BTS SIO SLAM

# Documentation Technique de Réalisation

## Gestion des mandats de représentation patronale

**Laravel 13 + Filament 5 — preprod-mandats.upv.org**

| | |
|---|---|
| **Développeur** | Cyrille COUR |
| **Organisation** | UPV (Union Patronale du Var) |
| **Stack technique** | PHP 8.3 / Laravel 13 / Filament 5 / MySQL |
| **Environnement** | Evolix (prod) |
| **Période** | Septembre 2026 |

# Sommaire

# 1. Mise en place de l'environnement

## 1.1 Création du projet Laravel 13

Le projet a été initialisé en local via Composer avec PHP 8.3 (Laragon) :

```
composer create-project laravel/laravel preprod-mandats
cd preprod-mandats
```

Vérification de la version de Laravel installée :

```
php artisan --version
# Laravel Framework 13.x
```

Configuration du fichier .env local (connexion à la base de données MySQL Laragon) :

```
DB_CONNECTION=mysql
DB_HOST=127.0.0.1
DB_PORT=3306
DB_DATABASE=preprod_mandats
DB_USERNAME=root
DB_PASSWORD=
```

Génération de la clé applicative :

```
php artisan key:generate
```

Configuration des drivers session, cache et queues sur database (nécessaire en environnement serveur sans Redis) :

```
SESSION_DRIVER=database
CACHE_STORE=database
QUEUE_CONNECTION=database
```

Création des tables système Laravel :

```
php artisan migrate
```

## 1.2 Installation de Filament 5

Filament 5 est le panneau d'administration qui constitue l'intégralité de l'interface de l'application Mandats. Installation via Composer :

```
composer require filament/filament:"^5.0" --with-all-dependencies
```

Installation du panneau par défaut :

```
php artisan filament:install --panels
```

Filament génère un fichier de configuration du panneau dans app/Providers/Filament/AdminPanelProvider.php. Lors de l'installation, Filament demande si l'on souhaite ajouter le trait FilamentUser au modèle User — réponse : oui.

Publication des assets Filament :

```
php artisan filament:assets
```

Accès au panneau en local :

```
http://localhost/preprod-mandats/public/admin
```

## 1.3 Structure du projet

Architecture MVC Laravel avec les dossiers propres à Filament :

| Dossier / Fichier | Rôle |
|---|---|
| app/Models/ | Modèles Eloquent (11 modèles métier) |
| app/Filament/Resources/ | Resources Filament (CRUD par entité) |
| app/Filament/Widgets/ | Widgets du dashboard (KPI, statistiques) |
| app/Providers/Filament/ | Configuration du panneau Filament |
| database/migrations/ | Migrations Laravel (création des tables) |
| database/seeders/ | Seeders pour les données initiales |
| .github/workflows/ | Pipeline CI/CD GitHub Actions |
| .env | Variables d'environnement (hors Git) |
| storage/ | Fichiers temporaires, logs, cache (chmod 777) |

## 1.4 Initialisation Git & dépôt GitHub

Initialisation du dépôt local et premier commit :

```
git init
git add .
git commit -m "feat: initialisation projet Laravel 13 + Filament 5"
```

Connexion au dépôt distant de l'organisation SI-UPV (dépôt privé) :

```
git remote add origin https://github.com/SI-UPV/preprod-mandats.git
git branch -M main
git push -u origin main
```

Le dépôt étant privé, le workflow de déploiement GitHub Actions nécessite un jeton d'accès personnel (PAT) intégré à l'URL du remote sur le serveur. Cette configuration est détaillée en section 5.

# 2. Base de données

## 2.1 Choix technique

Base de données MySQL hébergée localement sous Laragon (port 3306) pour le développement, et sur le serveur Evolix pour la recette. L'ORM Eloquent de Laravel gère l'intégralité des interactions avec la base via les migrations et les modèles.

La base de données de recette se nomme preprod_mandats et est provisionnée automatiquement lors du premier déploiement via la commande php artisan migrate exécutée dans le pipeline CI/CD.

