Journal of Emerging Information System and Business Intelligence
ISSN: 2774-3993
Journal homepage: https://ejournal.unesa.ac.id/index.php/JEISBI/

Rendering Performance Analysis of Astro JS, Next JS, Nuxt JS, and SvelteKit Frameworks Using Google Lighthouse, PageSpeed Insight, and JMeter

Ahmad Jourji Zaidan[1], Dwi Fatrianto Suyatno[2]
[1,2] *Surabaya State University, Surabaya, Indonesia*
ahmad.20023@mhs.unesa.ac.id, dwifatrianto@unesa.ac.id

ABSTRACT

The development and popularity of modern technology encourage innovation in developing web applications to increase loading speed and retain users. This research analyzes the performance of websites that use Astro JS, Next JS, Nuxt JS, and SvelteKit frameworks using the Server-Side Rendering technique. The advantage of the SSR technique is that it can improve website performance, SEO (Search Engine Optimization) optimization, and user experience. The tools used in this research are Google Lighthouse, PageSpeed Insight, and JMeter. The metrics measured in Google Lighthouse and PageSpeed Insight testing are FCP (First Contentful Paint), TBT (Total Blocking Time), SI (Speed Index), LCP (Large Contentful Paint), and CLS (Cumulative Layout Shift). While JMeter testing, the metrics measured are Response Time (Min, Max, Average), Error Rate, and Throughput. The development method used is the XP (Extreme Programming) method. The results of this study show Astro JS has superior performance in most Web Vitals metrics, followed by Next JS which shows superiority in several metrics. Nuxt JS and SvelteKit each only excelled in one Web Vitals metric. In stability and reliability of the system testing using JMeter, Nuxt JS showed the best performance by excelling in the response time, error rate, and throughput metrics. SvelteKit also performed well with dominance in several stability metrics, while Astro JS and Next JS only excelled in a small number of them. This research was conducted to provide insight for developers in choosing the right framework based on their needs.

Keyword: Astro JS, Next JS, Nuxt JS, SvelteKit, Google Lighthouse, PageSpeed Insight, JMeter.

Article Info:
Article history:
Received October 27, 2025
Revised March 24, 2025
Accepted April 02, 2025

Corresponding Author
Ahmad Jourji Zaidan
Universitas Negeri Surabaya, Surabaya, Indonesia
ahmad.20023@mhs.unesa.ac.id

1. INTRODUCTION

Along with the development and popularity of modern technology, web applications have grown increasingly rapidly. It is evident that there are currently 1.9 billion websites in the world and the number will continue to grow [1]. Web page loading speed is one of the factors to help retain users [2]. Currently, server-side rendering techniques (SSR) are an important consideration in modern web development because with its advantages it can improve website performance, Search Engine Optimization (SEO) optimization, and user experience [3].

The research "A Comparative Analysis of Next JS, SvelteKit, and Astro JS for E-commerce Web Development" by [4] and "Rendering Performance Comparison of React, Vue, Next, and Nuxt" by [5] both focused on comparing framework performance in web development. The first research highlights the impact of frameworks on performance metrics such as load time, interactivity, SEO, and resource efficiency, with results showing that Astro JS and SvelteKit are

superior to Next JS in load time and resource efficiency. Meanwhile, the second research study compared client- and server-side rendering performance of React, Vue, Next, and Nuxt using various test tools. WebPageTest recorded Vue as the fastest (0.9 seconds), PageSpeed Insight showed React ahead (2.8 seconds), while GTMetrix placed Next as the fastest (0.8 seconds).

In the context of the comparison between React Js and Next Js, the studies "React Js vs Next Js" by [6] and "Performance Rendering Analysis Between Server-Side and Client-Side Web Application" by [7] provide additional insights. The first research compares the popularity and performance of both technologies with Google Lighthouse as a test tool, where React JS is more popular and has an advantage in several performance aspects. The second research was more specific in comparing client- and server-side rendering, with the results showing that React Js scored 68, while Next Js only scored 35. Meanwhile, research by [8] in "Comparative Analysis Study of Front-End JavaScript Frameworks Performance Using Lighthouse Tool" examines the performance of web applications built with React, Angular, Vue, Svelte, and Solid Js. The results show that Vue has the best performance in implementing weather applications compared to other frameworks.

