Roadmap Technique SEO Actionnable

1. Accessibilité par les moteurs de recherche et les utilisateurs

1.1 Résolution des problèmes d'indexation

- •Identifier les URLs en erreur (404, explorées non indexées, échouées).
- •Supprimer ou rediriger les pages 404 vers les pages pertinentes.
- •Mettre à jour les pages explorées non indexées : enrichir le contenu, ajouter du maillage interne.
- •Fusionner les contenus similaires ou trop proches.

1.2 Configuration du fichier robots.txt

- •Implémenter le fichier recommandé suivant :

```
User-agent: *
Disallow: /wp-admin/
Disallow: /wp-login.php
Disallow: /?s=
Disallow: */trackback/
Disallow: */feed/
Disallow: */comments/
Disallow: */?*
Allow: /wp-admin/admin-ajax.php
Sitemap: <https://jemeremetsausport.com/sitemap_index.xml>
```

1.3 Traitement des pages orphelines

- •Répertorier toutes les pages orphelines identifiées.
- •Créer un plan de maillage interne vers ces pages depuis les pages piliers.
- •Intégrer les pages dans les menus ou les suggestions d'articles liés.

1.4 Sitemaps XML

- •Vérifier que la sitemap est bien générée et active.
- •Soumettre la sitemap à nouveau dans la Search Console.
- •Corriger les erreurs présentes dans la sitemap : pas de 404, noindex, ou redirections.

2. Architecture du site

2.1 Implémentation du fil d'Ariane

- •Ajouter un fil d'Ariane sur toutes les pages.
- •Positionner le fil d'Ariane au-dessus du contenu principal.
- •Rendre tous les niveaux cliquables.
- •Utiliser ">" comme séparateur entre les niveaux.

2.2 Refonte de la navigation principale

- Supprimer le double menu actuel.
- Créer un menu principal contenant uniquement les pages stratégiques :
  - Reprendre le sport
  - Programme d'entrainement
  - Perdre du poids / maigrir
  - Exercices ciblés
  - Devenir coach
  - Comprendre mon corps
  - Alimentation saine
- Créer un menu secondaire ou footer épuré pour les liens secondaires.

2.3 Refonte de la structure URL

- Adopter une architecture siloée :
  - /exercices-cibles/epaules/face-pull/
  - /programme-entrainement/4-jours/
- Mettre à jour les URLs existantes via des redirections 301.

2.4 Optimisation de la pagination

- Implémenter les balises rel="prev" et rel="next".
- Assurer que chaque page paginée ait un self canonical.
- Garder les pages paginées en index, follow.
- Structurer les URLs correctement : /page/2, /page/3, etc.

2.5 Maillage interne

- Ajouter du maillage interne contextualisé à la fin de chaque article.
- Créer des suggestions d'articles similaires ou blocs "vous aimerez aussi".
- Utiliser des ancres descriptives et optimisées sémantiquement.
- Mailler systématiquement chaque contenu vers sa catégorie pilar.

3. Audit SEO On-Page

3.1 Balises Title et Meta Description

- Réduire les balises Title > 70 caractères.
- Augmenter les Titles < 30 caractères en ajoutant des mots-clés.
- Éviter les meta descriptions trop longues ou trop courtes.
- Ajouter des descriptions engageantes orientées utilisateur.

3.2 Balises H1 et H2

- 1 seul H1 par page, contenant le mot-clé principal.
- Structurer les contenus avec des H2/H3 clairs, hiérarchisés.
- Éviter les balises H2 dupliquées ou vides.

3.3 Optimisation des images

- Compresser toutes les images > 100 Ko.
- Renommer les fichiers avec des noms descriptifs en kebab-case.
- Ajouter une balise ALT pertinente pour chaque image.

Planification recommandée (3 mois)

Semaine	Actions principales
S1-S2	Correction robots.txt, soumission sitemap, suppression 404
S3-S4	Refonte du menu principal, création du fil d'Ariane
S5-S6	Implémentation de l'architecture siloée + redirections
S7	Réécriture des balises Title & Meta + Maillage interne
S8	Refonte pagination (balises, canonicals, indexation)
S9-S10	Optimisation des balises Hn et contenus enrichis
S11	Compression et renommage des images
S12	Contrôle qualité final + soumission GSC + reporting

Roadmap Stratégie SEO Actionnable

1. Optimisation des contenus existants

Objectif : tirer profit des contenus déjà positionnés pour générer des quick wins SEO.

1.1 Audit sémantique et ciblage des bonnes intentions

- Identifier les pages positionnées en TOP 3/10/20 sur des requêtes non pertinentes.
- Recentrer les pages sur les bons mots-clés (ex. : "deltoïde antérieur" → viser "exercice épaule").
- Mettre à jour les balises Title & Meta pour aligner intention et contenu.

1.2 Enrichissement du contenu

- Ajouter des FAQ basées sur les recherches associées (PAA).
- Intégrer des éléments engageants (visuels, vidéos, schémas).
- Clarifier l'intention des contenus : informatif ? pratique ? transformationnel ?

1.3 Maillage interne

- Relier chaque contenu à sa page pilier.
- Ajouter des liens contextuels vers d'autres articles du même silo.
- Optimiser les ancres pour renforcer la sémantique.

1.4 Suivi

- Mesurer le CTR et l'évolution de position via la Search Console.
- Utiliser des outils type SERPMantics pour analyser la couverture lexicale.

---

2. Déploiement du cocon sémantique

Objectif : renforcer la structure SEO par thématiques (silotage éditorial).

? Fichier associé :

https://docs.google.com/spreadsheets/d/12pEu3zBR-3oot3_oi_nEnJJsbp4SR0OO/edit?gid=1178323184#gid=1178323184

2.1 Création des pages piliers

Créer des contenus profonds et très qualitatifs sur les thématiques suivantes :

- /reprendre-le-sport/
- /programme-entrainement/
- /perdre-du-poids/
- /exercices-musculation/
- /devenir-coach/
- /recettes-saines/

Ces pages doivent :

- Couvrir l'intention TOFU/MOFU/BOFU.
- Être riches, bien structurées, et différenciantes.
- Intégrer des liens vers les sous-pages du cocon.

2.2 Rédaction des contenus secondaires

Créer ou améliorer les articles enfants liés à chaque pilier :

- Ex : /reprendre-le-sport-a-40-ans/, /programme-musculation-debutant/, etc.
- Utiliser les mots-clés fournis dans le tableau pour définir les contenus à produire.
- Appliquer les bonnes pratiques SEO on-page : balisage, intention, maillage, FAQ, etc.

