils `active` → `proposals.status = sent` , notification au client.
- 3. Le client compare et accepte** — `GET /missions/{id}/comparer` puis `POST /propositions/{id}/accepter` → `EscrowService::acceptProposal()` (transaction) : crée le `Contract` (`accepted` , taux de commission figé), mission → `awarded` , les autres propositions → `declined` . Variante : offre directe sans mission `POST /chercheurs/{slug}/engager` .
- 4. Le chercheur fixe la date de démarrage** — `POST /espace/chercheur/contrats/{id}/demarrer` → `EscrowService::start()` → contrat `funding` , notification au client.

5. Le client verse le dépôt — `POST /contrats/{id}/depot` →

`EscrowService::deposit()` via `PaymentGateway`. Dépôt demandé : 2 semaines d'heures plafonnées (horaire) ou 1 mois (mensuel). Paiement reçu → dépôt `held`, contrat `active` ; virement → dépôt `pending` jusqu'à réception.

6. Le travail est validé

- Horaire : le chercheur déclare ses heures (`POST /espace/chercheur/contrats/{id}/heures` → `time_logs.pending`) ; le client valide ou conteste (`POST /heures/{id}/decision`). Sans réponse en 7 jours : `ubcg:auto-approve` → `auto_approved` .
- Mensuel : chaque mois du contrat (mois de démarrage inclus) est dû tant que le contrat est `active` ou `completed` .

7. Litige — une contestation passe le contrat en `dispute` . L'admin arbitre (`POST /admin/contrats/{id}/arbitrage`) : heures validées, rejetées ou partagées 50/50, décision motivée envoyée aux deux parties, contrat → `active` .

8. Clôture du mois — le 1er à 03:00 `ubcg:close-month` (ou bouton « Clôturer » dans Finance) → `PayoutService::closeMonth()` crée un relevé `draft` par chercheur : ventes d'études payées + heures validées + mensualités, moins commission, moins retenue à la source, plus report des mois précédents. Les mensualités sont libérées du séquestre (`escrow_deposits.status = released` , `released_month`).

9. Reversement — dans Admin › Finance (liste ou kanban) : `draft` → `approved` → `processing` → `paid` . Un relevé sous le seuil (`payout.min` , défaut 1 000 MAD) passe automatiquement en `carried` et s'ajoute au mois suivant. Le fichier de virements CSV ne contient que les relevés `approved` au-dessus du seuil ; `paid` envoie l'avis de paiement au chercheur.

10. Avis — après un contrat ou un achat, le client note le chercheur (`POST /chercheurs/{slug}/avis`) ; la note n'est affichée qu'à partir de 5 avis (`reviews.min`).

6.2 Achat d'une étude

1. Fiche étude → choix de la licence (individuelle ×1, équipe ×1,6, organisation ×2,5) → `GET /etudes/{slug}/acheter` .
2. `POST /etudes/{slug}/acheter` (`CheckoutController@store`) : CGV acceptées + renonciation expresse au droit de rétractation (contenu numérique) obligatoires ; invité = e-mail + nom.
3. `PaymentGateway::charge()` → `orders` (`paid` ou `awaiting` pour virement / bon de commande), TVA calculée, `commission_rate` figé (100 % pour une étude UBCG).
4. Redirection vers un lien signé valable 30 jours (`checkout.done`) ; l'invité le reçoit aussi par e-mail. Les fichiers sont servis par `FileController` uniquement à l'acheteur ou via ce lien.

5. Étude gratuite : `POST /etudes/{slug}/obtenir` (compte requis) → `downloads`. Étude « lien » : redirection vers `external_url`.

6.3 Devenir chercheur et être validé

1. Inscription avec le rôle chercheur ou `POST /devenir-chercheur` depuis un compte existant → `researcher_profiles.status = incomplete`.
2. Le chercheur complète profil, domaines, coordonnées de reversement et pièces justificatives (liste selon son statut fiscal : `ResearcherProfile::requiredDocuments()`), puis `POST /espace/chercheur/soumettre` → `pending`.
3. L'admin vérifie (4 cases à cocher obligatoires) : `approve` → `active` (documents marqués vérifiés), `reject` → `rejected`, `info` → `incomplete` avec message. Un déplacement kanban de `pending` vers `active` ou `rejected` renvoie 409 et ouvre la fiche : la décision passe toujours par la vérification.
4. Seul un profil `active` apparaît publiquement, propose sur des missions et publie des études.

6.4 Publier une étude

1. Chercheur `active` : `POST /espace/chercheur/etudes` (`StudyForm`) → `studies.status = review`, licences et fichiers synchronisés.
2. L'admin publie ou refuse avec motif (3 cases à cocher) → `published` / `rejected` + notification.
3. Toute modification importante repasse l'étude en `review`. Retirer une étude = suppression douce : les acheteurs gardent leur accès.
4. Études UBCG et partenaires : créées par l'admin (Admin › Études, éditeur en popup), publiées directement.

