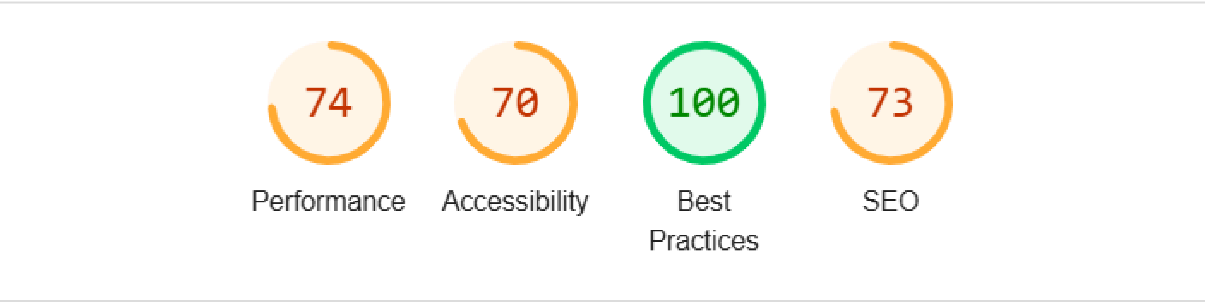
Rapport d’intervention

Nina Carducci

I - Score Lighthouse

Score Lighthouse avant optimisation

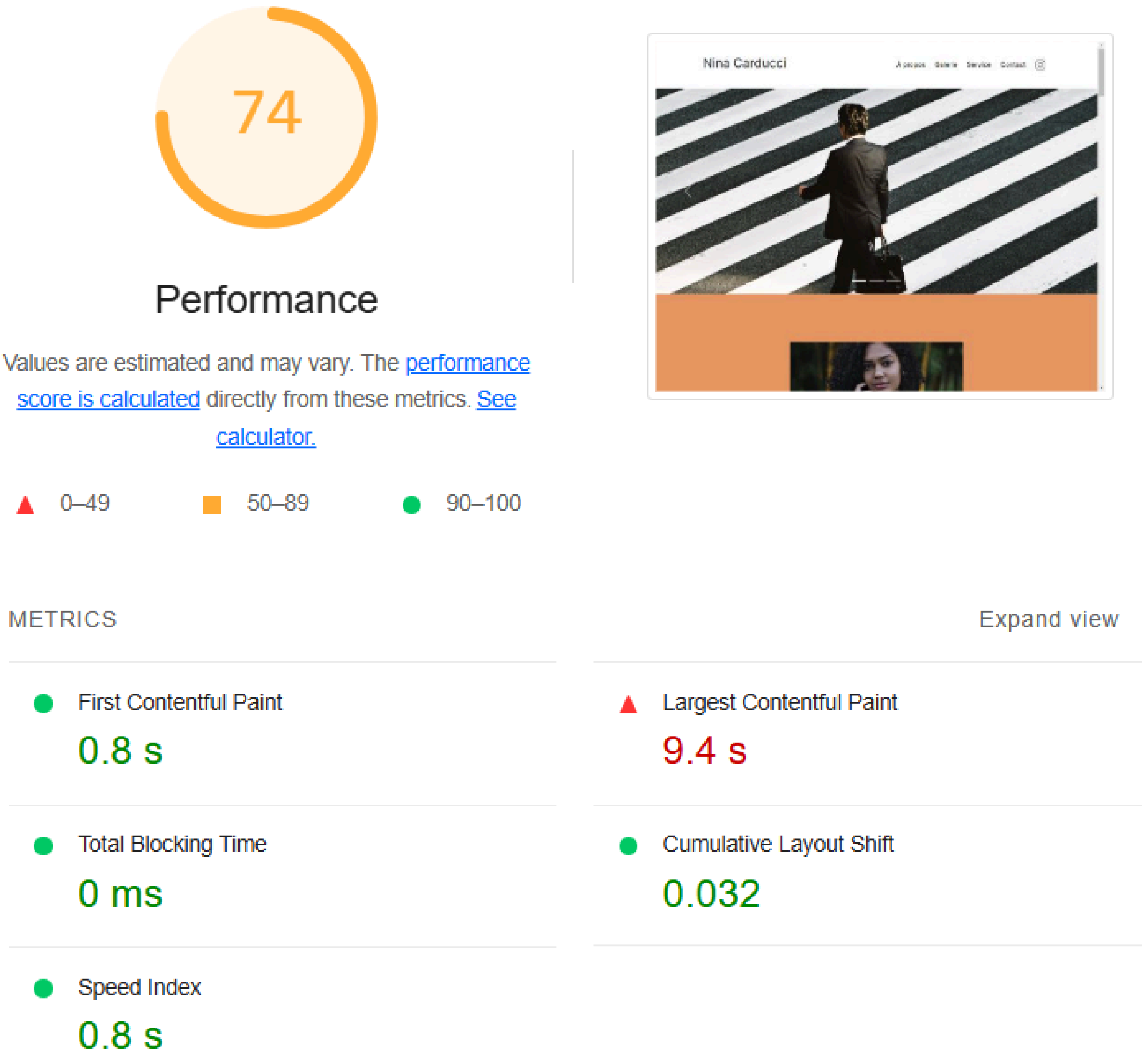


74 Performance

70 Accessibility

100 Best Practices

73 SEO



74

Performance

Values are estimated and may vary. The performance score is calculated directly from these metrics. See calculator.