2.3 Maillage entre les contenus du cocon

- Relier chaque page enfant à sa page pilier + autres enfants si cohérence.
- Mettre en place des blocs de navigation thématique dans chaque page.

---

3. Stratégie Netlinking

? Fichier de benchmark :

https://docs.google.com/spreadsheets/d/1uWJGOK_ULpd0MQkHMZ98d7iLFkJ6jA7j13yi2N5XX_E/edit

3.1 Objectifs

- Passer de 200 à 400 domaines référents en 12 mois.
- Augmenter le TrustFlow à 30+.
- Obtenir des liens depuis des sites dans les thématiques Fitness, Nutrition, Santé, Coaching, Lifestyle.

3.2 Acquisition de backlinks qualitatifs

- Articles invités sur des blogs spécialisés sport/santé.
- Partenariats avec coach sportifs, influenceurs, YouTubeurs.
- Backlinks depuis médias de vulgarisation santé (ex : Medisite, Doctissimo).

3.3 Création de contenus à fort potentiel de lien

- Guides complets (ex : "Comment reprendre le sport à 50 ans ?").
- Pages de ressources (ex : "Recettes saines pour sportifs").
- Visuels à valeur ajoutée (infographies, routines, vidéos).

3.4 Netlinking Tiers 2 & Tiers 3

- Booster les backlinks existants via Tiers 2.
- Créer des mini-sites ou blogs de renfort.
- Automatiser des signaux sociaux ou commentaires ciblés sur les pages citées.

Planification recommandée (3 mois)

Semaine	Actions principales
S1-S2	Sélection des pages existantes à retravailler + amélioration SEO on-page
S3	Réécriture des Titles & Meta Descriptions pour les quick wins
S4-S5	Lancement rédaction pages piliers du cocon
S6	Implémentation du maillage interne siloé
S7-S8	Création contenus enfants + enrichissements (FAQ, CTA, visuels)
S9	Rédaction contenus invités et partenariat netlinking
S10	Déploiement des premières campagnes de Netlinking (Linkuma ou autres)
S11	Analyse des performances via GSC + Majestic
S12	Ajustements + rapport de pilotage (gains, positions, CTR)

Audit

Bienvenue dans votre espace client. Vous trouverez sur ce document toutes les informations et supports liées à notre prestation.

Voici l’audit technique

ATTENTION NOTES AUX DÉVELOPPEURS WEB

Certaines recommandations doivent être réalisées sur un environnement de développement. La version de pré-production doit comporter un accès login et mot de passe. De plus, dans le header de cette page d'authentification, les meta doivent correspondrent comme ce qui suit :

<meta name=”robots” content=”noindex, follow”>

Bienvenue dans votre espace client. Vous trouverez sur ce document toutes les informations et supports liées à notre prestation.

Voici l’audit technique

ATTENTION NOTES AUX DÉVELOPPEURS WEB

Certaines recommandations doivent être réalisées sur un environnement de développement. La version de pré-production doit comporter un accès login et mot de passe. De plus, dans le header de cette page d'authentification, les meta doivent correspondrent comme ce qui suit :

<meta name=”robots” content=”noindex, follow”>

1. Audit des webs performance du site

1.1. Situation actuelle du site web

Cet audit SEO a pour but de lister l’ensemble des points à travailler. De ce fait, ce premier audit a pour but de centraliser l’ensemble des erreurs SEO et d’amener progressivement le site à un état plus optimisé d’un point de vue technique.

Pour structurer l’audit, nous allons utiliser un tableau de priorité.

Priorité à accorder	Type de priorité	Commentaires
1	**Décisif**	Risque très élevé de mauvais SEO
2	**Très important**	Risque élevé de mauvais SEO
3	**Important**	Risque présent mais raisonnable
4	**Utile**	Donner plus de poids à son site

1.2 Analyse des web performances du site web

État à date

Le site est globalement bien optimisé pour les performances web, notamment sur desktop grâce à l’usage de WP Rocket, Cloudflare et une version récente de PHP (8.2.27). Toutefois, certaines métriques montrent des axes d’optimisation possibles, en particulier sur mobile, où l’expérience utilisateur peut encore être fluidifiée.

Page d’accueil : https://jemeremetsausport.com/

Desktop :

Score performances : 94

Accessibilité : 96

Bonnes pratiques : 100

SEO : 92

- First Contentful Paint (FCP) : 0,4 s
- Largest Contentful Paint (LCP) : 0,6 s
- Total Blocking Time (TBT) : 10 ms
- Cumulative Layout Shift (CLS) : 0.171
- Speed Index (SI) : 1,2 s

Le site se charge en moins d'1s sur desktop. C'est excellent. L'optimisation est déjà très aboutie sur ce support.

Mobile :

Score performances : non précisé, mais indicateurs montrent des limites.

- First Contentful Paint (FCP) : 2,6 s
- Largest Contentful Paint (LCP) : 2,7 s
- Total Blocking Time (TBT) : 70 ms
- Cumulative Layout Shift (CLS) : 0.235
- Speed Index (SI) : 6,1 s

Les performances sont correctes mais perfectibles sur mobile, en particulier le Speed Index (>6s) et le CLS (>0.2), qui traduisent une instabilité visuelle et un chargement perçu comme long. Ces éléments peuvent impacter l'expérience utilisateur et le référencement mobile.

Page thématique : https://jemeremetsausport.com/circuit-training-perte-de-poids/

Desktop :

Score performances : 76

Accessibilité : 96

Bonnes pratiques : 100

SEO : 76

- First Contentful Paint (FCP) : 0,6 s
- Largest Contentful Paint (LCP) : 2,3 s
- Total Blocking Time (TBT) : 80 ms
- Cumulative Layout Shift (CLS) : 0.171
- Speed Index (SI) : 1,9 s

Les performances sont bonnes sur desktop. Le LCP est un peu plus long que sur la page d'accueil, mais reste dans une zone acceptable (<2,5s).

Mobile :

Score performances : 75

- First Contentful Paint (FCP) : 2,9 s
- Largest Contentful Paint (LCP) : 3,0 s
- Total Blocking Time (TBT) : 10 ms
- Cumulative Layout Shift (CLS) : 0.235
- Speed Index (SI) : 4,8 s

Le chargement reste fluide, mais le LCP à 3s et le CLS élevé montrent une marge d'amélioration sur la stabilité visuelle et la rapidité du contenu principal. On constate également une performance mobile plus faible que sur desktop, ce qui est un axe d'amélioration prioritaire sur ce type de site à usage mobile fréquent.