6.5 Premium, visibilité et offres

- **Premium client** (`POST /espace/premium`) et **Premium chercheur** (`POST /espace/chercheur/premium`) → `subscriptions` + `boost_orders` ; prix depuis `settings`, réduction étudiant, offre en cours appliquée par `Pricing`.
- **Positions sponsorisées** chercheur (par domaine et ville, en mois) ou mission (en semaines) → `placements`. Classement public : positions sponsorisées → Premium → Basic, ordre aléatoire dans chaque groupe (`Ranking`), toujours marqué « Sponsorisé ».
- **Offres** : l'admin crée une réduction en % et sa période (`offers`), puis l'annonce à une audience (`announcements`) : boîte « Messages » pour tous, e-mail uniquement aux comptes vérifiés avec `marketing_opt_in`, lien de désinscription signé.

6.6 Conformité

- **Signalement DSA** : `POST /signalement` (motif, description, bonne foi, e-mail) → `content_reports.open` + accusé de réception. L'admin décide avec un exposé des motifs (min. 20 caractères), peut masquer le contenu ; le déclarant est notifié.
- **Droits RGPD / loi 09-08** (Mon espace › Confidentialité) : export JSON immédiat ; demandes d'accès, rectification, opposition, limitation → `data_requests` à traiter sous 1 mois dans Admin › Conformité ; suppression du compte avec mot de passe, refusée s'il reste un contrat `funding`, `active` ou `dispute`.
- **Consentements** : inscription (CGU, confidentialité, marketing), cookies (bandeau → `POST /consentement`), chaque choix enregistré avec sa version dans `consent_records`. Changer la version des CGU dans `config/legal.php` redemande l'acceptation : le middleware `EnsureCurrentTerms` renvoie l'utilisateur connecté vers `/conditions` tant qu'il n'a pas accepté (pages légales, confidentialité et déconnexion restent accessibles).

7. Statuts et transitions

Chaque statut est une chaîne (`string(20)`) ; seules les transitions ci-dessous sont prévues par le code. Toute nouvelle transition passe par le contrôleur ou le service indiqué, jamais par une mise à jour directe en base.

Table	Statuts	Transitions et déclencheur
<code>researcher_profiles</code>	incomplete, pending, active, suspended, rejected	incomplete → pending (chercheur soumet) → active / rejected / incomplete (admin décide) ; active ↔ suspended (admin) ; suspended aussi après effacement RGPD ou signalement DSA
<code>studies</code>	draft, review, published, rejected	review → published / rejected (admin) ; modification importante → review ; signalement DSA → rejected ; retrait = <code>deleted_at</code>
<code>orders</code>	pending, awaiting, paid, refunded	paiement carte → paid ; virement ou bon de commande → awaiting puis paid ; refunded réservé au remboursement

Table	Statuts	Transitions et déclencheur
mission_requests	open, awarded, closed	open → awarded (proposition acceptée) ; open → closed (client ferme ou modération)
proposals	sent, accepted, declined	sent → accepted (client) ; les autres sent → declined automatiquement ; sent → declined (client refuse)
contracts	offer, counter, accepted, funding, active, paused, dispute, completed, declined	offer / counter (offre directe) → accepted → funding (date fixée) → active (dépôt payé) ; active → dispute (heures contestées) → active (arbitrage) ; active ↔ paused, → completed (admin)
escrow_deposits	pending, held, released, refunded	pending → held (paiement reçu) → released (mensualité à la clôture, released_month) ; refunded en cas de remboursement
time_logs	pending, approved, auto_approved, disputed, rejected	pending → approved / disputed (client) ; pending → auto_approved après 7 jours ; disputed → approved (heures entières ou divisées par 2) / rejected (arbitrage)
statements	draft, approved, processing, paid, carried, hold	draft → approved → processing → paid (Finance, liste ou kanban) ; approved ou processing sous le seuil → carried ; hold = bloqué (document, litige)
reviews	published, hidden	published → hidden (modération DSA)
subscriptions	active, cancelled, expired	active à la souscription ; cancelled / expired en fin de période
data_requests	open, done, rejected	open → done / rejected (admin, réponse envoyée par e-mail, échéance due_at = 1 mois)

Table	Statuts	Transitions et déclencheur
<code>content_reports</code>	open, actioned, rejected	open → actioned / rejected avec exposé des motifs (DSA art. 17)

Les trois kanbans de l'admin (chercheurs, contrats, relevés) appellent les mêmes routes de statut que les formulaires : `POST JSON {status} → {ok}` ou 409 avec l'URL de la fiche quand la transition exige une vérification humaine.