Based on previous research, after further analysis, it was found that there were limitations to the research, such as the absence of server specifications in the application used for testing, and one of the applications developed for testing did not meet the test metrics. This has an impact on the accuracy of the test results because these factors can affect the test results, especially in the metrics of load time, interactivity, and resource usage when applied in different environments. Therefore, this research aims to complement these limitations by providing a more in-depth analysis that is expected to provide more comprehensive information. In this research, we will compare the performance of rendering frameworks with Server-Side Rendering (SSR) technique. The frameworks compared are Astro JS, Next JS, Nuxt JS, and SvelteKit, the tools used are Google Lighthouse, PageSpeed Insight, and JMeter.

2. METHODS

There is a research flow that explains the stages that discuss the general description of the research from the first step to the last step that must be passed by the researcher to achieve the research objectives.

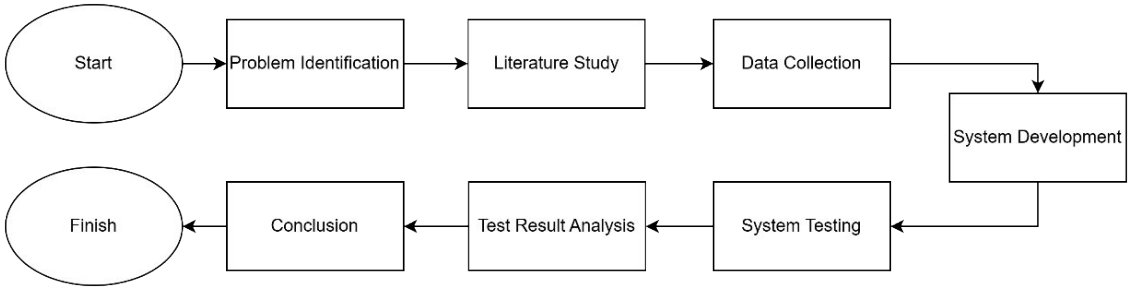

Figure 1. Research Flow

2.1 Problem Identification

There is a research flow that explains the stages that discuss the general description of the research from the initial stage to the final stage that the researcher must go through to achieve the research objectives.

2.2 Literature Study

Literature studies are used to gain in-depth knowledge and insights that can be used as a theoretical basis, reference, and consideration. At this stage, information collection is carried out by collecting various trusted sources such as journals, scientific articles, theses, and other related sources.

2.3 Data Collection

The data collection technique in this research is to utilize the Hotel Reviews dataset taken from an open-source site called *Kaggle*. The data has gone through an adjustment process for testing needs.

2.4 System Development

1. Development Method

   The application development method in this study is to use the *Extreme Programming* method. *Extreme Programming* method is a method that emphasizes flexibility in the face of change by focusing on software quality [9]. In addition, this method consists of several stages: planning, design, coding, and testing.

   a. Planning

   In this step, it aims to identify the application development needs that will be applied by the application, including analyzing system requirements, namely functional and non-functional needs.

   1) Functional Requirements

   Is a need in the form of data such as what processes or services the system must perform [10]. The following are the functional requirements of the application design that will be made as follows:

   a) The system can display text scenario reviews
   b) The system can display image scenario reviews based on the type of image resolution (480p, 720p, 1080p, 2K, and 4K)
   c) The system can display video scenario reviews based on the type of video resolution (480p, 720p, 1080p, 2K, and 4K)
   d) The system can display a review of text and image combination scenarios based on the type of image resolution (480p, 720p, 1080p, 2K, and 4K)
   e) The system can display a review of text and video combination scenarios based on the type of video resolution (480p, 720p, 1080p, 2K, and 4K)

   2) Non-functional Requirements

   Is a need owned by a system related to performance, platform operations, and so on [10]. The following are the non-functional needs of the application design that will be made as follows:

   a) Hardware

   The hardware used to build and test the application is as follows:

   - Laptop : HP 14S CF1051TU
   - Processor : Intel Celeron 4205U @1.8GHz
   - GPU : Intel UHD Graphics 610
   - Storage : 512 GB (SSD)
   - RAM : 8 GB

   b) Software

   The software used to build the application is as follows:

   - OS : Windows 10
   - Text Editor : Visual Studio Code
   - Web Browser : Chrome
   - API Testing : Postman

   c) Development Technology

The development technologies used to build each application are as follows:

- Astro JS v.4.10.2
- Next JS v.14.2.3
- Nuxt JS v.3.11.2
- SvelteKit v.2.0.0
- Tailwind CSS v.3
- Express JS v.4.18.2
- MySQL v.8.0.30

d) Server
Server specifications used for testing applications are as follows:
- Front-End
  - CPU : 2 Core
  - RAM : 4 GB
  - Storage : 60 GB
  - Location : West Java, Indonesia
- Back-End
  - CPU : 4 Core
  - RAM : 4 GB
  - Storage : 60 GB
  - Location : West Java, Indonesia

e) Cloud Storage
Used to store assets such as images and videos. The platform used is *Google Cloud Platform* with the Asia-Southeast2 region (Jakarta).

b. Design
In this step, system design is carried out with the aim of analyzing and knowing the main functions of the software. At this stage the researcher translates into UML form in the form of Use Case and Activity Diagrams.

c. Coding
In this step, the implementation of making applications is carried out by creating program code based on the design that has been made before. Application development uses each framework namely Astro JS, Next JS, Nuxt JS, and SvelteKit for the client side, Express JS for the server side and MySQL for the database.

d. Testing
In this step, testing of applications that have been developed using Chrome as a web browser for testing and testing tools using Google Lighthouse, PageSpeed Insight, and JMeter.

2. Applications to be tested
In this research, each application used for testing has the same interface, logic, and data used. In addition, each test scenario will render 1000 elements using the Server-Side Rendering technique.

2.5 System Testing

In this step, testing is carried out on each application with a predetermined scenario. Testing uses 3 tools namely Google Lighthouse, PageSpeed Insight, and JMeter. Google Lighthouse and PageSpeed Insight testing are used to measure performance and JMeter to measure system stability. The research by [7] on comparing performance between React Js and Next Js. In the test, it is explained that the test was carried out 5 times with the aim of getting maximum results. In testing, to get consistent and maximum results, testing using Google Lighthouse and PageSpeed Insight is carried out five times in each scenario. In addition, in the Google Lighthouse and PageSpeed Insight tests, the assessment reference uses the metrics in the Web Vitals model with 1 additional parameter, namely the Speed Index. Speed Index is a parameter used to visually measure the speed at which content is loaded during page loading [11].

The following are the Web Vitals metrics and their rendering time categorization:

Table 1. Rendering Time Metrics and Categorization Web Vitals

Parameter	Category	Rendering Time
First Contentful Paint	Fast (Green)	0 – 1.8 *seconds*
	Medium (Orange)	1.9 – 3 *seconds*
	Slow (Red)	> 3 *seconds*
Total Blocking Time	Fast (Green)	0 – 200 *milliseconds*
	Medium (Orange)	201 – 600 *milliseconds*
	Slow (Red)	> 600 *milliseconds*
Speed Index	Fast (Green)	0 – 3.4 *seconds*
	Medium (Orange)	3.5 – 5.8 *seconds*
	Slow (Red)	> 5.8 *seconds*
Large Contentful Paint	Fast (Green)	0 – 2.5 *seconds*
	Medium (Orange)	2.6 – 4 *seconds*
	Slow (Red)	> 4 *seconds*
Cumulative Layout Shift	Fast (Green)	0 – 0.1 %
	Medium (Orange)	0.2 – 0. 25 %
	Slow (Red)	> 0.25 %

In addition, system stability will be tested using the load test method to measure system stability and the ability of the system to handle users with predetermined specifications. The performance of a web-based application can be measured through the response time and throughput of a request, where the smaller the response time value and the greater the throughput value, the better the application performance [12]. Therefore, in this study, the metrics that will be measured for JMeter testing are response time and throughput metrics by adding 1 metric, namely error rate. In the testing process, this study will use 3 scenarios, namely the first scenario represents the condition of the website with low load, the second scenario represents medium load, and the third scenario represents high load.

The following is a test scenario using the load test method using JMeter:

Table 2. JMeter Load Testing Scenario

JMeter Testing Scenario		
Scenario 1	*Number of Threads (users)*	100
	Ramp-up Time (seconds)	100
	Loop Count	1
Scenario 2	*Number of Threads (users)*	300
	Ramp-up Time (seconds)	300
	Loop Count	1
Scenario 3	*Number of Threads (users)*	500
	Ramp-up Time (seconds)	500
	Loop Count	1

2.6 Test Result Analysis

In this step, an analysis of the test results that have been obtained is carried out. Rendering performance analysis using Google Lighthouse and PageSpeed Insight, focusing on Web Vitals metrics such as FCP (First Contentful Paint), TBT (Total Blocking Time), SI (Speed Index), LCP (Large Contentful Paint), and CLS (Cumulative Layout Shift). While the performance test analysis with the load test method, focuses on the metrics of response time, error rate, and throughput.

