Analyzing the Performance Impacts of Lazy Loading in Web Applications

BÂRA Răzvan-Mihail
POLITEHNICA National University for Science and Technology of Bucharest
Razvan_mihail.bara@stud.etti.upb.ro

BOIANGIU Costin Anton
POLITEHNICA National University for Science and Technology of Bucharest
costin.boiangiu@upb.ro

Abstract:

This scientific paper explores the effects of lazy loading on the performance of web applications. Lazy loading is a technique employed in web development to defer the loading of non-essential resources until they are actually needed, thereby optimizing initial page load times and resource utilization. This study aims to investigate the impact of lazy loading on various performance metrics, including page load speed, user experience, and overall efficiency of web applications.

1. Introduction

1.1 Background

Web applications have become integral to our digital lives, serving as dynamic platforms for information dissemination, e-commerce, and interactive user experiences. As user expectations continue to rise, so does the demand for web applications that deliver seamless performance. One critical aspect of optimizing web application performance is the loading time of resources during the initial page load. Users often associate a sluggish or delayed loading experience with poor application performance, which can lead to frustration and decreased user engagement.

In response to this challenge, web developers employ various techniques to enhance loading efficiency. Lazy loading has emerged as a prominent strategy, allowing developers to defer the loading of non-essential resources until they are required for user interaction. This deferred loading aims to streamline the initial page load, reduce bandwidth consumption, and improve overall user experience. The concept of lazy loading has garnered significant attention as a means to strike a balance between feature-rich web applications and optimal performance. [1]

1.2 Objectives

The primary objective of this study is to delve into the effects of lazy loading on the performance of web applications. By conducting a systematic analysis, we aim to uncover insights into how lazy loading influences key performance metrics, including page load speed, network utilization, and user experience. Through this investigation, we seek to provide web developers with a deeper understanding of the trade-offs associated with lazy loading, enabling them to make informed decisions in the pursuit of creating high-performance web applications.

To achieve this goal, we will employ a rigorous experimental methodology, utilizing real-world web applications and diverse test scenarios. The study will not only assess the quantitative impact on performance metrics but also explore the qualitative aspects of user interactions and satisfaction. By addressing these facets, we aim to contribute valuable knowledge that can guide best practices for implementing lazy loading in web development.

In the subsequent sections, we will review existing literature on lazy loading, detail our experimental setup and methodology, present and analyze results, and conclude with recommendations for developers and avenues for future research. Through this exploration, we aspire to contribute to the ongoing dialogue on web application optimization and advance the understanding of the practical implications of lazy loading in real-world scenarios.

2. Literature Review

The literature review aims to provide a comprehensive overview of existing research on lazy loading in web applications. It examines key studies and findings related to the impact of lazy loading on web application performance, user experience, and overall efficiency.

2.1 Lazy Loading in Web Development

Lazy loading has gained prominence as a technique for optimizing web application performance by deferring the loading of non-essential resources until they are needed. According to Simon Fray, lazy loading minimizes initial page load times and enhances user experience by prioritizing critical content. Additionally, emphasize the role of lazy loading in reducing server loads and conserving bandwidth, contributing to a more sustainable and scalable web infrastructure. [1]

2.2 Performance Metrics in Lazy Loading Studies

In an article written by Jamie Juviler highlights the importance of considering multiple performance metrics when evaluating the effects of lazy loading. In their study, they analyze the impact on page load speed, resource utilization, and user engagement. [2]

2.3 User Experience Implications

User experience is a crucial aspect of web application development. The study by Jamie Juviler [2] investigates the correlation between lazy loading and user satisfaction, emphasizing the positive impact on perceived performance. Contrarily, Felix and Rick [3] suggest potential drawbacks, such as delayed rendering of images during user interaction, raising concerns about the trade-offs associated with lazy loading.

3. Methodology

3.1 Experimental Setup

To assess the effects of lazy loading on web application performance, we conducted experiments on two distinct applications: a static website and a full-stack application representing an online library.

3.1.1 Static Website

The first application is a static website designed to showcase content without dynamic interactions. This serves as a baseline scenario to evaluate the impact of lazy loading on straightforward, content-focused web pages. The website is composed of only one page, that has 5 sections, each section containing a slider of 10 high quality images.