Le CLS mesuré sur mobile atteint 0,235, ce qui dépasse le seuil recommandé de 0,1. Cela signifie que des éléments visibles dans la fenêtre d'affichage subissent des décalages après le chargement initial, perturbant la stabilité visuelle de la page. Ce comportement impacte directement l'expérience utilisateur et les Core Web Vitals.

Pourquoi cette recommandation ?

Les temps de chargement d'un site web deviendront de plus en plus des éléments très importants et notamment avec la mise à jour de l'algorithme de Google en mai 2021 (Core Web Vitals pour les mobiles), puis en février 2022 (Desktop).

Nous devons donc travailler sur les webperformances (notamment pour les mobiles). On estime en 2022 à près de 80% d'utilisateurs mobile qui utilisent un moteur de recherche. Que ce soit pour rechercher de l'information, utiliser un assistant vocal ou acheter un produit / service, la majorité viennent d'utilisateurs mobile. Il est donc indispensable de proposer un site web adapté aux mobiles (rapidité, efficacité, architecture, HTML…).

1.2.1 Réduisez les ressources JavaScript inutilisées

Très important

État à date

Sur mobile, PageSpeed Insights identifie jusqu'à 292 Kio de JavaScript inutilisés sur certaines pages comme la page d'accueil et la page `circuit-training-perte-de-poids`. Ces ressources impactent directement les métriques LCP et FCP, en ralentissant le rendu du contenu visible, et participent à une augmentation de la consommation de bande passante, ce qui dégrade l'expérience utilisateur.

Problème détecté : Scripts tiers et scripts internes mal optimisés

? Principaux scripts concernés :

URL	Taille transférée	Économies potentielles
`show_ads_impl_with_ama_fy2021.js` (Doubleclick)	255,7 Kio	**154,8 Kio**
`adsbygoogle.js` (AdSense)	52,2 Kio	**34,0 Kio**
`0d6dc9d8e3.min.js` (propre au site)	96,1 Kio	**77,4 Kio**
`gtag/js` (Google Tag Manager)	147,1 Kio	**59,6 Kio**

Ces scripts sont chargés systématiquement, même lorsqu'ils ne sont pas utiles à l'affichage immédiat du contenu principal. Cela contribue à une surcharge du fil d'exécution principal et à une détérioration du Largest Contentful Paint (LCP), surtout sur mobile.

Solutions recommandées

1. WP Rocket – Reporter l'exécution des scripts JavaScript

- Activer l'option "Delay JavaScript Execution" dans WP Rocket.
  - Cela permet de reporter l'exécution des scripts jusqu'à l'interaction de l'utilisateur (scroll, clic ou tap).
  - À activer pour tous les scripts liés à :
    - Google AdSense
    - Google Tag Manager
    - DoubleClick (publicité)
    - Scripts personnalisés de thème ou de page qui ne participent pas au rendu initial.

? WP Rocket propose une configuration automatique intelligente, mais il est recommandé de tester manuellement les comportements après activation pour éviter les régressions.

---

2. Analyse de la couverture du code JavaScript

- Ouvrir Chrome DevTools > Coverage, puis charger les pages concernées.
- Identifier les blocs rouges, représentant les scripts JavaScript chargés mais inutilisés.
- Repérer les plugins WordPress ou les scripts Elementor injectant du code inutile.

? Objectif : désactiver le chargement de ces scripts sur les pages où ils ne sont ni visibles ni utilisés (par ex. carrousel, animations, widgets, tracking).

---

3. Optimisation des publicités AdSense

- Intégrer les scripts publicitaires de façon asynchrone uniquement.
- Utiliser `loading="lazy"` pour les blocs d'annonces, ou retarder leur affichage via un déclenchement sur scroll.
- Désactiver les annonces automatiques si elles ne sont pas stratégiquement utilisées.

---

4. Nettoyage des scripts injectés via plugins

•Vérifier si certains plugins (Jetpack, Site Kit, Elementor) ajoutent des fonctionnalités inutilisées (ex. sliders, stats frontend, etc.).
•Supprimer ou désactiver les modules internes qui ne sont pas critiques pour l'expérience utilisateur.

---

5. Scripts critiques à isoler

•Intégrer les scripts critiques in-line (très courts) et reporter le reste avec `defer` :

```
<script src="script.js" defer></script>
```

•Utiliser WP Rocket pour combiner les scripts non-bloquants si cela n'impacte pas le rendu visuel.

1.2.2 Élément identifié comme "Largest Contentful Paint"

Très important

État à date

Sur la page d'accueil de **https://jemeremetsausport.com/** (en version mobile), l'élément identifié comme Largest Contentful Paint (LCP) met 2 700 ms à s'afficher. Bien que ce chiffre soit dans les seuils acceptables de Google (inférieur à 2,5 s recommandé, 4 s toléré), il peut encore être optimisé pour améliorer la vitesse perçue et les Core Web Vitals.

Problème : image de fond en LCP (2 700 ms)

? Élément concerné :

```
<div class="elementor-cta__bg elementor-bg" style="background-image:
url('<https://jemeremetsausport.com/wp-content/uploads/>...')" role="img" aria-
label="entrainement&exercices">
```

? Analyse du temps de chargement :

Phase	% du LCP	Durée
TTFB (Time To First Byte)	22 %	**600 ms**
Délai de chargement	32 %	**850 ms**
Temps de chargement (download)	27 %	**720 ms**
Délai de rendu (render)	20 %	**530 ms**

Analyse

•L’élément LCP est une image de fond appliquée via un style `background-image`.
•Ces images ne sont pas détectées comme prioritaires par le navigateur contrairement aux balises `<img>`, ce qui retarde leur chargement.
•Même si le poids de l’image semble raisonnable, l’ordre de chargement et le rendu sont sous-optimaux.
•Elementor a tendance à charger beaucoup de styles CSS dynamiques, ce qui ajoute un délai de rendu pour les sections de type "Call To Action" avec background personnalisé.

Solutions recommandées

1. Transformer l’image de fond en `<img>` HTML visible

•Utiliser une balise `<img>` réelle à la place d’un background CSS, si cela ne compromet pas le design :

```
<img src="<https://jemeremetsausport.com/wp-content/uploads/.../visuel-entrainement.jpg>" alt="Entraînement & Exercices" width="..." height="..." fetchpriority="high">
```

•Cela permet au navigateur de détecter l’image comme LCP, de prioriser son chargement, et de l’afficher plus rapidement.
•Elementor permet d’ajouter une image dans un widget `Image` plutôt que dans une section de fond pour contourner ce problème.

---

2. Preload de l’image LCP dans le `<head>`

Si l’image ne peut pas être changée en `<img>`, il est fortement recommandé de la précharger manuellement :

```
<link rel="preload" as="image" href="<https://jemeremetsausport.com/wp-content/uploads/.../visuel-entrainement.jpg>" fetchpriority="high">
```

•Ce preload permet de forcer le navigateur à charger l’image immédiatement, même si elle est intégrée en `background-image`.

---

3. Optimisation de l’image elle-même

•Vérifier que l’image est bien servie avec une taille adaptée à la fenêtre d’affichage (éviter les images > 2000px de large sur mobile).
•Utiliser un format WebP ou AVIF si possible.
•Vérifier dans WP Rocket que l’option “Optimiser les images” (via Imagify ou un autre plugin) est bien activée.

---

4. CSS critique & priorisation du rendu

•Extraire le CSS critique de Elementor et l'injecter en `<head>` (WP Rocket propose une option automatique).
•Réduire le nombre de dépendances CSS (animations, effets de transition) sur la section contenant l'image.

---

5. Surveillance du LCP via PageSpeed et Search Console

•Suivre le rapport Core Web Vitals dans la Search Console pour vérifier si la métrique LCP passe dans le vertsuite aux modifications.
•Réitérer les tests après chaque changement, en particulier via PageSpeed Insights en mode mobile, plus sensible aux ralentissements.

1.2.3 Éliminez les ressources qui bloquent le rendu

Très important

État à date

PageSpeed Insights signale une économie potentielle de 1 580 ms en éliminant les ressources CSS et JavaScript qui bloquent le rendu initial. Sur mobile, cela impacte directement le First Contentful Paint (FCP) et le Largest Contentful Paint (LCP), avec un premier affichage retardé alors que la page est techniquement chargée en arrière-plan.

Problème : Ressources CSS bloquantes

? Ressource concernée (principale) :

`…/static/becada5b87.min.css?ver=174… (jemeremetsausport.com)`

•Taille transférée : 92,5 KiB
•Économie potentielle estimée : 600 ms

Cette feuille de style CSS — probablement générée par Elementor ou un plugin actif sur la page — est chargée de manière bloquante dans le `<head>`. Cela signifie que le navigateur attend le chargement complet du fichier avant de continuer le rendu, ce qui nuit fortement à la performance perçue.

Solutions recommandées

1. WP Rocket – Supprimer les CSS inutilisées et différer les scripts

Ĺ Activer les options suivantes dans WP Rocket :

- « Supprimer les CSS inutilisées »

  Cela permet de générer automatiquement du CSS critique par page, tout en supprimant les styles inutilisés ou chargés de manière globale par Elementor, Jetpack ou d'autres plugins.

- « Reporter l'exécution JavaScript »

  Cela différera les scripts JS non essentiels au rendu initial, comme :

  - les animations Elementor,
  - les widgets,
  - les scripts de suivi,
  - les effets visuels non critiques.

? WP Rocket crée une version inline du CSS critique directement dans le `<head>`, ce qui accélère fortement l'affichage initial.

¸ Attention : certains composants visuels peuvent disparaître ou se désaligner si un JS/CSS critique est mal détecté. Il est important de :

- Tester visuellement les pages importantes après activation.
- Ajouter manuellement certains fichiers dans la liste d'exclusion, si nécessaire.

---

2. Pré-chargement et différé des ressources critiques

Si certaines feuilles de style sont essentielles, tu peux envisager un preload CSS couplé à un chargement différé :

```
<link rel="preload" as="style" href="style.css" onload="this.rel='stylesheet'">
<noscript><link rel="stylesheet" href="style.css"></noscript>
```

Cela permet de :

- prioriser le téléchargement,
- tout en débloquant le rendu immédiatement.

WP Rocket s'en charge automatiquement pour la plupart des fichiers quand l'option CSS Critique est activée.

---

3. Optimisation manuelle des plugins Elementor / Jetpack

- Elementor et Jetpack injectent souvent des styles globaux même sur des pages qui n'en ont pas besoin.
- Utiliser un plugin comme Asset CleanUp ou Perfmatters (optionnel) pour :
  - désactiver les CSS/JS plugin par plugin, page par page,
  - supprimer les styles surchargés injectés globalement.

Objectif : ne charger que le strict nécessaire, et éviter les redondances entre les styles du thème, d'Elementor et des modules.

4. Alléger la structure HTML/CSS de Elementor

•Elementor ajoute une couche HTML/CSS relativement lourde.
•Réduire les effets visuels (parallax, overlay, animations) qui demandent beaucoup de CSS/JS pour le rendu initial.
•Privilégier les sections simples pour les éléments “Above the Fold” (au-dessus de la ligne de flottaison).

1.2.4 Réduisez les ressources CSS inutilisées

Très important

État à date

PageSpeed Insights détecte un volume significatif de CSS non utilisé, avec une économie potentielle estimée à 102 Kiosur certaines pages du site. Cela signifie que de nombreux styles sont chargés alors qu’ils ne sont pas utilisés pour le rendu du contenu au-dessus de la ligne de flottaison (above the fold). Cela impacte directement le First Contentful Paint (FCP)et ralentit l’affichage initial.

Problème : Feuilles de style CSS inutilisées chargées globalement

? Ressources concernées :

URL	Taille transférée	Économies potentielles
…/static/becada5b87.min.css?ver=174	107,2 KiB	**101,9 KiB**
…/static/d6ab61b498.min.css?ver=174	15,4 KiB	**14,7 KiB**

Ces feuilles de style sont probablement générées par Elementor ou d’autres composants dynamiques du thème WordPress. Elles sont chargées globalement sur toutes les pages, même lorsque seule une petite portion est réellement utilisée — ce qui entraîne une surcharge CSS inutile, surtout sur mobile.

Solutions recommandées

1. WP Rocket – Supprimer les CSS inutilisés

Ĺ Activer l’option “Supprimer les CSS inutilisés” dans WP Rocket (Remove Unused CSS)

•Cette fonctionnalité permet :
•D’analyser les styles nécessaires au rendu initial pour chaque page.
•De conserver uniquement le CSS utilisé et d’inliner ce CSS critique directement dans le `<head>`.
•De déporter les autres styles au bas de la page ou de les supprimer s’ils ne sont jamais utilisés.