8. Exploitation, sécurité et mise en production

L'application tourne sur MariaDB 10.11 : `migrate:fresh --seed` (43 tables, 54 clés étrangères, données de démo) et les 15 tests automatisés passent sur MariaDB comme sur SQLite.

Installation

1. Créer la base : `CREATE DATABASE ubcg_research CHARACTER SET utf8mb4 COLLATE utf8mb4_unicode_ci` ; + un utilisateur dédié avec les droits sur cette base.
2. `composer install`, `cp .env.example .env`, `php artisan key:generate`, renseigner `DB_*`, `MAIL_*`, `OWNER_PASSWORD`, `LEGAL_*`.
3. `php artisan migrate --seed` (en production : `--force`, sans données de démo car `DemoSeeder` ne tourne que hors production), puis `php artisan storage:link`.
4. Cron : `* * * * * php artisan schedule:run` ; worker : `php artisan queue:work` pour les e-mails.
5. Production : `APP_ENV=production`, `APP_DEBUG=false`, HTTPS, `php artisan config:cache route:cache view:cache`.

Tâches planifiées (`routes/console.php`)

Commande	Quand	Effet
<code>ubcg:auto-approve</code>	Tous les jours 02:00	Heures <code>pending</code> depuis 7 jours → <code>auto_approved</code>
<code>ubcg:close-month</code>	Le 1er du mois 03:00	Relevés du mois précédent, mensualités libérées du séquestre
<code>ubcg:retention</code>	Dimanche 04:00	Purge du journal (24 mois), signalements clos (24 mois), user-agent des consentements (13 mois)

Sécurité en place

- Mots de passe bcrypt, 10 caractères minimum avec majuscules, minuscules et chiffres ; limites de débit (connexion, formulaires, IA, AJAX) ; honeypots sur inscription et signalement.
- Admin : code à 6 chiffres par e-mail (10 minutes, 5 essais), droits par page, journal d'audit avec IP hachée.
- CSRF partout, session chiffrée, cookies `Secure` / `HttpOnly` / `SameSite=Lax`, en-têtes CSP avec nonce, HSTS, `X-Frame-Options`, `Referrer-Policy`, `Permissions-Policy`.
- Identifiant fiscal, RIB/IBAN et TIN DAC7 chiffrés en base ; pièces justificatives sur disque privé ; téléchargements par lien signé ou contrôle d'accès.
- Le pilote de paiement de démonstration lève une exception en production.

Conformité couverte par le code

Cadre	Ce qui est implémenté
RGPD / loi 09-08 (CNDP)	Consentements versionnés, re-acceptation des CGU, export JSON, demandes sous 1 mois, effacement par anonymisation, factures gardées 10 ans, purge planifiée
Cookies (ePrivacy, TDDDG)	Strict nécessaire par défaut ; Google Fonts et mesure d'audience après accord seulement
DSA	Formulaire de signalement, décision motivée notifiée, mention « Sponsorisé », transparence du classement, communications commerciales identifiées
Droit de la consommation UE	Prix TTC, renonciation expresse au droit de rétractation pour le contenu numérique, CGV et formulaire de rétractation
AI Act art. 50	Traductions IA étiquetées et modifiables, page « Utilisation de l'IA », journal d'usage sans contenu
DAC7	Collecte chiffrée des données fiscales des chercheurs
Impressum	UBCG SARL, RC Marrakech 132191, ICE 003211521000046, IF 53546929, Rue Imam Chafii, Quartier Hivernage, 40020 Marrakech

À compléter avant la mise en ligne

- ☐ Brancher un prestataire de paiement agréé dans `app/Services/Payments` ; le séquestre doit être tenu par un établissement habilité.
- ☐ Renseigner `LEGAL_TAXE_PRO` , `LEGAL_CNSS` , `LEGAL_CAPITAL` , `LEGAL_CNDP_NUMBER` , `LEGAL_EU_REP` , `LEGAL_HOST` (les pages légales affichent « à compléter » tant qu'ils sont vides).
- ☐ Faire relire pages légales, CGU chercheurs et mécanisme de séquestre par un avocat (droit marocain et droit de l'UE).
- ☐ Valider les taux de retenue à la source avec l'expert-comptable (Admin › Commission & fiscalité).
- ☐ Choisir le fournisseur d'IA (`AI_TRANSLATION_DRIVER=http`), avec contrat de sous-traitance RGPD.
- ☐ Auto-héberger les polices pour supprimer tout transfert vers Google.
- ☐ Sauvegardes chiffrées de la base et de `storage/app/private` .