3.1.2 Full-Stack Application for an Online Library

The second application represents a comprehensive full-stack online library, featuring dynamic content, user authentication, and real-time interactions. This complex application allows us to explore the scalability and adaptability of lazy loading in a more interactive and data-intensive environment.

3.2 Performance Metrics

Our methodology involves measuring various performance metrics to comprehensively evaluate the impact of lazy loading on both applications. The key metrics include:

3.2.1 Page Load Time

We will measure the time it takes for each web page to load completely, starting from the initiation of the request to the rendering of all visible content. This metric serves as a fundamental indicator of the overall responsiveness of the applications.

3.2.2 Bandwidth Usage

To assess the efficiency of lazy loading in conserving bandwidth, we will monitor the amount of data transferred between the server and the client during the page loading process. This measurement is crucial for understanding the potential benefits of lazy loading, particularly in scenarios with limited network resources.

3.2.3 User Experience Metrics

We will employ the Chrome DevTools and the Lighthouse extension to gather user experience metrics, including First Contentful Paint (FCP), Time to Interactive (TTI), and Cumulative Layout Shift (CLS). These metrics provide insights into the perceived performance and visual stability of the web applications.

3.3 Test Cases

For each application, we designed a series of test cases to simulate different user interactions and scenarios. These include:

3.3.1 Navigation Scenarios

- Initial page load
- Navigation between pages
- Loading additional content dynamically

3.3.2 User Interaction Scenarios

- Scrolling through content
- Interacting with dynamic elements (e.g., forms, modals)

3.4 Tools and Technologies

The experiments were conducted using Google Chrome's DevTools and the Lighthouse extension. The DevTools provide in-depth insights into network activity, rendering performance, and user interactions, while the Lighthouse extension automates the collection of performance metrics and provides a standardized evaluation of web page performance.

3.5 Test Environments

All experiments were conducted in controlled environments to minimize external factors affecting the measurements. The test environments included standardized hardware specifications and network configurations to ensure consistency across experiments.

4. Results and Analysis

4.1.1 Static Website

This application consists of a single html page with the purpose of showcasing different landforms. It has 5 distinct sections, each displaying 10 pictures of a landform. The results when comparing the lazy-loading performance versus the eager-loading performance are the following.

4.1.1 Page Load Time

For all the 50 pictures to load, using the eager loading strategy, the initial page load time is 4.2 seconds while using lazy loading, the initial load time is only 342 milliseconds. The load time decreases drastically.

This measurement is heavily impacted by the type of network used. For example, when the user is 3G, the total time for eager loading is 1.7 minutes while for lazy loading is only 1.7 seconds.

4.1.2 Network Utilization

The page transfers about 18MB worth of image content through the network. The same amount is transferred when using the lazy loading strategy, but only if the user explores the whole page. If not, the amount of transferred data is linked to what percentage of the images are requested by the user when navigating to a certain part of the page.

4.1.3 User Experience

While the slow initial page load time of eager loading would make a user leave the website due to the impression that the application isn’t working properly, the navigation experience would prove itself to be a pleasant one. All the content is already loaded onto the page, so scrolling and exploring through the different sliders of landforms presents no issue.

On the other hand, for lazy loading, the initial page load time is fast, but the user might encounter some issues while exploring the page. If his network is slow, he might end up leaving because of the on-demand loading time of a picture. If the network speed is fast enough, the experience of exploring the sliders would be like the one of eager loading.

4.1.4 Lighthouse extension reports

For eager loading, the first contentful paint along with the largest contentful paint took about 1.3 seconds to render, while for the other strategy took 0.5 seconds and 0.6 seconds. The first metric is strongly linked to how fast a user can see an element on the page, while the second metric refers to how fast a user can see the largest element.

Because of its static web page nature, the SEO score is the same for both loading strategies.

REFERENCES

[1] S. Fray, "Lazy Loading VS Eager Loading || What Impact Do They Have?," [Online]. Available: https://simon-frey.com/blog/lazy-loading-vs-eager-loading/.

[2] J. Juviler, "Lazy Loading: How It Decreases Load Time and Increases Engagement," 28 09 2023. [Online]. Available: https://blog.hubspot.com/website/lazy-loading-eager-loading.

[3] F. Arntz, "The performance effects of too much lazy loading," 15 07 2021. [Online]. Available: https://web.dev/articles/lcp-lazy-loading#causal_performance.