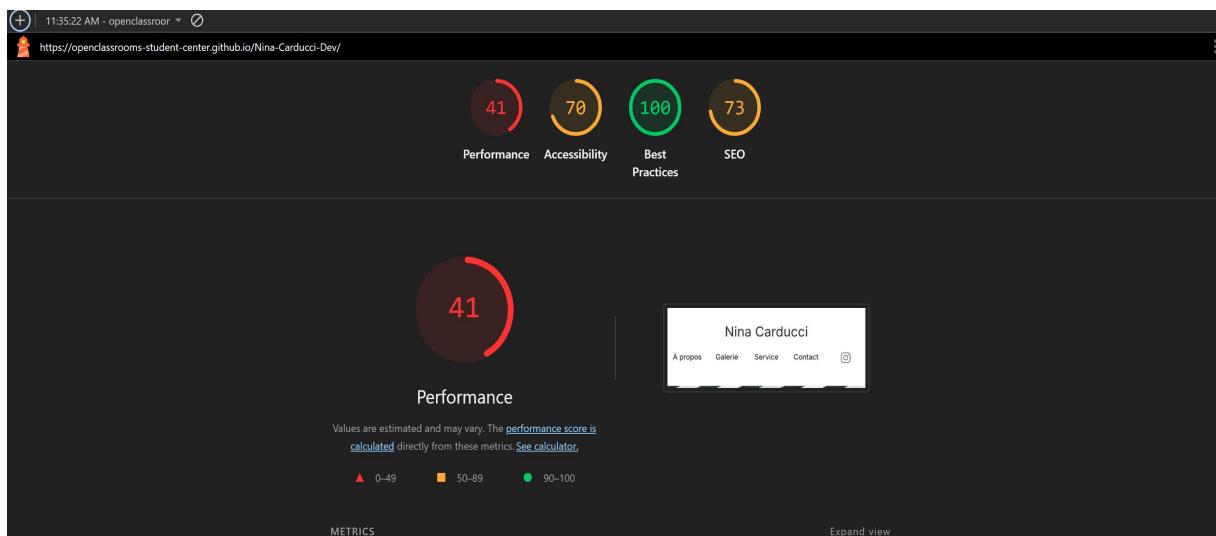


# *Intervention Report-*

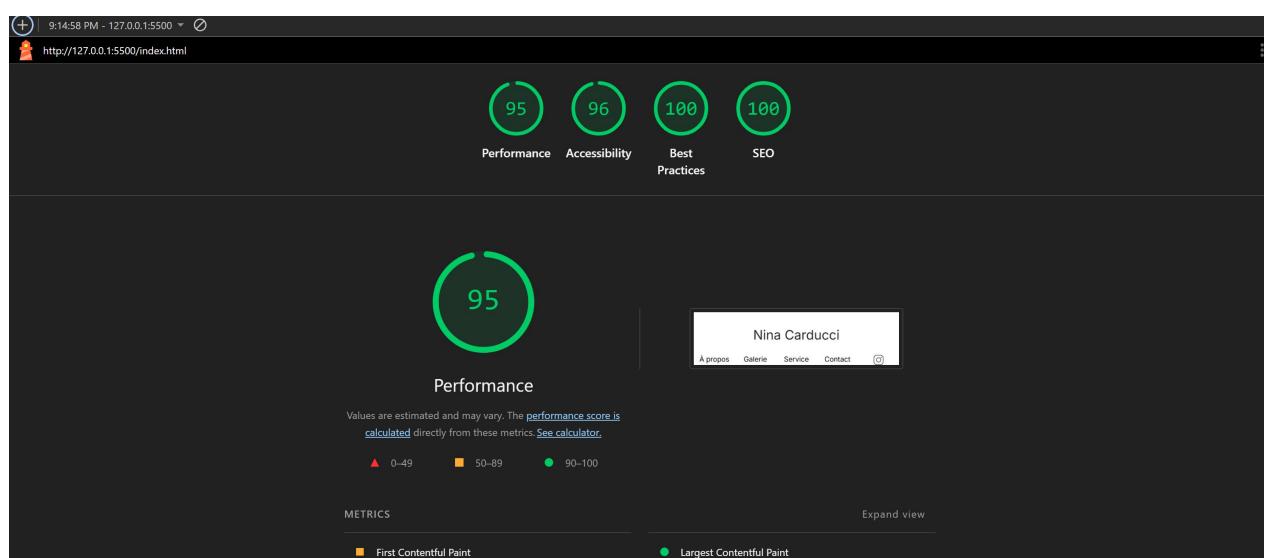
## *Nina Carducci*

### Part One: Nina Carducci Google Lighthouse Scores

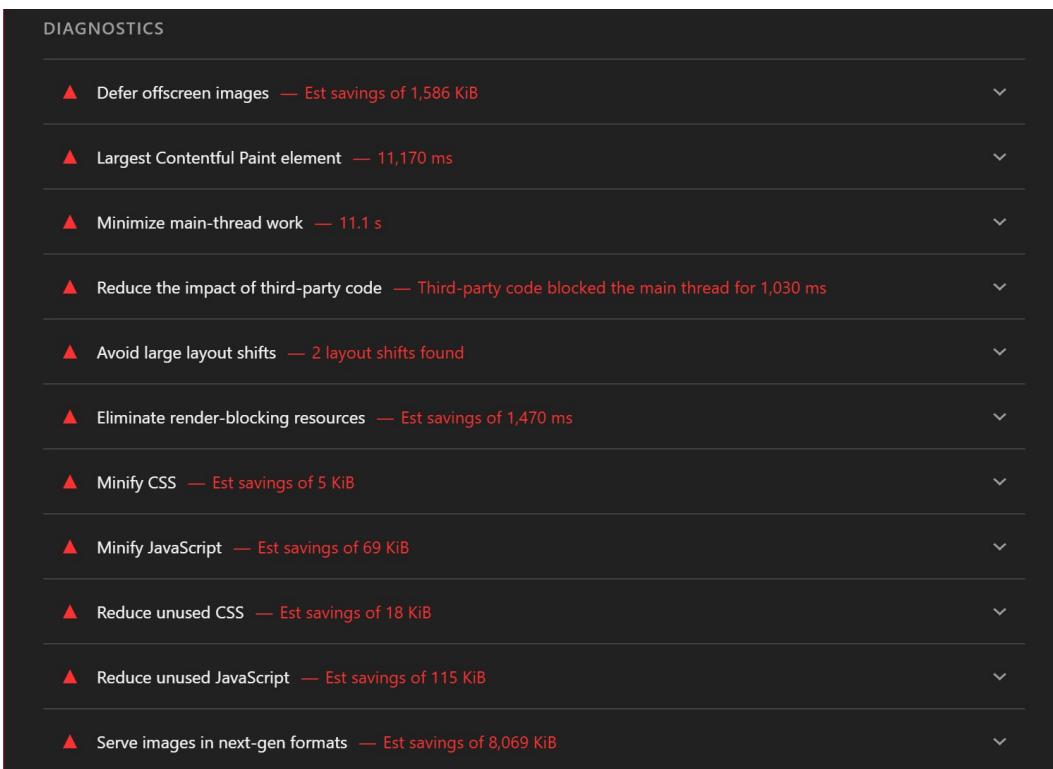
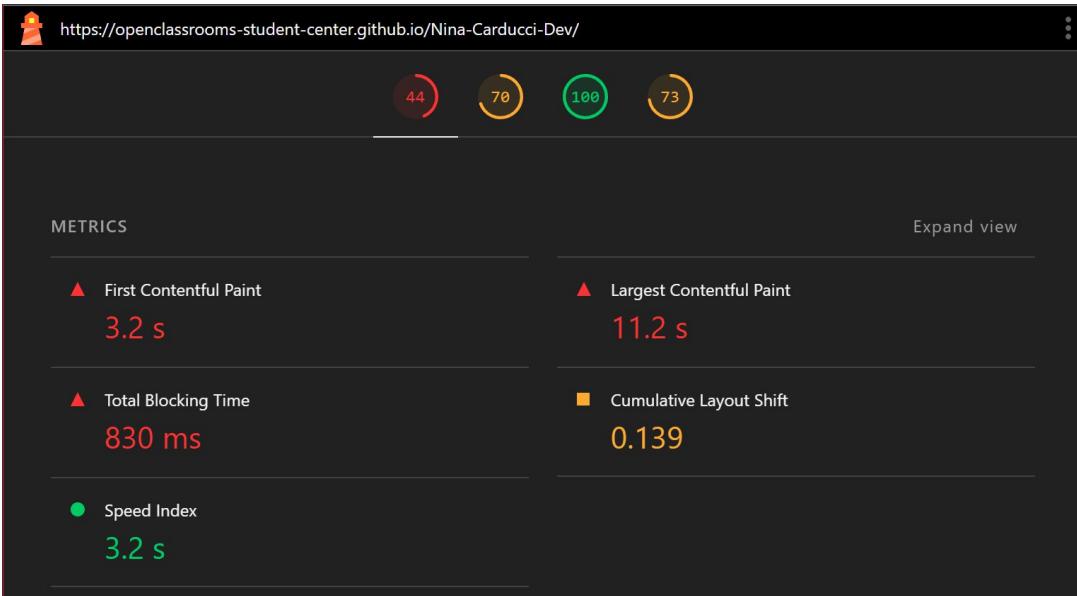
#### Pre-Optimized Google Lighthouse Scores:



#### Google Lighthouse highest Scores Post-Optimization:



## Part Two: Poor Performance Score Causes & Resolutions:



By utilizing Chrome DevTools and reviewing the **diagnostic breakdown** we can identify the major **issues** related to the site's **performance**.