▲ 0–49 ■ 50–89 ● 90–100

METRICS Expand view

● First Contentful Paint
0.8 s

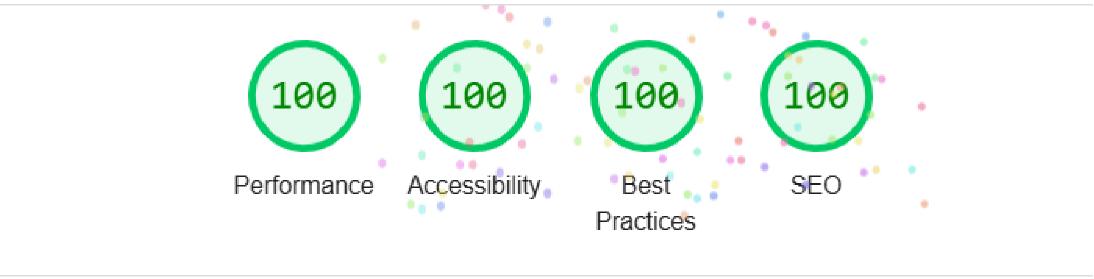
▲ Largest Contentful Paint
9.4 s

● Total Blocking Time
0 ms

● Cumulative Layout Shift
0.032

● Speed Index
0.8 s

Score Lighthouse après optimisation

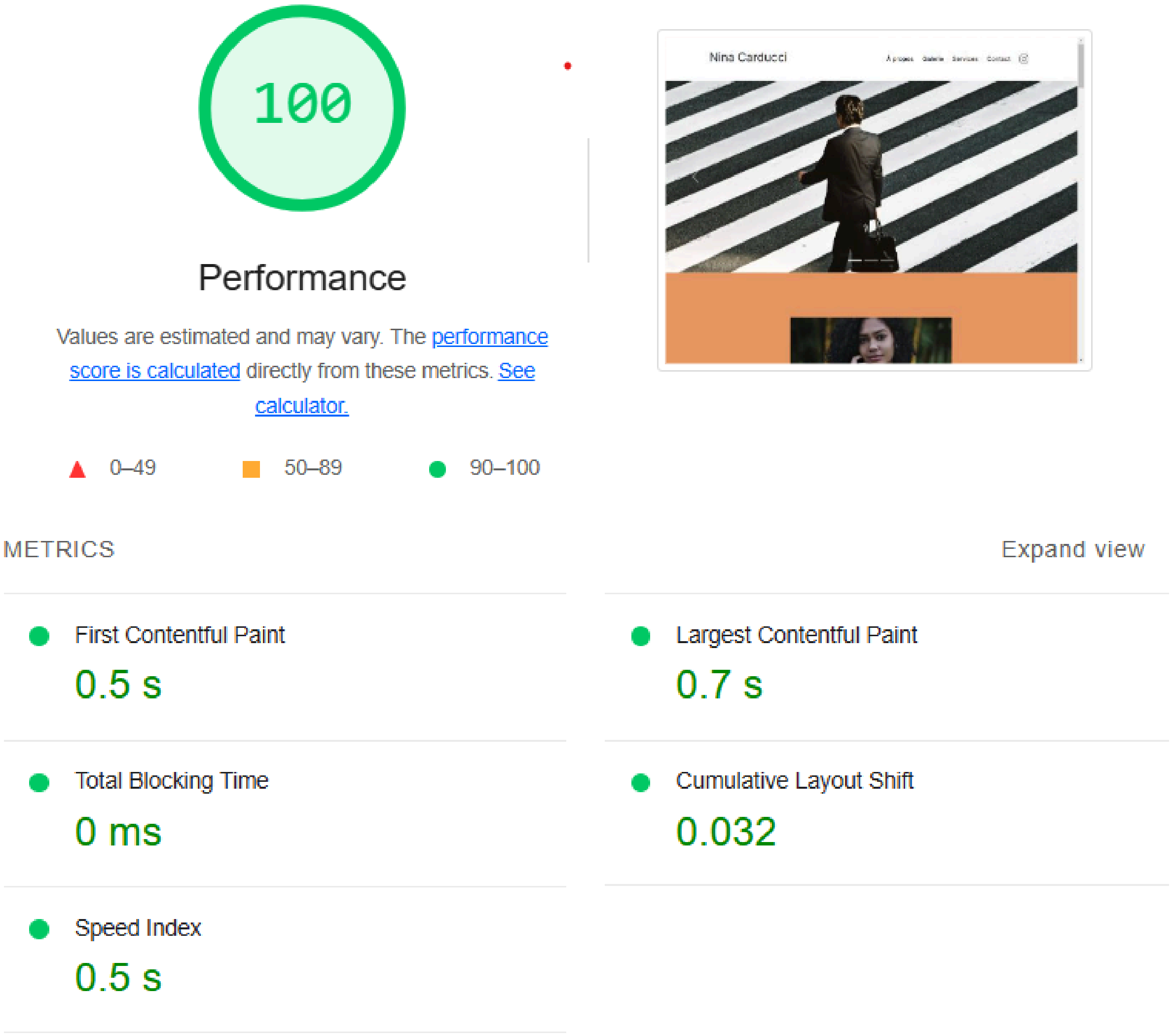


100 Performance

100 Accessibility

100 Best Practices

100 SEO



100

Performance

Values are estimated and may vary. The performance score is calculated directly from these metrics. See calculator.

▲ 0–49 ■ 50–89 ● 90–100

METRICS

Expand view

● First Contentful Paint
0.5 s

● Largest Contentful Paint
0.7 s

● Total Blocking Time
0 ms

● Cumulative Layout Shift
0.032

● Speed Index
0.5 s

II - Détails des optimisations et interventions effectuées

1 - Les images

Le projet comporte originellement 14 images pour un poids total de *30.9* MB. Nous avons effectué les modifications suivantes aux images :

- **Conversion en format webp** : ce format permet de compresser la taille des images en conservant une qualité d’image acceptable.
- **Création de différentes versions des images** pour intégrer des images adaptatives au html. Ainsi, en fonction de la taille de l’écran de l’utilisateur, la photo chargée sera plus ou moins grande. Pour ce faire, nous avons remplacé les balises img par des balises pictures dans le html.

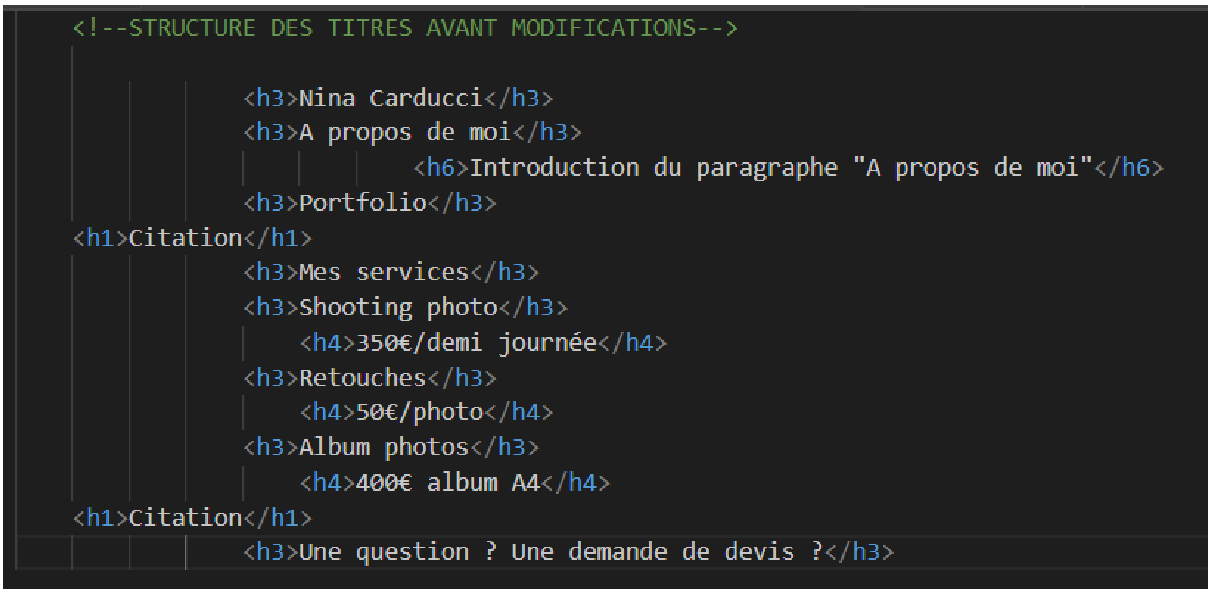
Après ces modifications, le poids total des images est de *314.1 kB*, soit un gain de *98,98* %.

2 - Correction de la structure html

Voici les éléments que nous avons corrigé et amélioré dans la structure du html :

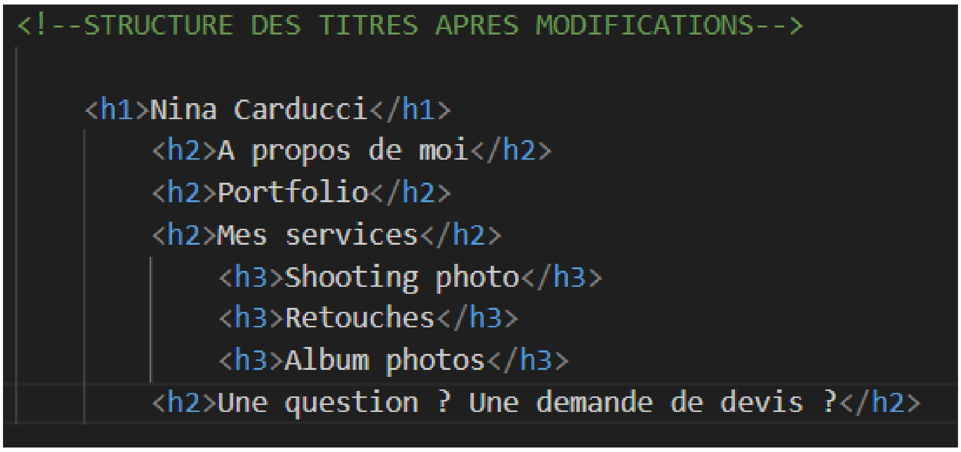
- **Réorganisation des titres** : le titre principal du site était en h3, nous l’avons corrigé en h1. Le h1 doit être unique car il indique le sujet principal du site aux utilisateurs et aux moteurs de recherche.

  Nous avons également corrigé le reste des titres dont la structure n’était pas cohérente (cf. structure ci-dessous). Il est important de respecter la hiérarchie des titres pour que l’organisation du contenu soit clair pour l’utilisateur et les moteurs de recherche.



```
<!--STRUCTURE DES TITRES AVANT MODIFICATIONS-->

      <h3>Nina Carducci</h3>
      <h3>A propos de moi</h3>
            <h6>Introduction du paragraphe "A propos de moi"</h6>
      <h3>Portfolio</h3>
<h1>Citation</h1>
      <h3>Mes services</h3>
      <h3>Shooting photo</h3>
        <h4>350€/demi journée</h4>
      <h3>Retouches</h3>
        <h4>50€/photo</h4>
      <h3>Album photos</h3>
        <h4>400€ album A4</h4>
<h1>Citation</h1>
      <h3>Une question ? Une demande de devis ?</h3>
```



```
<!--STRUCTURE DES TITRES APRES MODIFICATIONS-->

    <h1>Nina Carducci</h1>
        <h2>A propos de moi</h2>
        <h2>Portfolio</h2>
        <h2>Mes services</h2>
            <h3>Shooting photo</h3>
            <h3>Retouches</h3>
            <h3>Album photos</h3>
        <h2>Une question ? Une demande de devis ?</h2>
```

Ces corrections ont un impact sur l’expérience utilisateur, sur le SEO et également sur l’accessibilité car les lecteurs d’écran naviguent sur les sites via les titres.

- **Correction et ajout de balises sémantiques** :
    - Ajout de la balise **header** : elle permet d'identifier de manière claire l’en-tête de la page, qui comprend généralement les éléments principaux de navigation, le titre et le logo.
      Note : La div du carrousel avait d’ailleurs “header” pour nom de classe, ce qui nous paraissait incohérent. Nous avons donc supprimé cette div qui était inutile.
    - Ajout de la balise **nav** : elle englobe les sections de liens de navigation du site et se situent généralement dans le header et le footer.
    - Ajout de la balise **main** : elle englobe le contenu principal du site.
    - Ajout des balises **section**, **article** et **aside** : elles permettent de structurer le contenu principal du site.
      Les sections indiquent les thématiques principales du site et permettent de de donner une structure logique à la page.
      Les articles sont des contenus autonomes qui peut être géré indépendamment du reste de la page. Dans notre cas, nous avons ajouté ces balises pour les services proposés.
      Les aside contiennent généralement des informations complémentaires ou secondaires. C’est ici le cas des citations.

L’utilisation de ces balises sémantiques améliore l’organisation et la clarté du code pour l’utilisateur, les moteurs de recherche et les technologies d’assistance. Elle a donc un impact direct sur l’expérience utilisateur, l’indexation de notre site donc le SEO, et l’accessibilité.

3 - Mise en place du référencement local

Nous avons effectué plusieurs ajustements pour améliorer le référencement local :

- **Ajout de mots clés géographiques dans les balises** title, meta description ainsi que dans les balises Open Graph (cf. partie 4 - Ajout des balises meta) : ces balises comprennent désormais les mots-clés Bordeaux et Nouvelle-Aquitaine, ce qui indique aux moteurs de recherche et aux réseaux sociaux que le contenu est pertinent pour les recherches locales.
- Ajout de **données structurées Schema.org** : elles incluent des données locales spécifiques comme l’adresse, la région desservie et les informations de contact. Cela permet aux moteurs de recherche d’afficher les informations localisées dans les résultats de recherche et d’augmenter ses chances de voir son site apparaître dans les résultats de recherche des utilisateurs situés à proximité.

Globalement, ces actions renforcent le positionnement de l’entreprise en local.

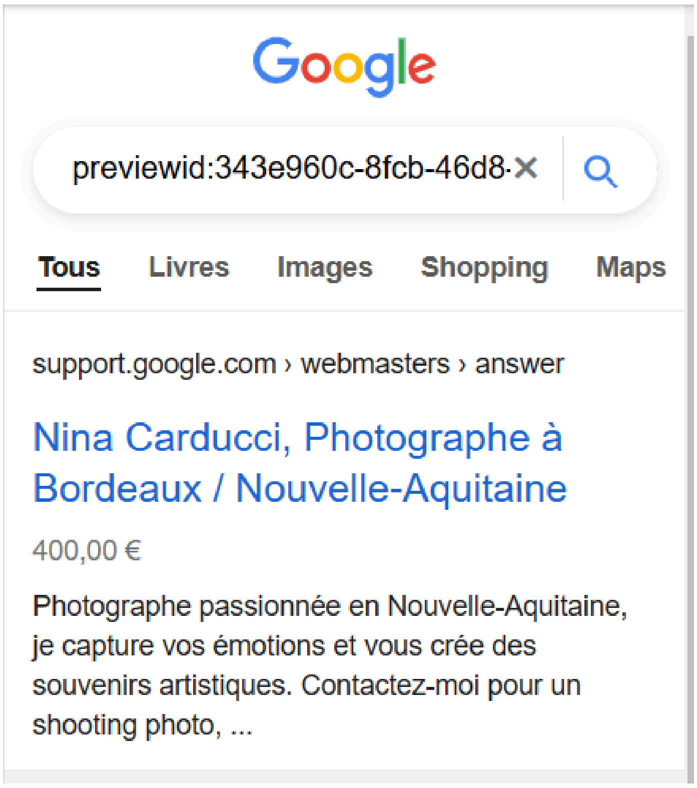
4 - Ajout des balises meta

Le code source comprenait déjà deux balises méta indispensables. La balise meta-charset, qui définit l’encodage des caractères, et la balise meta viewport qui contrôle la mise en page sur les mobiles. Ces deux balises ont un impact direct sur l’affichage de notre page.

Nous avons ajouté les balises suivantes :

- **Meta description** : c’est la courte description de la page qui apparaît généralement dans les résultats de recherche. Une bonne description influencera forcément le taux de visites sur le site.
- **Meta robots** : ces balises indiquent aux moteurs de recherche comment indexer votre page. Dans notre cas, nous avons demandé aux moteurs de recherche de ne pas conserver de version en cache de notre page (nocache pour Internet explorer et noarchive pour les autres navigateurs), sans limite d’affichage de snippets (cf. balises JSON-LD) et d’afficher les images en taille standard dans les résultats de recherche.
- **Meta language** : on spécifie la langue de la page. Cela peut aider les technologies d’assistance, les moteurs de recherche et les navigateurs.
- **Open Graph** : ce sont les balises qui sont utilisées pour optimiser l’affichage des partages sur les réseaux sociaux. Nous avons donc défini via ces balises le titre, la description, l’image et l’URL qui s’afficheront sur les réseaux.
- **Meta twitter** : ces balises permettent de configurer l’affichage des informations de notre entreprise sous forme de carte lors des partages sur twitter.
- **JSON-LD** : il s’agit des données structurées (rich snippet) schema.org. Nous avons déjà abordé la première balise de ce type dans la partie référencement local. Nous en avons ajouté trois autres qui sont des snippet de produits et de services. Cela permettra d’afficher dans les résultats de recherche les produits et services que vous proposez avec titre, description et prix.

Voici un aperçu de la forme que peut prendre le résultat d’une recherche suite à la mise en place de contenu structuré (aperçu via google Rich Snippet) :



support.google.com › webmasters › answer

Nina Carducci, Photographe à Bordeaux / Nouvelle-Aquitaine

400,00 €

Photographe passionnée en Nouvelle-Aquitaine, je capture vos émotions et vous crée des souvenirs artistiques. Contactez-moi pour un shooting photo, ...

5 - Chargement différé

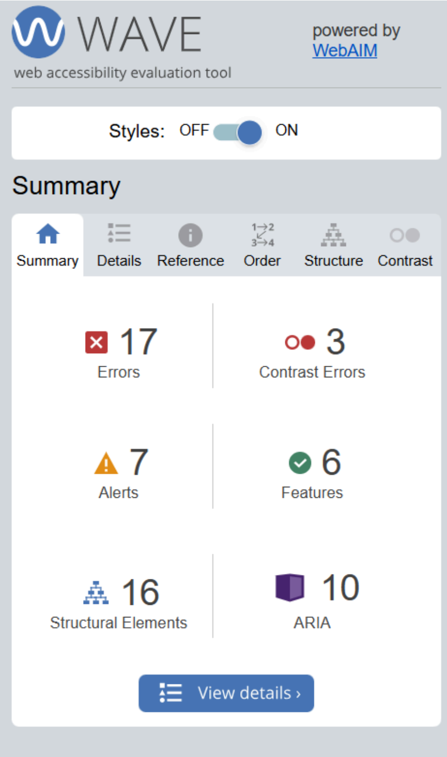
Les scripts javascript étaient initialement chargés en même temps que la page html. Cela peut créer des blocages de rendu, il est donc de bonne pratique de différer le chargement des javascripts. On s’assure que tout le contenu html soit bien affiché avant de charger les scripts. Nous avons donc placé les scripts en fin de body.

6 - Minification des fichiers

Pour réduire le temps de chargement et ainsi améliorer les performances du site, nous avons minifié les fichiers bootstrap.bundle.js et maugallery.js. Les fichiers plus petits seront chargés plus rapidement (important pour les utilisateurs mobiles ou dont la connexion est plus lente) et l’expérience utilisateur sera ainsi améliorée.

III - Accessibilité du site

Accessibilité avant optimisation



Styles: OFF ON

Summary

17 Errors

3 Contrast Errors

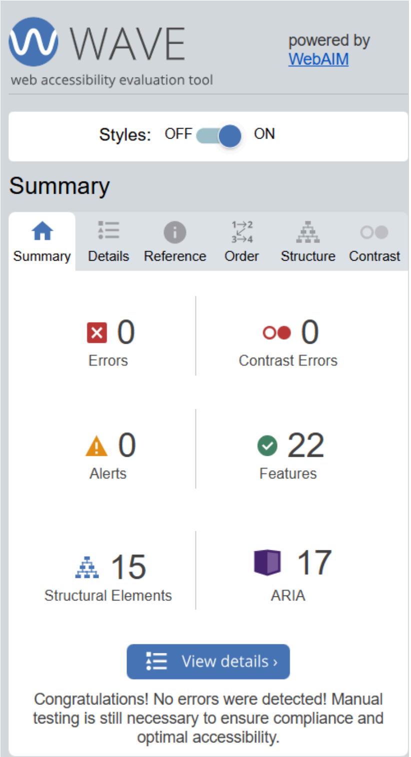
7 Alerts

6 Features

16 Structural Elements

10 ARIA

Accessibilité après optimisation



Styles: OFF ON

Summary

0 Errors

0 Contrast Errors

0 Alerts

22 Features

15 Structural Elements

17 ARIA

Congratulations! No errors were detected! Manual testing is still necessary to ensure compliance and optimal accessibility.