## 2.2 Modèle de données

L'application repose sur 11 tables métier structurées autour de deux axes : la hiérarchie organisationnelle et la chaîne de représentation (mandat → mandature → mandataire).

| Table | Rôle métier | Relations clés |
|---|---|---|
| domaines | Grands domaines thématiques (Économie, Social…) | hasMany → poles |
| poles | Pôles rattachés à un domaine | belongsTo → domaine, hasMany → instances |
| instances | Instances de représentation | belongsTo → pole, hasMany → entites |
| entites | Entités (organismes, entreprises) | belongsTo → instance, hasMany → bloc_quotas |
| bloc_quotas | Quota de sièges par entité et syndicat | belongsTo → entite, hasMany → mandats |
| mandats | Siège individuel (titulaire / suppléant) | belongsTo → bloc_quota, hasMany → mandatures |
| mandatures | Période d'occupation d'un siège | belongsTo → mandat + mandataire |
| mandataires | Personnes représentantes | hasMany → mandatures |
| alertes | Alertes de non-conformité (polymorphe) | morphTo → mandat ou mandataire |
| regle_gestions | Règles qui génèrent les alertes | hasMany → alertes |
| users | Comptes utilisateurs de l'application | — |

## 2.3 Migrations principales

Chaque modèle dispose de sa migration dédiée. Exemple de la migration des mandataires :

```
Schema::create('mandataires', function (Blueprint $table) {
    $table->id();
    $table->string('civilite', 10)->nullable();
    $table->string('nom');
    $table->string('prenom');
    $table->enum('genre', ['H', 'F'])->nullable();
    $table->date('date_naissance')->nullable();
    $table->string('email')->nullable();
    $table->string('telephone', 20)->nullable();
    $table->string('entreprise')->nullable();
    $table->boolean('adherent')->default(false);
    $table->boolean('cotisation_a_jour')->default(false);
    $table->boolean('actif')->default(true);
    $table->softDeletes();
    $table->timestamps();
});
```

La colonne softDeletes() est présente sur tous les modèles métier : les suppressions sont logiques (champ deleted_at), ce qui préserve l'historique des données.

Lancement des migrations :

```
php artisan migrate
# ou en environnement de recette (via pipeline) :
php artisan migrate --force
```

## 2.4 Données initiales (seeders)

Un seeder DatabaseSeeder.php peuple les tables de référence au premier déploiement. Les règles de gestion par défaut sont également insérées via le modèle RegleGestion :

```
php artisan db:seed
```

Les six règles de gestion insérées par défaut sont :

| Code règle | Libellé |
|---|---|
| AGE_DEFAULT | Limite d'âge par défaut (70 ans) |
| AGE_SS | Limite d'âge Sécurité Sociale (66 ans) |
| AGE_TC | Limite d'âge Tribunal de Commerce (75 ans) |
| PARITE | Non-conformité parité H/F |
| COTISATION | Décotisation entreprise |
| RPA | Règle de priorité absolue (siège titulaire vacant) |

# 3. Modèle métier & logique applicative

## 3.1 Hiérarchie organisationnelle

L'application structure la représentation patronale à travers quatre niveaux imbriqués. Chaque niveau est géré indépendamment via Filament, avec des relations Eloquent enchaînées :

**Domaine → Pôle → Instance → Entité**

| Niveau | Exemple concret |
|---|---|
| Domaine | ÉCONOMIE |
| Pôle | Emploi & Formation |
| Instance | France Travail PACA |
| Entité | France Travail PACA (organisme paritaire) |

La relation hasManyThrough sur le modèle Domaine permet d'accéder directement aux instances sans passer par les pôles intermédiaires :

```
// Domaine.php
public function instances(): HasManyThrough
{
    return $this->hasManyThrough(Instance::class, Pole::class);
}
```

## 3.2 Gestion des sièges (BlocQuota)

Un BlocQuota définit le nombre de sièges (titulaires et suppléants) qu'une entité doit pourvoir, pour un syndicat donné (MEDEF, CPME, ou Autre). Il génère automatiquement les mandats correspondants via la méthode genererMandats() :

```
// BlocQuota.php
public function genererMandats(): void
{
    $this->syncSieges('titulaire', $this->nb_titulaires);
    $this->syncSieges('suppleant', $this->nb_suppleants);
}
```

Le genre attendu est automatiquement calculé pour respecter la parité (H/F en alternance selon le numéro d'ordre du siège). Chaque Mandat dispose d'un statut vacant ou pourvu, mis à jour automatiquement lors de la création ou suppression d'une Mandature via les événements Eloquent (booted).

## 3.3 Mandataires et Mandatures

Un Mandataire est une personne physique qui peut occuper plusieurs sièges simultanément. Une Mandature est la matérialisation de l'occupation d'un siège par un mandataire sur une période donnée :

| Champ Mandataire | Description |
|---|---|
| civilite / nom / prenom | Identité civile |
| genre / date_naissance | Données démographiques (contrôle parité et âge) |
| entreprise / adherent | Appartenance syndicale et statut de cotisation |
| actif | Mandataire actif ou archivé |

La fermeture d'une Mandature (renseignement de date_fin + motif_fin) bascule automatiquement le siège correspondant en statut vacant via les observers Eloquent :

```
// Mandature.php — booted()
static::updated(function (self $m) {
    if ($m->isDirty('date_fin') && $m->date_fin !== null) {
        // Vérifie qu'aucune autre mandature active n'occupe le siège
        $m->mandat->update(['statut' => 'vacant']);
    }
});
```

## 3.4 Système d'alertes

Le moteur d'alertes détecte les non-conformités et génère des instances d'Alerte rattachées polymorphiquement à un Mandat ou à un Mandataire. Chaque alerte est liée à une RegleGestion et dispose d'un niveau de gravité (info, warning, critical) ainsi qu'un statut de traitement :

| Statut alerte | Signification |
|---|---|
| ouverte | Non traitée — visible sur le dashboard |
| resolue | Traitée par un utilisateur avec commentaire |
| ignoree | Explicitement mise de côté avec justification |

La résolution ou l'ignorance d'une alerte est tracée avec l'identifiant de l'utilisateur traitant et la date d'intervention :

```
// Alerte.php
public function resoudre(int $userId, string $commentaire = ''): void
{
    $this->update([
        'statut'      => 'resolue',
        'traitee_par' => $userId,
        'traitee_le'  => now(),
        'commentaire_traitement' => $commentaire,
    ]);
}
```

# 4. Interface Filament 5

## 4.1 Dashboard et indicateurs KPI

La page d'accueil de l'application présente quatre indicateurs de pilotage mis à jour en temps réel via des widgets Filament :

| Indicateur | Calcul | Couleur alerte |
|---|---|---|
| Complétude des mandats | Ratio sièges pourvus / total sièges | Rouge si < 100% |
| Sièges vacants | Mandats au statut "vacant" | Vert si 0 |
| Alertes critiques | Alertes niveau critical ouvertes | Vert si 0 |
| Mandataires actifs | Mandataires avec actif = true | Informatif |

Le panneau Filament affiche également un widget de bienvenue avec le compte connecté et un lien vers la documentation Filament. La barre de navigation latérale regroupe tous les modules de l'application.

## 4.2 Référentiels (Domaines, Pôles, Instances, Entités)

Les quatre niveaux de la hiérarchie organisationnelle sont gérés via des Resources Filament distinctes. Chaque Resource offre une liste paginée avec recherche et filtres, ainsi qu'un formulaire de création/édition.

Le module Domaines et Pôles sont des référentiels simples (nom, code, description). Les Instances et Entités disposent en plus d'informations de localisation (adresse, ville, code postal) et d'un indicateur de statut actif.

La colonne Domaine visible dans la liste des Bloc Quotas est obtenue par une relation enchaînée : Bloc Quota → Entité → Instance → Pôle → Domaine.

## 4.3 Bloc Quotas

Le module Bloc Quotas est au cœur de la configuration des mandats. Chaque enregistrement définit :

| Champ | Description |
|---|---|
| entite_id | Entité concernée (avec affichage du domaine) |
| syndicat | Organisation patronale : MEDEF, CPME ou Autre |
| nb_titulaires | Nombre de sièges titulaires à pourvoir |
| nb_suppleants | Nombre de sièges suppléants à pourvoir |
| tolerance_parite | Tolérance sur la contrainte de parité H/F |
| limite_age_specifique | Surcharge de la limite d'âge par défaut |
| debut_mandature / fin_mandature | Période de validité du bloc |

La liste affiche les colonnes Entité (triable), Domaine, Syndicat (badge coloré), Titulaires, Suppléants, Début et Fin de mandature. La pagination est réglée sur 10 résultats par page.