WP Rocket génère une version page par page du CSS, ce qui évite les lourdes feuilles de style globales typiques d'Elementor et des thèmes WordPress.

---

2. Réduction des dépendances CSS côté plugins

- Identifier les plugins injectant du CSS globalement (Jetpack, Elementor, Site Kit, etc.).
- Utiliser un plugin comme Asset CleanUp ou Perfmatters pour :
  - Désactiver les styles CSS plugin par plugin et page par page.
  - Empêcher le chargement des fichiers CSS si la fonctionnalité associée n'est pas utilisée sur la page.

Objectif : n'inclure que ce qui est nécessaire au rendu immédiat de chaque page.

---

3. Optimiser les sections Elementor

- Elementor ajoute des classes utilitaires et des styles dynamiques même sur des widgets inactifs.
- Réduire l'utilisation :
  - de widgets complexes ou animés,
  - des effets de survol ou de transitions CSS qui nécessitent des styles additionnels,
  - des templates avec nombreux conteneurs ou duplications de sections.

Moins d'éléments = moins de styles injectés = meilleure performance CSS.

---

4. Inspection avec Chrome DevTools – Couverture du CSS

- Ouvrir DevTools > Coverage et analyser le CSS téléchargé sur les pages clés.
- Repérer les fichiers avec une majorité de styles non utilisés (en rouge).
- Identifier les sources (plugins, thème, Elementor global, etc.) et envisager :
  - Une exclusion via WP Rocket,
  - Ou une suppression via des optimisations de template.

---

5. Optimisation du thème enfant (le cas échéant)

- Si un thème enfant est utilisé, s'assurer qu'il ne charge pas des CSS redondants.
- Supprimer les imports inutiles ou combiner les fichiers CSS personnalisés dans un fichier unique et minifié.

1.2.5 Diffusez des éléments statiques grâce à des règles de cache efficaces

Important

État à date

PageSpeed Insights indique que 15 ressources statiques du site ne bénéficient pas de règles de cache optimales. Bien que certaines aient déjà une durée de vie de cache de 7 jours, d'autres, comme les fichiers liés à Cloudflare Rocket Loader, ne sont stockées que pour moins d'une heure.

Une configuration de cache trop courte oblige le navigateur à re-télécharger fréquemment des fichiers qui ne changent pourtant jamais, ce qui dégrade les performances pour les visiteurs réguliers et augmente inutilement la consommation de bande passante.

Problème : règles de cache trop courtes sur des ressources statiques

? Ressources concernées (exemples) :

URL	TTL actuel	Taille transférée
`/cloudflare-static/rocket-loader.min.js`	**47 min**	4 KiB
`/cloudflare-static/email-decode.min.js`	**47 min**	1 KiB
`/jquery/jquery.min.js?ver=3.7.1`	**7 jours**	29 KiB
`/js/frontend.min.js?ver=3.28.3`	**7 jours**	13 KiB
`/sharedaddy/sharing.min.js?ver=14.5`	**7 jours**	3 KiB
`/beacon.min.js` (Cloudflare Insights)	**1 jour**	7 KiB

Ces fichiers sont statiques, rarement mis à jour, et peuvent sans risque être mis en cache pour une durée plus longue.

Solutions recommandées

1. Améliorer la politique de cache côté serveur avec WP Rocket et Cloudflare

Ĺ Activer / Vérifier dans WP Rocket :

- L'option "Activer la mise en cache du navigateur" : cela ajoute automatiquement des en-têtes HTTP de type `Cache-Control` pour toutes les ressources statiques.
- Cela permet de définir une politique de cache par défaut de 365 jours (1 an) pour les fichiers `.js`, `.css`, `.jpg`, `.webp`, `.woff`, etc.

? En-tête recommandé :

```
Cache-Control: public, max-age=31536000, immutable
```

Si tu utilises Cloudflare, tu peux également :

- Aller dans Caching > Browser Cache TTL
- Choisir 1 month ou plus pour les ressources statiques.
- Pour une granularité maximale, activer les "Page Rules" pour appliquer ce TTL aux fichiers `.js`, `.css`, `.woff`, `.svg`, etc.

---

2. Exclure Rocket Loader si inutilisé ou non critique

Certains fichiers comme :

`/cloudflare-static/rocket-loader.min.js`

ont un TTL très faible (47 minutes), car Cloudflare ne le considère pas comme une ressource pleinement statique côté utilisateur.

Si Rocket Loader n'apporte pas de gain mesurable sur le site (ce qui est souvent le cas avec Elementor), tu peux envisager de le désactiver dans Cloudflare pour :

- Réduire les conflits JS,
- Supprimer une ressource non cache-friendly.

---

3. Fusion et minification via WP Rocket

- Activer la minification et la combinaison des fichiers CSS et JS permet de réduire le nombre de requêtes et la taille totale à charger.
- Une fois fusionnés, ces fichiers peuvent être mis en cache plus longtemps, puisqu'ils sont générés dynamiquement avec un hash (`ver=xxx`) qui change lors de la mise à jour.

---

4. Pré-chargement opportun des scripts essentiels

- WP Rocket peut automatiquement précharger certains fichiers JS/CSS critiques.
- Tu peux forcer le préchargement de scripts vitaux via des balises :

```
<link rel="preload" as="script" href="/js/frontend.min.js?ver=3.28.3">
```

Cela accélère le chargement initial, sans empêcher le cache long terme.

---

5. Surveillance post-optimisation

- Utiliser PageSpeed Insights + **WebPageTest.org** pour vérifier les nouveaux TTL appliqués.
- Contrôler les en-têtes HTTP via l'onglet Network de DevTools pour t'assurer que la règle `max-age` est bien propagée sur toutes les ressources.

1.2.6 Réduire le temps de réponse initial du serveur

Important

État à date

Sur mobile, le document racine (https://jemeremetsausport.com) met 1 150 ms à répondre. Ce délai correspond au Time to First Byte (TTFB), qui mesure le temps entre la requête de l'utilisateur et la

réception du premier octet de la page HTML. Idéalement, ce délai doit être inférieur à 500 ms pour une performance optimale.

Un TTFB trop long ralentit l'ensemble du processus de chargement : aucun CSS, JS ou image ne peut être téléchargé avant que le HTML initial ne soit reçu. Cela affecte directement le FCP, le LCP, et l'expérience perçue, en particulier pour les visiteurs mobiles ou les zones à faible débit.