2.7 Conclusion

After all analysis data is obtained, the next step is to draw conclusions from the results of the analysis carried out based on Web Vitals and Load Test measurements.

3. RESULTS AND DISCUSSION

In the development process, the dataset used comes from the Hotel Reviews dataset taken from the Kaggle platform. The data has a csv (Comma Separated Values) format. Furthermore, data adjustment is carried out with the aim of facilitating the data import process. After that, import data into the MySQL database.

The test scenarios used are text scenarios, image scenarios (480p, 720p, 1080p, 2K, and 4K), video scenarios (480p, 720p, 1080p, 2K, and 4K), text and image combination scenarios (480p, 720p, 1080p, 2K, and 4K), and text and video combination scenarios (480p, 720p, 1080p, 2K, and 4K). From a total of 21 scenarios, each scenario was tested 5 times, then the results were averaged to obtain a more accurate value. The average value obtained is used as a representation of each metric and is used as a basis for further analysis. In addition, in some conditions, there are some frameworks that show identical results after the average calculation. Therefore, to maintain consistency in the analysis, the same metric values are still included in the calculation results. The following are the test results using Google Lighthouse, PageSpeed Insight and JMeter.

3.1 Google Lighthouse

1. *First Contentful Paint* (SI)

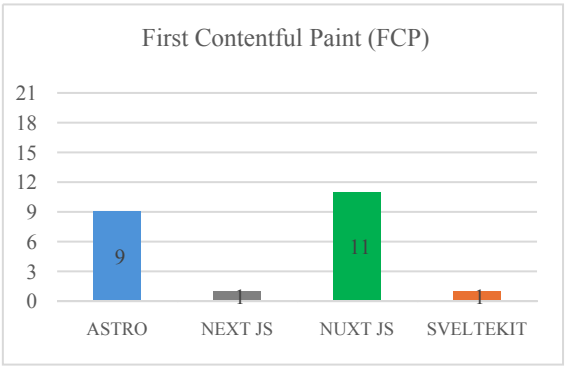

Figure 2. Google Lighthouse FCP Metrics Testing Results

The test results show that on the FCP metric Nuxt JS has the best FCP value by winning in 11 out of 21 scenarios. Followed by Astro JS, Next JS and SvelteKit.

2. *Total Blocking Time* (TBT)

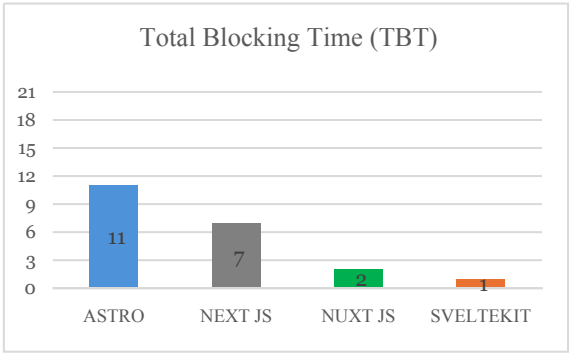

Figure 3. Google Lighthouse TBT Metrics Testing Results

The test results show that on the TBT metric Astro JS has the best TBT value by winning in 11 out of 21 scenarios. Followed by Next JS, Nuxt JS and SvelteKit.

3. *Speed Index* (SI)

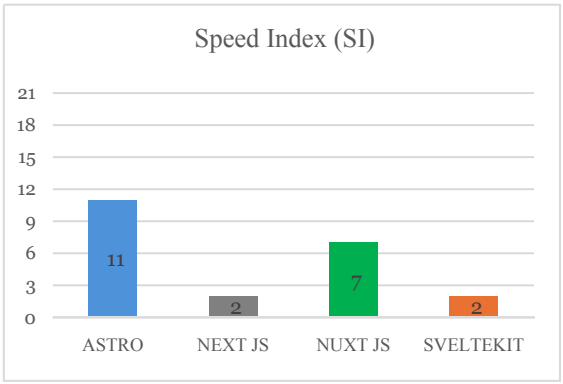

Figure 4. Google Lighthouse SI Metrics Testing Results

The test results show that on the SI metric Astro JS has the best SI value by winning in 11 scenarios out of 21 scenarios. Followed by Nuxt JS, Next JS and SvelteKit.

4. *Large Contentful Paint* (LCP)

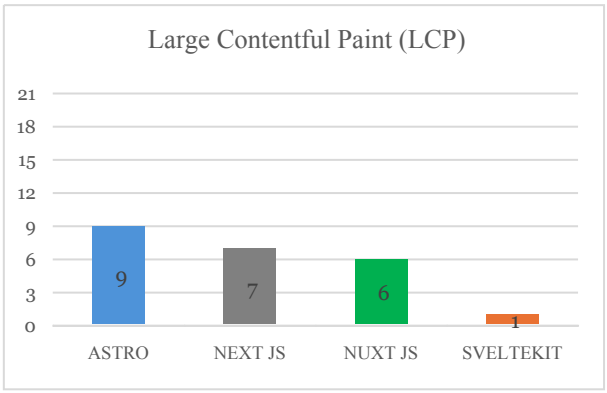

Figure 5. Google Lighthouse LCP Metrics Testing Results

The test results show that on the LCP metric Astro JS has the best LCP value by winning in 9 scenarios out of 21 scenarios. Followed by Next JS, Nuxt JS and SvelteKit.

5. *Cumulative Layout Shift* (CLS)

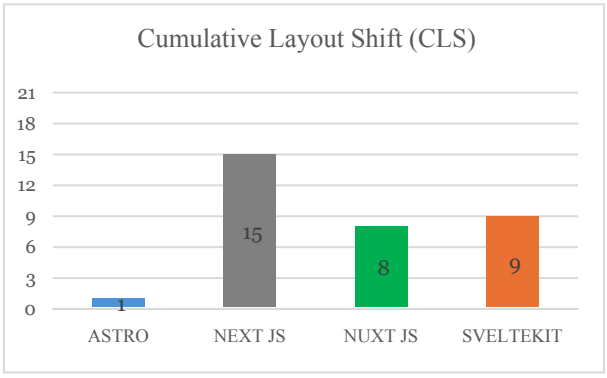

Figure 6. Google Lighthouse CLS Metrics Testing Results

The test results show that on the CLS metric Next JS has the best CLS value by winning in 15 out of 21 scenarios. Followed by SvelteKit, Nuxt JS, and Astro JS.

3.2 PageSpeed Insight

1. *First Contentful Paint* (FCP)

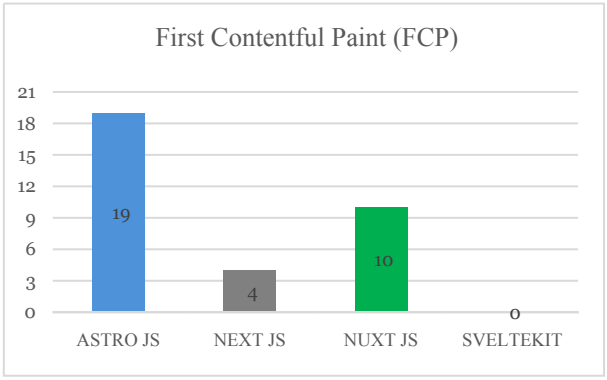

Figure 7. PageSpeed Insight FCP Metrics Testing Results

The test results show that Astro JS FCP padametric has the best FCP value by winning in 19 scenarios out of 21 scenarios. Followed by Nuxt JS, Next JS. While SvelteKit, did not excel in any scenario.

2. *Total Blocking Time* (TBT)

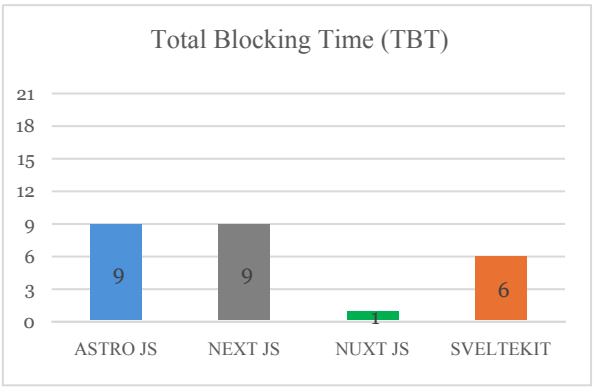

Figure 8. PageSpeed Insight TBT Metrics Testing Results

The test results show that on the TBT metric Astro JS and Next JS have the same TBT value by winning in 9 out of 21 scenarios. Followed by SvelteKit and Nuxt JS.

3. *Speed Index* (SI)

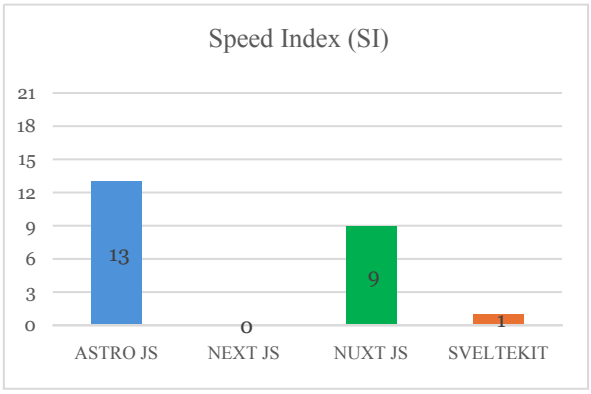

Figure 9. PageSpeed Insight SI Metrics Testing Results

The test results show that on the SI metric Astro JS has the best SI value by winning in 13 out of 21 scenarios. Followed by Nuxt JS and SvelteKit. Next JS, on the other hand, did not excel in any scenario.

4. *Large Contentful Paint* (LCP)

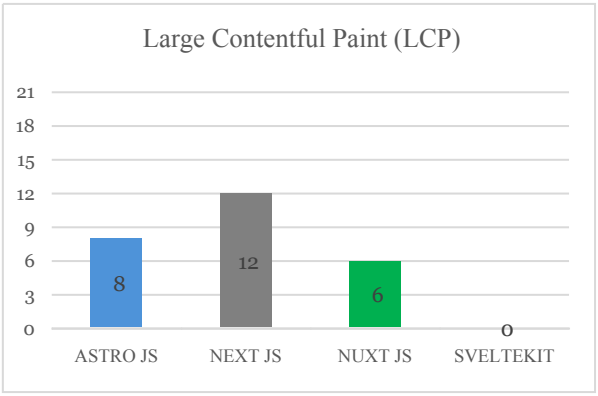

Figure 10. PageSpeed Insight LCP Metrics Testing Results

The test results show that on the LCP metric Next JS has the best LCP value by winning in 12 out of 21 scenarios. Followed by Astro JS, and Nuxt JS. SvelteKit, on the other hand, did not excel in any scenario.

5. *Cumulative Layout Shift* (CLS)

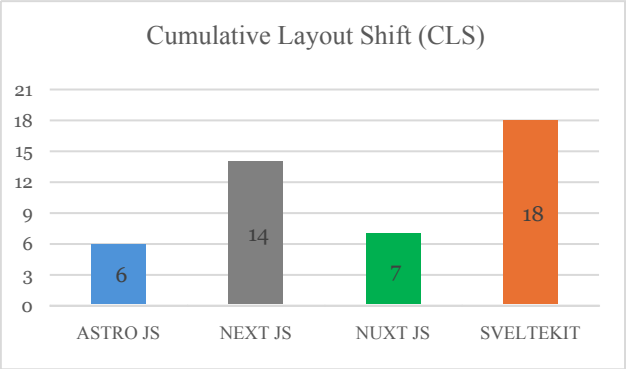

Figure 11. PageSpeed Insight CLS Metrics Testing Results

The test results show that on the CLS metric SvelteKit has the best CLS value by winning in 18 out of 21 scenarios. Followed by Next JS, Nuxt JS, and Astro JS.

3.3 JMeter

1. Simulated Load 100 Users

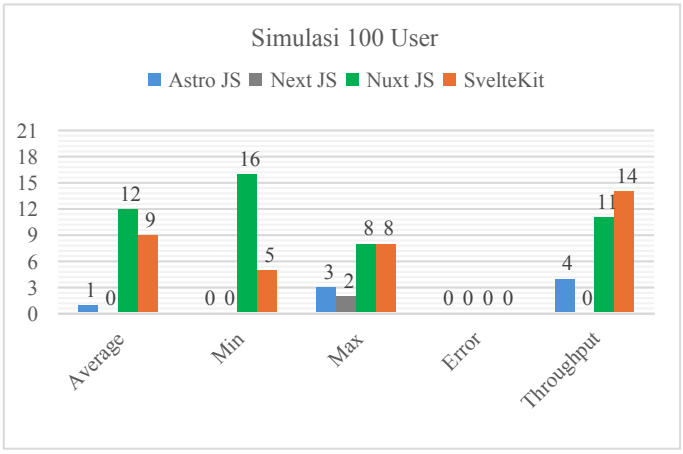

Figure 12. Load Testing Results 100 Users

The test results show that at a low load simulation of 100 users, Next JS has the best consistency in system stability and reliability testing by excelling in most metrics. Followed by SvelteKit, Astro Js, and Next JS.

2. Simulated Load 300 Users

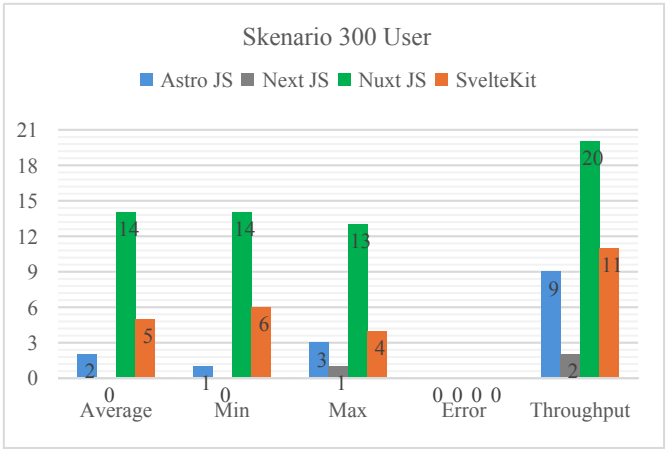

Figure 13. Load Testing Results 300 Users

The test results show that at a medium load simulation of 300 users, Next JS has the best consistency in testing system stability and reliability by excelling in all metrics. Followed by SvelteKit, Astro Js, and Next JS.

3. Simulated Load 500 Users

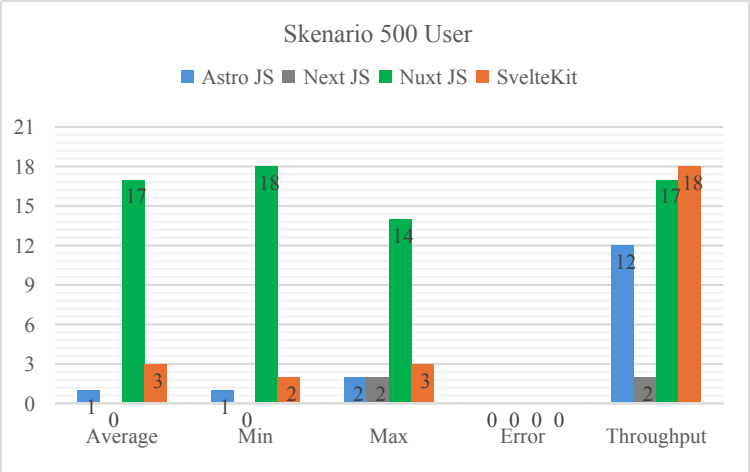

Figure 14. Load Testing Results 500 Users

The test results show that in the 500 users high load simulation, Next JS has the best consistency in system stability and reliability testing by excelling in most metrics. Followed by SvelteKit, Astro Js, and Next JS.

Based on the previous test results, the following is an overall analysis presented in the form of graph visualization. This approach aims to identify performance patterns, performance consistency between testing tools, and the characteristics of each framework in various metrics.

1. Analysis of Google Lighthouse and PageSpeed Insight Results

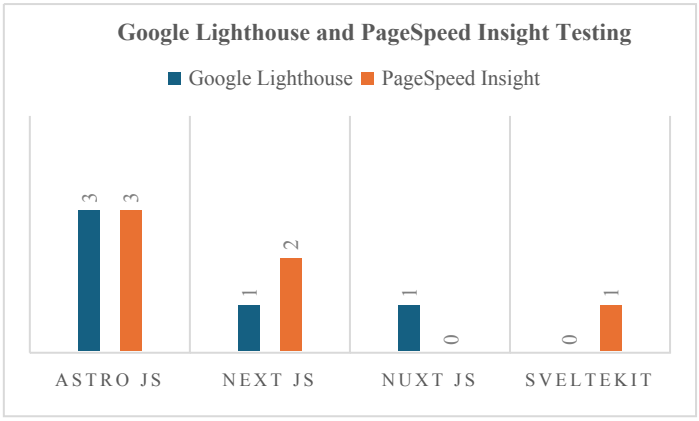

Figure 15. Google Lighthouse and PageSpeed Insight Testing Comparison Results

Based on the visualization data above, performance testing using Web Vitals metrics shows that Astro JS excels in most metrics, namely in the FCP, TBT, LCP, and SI metrics. This means that Astro JS has a good ability to handle the speed of rendering content in the form of text, visuals and content with large layout sizes. In addition, Astro JS also has a good ability to handle blocking time or handle the waiting time for users to interact with the page. Next JS shows good ability in TBT, LCP, and CLS metrics. This means Next JS has a good ability to handle blocking time, rendering content with large layout sizes and maintaining the visual stability of the page. Next, Nuxt JS and SvelteKit, Nuxt JS has an advantage in the FCP metric, which means the ability to render content the first time quickly, while SvelteKit has an advantage in the CLS metric, which means a good ability to maintain the visual stability of the page.

2. Analysis of JMeter Load Testing Results

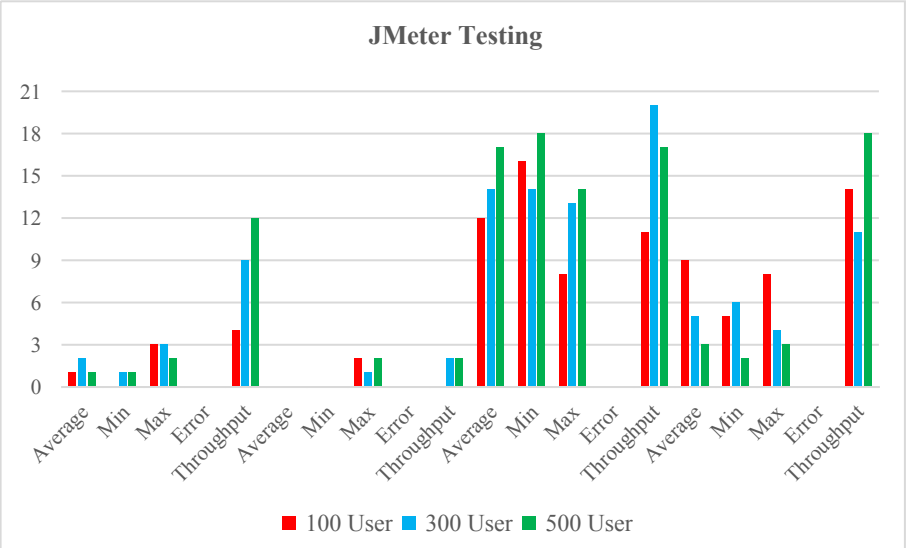

Figure 16. JMeter Testing Comparison Results

Based on the visualization data above, testing system stability and reliability using varying loads with Response time (Min, Max, Average), Error rate, and Throughput metrics, shows that Nuxt JS has good performance and consistency by excelling in various test metrics. Followed by SvelteKit, astro JS, and Next JS. In response time metrics (Min, Max and Average), Nuxt JS has good consistency, which means it has a good ability to handle requests quickly compared to other frameworks. In the error rate metric, all frameworks show good consistency with 0% error rate, which means they have good stability and reliability. And on the throughput metric, Nuxt JS and SvelteKit have good consistency, this means they have a good ability to handle high loads. While Astro JS and Next JS show poor consistency in various metrics.

CONCLUSION

This research was conducted with the aim of knowing the comparison of rendering performance. Based on the results of analyzing the rendering performance of the Astro JS, Next JS, Nuxt JS, and SvelteKit frameworks using Google Lighthouse, PageSpeed Insight, and JMeter, the following conclusions can be drawn:

1. The analysis process begins by creating 4 applications with the same interface design and data used, using the Hotel Reviews dataset. The application will render 1000 elements at once using the SSR (Server-Side Rendering) technique. Furthermore, each application is tested five times for each scenario using Google Lighthouse and PageSpeed Insight with an incognito mode browser configuration and a disabled cache. While JMeter testing, one test is carried out in each scenario and is carried out in stages based on the user simulation load scenario.
2. The results showed that overall, in performance testing using Web Vitals metrics, Astro JS performed very well by excelling in most metrics. