Liste de tâches pour la bibliothèque CMS Laravel

1. Structure du projet et conventions

- Créer une librairie Composer conforme au standard PSR-4.
- Fournir un ServiceProvider pour :
    - Publier les fichiers de configuration.
    - Charger les vues du thème par défaut et permettre leur surcharge.
    - Charger les routes pour le front et le back office.
    - Enregistrer les événements et les écouteurs nécessaires.
- Respecter les standards PSR (PSR-1/PSR-4) pour la structure, la nomenclature et l'autoloading.

2. Core CMS

- Mettre en place la gestion des thèmes, pages et menus :
    - Créer un thème par défaut dans la librairie.
    - Permettre la surcharge des vues via `/resources/views`.
- Créer les modèles et migrations pour les entités *Theme*, *Page* et *Menu*.
- Implémenter un système d'extensions par plugins s'appuyant sur des événements personnalisés.
- Développer un parseur de shortcodes avec support des imbrications infinies.

3. Back-Office

- Concevoir un tableau de bord moderne :
    - Sidebar gauche repliable.
    - Barre supérieure avec avatar et menu utilisateur.
    - Section principale avec titre, fil d'ariane, contenu principal et footer.
- Mettre en place une gestion des droits d'accès via les *Policies* et *Gates* de Laravel.
- Gérer le multilingue à l'aide du système de traduction de Laravel.
- Implémenter des fonctions d'import/export (JSON/CSV) pour les contenus et configurations.
- Construire des interfaces d'administration pour les thèmes et plugins (activation,

désactivation, aperçu).

4. Système de plugins

- Définir et développer les plugins suivants :
    - SEO (gestion des métadonnées, sitemap…)
    - Optimisation des images (compression à l'upload)
    - Solutions de paiement (intégration de passerelles de paiement)
    - Immobilier (gestion de biens, formulaires de contact)
    - Slider (carrousels d'images)
    - Éditeur de texte WYSIWYG
    - Gestion des shortcodes et de leur imbrication
    - Tout autre plugin d'optimisation nécessaire
- Permettre l'enregistrement dynamique des plugins via un `PluginServiceProvider` et des hooks.

5. Performance et optimisation

- Mettre en cache les pages et menus via les systèmes de cache de Laravel (Redis, Memcached…).
- Optimiser les requêtes SQL et ajouter les index nécessaires.
- Compiler et minifier les assets (CSS/JS) à l'aide de Vite ou Laravel Mix.
- Mettre en place des tests automatisés pour mesurer les temps de réponse et assurer des performances supérieures à WordPress.

Cette liste de tâches servira de guide pour suivre l'évolution du développement de votre bibliothèque CMS Laravel.