Problème : serveur trop lent à répondre

? Données relevées :

URL	TTFB mesuré
`https://jemeremetsausport.com`	**1 150 ms**

Ce délai est élevé pour un site WordPress optimisé, et peut provenir :

- De plugins trop lourds ou inutiles,
- D'un thème complexe ou mal codé (Elementor ajoute une couche importante),
- D'un hébergement mutualisé peu performant,
- D'un manque de cache HTML efficace côté serveur.

Solutions recommandées

1. Améliorer la mise en cache HTML complète

Activer la mise en cache de la page entière avec WP Rocket :

- Vérifie que l'option "Mise en cache des pages" est bien activée.
- WP Rocket crée une version statique HTML de chaque page, stockée et servie instantanément, sans recalcul WordPress/PHP.
- Activer également le cache mobile (version distincte si du contenu dynamique est utilisé via Elementor).

Si le cache est déjà activé, assure-toi qu'il n'est pas contourné par :

- Des cookies de session,
- Des paramètres URL dynamiques,
- Des headers ou conditions serveur.

---

2. Optimiser l'hébergement (Hostinger)

Le site est actuellement hébergé chez Hostinger. Bien que correct pour des projets personnels, il peut rapidement atteindre ses limites sur WordPress + Elementor + Ads + Analytics.

Recommandations :

- Passer à une offre "Business WordPress" ou "Cloud Premium" chez Hostinger.
- Ou migrer vers un hébergement géré spécialisé WordPress (Kinsta, O2Switch Pro+, PlanetHoster, etc.).
- Vérifier l'activation du cache serveur côté Hostinger (LiteSpeed Cache ou NGINX) si disponible.

---

3. Réduire les appels serveur WordPress/PHP

•Désactiver les plugins non utilisés, même s'ils sont inactifs.
•Remplacer les plugins lourds ou multifonctions par des solutions plus légères (ex. éviter Jetpack si peu utilisé).
•Éviter les plugins qui effectuent des requêtes externes à chaque chargement (social, statistiques, etc.).

---

4. Choisir un thème plus léger (si refonte envisagée)

Elementor est puissant mais gourmand. À moyen terme, envisager :

•De passer à un thème basé sur les blocs (Full Site Editing) comme GeneratePress, Blocksy ou Kadence.
•De recréer certaines pages sans Elementor via Gutenberg et des blocs réutilisables.
•Cela réduit le nombre de requêtes PHP + JS nécessaires pour afficher chaque page.

---

5. Surveillance continue du TTFB

•Utiliser GTmetrix, WebPageTest et l'onglet "Network" de Chrome DevTools pour vérifier le TTFB.
•Activer le rapport Core Web Vitals dans Google Search Console pour suivre les impacts sur l'indexation et l'UX.

1.2.7 Éviter les changements de mise en page importants

Important

État à date

Problème : Décalage de mise en page dans l'élément principal

? Élément concerné :

```
<main id="content" class="site-main post-1734 page type-page status-publish has-post-thumbnail hentry" style="height: auto !important;">
```

? Score de décalage de mise en page : 0,235

? Contenu décalé identifié :

CONSEILS POUR SE REMETTRE AU SPORT EN TOUTE CONFIANCE ENTRAINEMENT & EXERCICES

Ce bloc semble subir un changement de position ou de hauteur au chargement, probablement causé

par :

- Des images ou polices Web chargées de manière asynchrone sans dimensions fixes.
- Une animation ou un effet Elementor / CSS qui modifie la hauteur du contenu après le rendu initial.
- Un style CSS ou un JavaScript tardif qui modifie la taille ou la structure du bloc principal.

Solutions recommandées

1. Réserver l'espace pour les images, vidéos et bannières

- Toutes les images (notamment en `<div style="background-image">` ou en `<img>`) doivent inclure des attributs `width` et `height`, ou être contenues dans un bloc avec des dimensions explicites via CSS :

```
.elementor-widget-image img {
  width: 100%;
  height: auto;
  aspect-ratio: 16/9;
}
```

- Si l'image est chargée en `background-image` dans un bloc Elementor (comme pour l'en-tête actuel), définir une hauteur CSS fixe ou min-height, au lieu de `auto` :

```
.site-main {
  min-height: 400px;
}
```

---

2. Charger les polices Web de façon non bloquante

- Utiliser la règle `font-display: swap` dans les fichiers CSS pour éviter que le texte ne reste invisible en attendant les polices personnalisées :

```
@font-face {
  font-family: 'YourFont';
  src: url('font.woff2') format('woff2');
  font-display: swap;
}
```

- Cela évite les décalages de blocs texte à l'affichage.

---

3. Réduire ou supprimer les effets dynamiques Elementor au-dessus de la ligne de flottaison

- Désactiver :
  - Les animations à l'apparition sur les titres, images ou CTA,
  - Les transitions de chargement Elementor (scroll, fade-in, etc.).
- Privilégier des blocs statiques au chargement pour la section d'en-tête principale.

---

4. Vérifier et corriger les styles `height: auto !important;`

- Le style forcé `height: auto !important;` peut provoquer des effets de cascade imprévus, surtout si des éléments sont injectés via JS ou que la taille dépend d'images externes.
- Préférer des hauteurs explicites ou responsives (`min-height`, `vh`, `clamp()`).

---

5. WP Rocket – Optimisation complémentaire

- Vérifier dans WP Rocket que l'option "Optimiser les polices Google" est activée.
- Cela réduit le FOUT (flash de texte non stylisé) qui peut contribuer au CLS.
- Coupler avec "Précharger les polices" si tu identifies celles critiques au design.

Exemple :

```
<link rel="preload" href="<https://fonts.googleapis.com/css2?
family=Montserrat&display=swap>" as="style"
onload="this.onload=null;this.rel='stylesheet'">
```

---

6. Audit visuel des décalages

- Utiliser Chrome DevTools > Lighthouse > Layout Shifts ou l'outil Web Vitals Chrome Extension pour :
  - Identifier visuellement les zones qui bougent au chargement.
  - Appliquer des correctifs CSS ou de structure HTML pour stabiliser le rendu.

2. Corrections techniques

Ce rapport est en lien avec l'analyse technique.

https://docs.google.com/spreadsheets/d/1yOgTp02xdpXsXhSIpgWUmQt4sKrB8IAg/edit?gid=366580973#gid=366580973

1. Audit de l'accessibilité par les moteurs de recherche et les utilisateurs

---

1. État de l'indexation

? Fichier associé :

https://docs.google.com/spreadsheets/d/1daRjSjzAnUUG8BaXM1Tad1FthtO5V3o68SbqP0XYOSQ/edit?gid=1755378798#gid=1755378798

Constat :

•De nombreuses URLs sont en statut “Explorée mais non indexée” ou 404, ce qui indique un problème de qualité ou de pertinence du contenu perçu par Google.
•Cela peut également indiquer un budget crawl gaspillé sur des pages pauvres ou non consolidées.
•Certaines pages sont en “Échoué”, ce qui signifie qu’elles ne sont même pas explorées correctement par Googlebot.