Modifications effectuées

- Ajout de descriptions sur les photos avec l’attribut **alt** : permet aux utilisateurs qui ne peuvent pas voir l’image d’avoir une description de l’image. Elle est également utile pour les moteurs de recherche qui se servent de cet attribut pour indexer les images.

- Ajout d’un **label aux champs de formulaire** de contact : cela permet aux lecteurs d’écran d’identifier clairement les champs de saisie.

- **Correction de contrastes** : cela rend la page plus lisible, moins fatigante à lire et accessible aux personnes ayant des déficiences visuelles. Nous avons corrigé la couleur du texte des filtres, qui étaient blancs lorsqu’ils étaient actifs.

- **Ajout de lang=fr au head du html** : c’est un élément indispensable pour indiquer aux moteurs de recherche et lecteurs d’écran la langue principale du site. Cela aide également les outils de traduction.

- **Ajout d’attributs aria-label** : il y avait déjà des attributs aria-label sur le site, nous en avons ajouté quelques-uns pour améliorer davantage l’accessibilité pour les utilisateurs de technologies d’assistance.

Ces changements contribuent à rendre le site plus performant et plus inclusif en le rendant plus compréhensible pour les utilisateurs, qu’ils aient un handicap ou non. Par ailleurs, les descriptions générées par les attributs alt et label auront un impact positif sur le référencement du site auprès des moteurs de recherche.

IV - Détails de réalisations additionnelles à la demande du client

1 - Correction du bug de navigation dans la modale

La navigation dans la modale de la galerie entre image précédente et suivante ne fonctionnait pas. Nous avons trouvé la cause du problème dans le fichier maugallery.js :
Dans les fonctions prevImage et nextImage, les images précédentes ou suivantes sont déterminées par la variable **next**.
Voici la construction de la variable dans le fichier initial pour la fonction prevImage :

```
      next =
        imagesCollection[index] ||
        imagesCollection[imagesCollection.length - 1];
```

Si on traduit, cela veut dire que l’image suivante qui va s’afficher doit être l’image de la collection qui a le même index que l’image active (imagesCollection[index]) , ou s’il ne la trouve pas, la dernière de la collection (imagesCollection[imagesCollection.length - 1) pour créer une boucle. Or cette fonction ne fonctionne pas puisqu’elle trouve systématiquement l’image active. Ce que l’on souhaite, c’est afficher l’image dont l’index précède l’image active. Nous avons donc corrigé la fonction comme suit pour régler le problème :

```
      next =
        imagesCollection[index -1] ||
        imagesCollection[imagesCollection.length - 1];
```

Le problème était le même dans la fonction nextImage. La variable était définie comme cela :

```
next = imagesCollection[index] || imagesCollection[0];
```

Or nous voulons que l’image affichée soit celle avec l’index supérieur. On corrige donc de cette manière :

```
 next = imagesCollection[index +1] || imagesCollection[0];
```

2 - Problème d’affichage des filtres actifs

Vous souhaitez que le filtre actif de catégorie ait un fond de couleur doré comme pour le filtre par défaut. Dans la version initiale, le fond apparaît bien sur le filtre Tous au chargement de la page. Puis lorsque l’on clique sur les autres filtres, il disparaît pour ne jamais réapparaître nulle part.

Nous avons également trouvé la source du problème dans le fichier maugallery.js, ainsi que dans le style CSS :
Dans la fonction filterByTag du fichier initial, le filtre Tous avait deux classes par défaut : la classe *active* et la classe *active-tag*.
Voici le fonctionnement de la fonction de filtre : on commence par vérifier si l’élément cliqué a une class *active-tag*. Si c’est le cas, cela veut dire que le filtre cliqué est le filtre déjà actif. Si ce n’est pas le filtre actif, on supprime de l’élément actif les classes *active* et *active-tag*, puis on ajoute une classe active-tag à l’élément cliqué.
Nous pointons déjà une incohérence puisque le code supprime deux classes mais n’en rajoute qu’une des deux au clic. Il s’avère que la classe active n’est pas nécessaire, nous l’avons donc supprimé du code.
Par ailleurs, la mise en forme du filtre était appliquée sur la classe *active*, cela ne pouvait donc pas fonctionner. Nous avons donc modifié le fichier CSS et appliqué cette mise en forme aux éléments ayant la classe *active-tag*.

V - Cahier de recette

Détail des fonctionnalités débuggées et de leur statut :

ID	Action	Résultat initial	Résultat attendu	Statut	Remarques et commentaires
1	Naviguer entre les images de la galerie avec les flèches précédent et suivant	Rien ne se passe au clic	Navigation fonctionnelle	À faire / Résolu	Résolu
2	Problème d’affichage des filtres actifs	Au clic sur les filtres, pas de fond doré sur le filtre actif	Le fond doré apparaît sur le filtre sélectionné. Si aucun filtre n’est sélectionné, le filtre Tous a ce fond par défaut		Résolu

Annexe

Rapport complet de l’audit Lighthouse

Nina Carducci

https://heloiseqsn.github.io/HeloiseQsn-Nina-Carducci-Dev/

100 Performance

100 Accessibility

100 Best Practices

100 SEO

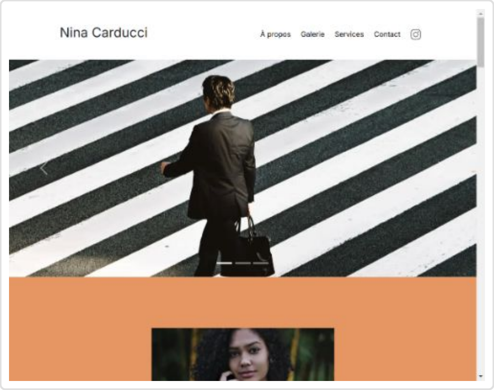
SI FCP LCP TBT CLS

100

Performance

Values are estimated and may vary. The performance score is calculated directly from these metrics. See calculator.

▲ 0–49 50–89 90–100



METRICS Expand view

First Contentful Paint
0.5 s

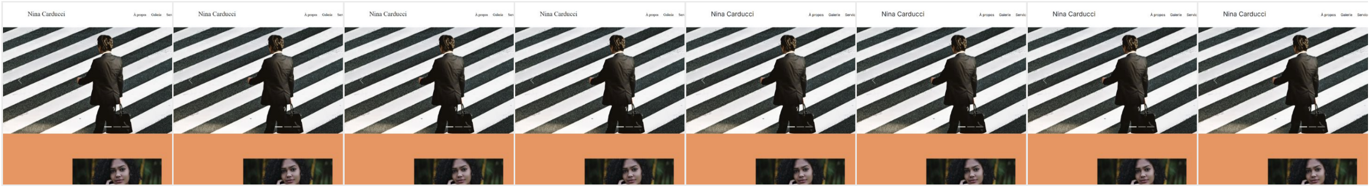
Largest Contentful Paint
0.7 s

Total Blocking Time
0 ms

Cumulative Layout Shift
0.032

Speed Index
0.5 s

View Treemap



Show audits relevant to: All FCP LCP TBT CLS

DIAGNOSTICS

▲ Eliminate render-blocking resources — Potential savings of 240 ms

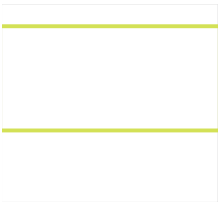
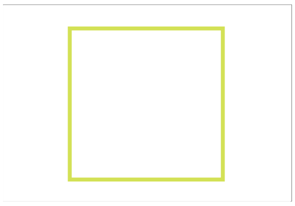
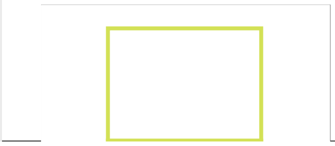
Resources are blocking the first paint of your page. Consider delivering critical JS/CSS inline and deferring all non-critical JS/styles. Learn how to eliminate render-blocking resources. FCP LCP

☑ Show 3rd-party resources (2)

URL	Transfer Size	Potential Savings
GitHub Utility 1st Party	**58.0 KiB**	**160 ms**
…bootstrap/bootstrap.css (heloiseqsn.github.io)	26.6 KiB	80 ms
…bootstrap/bootstrap.bundle.js (heloiseqsn.github.io)	31.3 KiB	80 ms
Google Fonts Cdn	**1.3 KiB**	**230 ms**
/css2?family=… (fonts.googleapis.com)	1.3 KiB	230 ms
jQuery CDN Cdn	**30.2 KiB**	**320 ms**
/jquery-3.4.1.min.js (code.jquery.com)	30.2 KiB	320 ms

Image elements do not have explicit `width` and `height`

Set an explicit width and height on image elements to reduce layout shifts and improve CLS. Learn how to set image dimensions CLS

		URL
GitHub Utility 1st Party		
	img.d-block.w-100	…slider/ryoji-iwata-1200.webp (heloiseqsn.github.io)
	img	…images/photo-Nin….webp (heloiseqsn.github.io)
	img	…images/photo-camera-400.webp (heloiseqsn.github.io)

	URL
img	…images/instagram.png (heloiseqsn.github.io)

Minify CSS — Potential savings of 5 KiB

Minifying CSS files can reduce network payload sizes. Learn how to minify CSS. FCP LCP

URL	Transfer Size	Potential Savings
GitHub Utility 1st Party	**26.6 KiB**	**5.2 KiB**
…bootstrap/bootstrap.css (heloiseqsn.github.io)	26.6 KiB	5.2 KiB

Minify JavaScript — Potential savings of 7 KiB

Minifying JavaScript files can reduce payload sizes and script parse time. Learn how to minify JavaScript. FCP LCP

URL	Transfer Size	Potential Savings
`chrome-extension://amaaokahonnfjjemodnpmeenfpnnbkco/content_page2.js`	24.3 KiB	6.6 KiB

Serve static assets with an efficient cache policy — 20 resources found

A long cache lifetime can speed up repeat visits to your page. Learn more about efficient cache policies.

URL	Cache TTL	Transfer Size
GitHub Utility 1st Party		**352 KiB**
…slider/ryoji-iwata-1200.webp (heloiseqsn.github.io)	10m	81 KiB
…slider/nicholas-green-1200.webp (heloiseqsn.github.io)	10m	47 KiB

URL	Cache TTL	Transfer Size
…slider/edward-cisneros-1200.webp (heloiseqsn.github.io)	10m	46 KiB
…bootstrap/bootstrap.bundle.js (heloiseqsn.github.io)	10m	31 KiB
…bootstrap/bootstrap.css (heloiseqsn.github.io)	10m	27 KiB
…concerts/photo-concerts-aaron-paul-320.webp (heloiseqsn.github.io)	10m	21 KiB
…images/photo-Nin….webp (heloiseqsn.github.io)	10m	16 KiB
…mariage/photo-mariages-jakob-owens-320.webp (heloiseqsn.github.io)	10m	15 KiB
…portraits/photo-portraits-nino-van-prattenburg-320.webp (heloiseqsn.github.io)	10m	13 KiB
…portraits/photo-portraits-ade-tunji-320.webp (heloiseqsn.github.io)	10m	12 KiB
…images/photo-camera-400.webp (heloiseqsn.github.io)	10m	9 KiB
…entreprise/photo-entreprise-mateus-campos-felipe-320.webp (heloiseqsn.github.io)	10m	8 KiB
…concerts/photo-concerts-austin-neill-320.webp (heloiseqsn.github.io)	10m	7 KiB
…entreprise/photo-entreprise-jason-goodman-320.webp (heloiseqsn.github.io)	10m	6 KiB
…mariage/photo-mariages-hannah-busing-320.webp (heloiseqsn.github.io)	10m	5 KiB
…entreprise/photo-entreprise-ali-morshedlou-320.webp (heloiseqsn.github.io)	10m	5 KiB
…assets/maugallery.js (heloiseqsn.github.io)	10m	2 KiB
…assets/style.css (heloiseqsn.github.io)	10m	2 KiB
…images/instagram.png (heloiseqsn.github.io)	10m	1 KiB
…assets/scripts.js (heloiseqsn.github.io)	10m	1 KiB

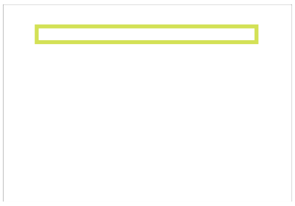
Reduce unused CSS — Potential savings of 10 KiB

Reduce unused rules from stylesheets and defer CSS not used for above-the-fold content to decrease bytes consumed by network activity. Learn how to reduce unused CSS. FCP LCP

URL	Transfer Size	Potential Savings
GitHub Utility 1st Party	**10.5 KiB**	**10.1 KiB**
…bootstrap/bootstrap.css (heloiseqsn.github.io)	10.5 KiB	10.1 KiB

○ Avoid large layout shifts — 2 layout shifts found

These are the largest layout shifts observed on the page. Each table item represents a single layout shift, and shows the element that shifted the most. Below each item are possible root causes that led to the layout shift. Some of these layout shifts may not be included in the CLS metric value due to windowing. Learn how to improve CLS CLS

Element		Layout shift score
div#about-me		0.031
 nav		0.000
…v13/rnCu-xNNw….woff2 (fonts.gstatic.com)	Web font loaded	
…v18/UcCO3FwrK….woff2 (fonts.gstatic.com)	Web font loaded	
…assets/maugallery.js (heloiseqsn.github.io)	A late network request adjusted the page layout	
…assets/scripts.js (heloiseqsn.github.io)	A late network request adjusted the page layout	
/jquery-3.4.1.min.js (code.jquery.com)	A late network request adjusted the page layout	
/css2?family=… (fonts.googleapis.com)	A late network request adjusted the page layout	
…assets/style.css (heloiseqsn.github.io)	A late network request adjusted the page layout	
…bootstrap/bootstrap.bundle.js (heloiseqsn.github.io)	A late network request adjusted the page layout	
…bootstrap/bootstrap.css (heloiseqsn.github.io)	A late network request adjusted the page layout	

○ Initial server response time was short — Root document took 20 ms

Keep the server response time for the main document short because all other requests depend on it. Learn more about the Time to First Byte metric. FCP LCP

URL	Time Spent
GitHub Utility 1st Party	**20 ms**
/HeloiseQsn-Nina-Carducci-Dev/ (heloiseqsn.github.io)	20 ms

○ Avoids enormous network payloads — Total size was 456 KiB

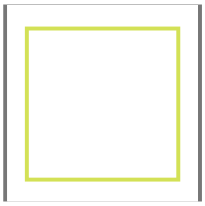
Large network payloads cost users real money and are highly correlated with long load times. Learn how to reduce payload sizes.

☑ Show 3rd-party resources (3)

URL	Transfer Size
GitHub Utility 1st Party	**268.9 KiB**
…slider/ryoji-iwata-1200.webp (heloiseqsn.github.io)	81.4 KiB
…slider/nicholas-green-1200.webp (heloiseqsn.github.io)	46.9 KiB
…slider/edward-cisneros-1200.webp (heloiseqsn.github.io)	46.1 KiB
…bootstrap/bootstrap.bundle.js (heloiseqsn.github.io)	31.3 KiB
…bootstrap/bootstrap.css (heloiseqsn.github.io)	26.6 KiB
…concerts/photo-concerts-aaron-paul-320.webp (heloiseqsn.github.io)	20.5 KiB
…images/photo-Nin….webp (heloiseqsn.github.io)	16.0 KiB
Google Fonts Cdn	**39.2 KiB**
…v18/UcCO3FwrK….woff2 (fonts.gstatic.com)	23.8 KiB
…v13/rnCu-xNNw….woff2 (fonts.gstatic.com)	15.4 KiB
jQuery CDN Cdn	**30.2 KiB**
/jquery-3.4.1.min.js (code.jquery.com)	30.2 KiB

○ Avoids an excessive DOM size — 190 elements

A large DOM will increase memory usage, cause longer style calculations, and produce costly layout reflows. Learn how to avoid an excessive DOM size. TBT

Statistic	Element	Value
Total DOM Elements		190
Maximum DOM Depth	div.mg-prev	10
Maximum Child Elements	 div.gallery-items-row.row	9

○ Avoid chaining critical requests — 10 chains found

The Critical Request Chains below show you what resources are loaded with a high priority. Consider reducing the length of chains, reducing the download size of resources, or deferring the download of unnecessary resources to improve page load. Learn how to avoid chaining critical requests.

Maximum critical path latency: **216.424 ms**

Initial Navigation

- /HeloiseQsn-Nina-Carducci-Dev/ (heloiseqsn.github.io)
  - …bootstrap/bootstrap.css (heloiseqsn.github.io) **- 21.903 ms, 26.64 KiB**
  - …assets/style.css (heloiseqsn.github.io) **- 32.938 ms, 1.73 KiB**
  - /css2?family=… (fonts.googleapis.com)
    - …v18/UcCO3FwrK….woff2 (fonts.gstatic.com) **- 34.169 ms, 23.78 KiB**
    - …v13/rnCu-xNNw….woff2 (fonts.gstatic.com) **- 38.19 ms, 14.58 KiB**
    - …v13/rnCu-xNNw….woff2 (fonts.gstatic.com) **- 47.236 ms, 15.38 KiB**
    - …v13/rnCr-xNNw….woff2 (fonts.gstatic.com) **- 42.687 ms, 13.57 KiB**
  - …bootstrap/bootstrap.bundle.js (heloiseqsn.github.io) **- 22.13 ms, 31.32 KiB**
  - /jquery-3.4.1.min.js (code.jquery.com) **- 45.377 ms, 30.25 KiB**
  - …assets/maugallery.js (heloiseqsn.github.io) **- 18.198 ms, 1.95 KiB**
  - …assets/scripts.js (heloiseqsn.github.io) **- 18.238 ms, 0.59 KiB**

○ JavaScript execution time — 0.0 s

Consider reducing the time spent parsing, compiling, and executing JS. You may find delivering smaller JS payloads helps with this. Learn how to reduce Javascript execution time. TBT

☑ Show 3rd-party resources (1)

URL	Total CPU Time	Script Evaluation	Script Parse
GitHub Utility 1st Party	**127 ms**	**10 ms**	**0 ms**

URL	Total CPU Time	Script Evaluation	Script Parse
/HeloiseQsn-Nina-Carducci-Dev/ (heloiseqsn.github.io)	127 ms	10 ms	0 ms
jQuery CDN Cdn	**67 ms**	**37 ms**	**1 ms**
/jquery-3.4.1.min.js (code.jquery.com)	67 ms	37 ms	1 ms
Unattributable	**51 ms**	**2 ms**	**0 ms**
`Unattributable`	51 ms	2 ms	0 ms

○ Minimizes main-thread work — 0.3 s

Consider reducing the time spent parsing, compiling and executing JS. You may find delivering smaller JS payloads helps with this. Learn how to minimize main-thread work TBT

Category	Time Spent
Other	100 ms
Style & Layout	84 ms
Script Evaluation	59 ms
Parse HTML & CSS	10 ms
Rendering	10 ms
Script Parsing & Compilation	6 ms

○ Minimize third-party usage — Third-party code blocked the main thread for 0 ms

Third-party code can significantly impact load performance. Limit the number of redundant third-party providers and try to load third-party code after your page has primarily finished loading. Learn how to minimize third-party impact. TBT

Third-Party	Transfer Size	Main-Thread Blocking Time
Google Fonts Cdn	**69 KiB**	**0 ms**
…v18/UcCO3FwrK….woff2 (fonts.gstatic.com)	24 KiB	0 ms
…v13/rnCu-xNNw….woff2 (fonts.gstatic.com)	15 KiB	0 ms
…v13/rnCu-xNNw….woff2 (fonts.gstatic.com)	15 KiB	0 ms

Third-Party	Transfer Size	Main-Thread Blocking Time
…v13/rnCr-xNNw….woff2 (fonts.gstatic.com)	14 KiB	0 ms
/css2?family=… (fonts.googleapis.com)	1 KiB	0 ms
jQuery CDN Cdn	**30 KiB**	**0 ms**
/jquery-3.4.1.min.js (code.jquery.com)	30 KiB	0 ms

○ Largest Contentful Paint element — 670 ms

This is the largest contentful element painted within the viewport. Learn more about the Largest Contentful Paint element LCP

Element
img.d-block.w-100

Phase	% of LCP	Timing
TTFB	26%	170 ms
Load Delay	0%	0 ms
Load Time	11%	70 ms
Render Delay	63%	430 ms

More information about the performance of your application. These numbers don't directly affect the Performance score.

PASSED AUDITS (23) Hide

Properly size images

Serve images that are appropriately-sized to save cellular data and improve load time. Learn how to size images. FCP LCP

Defer offscreen images

Consider lazy-loading offscreen and hidden images after all critical resources have finished loading to lower time to interactive. Learn how to defer offscreen images. FCP LCP

Reduce unused JavaScript

Reduce unused JavaScript and defer loading scripts until they are required to decrease bytes consumed by network activity. Learn how to reduce unused JavaScript. FCP LCP

Efficiently encode images

Optimized images load faster and consume less cellular data. Learn how to efficiently encode images. FCP LCP

Serve images in next-gen formats

Image formats like WebP and AVIF often provide better compression than PNG or JPEG, which means faster downloads and less data consumption. Learn more about modern image formats. FCP LCP

Enable text compression

Text-based resources should be served with compression (gzip, deflate or brotli) to minimize total network bytes. Learn more about text compression. FCP LCP

Preconnect to required origins

Consider adding `preconnect` or `dns-prefetch` resource hints to establish early connections to important third-party origins. Learn how to preconnect to required origins. LCP FCP

Avoid multiple page redirects

Redirects introduce additional delays before the page can be loaded. Learn how to avoid page redirects. LCP FCP

Use HTTP/2

HTTP/2 offers many benefits over HTTP/1.1, including binary headers and multiplexing. Learn more about HTTP/2. LCP FCP

Use video formats for animated content

Large GIFs are inefficient for delivering animated content. Consider using MPEG4/WebM videos for animations and PNG/WebP for static images instead of GIF to save network bytes. Learn more about efficient video formats FCP LCP

Remove duplicate modules in JavaScript bundles

Remove large, duplicate JavaScript modules from bundles to reduce unnecessary bytes consumed by network activity. FCP LCP

Avoid serving legacy JavaScript to modern browsers

Polyfills and transforms enable legacy browsers to use new JavaScript features. However, many aren't necessary for modern browsers. For your bundled JavaScript, adopt a modern script deployment strategy using module/nomodule feature detection to reduce the amount of code shipped to modern browsers, while retaining support for legacy browsers. Learn how to use modern JavaScript FCP LCP

Preload Largest Contentful Paint image

If the LCP element is dynamically added to the page, you should preload the image in order to improve LCP. Learn more about preloading LCP elements. LCP

○ User Timing marks and measures

Consider instrumenting your app with the User Timing API to measure your app's real-world performance during key user experiences. Learn more about User Timing marks.

All text remains visible during webfont loads

Leverage the `font-display` CSS feature to ensure text is user-visible while webfonts are loading. Learn more about `font-display`.

○ Lazy load third-party resources with facades

Some third-party embeds can be lazy loaded. Consider replacing them with a facade until they are required. Learn how to defer third-parties with a facade. TBT

Largest Contentful Paint image was not lazily loaded

Above-the-fold images that are lazily loaded render later in the page lifecycle, which can delay the largest contentful paint. Learn more about optimal lazy loading. LCP

Element	
	img.d-block.w-100

Uses passive listeners to improve scrolling performance

Consider marking your touch and wheel event listeners as `passive` to improve your page's scroll performance. Learn more about adopting passive event listeners.

Avoids `document.write()`

For users on slow connections, external scripts dynamically injected via `document.write()` can delay page load by tens of seconds. Learn how to avoid document.write().

○ Avoid long main-thread tasks

Lists the longest tasks on the main thread, useful for identifying worst contributors to input delay. Learn how to avoid long main-thread tasks TBT

○ Avoid non-composited animations

Animations which are not composited can be janky and increase CLS. Learn how to avoid non-composited animations CLS

Has a `<meta name="viewport">` tag with `width` or `initial-scale`

A `<meta name="viewport">` not only optimizes your app for mobile screen sizes, but also prevents a 300 millisecond delay to user input. Learn more about using the viewport meta tag.

Page didn't prevent back/forward cache restoration

Many navigations are performed by going back to a previous page, or forwards again. The back/forward cache (bfcache) can speed up these return navigations. Learn more about the bfcache

100

Accessibility

These checks highlight opportunities to improve the accessibility of your web app. Automatic detection can only detect a subset of issues and does not guarantee the accessibility of your web app, so manual testing is also encouraged.

ADDITIONAL ITEMS TO MANUALLY CHECK (10) Hide

○ Interactive controls are keyboard focusable

Custom interactive controls are keyboard focusable and display a focus indicator. Learn how to make custom controls focusable.

○ Interactive elements indicate their purpose and state

Interactive elements, such as links and buttons, should indicate their state and be distinguishable from non-interactive elements. Learn how to decorate interactive elements with affordance hints.

○ The page has a logical tab order

Tabbing through the page follows the visual layout. Users cannot focus elements that are offscreen. Learn more about logical tab ordering.

○ Visual order on the page follows DOM order

DOM order matches the visual order, improving navigation for assistive technology. Learn more about DOM and visual ordering.

○ User focus is not accidentally trapped in a region

A user can tab into and out of any control or region without accidentally trapping their focus. Learn how to avoid focus traps.

○ The user's focus is directed to new content added to the page

If new content, such as a dialog, is added to the page, the user's focus is directed to it. Learn how to direct focus to new content.

○ HTML5 landmark elements are used to improve navigation

Landmark elements (`<main>`, `<nav>`, etc.) are used to improve the keyboard navigation of the page for assistive technology. Learn more about landmark elements.

○ Offscreen content is hidden from assistive technology

Offscreen content is hidden with display: none or aria-hidden=true. Learn how to properly hide offscreen content.

○ Custom controls have associated labels

Custom interactive controls have associated labels, provided by aria-label or aria-labelledby. Learn more about custom controls and labels.

○ Custom controls have ARIA roles

Custom interactive controls have appropriate ARIA roles. Learn how to add roles to custom controls.

These items address areas which an automated testing tool cannot cover. Learn more in our guide on conducting an accessibility review.

PASSED AUDITS (26) Hide

`[aria-*]` attributes match their roles

Each ARIA `role` supports a specific subset of `aria-*` attributes. Mismatching these invalidates the `aria-*` attributes. Learn how to match ARIA attributes to their roles.

`[aria-hidden="true"]` is not present on the document `<body>`

Assistive technologies, like screen readers, work inconsistently when `aria-hidden="true"` is set on the document `<body>`. Learn how `aria-hidden` affects the document body.

`[role]`s have all required `[aria-*]` attributes

Some ARIA roles have required attributes that describe the state of the element to screen readers. Learn more about roles and required attributes.

`[aria-*]` attributes have valid values

Assistive technologies, like screen readers, can't interpret ARIA attributes with invalid values. Learn more about valid values for ARIA attributes.

`[aria-*]` attributes are valid and not misspelled

Assistive technologies, like screen readers, can't interpret ARIA attributes with invalid names. Learn more about valid ARIA attributes.

Buttons have an accessible name

When a button doesn't have an accessible name, screen readers announce it as "button", making it unusable for users who rely on screen readers. Learn how to make buttons more accessible.

Image elements have `[alt]` attributes

Informative elements should aim for short, descriptive alternate text. Decorative elements can be ignored with an empty alt attribute. Learn more about the `alt` attribute.

Input buttons have discernible text.

Adding discernable and accessible text to input buttons may help screen reader users understand the purpose of the input button. Learn more about input buttons.

`[user-scalable="no"]` is not used in the `<meta name="viewport">` element and the `[maximum-scale]` attribute is not less than 5.

Disabling zooming is problematic for users with low vision who rely on screen magnification to properly see the contents of a web page. Learn more about the viewport meta tag.

ARIA attributes are used as specified for the element's role

Some ARIA attributes are only allowed on an element under certain conditions. Learn more about conditional ARIA attributes.

`[aria-hidden="true"]` elements do not contain focusable descendents

Focusable descendents within an `[aria-hidden="true"]` element prevent those interactive elements from being available to users of assistive technologies like screen readers. Learn how `aria-hidden` affects focusable elements.

Elements use only permitted ARIA attributes

Using ARIA attributes in roles where they are prohibited can mean that important information is not communicated to users of assistive technologies. Learn more about prohibited ARIA roles.

`[role]` values are valid

ARIA roles must have valid values in order to perform their intended accessibility functions. Learn more about valid ARIA roles.

Background and foreground colors have a sufficient contrast ratio

Low-contrast text is difficult or impossible for many users to read. Learn how to provide sufficient color contrast.

Document has a `<title>` element

The title gives screen reader users an overview of the page, and search engine users rely on it heavily to determine if a page is relevant to their search. Learn more about document titles.

`<html>` element has a `[lang]` attribute

If a page doesn't specify a `lang` attribute, a screen reader assumes that the page is in the default language that the user chose when setting up the screen reader. If the page isn't actually in the default language, then the screen reader might not announce the page's text correctly. Learn more about the `lang` attribute.

`<html>` element has a valid value for its `[lang]` attribute

Specifying a valid BCP 47 language helps screen readers announce text properly. Learn how to use the `lang` attribute.

Form elements have associated labels

Labels ensure that form controls are announced properly by assistive technologies, like screen readers. Learn more about form element labels.

Links have a discernible name

Link text (and alternate text for images, when used as links) that is discernible, unique, and focusable improves the navigation experience for screen reader users. Learn how to make links accessible.

Lists contain only `<li>` elements and script supporting elements (`<script>` and `<template>`).

Screen readers have a specific way of announcing lists. Ensuring proper list structure aids screen reader output. Learn more about proper list structure.

List items (`<li>`) are contained within `<ul>`, `<ol>` or `<menu>` parent elements

Screen readers require list items (`<li>`) to be contained within a parent `<ul>`, `<ol>` or `<menu>` to be announced properly. Learn more about proper list structure.

Touch targets have sufficient size and spacing.

Touch targets with sufficient size and spacing help users who may have difficulty targeting small controls to activate the targets. Learn more about touch targets.

Heading elements appear in a sequentially-descending order

Properly ordered headings that do not skip levels convey the semantic structure of the page, making it easier to navigate and understand when using assistive technologies. Learn more about heading order.

Uses ARIA roles only on compatible elements

Many HTML elements can only be assigned certain ARIA roles. Using ARIA roles where they are not allowed can interfere with the accessibility of the web page. Learn more about ARIA roles.

Deprecated ARIA roles were not used

Deprecated ARIA roles may not be processed correctly by assistive technology. Learn more about deprecated ARIA roles.

Image elements do not have `[alt]` attributes that are redundant text.

Informative elements should aim for short, descriptive alternative text. Alternative text that is exactly the same as the text adjacent to the link or image is potentially confusing for screen reader users, because the text will be read twice. Learn more about the `alt` attribute.

NOT APPLICABLE (31)

Hide

○ `[accesskey]` values are unique

Access keys let users quickly focus a part of the page. For proper navigation, each access key must be unique. Learn more about access keys.

○ `button`, `link`, and `menuitem` elements have accessible names

When an element doesn't have an accessible name, screen readers announce it with a generic name, making it unusable for users who rely on screen readers. Learn how to make command elements more accessible.

○ Elements with `role="dialog"` or `role="alertdialog"` have accessible names.

ARIA dialog elements without accessible names may prevent screen readers users from discerning the purpose of these elements. Learn how to make ARIA dialog elements more accessible.

○ ARIA input fields have accessible names

When an input field doesn't have an accessible name, screen readers announce it with a generic name, making it unusable for users who rely on screen readers. Learn more about input field labels.

○ ARIA `meter` elements have accessible names

When a meter element doesn't have an accessible name, screen readers announce it with a generic name, making it unusable for users who rely on screen readers. Learn how to name `meter` elements.

○ ARIA `progressbar` elements have accessible names

When a `progressbar` element doesn't have an accessible name, screen readers announce it with a generic name, making it unusable for users who rely on screen readers. Learn how to label `progressbar` elements.

○ Elements with an ARIA `[role]` that require children to contain a specific `[role]` have all required children.

Some ARIA parent roles must contain specific child roles to perform their intended accessibility functions. Learn more about roles and required children elements.

○ `[role]`s are contained by their required parent element

Some ARIA child roles must be contained by specific parent roles to properly perform their intended accessibility functions. Learn more about ARIA roles and required parent element.

○ Elements with the `role=text` attribute do not have focusable descendents.

Adding `role=text` around a text node split by markup enables VoiceOver to treat it as one phrase, but the element's focusable descendents will not be announced. Learn more about the `role=text` attribute.

○ ARIA toggle fields have accessible names

When a toggle field doesn't have an accessible name, screen readers announce it with a generic name, making it unusable for users who rely on screen readers. Learn more about toggle fields.

○ ARIA `tooltip` elements have accessible names

When a tooltip element doesn't have an accessible name, screen readers announce it with a generic name, making it unusable for users who rely on screen readers. Learn how to name `tooltip` elements.

○ ARIA `treeitem` elements have accessible names

When a `treeitem` element doesn't have an accessible name, screen readers announce it with a generic name, making it unusable for users who rely on screen readers. Learn more about labeling `treeitem` elements.

○ The page contains a heading, skip link, or landmark region

Adding ways to bypass repetitive content lets keyboard users navigate the page more efficiently. Learn more about bypass blocks.

○ `<dl>`'s contain only properly-ordered `<dt>` and `<dd>` groups, `<script>`, `<template>` or `<div>` elements.

When definition lists are not properly marked up, screen readers may produce confusing or inaccurate output. Learn how to structure definition lists correctly.

○ Definition list items are wrapped in `<dl>` elements

Definition list items (`<dt>` and `<dd>`) must be wrapped in a parent `<dl>` element to ensure that screen readers can properly announce them. Learn how to structure definition lists correctly.

○ ARIA IDs are unique

The value of an ARIA ID must be unique to prevent other instances from being overlooked by assistive technologies. Learn how to fix duplicate ARIA IDs.