## 4.4 Mandataires

La Resource Mandataires permet de gérer le répertoire des personnes susceptibles d'occuper des sièges. Le formulaire de création regroupe les informations en sections thématiques :

| Section formulaire | Champs inclus |
|---|---|
| Identité | Civilité, nom, prénom, genre, date de naissance |
| Contact | Email, téléphone |
| Appartenance | Entreprise, fonction, expertise métier |
| Adhésion | Statut adhérent, date d'adhésion, cotisation à jour |
| Statut | Actif / inactif, notes libres |

L'attribut calculé nom_complet (concaténation civilité + prénom + nom) est affiché dans toutes les listes de sélection de mandataires à travers l'application. L'âge est calculé dynamiquement depuis la date de naissance.

## 4.5 Mandats & Mandatures

La Resource Mandats liste les sièges créés par les Bloc Quotas. Chaque mandat affiche son label (T1, T2, S1, S2…), son type (titulaire/suppléant) et son statut (vacant/pourvu) via un badge coloré.

La Resource Mandatures gère les affectations : pour chaque mandat, on peut créer une mandature en sélectionnant un mandataire et en renseignant la date de début. La date de fin et le motif de fin sont renseignés lors de la clôture de la mandature. L'indicateur en_cours est calculé dynamiquement (date_fin nulle = en cours).

## 4.6 Alertes & Règles de gestion

La Resource Alertes liste toutes les alertes générées par le moteur de conformité. Les colonnes affichées sont : Niveau (badge), Titre, Règle liée, Statut, Traité par, Détecté le.

Le bouton "New alerte" permet de créer manuellement une alerte. La liste est vide si aucune non-conformité n'est détectée, ce qui est le cas d'une application correctement configurée et à jour.

La Resource Règle Gestions affiche et permet de modifier les six règles de gestion prédéfinies ainsi que leurs seuils. Chaque règle dispose d'un template de message paramétré (variables {nom}, {age}, {bloc}, etc.).

# 5. Déploiement CI/CD

## 5.1 Workflow GitHub Actions

Le pipeline de déploiement est défini dans .github/workflows/deploy.yml. Il se déclenche à chaque push sur la branche main et réalise le déploiement automatique sur le serveur Evolix via SSH.

Structure du workflow :

```
name: Deploy preprod-mandats

on:
  push:
    branches: [main]

jobs:
  deploy:
    runs-on: ubuntu-latest
    steps:
      - name: Deploy via SSH
        uses: appleboy/ssh-action@v1.0.0
        with:
          host: ${{ secrets.PROD_SSH_HOST }}
          username: ${{ secrets.PROD_SSH_USER }}
          key: ${{ secrets.PROD_SSH_KEY }}
          port: 2222
          script: |
            cd /var/www/preprod-mandats.upv.org/
            git pull origin main
            /usr/bin/php8.3 artisan migrate --force
            /usr/bin/php8.3 artisan config:cache
            /usr/bin/php8.3 artisan route:cache
            /usr/bin/php8.3 artisan view:cache
```

Le port 2222 est le port SSH standard de l'infrastructure Evolix (le port 22 standard n'est pas utilisé). L'action appleboy/ssh-action@v1.0.0 gère la connexion SSH et l'exécution des commandes à distance.

## 5.2 Secrets GitHub configurés

Les informations sensibles sont stockées dans GitHub → Settings → Secrets and variables → Actions du dépôt SI-UPV/preprod-mandats :

| Secret GitHub | Valeur stockée |
|---|---|
| PROD_SSH_HOST | Adresse IP ou hostname du serveur Evolix |
| PROD_SSH_USER | Nom de l'utilisateur SSH sur le serveur |
| PROD_SSH_KEY | Clé privée SSH ed25519 sans passphrase |

La clé publique correspondante (PROD_SSH_KEY.pub) a été ajoutée au fichier ~/.ssh/authorized_keys de l'utilisateur sur le serveur Evolix.