? Fichier complémentaire (pages échouées) :

http://e.com/spreadsheets/d/1daRjSjzAnUUG8BaXM1Tad1FthtO5V3o68SbqP0XYOSQ/edit?gid=1399132644#gid=1399132644

Recommandations :

•Analyser les pages non indexées ou en erreur 404 : sont-elles anciennes, inutiles ou mal maillées ?
•Rediriger ou supprimer les pages 404 si elles ne doivent plus exister.
•Améliorer la qualité des contenus non indexés : enrichissement, réécriture, maillage interne.
•Consolider les pages proches (ex : plusieurs articles sur un même sujet) pour éviter les doublons de faible valeur.

---

2. Fichier robots.txt

Constat :

Le fichier robots.txt est mal configuré, ce qui peut nuire à l’exploration efficace du site par Googlebot.

Recommandation – Exemple de fichier optimisé pour WordPress :

```
User-agent: *
Disallow: /wp-admin/
Disallow: /wp-login.php
Disallow: /?s=
Disallow: */trackback/
Disallow: */feed/
Disallow: */comments/
Disallow: */?*
Allow: /wp-admin/admin-ajax.php

Sitemap: <https://jemeremetsausport.com/sitemap_index.xml>
```

Ce fichier bloque les ressources inutiles à crawler tout en autorisant l’accès aux contenus principaux, et déclare la sitemap XML pour guider Googlebot.

---

3. Pages orphelines

? Fichier associé :

https://docs.google.com/spreadsheets/d/11NyV6rvxb9N1WIhdSgv0Jd_StSz6vxOi/edit?gid=1727018047#gid=1727018047

Constat :

Le site contient un nombre important de pages orphelines, c'est-à-dire non maillées depuis le reste du site. Ces pages sont difficiles à découvrir pour Google et invisibles pour les utilisateurs.

Recommandations :

- Créer un plan de maillage interne depuis les pages piliers (ex. : /exercices-cibles/, /reprendre-le-sport/).
- Intégrer les pages orphelines dans des listes, menus, blocs de navigation contextuelle.
- Prioriser les pages stratégiques et positionnées pour ne pas les laisser isolées.

---

4. Sitemaps XML

Constat :

- Aucune page découverte par le sitemap XML dans la Search Console.
- Cela indique une mauvaise soumission ou configuration du fichier sitemap ou un oubli de déclaration.

Recommandations :

- Vérifier que la sitemap est bien générée (via RankMath, Yoast ou autre).
- Vérifier que l'URL du sitemap est correctement indiquée dans le fichier robots.txt.
- Aller dans Google Search Console > Index > Sitemaps et soumettre à nouveau le sitemap.
- Veiller à ce que le sitemap ne contienne pas d'erreurs (404, noindex, redirections).

2. Audit de l'architecture du site

? Résumé général

L'architecture du site JeMeRemetsAuSport.com souffre actuellement de multiples lacunes structurelles, tant du point de vue de l'expérience utilisateur que de l'optimisation SEO. L'absence de logique de navigation, de hiérarchisation des contenus et de maillage clair entrave le potentiel d'exploration par Google et pénalise l'expérience utilisateur.

---

1. Analyse du fil d'Ariane

Critère		État	Observation
Présence d'un fil d'Ariane	φ	Échoué	Aucune mise en place visible
Position en haut de page	φ	Échoué	Inexistant, donc absent du header
Liens cliquables sur mobile	φ	Échoué	Impossible de remonter facilement dans la hiérarchie
Utilisation des signes « >	φ	Échoué	Non implémenté

Critère	État	Observation
»		
Libellés courts et descriptifs	φ Échoué	Aucun libellé clair car aucun fil affiché
Affichage de la page en cours	φ Échoué	La page active n’est pas mise en valeur dans une logique de hiérarchie

Recommandations :

•Implémenter un fil d’Ariane complet, visible au-dessus du contenu principal, sur desktop et mobile.

•Utiliser une structure claire, hiérarchique et descriptive, du type :

```
Accueil > Exercices ciblés > Épaules > Développé Arnold avec haltères
```

•Rendre chaque niveau cliquable pour améliorer la navigation et renforcer le maillage interne contextuel.

---

2. Navigation principale et structure

Élément analysé	État	Observation
Logo cliquable vers la page d’accueil	¸ Partiellement fonctionnel	Double bouton "Accueil" peu élégant, non UX-friendly
Clarté des libellés de menu	φ Échoué	Les intitulés ne sont pas toujours explicites, double menu peu ergonomique
Structure en silo visible dans le menu	φ Échoué	Aucun silo visible, menu désorganisé
Architecture claire et cohérente	φ Échoué	Aucune logique thématique, URLs à plat, sans arborescence
Mise en valeur de la page active	φ Échoué	La page visitée n’est pas mise en évidence dans le menu
Navigation en pied de page	φ Échoué	Double menu identique à l’en-tête, surcharge inutile

Recommandations :

•Repenser la navigation principale pour y intégrer les 7 grandes catégories définies dans la stratégie (cf. plan stratégique Notion).

•Supprimer le double menu et adopter une navigation claire, fluide, logique.

•Mettre en place une architecture en silo :

Par exemple :

- `/reprendre-le-sport/`
- `/programme-entrainement/`
- `/exercices-cibles/epaules/face-pull/`

•Mettre en évidence la page active dans le menu (ex. couleur différente, surlignage, soulignement).

•Alléger le footer pour n’y laisser que les liens essentiels (infos légales, contact, plan du site, réseaux sociaux).