- ❖ **Issue: Off-screen images** are a large contributing factor to the reduced performance of the page.
  - **Solution:** Images that aren't immediately visible to the user need to be **deferred**. This has been done via **lazy loading** by giving said images the `loading="lazy"` attribute which tells the browser to only render/load the image until their within a certain range of the viewport
- ❖ **Issue: Serve images in next-gen formats** this can also heavily affect load times of pages due to the overall size of all images being displayed.
  - **Solution:** Conversion of all images to the **WebP format** for better compression while retaining image quality, and efficient download speeds, data consumption than PNG or JPEG formats
- ❖ **Issue: Eliminate Render-Blocking Resources:**

URL	Transfer Size	Est Savings
jQuery CDN <span>Cdn</span>	<b>30.2 KiB</b>	<b>1,600 ms</b>
/jquery-3.4.1.min.js (code.jquery.com)	30.2 KiB	1,600 ms
GitHub <span>Utility</span> <span>1st Party</span>	<b>74.2 KiB</b>	<b>1,320 ms</b>
...bootstrap/bootstrap.bundle.js (openclassrooms-student-center.github.io)	43.9 KiB	830 ms
...assets/maugallery.js (openclassrooms-student-center.github.io)	2.1 KiB	170 ms
...bootstrap/bootstrap.css (openclassrooms-student-center.github.io)	26.6 KiB	170 ms
...assets/style.css (openclassrooms-student-center.github.io)	1.6 KiB	170 ms
Google Fonts <span>Cdn</span>	<b>1.3 KiB</b>	<b>890 ms</b>
/css?family=... (fonts.googleapis.com)	1.3 KiB	890 ms

- **Solution:** Deferred any **non-critical** JavaScript (defer & async) & CSS and/or preconnected/preloaded necessary resources for better performance results (Google Fonts, Bootstrap CSS, style.css)

## ❖ Issue: JavaScript & CSS file bloat

The screenshot shows the Google Chrome DevTools interface. The top navigation bar has tabs for Elements, Console, Sources, Network, Performance, Memory, and Application. The Sources tab is active, showing a tree view of files under 'top' and 'bootstrap'. The 'bootstrap.css' file is selected, displaying its CSS code. The right sidebar contains sections for Watch, Breakpoints, Scope, Call Stack, and various XHR/fetch, DOM, and event listener breakpoints. Below the Sources tab, there's a 'Coverage' tab which is also active. This tab displays a table of unused bytes for different resources. The table includes columns for URL, Type, Total Bytes, Unused Bytes, and Usage Visualized (represented by a progress bar). The table data is as follows:

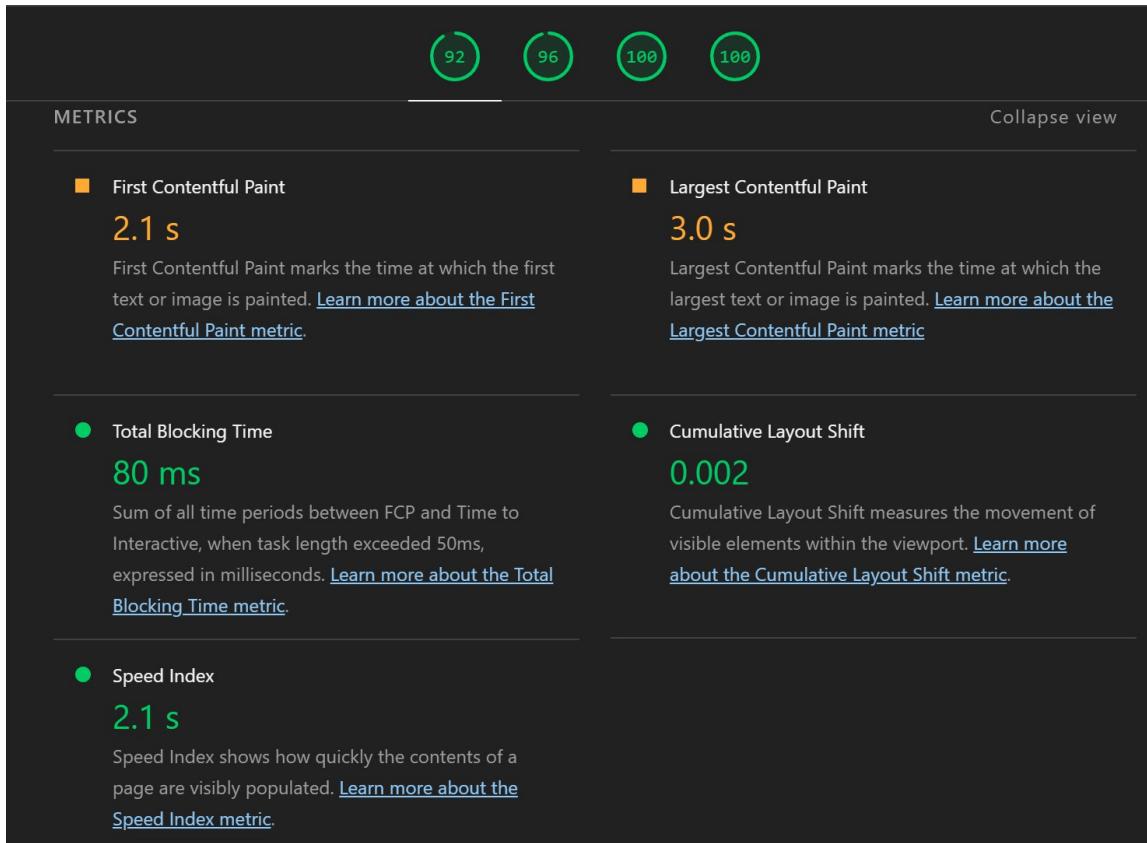
URL	Type	Total Bytes	Unused Bytes	Usage Visualized
<a href="https://openclassrooms-student-c.../bootstrap.css">https://openclassrooms-student-c.../bootstrap.css</a>	CSS	205,481	196,383	95.6% <div style="width: 95.6%;"></div>
<a href="https://openclassrooms-student-c.../bootstrap.bundle.js">https://openclassrooms-student-c.../bootstrap.bundle.js</a>	JS (per function)	209,713	138,132	65.9% <div style="width: 65.9%;"></div>
<a href="https://code.jquery.com/jquery-3.4.1.min.js">https://code.jquery.com/jquery-3.4.1.min.js</a>	JS (per function)	88,145	40,913	46.4% <div style="width: 46.4%;"></div>
<a href="chrome-extension://eimadpbcbfnm.../proxy">chrome-extension://eimadpbcbfnm.../proxy</a>	JS (per function)	23,847	18,485	77.5% <div style="width: 77.5%;"></div>
<a href="/css?family=Inter&amp;family=Spectralital,wght">/css?family=Inter&amp;family=Spectralital,wght</a>	CSS	10,088	10,088	100% <div style="width: 100%;"></div>
<a href="https://openclassrooms-student-c.../maugallery">https://openclassrooms-student-c.../maugallery</a>	JS (per function)	8,572	3,434	40.1% <div style="width: 40.1%;"></div>
<a href="https://openclassrooms-student-c.../style.css">https://openclassrooms-student-c.../style.css</a>	CSS	5,081	808	15.9% <div style="width: 15.9%;"></div>
<a href="https://openclassrooms-student-c.../scripts">https://openclassrooms-student-c.../scripts</a>	JS (per function)	315	0	0% <div style="width: 0%;"></div>

❖ **Solution:** Consulting the **Sources tab** in DevTools will allow us to track the **coverage** of both CSS & JavaScript files to accurately determine how much and which content within the files is going unused before completely removing them manually (this can be a long process depending on the file size).

- The removal process can be simplified with tools like PurgeCSS or PurifyCSS.
- However, do keep in mind that the page's DOM elements for this page are also being **dynamically adjusted** via JavaScript both locally within the site and from a CDN (jQuery) so it isn't recommended to only reference the HTML elements and classes that are initially and only visible within the HTML file itself, otherwise it will **cause negative unexpected behavior**.

- **Minification** is also recommended for JavaScript and CSS files as it allows the browser to read the code more efficiently

## Part Three: Summary of Fixes



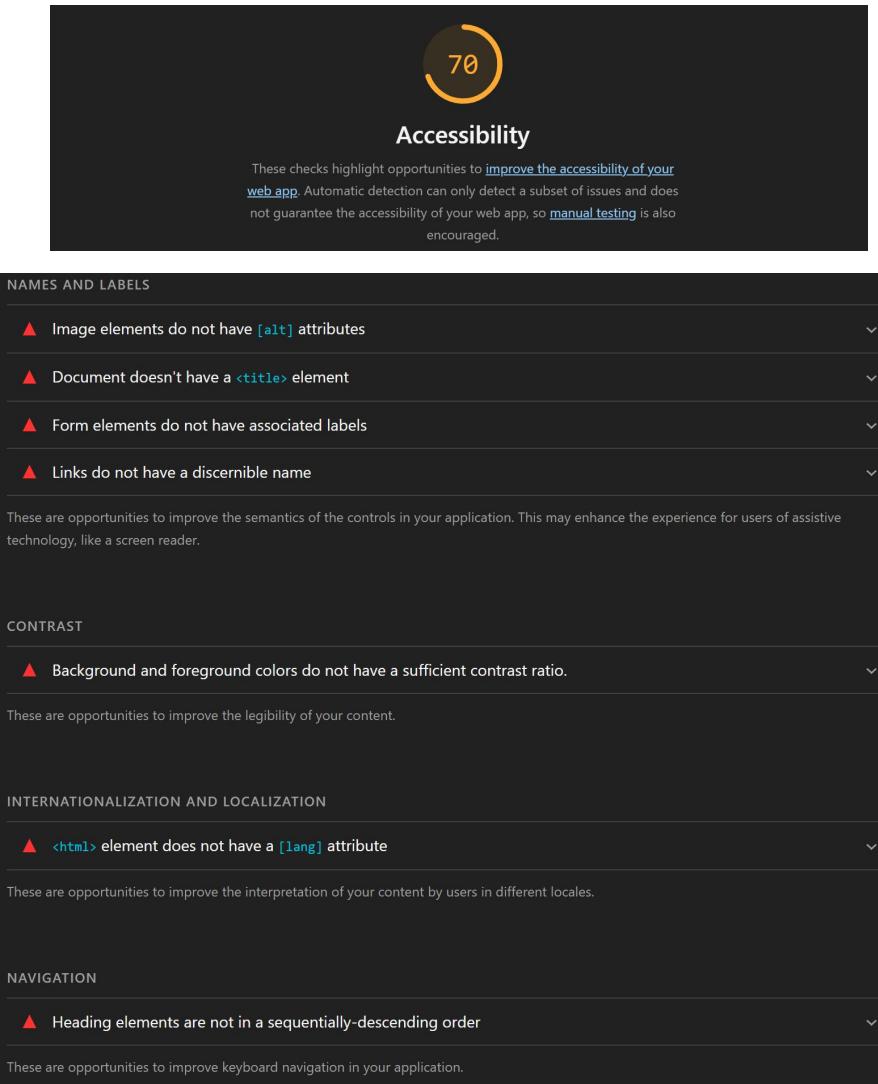
- ❖ Conversion and compression of all images to WebP greatly improved performance of the site, **positively** influencing the **speed index**, **FCP**, **LCP**, and **total blocking time**
- **Resizing images** and giving them explicit height & width properties **aid** in improving performance as well while reducing unnecessary **layout shifts**, which if left unchecked, would force browsers to guess the appropriate size(s) of visible elements
  - Images within the slider that are immediately visible heavily impacted the FCP & LCP scores originally due to their large sizes (ranging

from **1.54MB** to **5.43MB**)(now ranging from **99.6KB** to **229KB** while retaining as much of the original image quality as possible)

- ❖ **Deferring images** not immediately visible to viewports greatly improves performance scores, e.g., the images found within the **gallery**
- ❖ **Minification** and **removal** of unused code within CSS & JavaScript also greatly improved Lighthouse Performance scores
- ❖ **Preconnecting/Preloading** assets such as **Bootstrap.css & Google fonts** positively affect website performance
- ❖ Appropriately applying **async/defer** to <script> tags for JavaScript files that aren't immediately needed effectively reduced blocking time greatly
- ❖ **In conclusion**, all of the aforementioned changes being implemented, have made the site **consistently** reach a Google Lighthouse performance score of at least **90** with only slight deviations that stem from a connection to the jQuery CDN. The page contents can now be loaded approximately within **2.1 seconds** with **minimal blocking time (~80ms)**, a **low CLS score**. FCP & LCP times have been effectively reduced from ~**3.0+s** (FCP) & **10+s** (LCP), to now acceptable times of ~**2.1s** (FCP) & ~**2.4-3.0s** (LCP).

## Part Four: Accessibility Issues

### Pre-optimized Accessibility



The screenshot shows a dark-themed accessibility audit interface. At the top, a circular progress bar is at 70%, with the number '70' in the center. Below it, the word 'Accessibility' is displayed. A note below the progress bar reads: '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.'

**NAMES AND LABELS**

- ▲ Image elements do not have `[alt]` attributes
- ▲ Document doesn't have a `<title>` element
- ▲ Form elements do not have associated labels
- ▲ Links do not have a discernible name

These are opportunities to improve the semantics of the controls in your application. This may enhance the experience for users of assistive technology, like a screen reader.

**CONTRAST**

- ▲ Background and foreground colors do not have a sufficient contrast ratio.

These are opportunities to improve the legibility of your content.