Le dépôt étant privé, la commande git pull nécessite un jeton d'accès personnel (PAT) intégré à l'URL du remote sur le serveur :

```
git remote set-url origin \
  https://ghp_XXXXXXXXXXXXX@github.com/SI-UPV/preprod-mandats.git
```

*Cette URL est à configurer une seule fois en SSH sur le serveur. Le PAT utilisé doit avoir les permissions repo (lecture).*

## 5.3 Déploiement SSH sur serveur Evolix

Initialisation du dépôt sur le serveur lors du premier déploiement. Le dossier .git a été transféré via WinSCP (glisser-déposer avec inclusion des fichiers cachés) pour contourner l'absence d'accès direct à git clone sur le serveur.

Étapes d'initialisation réalisées manuellement en SSH :

```
# 1. Vérification de la version PHP disponible
/usr/bin/php8.3 --version

# 2. Installation des dépendances
/usr/bin/php8.3 /usr/bin/composer install \
    --no-dev --optimize-autoloader

# 3. Configuration du fichier .env
cp .env.example .env
# (édition manuelle des variables de production)

# 4. Génération de la clé applicative
/usr/bin/php8.3 artisan key:generate

# 5. Migrations et cache
/usr/bin/php8.3 artisan migrate --force
/usr/bin/php8.3 artisan config:cache

# 6. Permissions sur le dossier storage
chmod -R 777 storage/
```

Le chmod 777 récursif sur storage/ est indispensable : Apache doit pouvoir écrire dans tous les sous-dossiers (framework/cache, framework/sessions, framework/views, logs). L'option "appliquer récursivement" doit être cochée lors de la modification via WinSCP.

## 5.4 Configuration Apache & .htaccess

Le virtualhost Apache pointe vers le dossier public/ du projet Laravel. La configuration Evolix impose une restriction sur la directive AllowOverride qui empêche l'utilisation de certaines directives dans le .htaccess.

Modification nécessaire du fichier public/.htaccess pour commenter les lignes interdites :

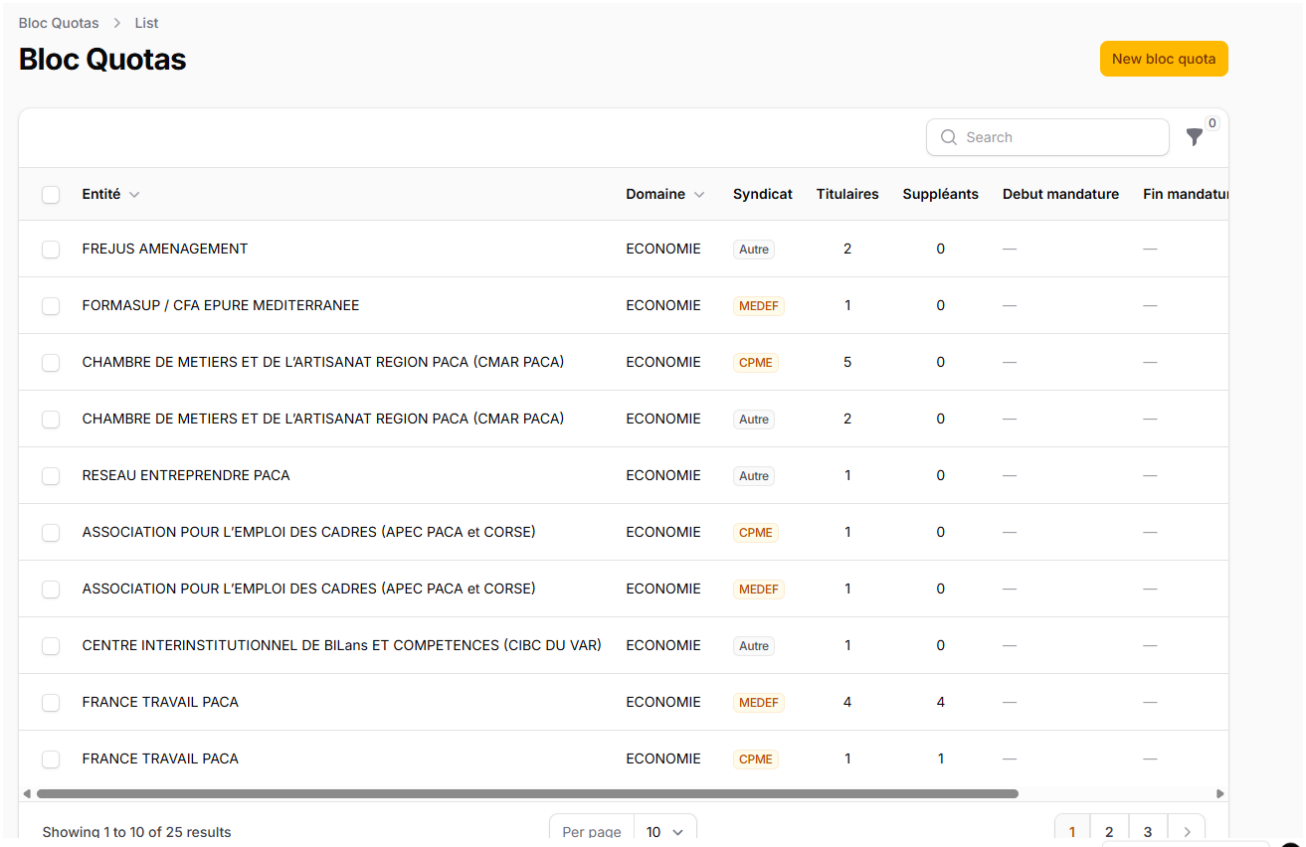
```
<IfModule mod_rewrite.c>
    <IfModule mod_negotiation.c>
        # Options -MultiViews -Indexes
        # ↑ Commenté : interdit par AllowOverride Evolix
    </IfModule>

    RewriteEngine On
    RewriteCond %{HTTP:Authorization} .
    RewriteRule .* - [E=HTTP_AUTHORIZATION:%{HTTP:Authorization}]
    RewriteCond %{REQUEST_FILENAME} !-d
    RewriteCond %{REQUEST_FILENAME} !-f
    RewriteRule ^ index.php [L]
</IfModule>
```