3. Analyse de la hiérarchie et de la structure interne

Élément audité	État	Observation
Présence de pages peu importantes dans le menu principal	φ Échoué	Des pages secondaires (blog générique, doublons) apparaissent dans la navigation
Canonicalisation des filtres/facettes	ü Non applicable	Pas de navigation à facettes identifiée
Code de réponse des pages paginées	φ Échoué	Plusieurs pages de type pagination retournent des erreurs (non indexées ou 404)
Ordre logique de la pagination	φ Échoué	Désordre constaté, absence de logique séquentielle claire (ex : 1 > 3, ou absence de 2)
Instructions rel="prev/next" correctes	φ Échoué	Instructions absentes ou incorrectes
Canonicalisation des pages paginées (self canonical)	φ Échoué	Mauvais usage, certaines pages paginées pointent vers la page 1
Pages paginées bien en index/follow	φ Échoué	De nombreuses pages sont en noindex ou désindexées
Structure propre des URL paginées	φ Échoué	Mauvaise construction (absence de `?page=2` ou URL dupliquée)

---

4. Problèmes de maillage interne

Élément audité	État	Observation
Pages avec peu ou pas de liens internes	φ Échoué	Nombreuses pages de blog sans maillage interne ou mal liées entre elles
Variété des ancres de liens internes	φ Échoué	Les ancres sont souvent génériques ("cliquez ici", "voir plus") ou peu optimisées sémantiquement

? Fichier associé : Screaming Frog + Analyse manuelle

Référence stratégie maillage interne :

https://www.notion.so/Maillage-interne-1c9c2bd81dab81ba8d58cba521f0d3b9

Recommandations techniques et structurelles

Repenser la hiérarchie du menu principal

•Ne garder dans la navigation principale que les pages stratégiques (catégories pilier, pages à fort volume SEO ou conversions).
•Supprimer les entrées peu informatives ou redondantes du menu (ex. "Accueil", doublons d’articles).
•Ajouter un menu secondaire ou pied de page complémentaire pour les liens moins stratégiques.

Corriger la pagination

•Uniformiser le balisage HTML : utiliser les attributs `rel="prev"` et `rel="next"`.
•Vérifier que chaque page paginée ait une URL propre (type `/page/2`) et une balise canonique autoréférencée.
•Garder les pages paginées en index, follow sauf cas de duplication.
•Respecter une logique séquentielle claire (1 → 2 → 3 → ...).

Optimiser le maillage interne

•Créer un maillage contextuel riche, notamment à la fin des articles (suggestions, "à lire aussi", blocs FAQ interliés).
•Utiliser des ancres optimisées sémantiquement liées à l'intention utilisateur et au mot-clé cible (ex. "exercices pour triceps", "programme 4 jours musculation").
•Relier systématiquement chaque article à sa catégorie principale et à d'autres contenus complémentaires.
•Mettre en place un plan de maillage par silo pour structurer le flux de PageRank interne.

? Pourquoi cette structure est essentielle au SEO ?

•Une mauvaise pagination empêche l'indexation complète du site et peut pénaliser le crawl budget.
•Un maillage interne déficient isole des contenus pourtant intéressants → perte d'autorité et d'opportunités de ranking.
•Une hiérarchie mal pensée dilue le signal de pertinence SEO et nuit à la navigation des utilisateurs.
•Un menu trop riche ou mal ciblé perturbe la lecture des robots comme des visiteurs.

---

5. Audit SEO On-Page

État des lieux global

L'audit SEO On-Page réalisé via Screaming Frog montre un ensemble important d'optimisations manquantes ou incorrectes sur les balises fondamentales (Title, H1, H2, Meta Description), ainsi que sur la gestion des images. Ces erreurs freinent considérablement le CTR (taux de clic), la compréhension sémantique par Google et l'accessibilité du contenu.

Fichier associé :

https://docs.google.com/spreadsheets/d/1daRjSjzAnUUG8BaXM1Tad1FthtO5V3o68SbqP0XYOSQ/edit?gid=375237502#gid=375237502

Optimisation des balises Title & Meta Description

Élément analysé	Statut	Détail
Balises Title > 561 px (≈ 70 caractères)	φ Échoué	Titles trop longs tronqués dans la SERP
Balises Title < 30 caractères	φ Échoué	Titles trop courts et peu explicites

Élément analysé	Statut	Détail
Meta descriptions > 985 px (≈ 155-165 caractères)	φ Échoué	Trop longues, Google les réécrit automatiquement
Meta descriptions < 70 caractères	φ Échoué	Trop courtes pour être impactantes

? Problèmes constatés :

- De nombreux Titles sont non optimisés pour le clic (CTR).
- Manque d'intégration de mots-clés dans les balises.
- Certaines balises Meta sont absentes ou génériques.

Recommandations :

- Réécrire les Titles en respectant une longueur optimale de 50-60 caractères, avec un mot-clé primaire en début de phrase.
- Rédiger des Meta Descriptions engageantes et différenciantes, orientées bénéfice utilisateur.

Optimisation des balises Hn

Élément analysé	Statut	Détail
Balise H1 dupliquée ou identique	φ Échoué	Plusieurs pages partagent la même H1
Balise H1 > 70 caractères	φ Échoué	Trop longue → perte d'impact sémantique
Pages avec plusieurs H1	φ Échoué	Mauvaise hiérarchisation sémantique
Balise H2 manquante ou vide	φ Échoué	Rupture dans la structuration du contenu
Balise H2 dupliquée	φ Échoué	Usage répétitif des mêmes H2
Balise H2 > 70 caractères	φ Échoué	Trop longues, peu efficaces pour le SEO

? Problèmes constatés :

- Le site utilise parfois plusieurs H1, ce qui dilue la thématique principale.
- Des balises H2 sont absentes ou non hiérarchisées correctement.

Recommandations :

- 1 H1 unique par page, intégrant le mot-clé principal.
- Structurer les H2/H3 autour des intentions secondaires ou sous-questions.
- Utiliser les H2 pour organiser logiquement le contenu et faciliter le scan utilisateur/Google.

Optimisation des images

Élément analysé	Statut	Détail
Images > 100 Ko	φ Échoué	Trop lourdes → dégradation du temps de chargement
Noms de fichiers non descriptifs	φ Échoué	Exemple : `image1.jpg`, `screenshot-2023.png`
Noms de fichiers non optimisés (caractères spéciaux, underscores, etc.)	φ Échoué	Présence de majuscules, chiffres, ponctuations
Noms de fichiers trop longs	φ Échoué	Manque d'impact SEO

? Problèmes constatés :

- Images non compressées : impact négatif sur la web performance mobile.
- Mauvais noms de fichiers = opportunité SEO perdue.

Recommandations :

- Compresser toutes les images à <100 Ko (via TinyPNG, ImageOptim, etc.).
- Renommer les fichiers images avec des mots-clés clairs et pertinents, en kebab-case (`exercice-triceps-halteres.jpg`).
- Ajouter systématiquement des balises ALT descriptives, orientées accessibilité + SEO.