**INTERNATIONALIZATION AND LOCALIZATION**

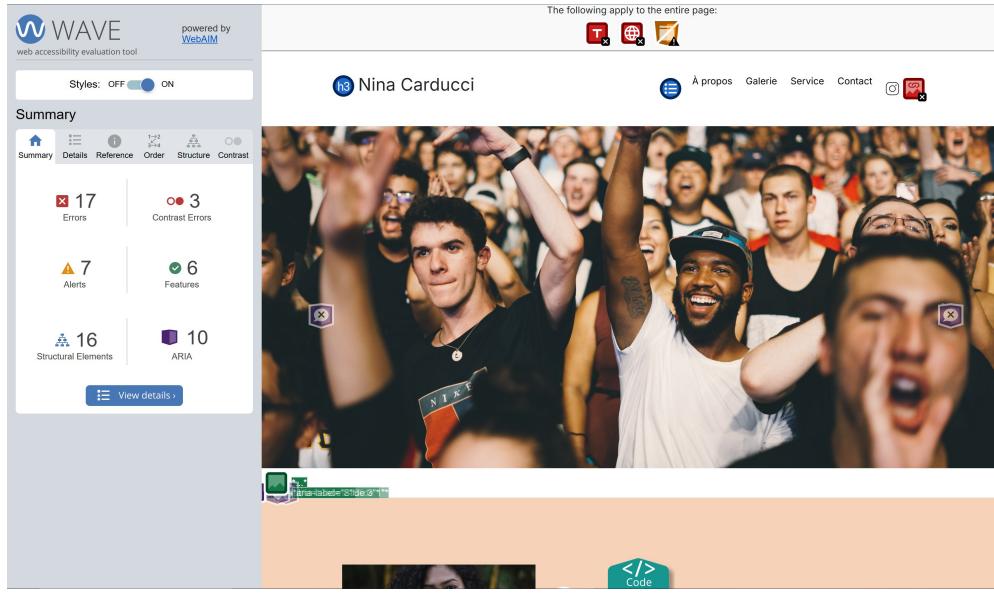
- ▲ `<html>` element does not have a `[lang]` attribute

These are opportunities to improve the interpretation of your content by users in different locales.

**NAVIGATION**

- ▲ Heading elements are not in a sequentially-descending order

These are opportunities to improve keyboard navigation in your application.

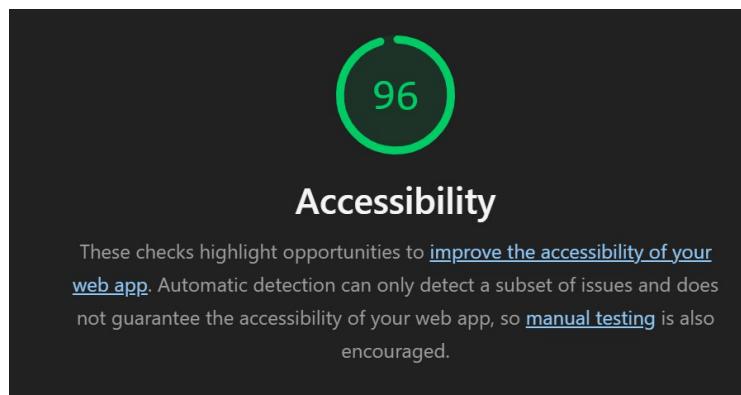


❖ Identified Issues:

- Several **image elements** in on the web page **lack** an **alt** attribute
  - **Solution:** Give a respective **succinct description** to each image, which will assist those who use screen readers when navigating/browsing websites
- Form elements lack related **labels (<label>)**, limiting the scope of assistive technologies that can announce form controls
  - **Solution:** Create **<label>** elements with '**for="..."**' that relate to the respective **<input>** id
- HTML document **lacks a <title> element**, diminishing SEO and limiting the scope of assistive technology capabilities
  - **Solution:** Within the **<head>** element of the HTML document, this **<title>** element was added: '**<title>Photographe Nina Carducci | Photographie Professionnelle à Bordeaux</title>**'
- **[lang]** or **[hreflang]** attribute is **not found** within **<html>** element
  - **Solution:** Simply give the page '**<html lang="fr-fr">**' to aid screen readers in **properly announcing** the page's content (particularly text content) in the appropriate language, in this case, French.
- Heading elements **aren't** in **sequentially-descending order**
  - **Solution:** Change heading elements like the main page's title from **<h3>** to **<h1>**, change 'about' section headers from **<h3>** to **<h2>** (**about-me\_title**) and **<h6>** to **<h3>** (**about-me\_introduction**), etc.

- These changes help **maintain a coherent semantic flow** throughout the page that won't be confusing to navigate through when utilizing assistive technologies
- **Lack of semantic elements**
- **Solution:** wrapping primary page content within a <**main**> element and having elements such as <**header**>, <**nav**>, and <**section**> help greatly with **keyboard navigation** and how screen readers will recognize the structure of the page.

### Accessibility Results: Post-intervention & Optimization



The screenshot shows the WAVE web accessibility evaluation tool interface. On the left, there's a sidebar with a summary of findings: 0 Errors, 3 Contrast Errors, 0 Alerts, 22 Features, 19 Structural Elements, and 9 ARIA. A 'View details' button is at the bottom of this sidebar. The main content area features a large image of a bride and groom sharing a kiss in front of a crowd. A red overlay on the image highlights several accessibility issues with icons like a person in a wheelchair and a person with a hearing aid. At the bottom of the page, there's a code editor window showing the HTML code for the image, with annotations like 'Photo' and 'aria-label="Slide 3"'.

- ❖ The only thing that can be adjusted for a better score is the color of the filter to a **darker color** to contrast with the white background

## Part Five: Search Engine Optimization (SEO)

### SEO Pre-optimization/fixes

The screenshot shows the Lighthouse SEO audit results. The overall score is 73. The category being viewed is "SEO". A note states: "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." Below this, under "CONTENT BEST PRACTICES", there are three issues listed with dropdown arrows:

- ▲ Document doesn't have a `<title>` element
- ▲ Document does not have a meta description
- ▲ Image elements do not have `[alt]` attributes

At the bottom, a note says: "Format your HTML in a way that enables crawlers to better understand your app's content."

#### ❖ Identified Issues:

- As previously mentioned a `<title>` element is essential for website pages. **Having no `<title>`** element makes it **difficult** for search engine users to determine if the page is relevant to their search
  - **Solution:** Give the page a `<title>` using **relevant keywords**:  
`'<title>Photographe Nina Carducci | Photographie Professionnelle  
a Bordeaux</title>'`
- HTML Document **lacks** a meta description
  - **Solution:** provide a meta description that gives a **concise summary** of the page content while **naturally including** relevant keywords:
    - “Professional photographer in Bordeaux, France, Nina Carducci captures weddings, portraits, and live events with creativity, passion, and a sharp eye for detail.”
- No social media presence

- **Solution:** <meta> elements were included in the <title> of the web page for both Open Graphs & Twitter cards
- ❖ Additional Improvements:
  - As stated before image element [alt] attributes have been added with a brief description of their respective image.
  - Local SEO **Structured Data Script** was created and added to document's <head>, incorporating **Schema.org microdata** to describe the business and provide information that would be useful to potential customers
  - **Google Rich Result Test:**

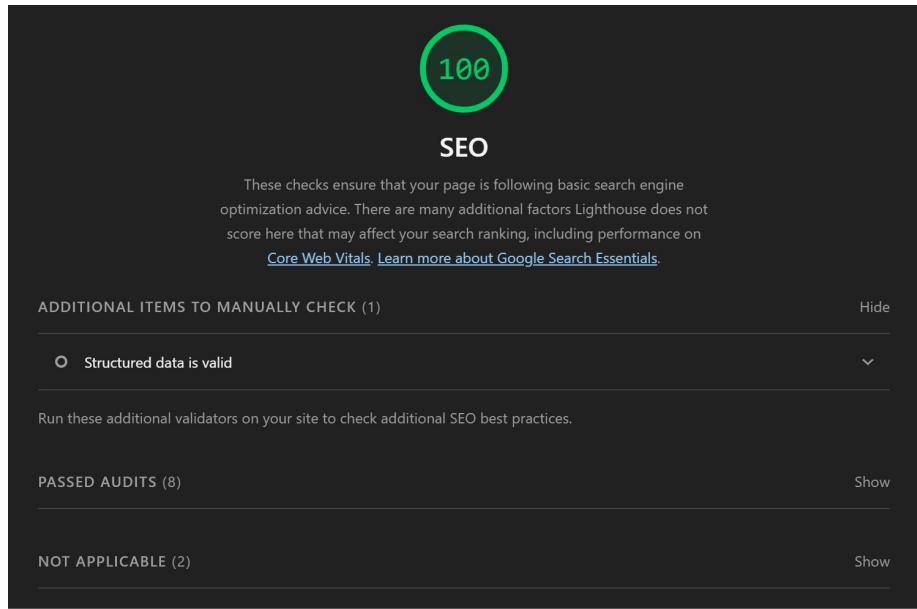
The screenshot shows the Google Rich Result Test interface. On the left, a code editor displays a JSON-LD structured data script. On the right, the results panel indicates "2 valid items detected" for "Local businesses" and "Organization". It also shows "1 valid item detected" for "Non-critical issues detected". Below this, there are sections for "Detected structured data" and "Additional resources". At the bottom, links for "Privacy" and "Terms" are visible.

```

1 <script type="application/ld+json">
2   {
3     "@context": "https://schema.org",
4     "@type": ["LocalBusiness", "Service"],
5     "name": "Nina Carducci",
6     "description": "Nina Carducci is a professional photographer in
Bordeaux, France, specializing in wedding photography, concert coverage, corporate
& professional portraits, and custom photo albums. With a creative eye and refined
technique, she captures life's most precious moments with emotion, grace, and
authenticity, turning each image into a timeless memory.",
7     "sameAs": ["https://www.instagram.com/ninacarducci.photo/?hl=fr"],
8     "address": {
9       "@type": "PostalAddress",
10      "streetAddress": "68 Avenue Alsace-Lorraine",
11      "addressLocality": "Bordeaux",
12      "addressRegion": "Nouvelle-Aquitaine",
13      "postalCode": "33200",
14      "addressCountry": "FR"
15    },
16    "areaServed": "Bordeaux",
17    "priceRange": "50€-400€",
18    "telephone": "+33 (0)5 56 67 78 89",
19    "openingHoursSpecification": [
20      {
21        "@type": "OpeningHoursSpecification",
22        "dayOfWeek": [
23          "Monday",
24          "Tuesday",
25          "Wednesday",
26          "Thursday",
27          "Friday"
28        ],
29        "opens": "10:00",
30        "closes": "19:00"
31      }
32    }
33  </script>

```

- ❖ **SEO Results Post-optimization & Fixes:**



## Part Six: Debugging & The Fixes

- ❖ It was requested by Nina Carducci, that the code related to the **gallery's image navigation** and **filter buttons** were reviewed for bug fixes.
  - “The gallery’s previous and next image navigation doesn’t work.”
    - **Resolved.** Both modal navigation arrows now function properly without error
  - “When you change filters to display images, the selected category is not visible. The category should normally have a gold background. The same as the default filter.”
    - **Resolved.** Clicking a different filter now highlights the active tag
    - **Suggestion:** If a better accessibility score is desired for the page, consider changing the active tag color to a darker color that contrasts well with the white background.
- ❖ Breakdown of the Debugging Process:
  - **1) Objective:** Identify and fix the bug that’s preventing **gallery image navigation** within the lightbox/modal.
    - Identified the relevant `<script>` for the gallery image navigation: `'maugallery.js'` and found the relevant functions `'prevImage()'` & `'nextImage()'`.

- Upon identifying the related functions, developer tools were utilized to add breakpoints in the code for both functions in order to trace logic.
- `console.log()` statements were used to check variables such as `activeImage`, `index`, and `imagesCollection`
- After some investigation, it was identified that the issue **primarily** stems from how the **index** variable was being used in relation to the `imagesCollection` array. Specifically, the **index is correctly calculated** as the position of the **current active image** within `imagesCollection`, however, the code **fails to increment** or **decrement** the **index** respectively.
- As a result, the **index** variable remains the same and isn't properly updated when the user clicks on either directional button. The fix to this is to simply adjust the **index** by either **subtracting(-) 1** for `prevImage()` and **adding(+) 1** for `nextImage()`. Examples below:
  - **Error** (`prevImage()`): `next = imagesCollection[index] || imagesCollection[0 - 1]`
  - **Solution** (`prevImage()`): `next = imagesCollection[index - 1] || imagesCollection[0 -1]`
- **Result Summary:** Both modal/lightbox navigation arrows now function properly as the index is now dynamically being adjusted

- **2) Objective:** Identify and fix the bug related to the **active filter highlight** not functioning properly when switching to a different filter category (Tous, Concert, Enterprise, etc.).
- The files related to this issue that were reviewed are '**maugallery.js**' & '**style.css**'
  - The **filterByTag()** function is responsible for the necessary class removals and additions that correspond to the linked CSS file
  - The classes **active** & **active-tag** are related to the **issue**
  - The **problem:** ``$(".active-tag").removeClass("active active-tag");``  
``$(this).addClass("active-tag");``
    - The ``.active-tag`` class that was added doesn't exist anywhere within the CSS file, making it completely useless. However,

`**.active**`, does exist and has the relevant CSS rules related to highlighting the active filter. So all that needs to be done replace `.**active-tag**` with `.**active**`.

- The **solution**: `\$(“.nav-link”).removeClass(“active”);`  
`\$(this).addClass(“active”);`
- **Result**: Clicking any filter option will now be highlighted in gold due to the `.**active**` class being properly applied.

## The Full Lighthouse Reports:

Final report for mobile can be found at:

[https://github.com/EAE00-src/Nina-Carducci\\_Website-Intervention/blob/master/Report%20Documentation/Nina-Carducci\\_Lighthouse-Report\\_mobile.pdf](https://github.com/EAE00-src/Nina-Carducci_Website-Intervention/blob/master/Report%20Documentation/Nina-Carducci_Lighthouse-Report_mobile.pdf)

Final report for desktops can be found at:

[https://github.com/EAE00-src/Nina-Carducci\\_Website-Intervention/blob/master/Report%20Documentation/Nina-Carducci\\_Lighthouse-Report\\_desktop.pdf](https://github.com/EAE00-src/Nina-Carducci_Website-Intervention/blob/master/Report%20Documentation/Nina-Carducci_Lighthouse-Report_desktop.pdf)