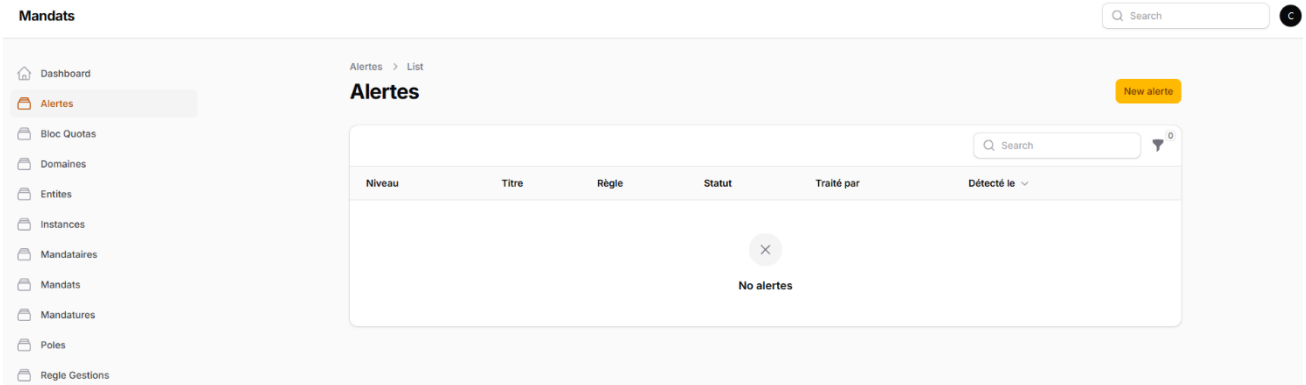
Sans cette modification, Apache renvoie une erreur 500 car la directive Options est restreinte au niveau de la configuration serveur Evolix (AllowOverride ne l'autorise pas).

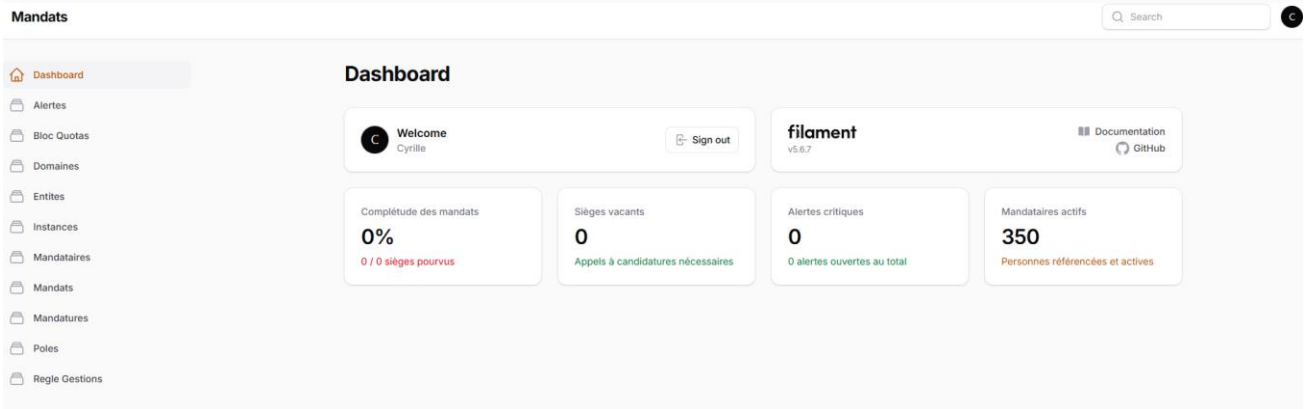
## Synthèse du déploiement

| Étape | Résultat |
|---|---|
| Projet Laravel 13 + Filament 5 | Fonctionnel |
| Pipeline GitHub Actions (push → deploy) | Actif |
| Déploiement SSH Evolix (port 2222) | Opérationnel |
| Migrations automatiques | Appliquées via --force |
| Configuration Apache (.htaccess) | Corrigée (Options commentées) |
| Permissions storage (chmod 777 -R) | Appliquées récursivement |
| URL de recette | https://preprod-mandats.upv.org — Accessible |

*Le déploiement de l'application Mandats est considéré comme opérationnel. Chaque push sur la branche main déclenche automatiquement la mise à jour de l'environnement de recette.*


Bloc Quotas > List
Bloc Quotas
New bloc quota
Search
Entité
Domaine
Syndicat
Titulaires
Suppléants
Debut mandature
Fin mandatu
FREJUS AMENAGEMENT ECONOMIE Autre 2 0
FORMASUP / CFA EPURE MEDITERRANEE ECONOMIE MEDEF 1 0
CHAMBRE DE METIERS ET DE L'ARTISANAT REGION PACA (CMAR PACA) ECONOMIE CPME 5 0
CHAMBRE DE METIERS ET DE L'ARTISANAT REGION PACA (CMAR PACA) ECONOMIE Autre 2 0
RESEAU ENTREPRENDRE PACA ECONOMIE Autre 1 0
ASSOCIATION POUR L'EMPLOI DES CADRES (APEC PACA et CORSE) ECONOMIE CPME 1 0
ASSOCIATION POUR L'EMPLOI DES CADRES (APEC PACA et CORSE) ECONOMIE MEDEF 1 0
CENTRE INTERINSTITUTIONNEL DE BILans ET COMPETENCES (CIBC DU VAR) ECONOMIE Autre 1 0
FRANCE TRAVAIL PACA ECONOMIE MEDEF 4 4
FRANCE TRAVAIL PACA ECONOMIE CPME 1 1
Showing 1 to 10 of 25 results
Per page 10
1 2 3


Mandats
Search
Dashboard
Alertes
Bloc Quotas
Domaines
Entites
Instances
Mandataires
Mandats
Mandatures
Poles
Regle Gestions
Alertes > List
Alertes
New alerte
Search
Niveau
Titre
Règle
Statut
Traité par
Détecté le
No alertes


Mandats
Search
Dashboard
Alertes
Bloc Quotas
Domaines
Entites
Instances
Mandataires
Mandats
Mandatures
Poles
Regle Gestions
Dashboard
Welcome
Cyrille
Sign out
filament
v5.6.7
Documentation
GitHub
Complétude des mandats
0%
0 / 0 sièges pourvus
Sièges vacants
0
Appels à candidatures nécessaires
Alertes critiques
0
0 alertes ouvertes au total
Mandataires actifs
350
Personnes référencées et actives