○ No form fields have multiple labels

Form fields with multiple labels can be confusingly announced by assistive technologies like screen readers which use either the first, the last, or all of the labels. Learn how to use form labels.

○ `<frame>` or `<iframe>` elements have a title

Screen reader users rely on frame titles to describe the contents of frames. Learn more about frame titles.

○ `<html>` element has an `[xml:lang]` attribute with the same base language as the `[lang]` attribute.

If the webpage does not specify a consistent language, then the screen reader might not announce the page's text correctly. Learn more about the `lang` attribute.

○ `<input type="image">` elements have `[alt]` text

When an image is being used as an `<input>` button, providing alternative text can help screen reader users understand the purpose of the button. Learn about input image alt text.

○ Links are distinguishable without relying on color.

Low-contrast text is difficult or impossible for many users to read. Link text that is discernible improves the experience for users with low vision. Learn how to make links distinguishable.

○ The document does not use `<meta http-equiv="refresh">`

Users do not expect a page to refresh automatically, and doing so will move focus back to the top of the page. This may create a frustrating or confusing experience. Learn more about the refresh meta tag.

○ `<object>` elements have alternate text

Screen readers cannot translate non-text content. Adding alternate text to `<object>` elements helps screen readers convey meaning to users. Learn more about alt text for `object` elements.

○ Select elements have associated label elements.

Form elements without effective labels can create frustrating experiences for screen reader users. Learn more about the `select` element.

○ Skip links are focusable.

Including a skip link can help users skip to the main content to save time. Learn more about skip links.

○ No element has a `[tabindex]` value greater than 0

A value greater than 0 implies an explicit navigation ordering. Although technically valid, this often creates frustrating experiences for users who rely on assistive technologies. Learn more about the `tabindex` attribute.

○ Tables have different content in the summary attribute and `<caption>`.

The summary attribute should describe the table structure, while `<caption>` should have the onscreen title. Accurate table mark-up helps users of screen readers. Learn more about summary and caption.

○ Cells in a `<table>` element that use the `[headers]` attribute refer to table cells within the same table.

Screen readers have features to make navigating tables easier. Ensuring `<td>` cells using the `[headers]` attribute only refer to other cells in the same table may improve the experience for screen reader users. Learn more about the `headers` attribute.

○ `<th>` elements and elements with `[role="columnheader"/"rowheader"]` have data cells they describe.

Screen readers have features to make navigating tables easier. Ensuring table headers always refer to some set of cells may improve the experience for screen reader users. Learn more about table headers.

○ `[lang]` attributes have a valid value

Specifying a valid BCP 47 language on elements helps ensure that text is pronounced correctly by a screen reader. Learn how to use the `lang` attribute.

○ `<video>` elements contain a `<track>` element with `[kind="captions"]`

When a video provides a caption it is easier for deaf and hearing impaired users to access its information. Learn more about video captions.

100

Best Practices

TRUST AND SAFETY

○ Ensure CSP is effective against XSS attacks

A strong Content Security Policy (CSP) significantly reduces the risk of cross-site scripting (XSS) attacks. Learn how to use a CSP to prevent XSS

Description	Directive	Severity
No CSP found in enforcement mode		High

GENERAL

○ Detected JavaScript libraries

All front-end JavaScript libraries detected on the page. Learn more about this JavaScript library detection diagnostic audit.

Name	Version
Bootstrap	5.1.3
jQuery	3.4.1

PASSED AUDITS (14)

Hide

Uses HTTPS

All sites should be protected with HTTPS, even ones that don't handle sensitive data. This includes avoiding mixed content, where some resources are loaded over HTTP despite the initial request being served over HTTPS. HTTPS prevents intruders from tampering with or passively listening in on the communications between your app and your users, and is a prerequisite for HTTP/2 and many new web platform APIs. Learn more about HTTPS.

Avoids deprecated APIs

Deprecated APIs will eventually be removed from the browser. Learn more about deprecated APIs.

Avoids third-party cookies

Chrome is moving towards a new experience that allows users to choose to browse without third-party cookies. Learn more about third-party cookies.

Allows users to paste into input fields

Preventing input pasting is a bad practice for the UX, and weakens security by blocking password managers.Learn more about user-friendly input fields.

Avoids requesting the geolocation permission on page load

Users are mistrustful of or confused by sites that request their location without context. Consider tying the request to a user action instead. Learn more about the geolocation permission.

Avoids requesting the notification permission on page load

Users are mistrustful of or confused by sites that request to send notifications without context. Consider tying the request to user gestures instead. Learn more about responsibly getting permission for notifications.

Displays images with correct aspect ratio

Image display dimensions should match natural aspect ratio. Learn more about image aspect ratio.

Serves images with appropriate resolution

Image natural dimensions should be proportional to the display size and the pixel ratio to maximize image clarity. Learn how to provide responsive images.

Has a `<meta name="viewport">` tag with `width` or `initial-scale`

A `<meta name="viewport">` not only optimizes your app for mobile screen sizes, but also prevents a 300 millisecond delay to user input. Learn more about using the viewport meta tag.

Page has the HTML doctype

Specifying a doctype prevents the browser from switching to quirks-mode. Learn more about the doctype declaration.

Properly defines charset

A character encoding declaration is required. It can be done with a `<meta>` tag in the first 1024 bytes of the HTML or in the Content-Type HTTP response header. Learn more about declaring the character encoding.

No browser errors logged to the console

Errors logged to the console indicate unresolved problems. They can come from network request failures and other browser concerns. Learn more about this errors in console diagnostic audit

No issues in the `Issues` panel in Chrome Devtools

Issues logged to the `Issues` panel in Chrome Devtools indicate unresolved problems. They can come from network request failures, insufficient security controls, and other browser concerns. Open up the Issues panel in Chrome DevTools for more details on each issue.

Page has valid source maps

Source maps translate minified code to the original source code. This helps developers debug in production. In addition, Lighthouse is able to provide further insights. Consider deploying source maps to take advantage of these benefits. Learn more about source maps.

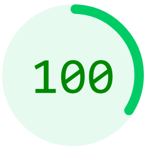
NOT APPLICABLE (2) Hide

○ Redirects HTTP traffic to HTTPS

Make sure that you redirect all HTTP traffic to HTTPS in order to enable secure web features for all your users. Learn more.

○ Document uses legible font sizes

Font sizes less than 12px are too small to be legible and require mobile visitors to “pinch to zoom” in order to read. Strive to have >60% of page text ≥12px. Learn more about legible font sizes.



SEO

These checks ensure that your page is following basic search engine optimization advice. There are many additional factors Lighthouse does not score here that may affect your search ranking, including performance on Core Web Vitals. Learn more about Google Search Essentials.

ADDITIONAL ITEMS TO MANUALLY CHECK (1)

Hide

○ Structured data is valid

Run the Structured Data Testing Tool and the Structured Data Linter to validate structured data. Learn more about Structured Data.

Run these additional validators on your site to check additional SEO best practices.

PASSED AUDITS (8)

Hide

Page isn’t blocked from indexing

Search engines are unable to include your pages in search results if they don't have permission to crawl them. Learn more about crawler directives.

Document has a `<title>` element

The title gives screen reader users an overview of the page, and search engine users rely on it heavily to determine if a page is relevant to their search. Learn more about document titles.

Document has a meta description

Meta descriptions may be included in search results to concisely summarize page content. Learn more about the meta description.

Page has successful HTTP status code

Pages with unsuccessful HTTP status codes may not be indexed properly. Learn more about HTTP status codes.

Links have descriptive text

Descriptive link text helps search engines understand your content. Learn how to make links more accessible.

Links are crawlable

Search engines may use `href` attributes on links to crawl websites. Ensure that the `href` attribute of anchor elements links to an appropriate destination, so more pages of the site can be discovered. Learn how to make links crawlable

Image elements have `[alt]` attributes

Informative elements should aim for short, descriptive alternate text. Decorative elements can be ignored with an empty alt attribute. Learn more about the `alt` attribute.

Document has a valid `hreflang`

hreflang links tell search engines what version of a page they should list in search results for a given language or region. [Learn more about `hreflang`].

NOT APPLICABLE (2) Hide

○ robots.txt is valid

If your robots.txt file is malformed, crawlers may not be able to understand how you want your website to be crawled or indexed. [Learn more about robots.txt].

○ Document has a valid `rel=canonical`

Canonical links suggest which URL to show in search results. [Learn more about canonical links].

Captured at Oct 25, 2024, 2:23 PM GMT+2
Initial page load

Emulated Desktop with Lighthouse 12.2.1
Custom throttling

Single page session

Using Chromium 130.0.0.0 with devtools

Generated by **Lighthouse** 12.2.1 | File an issue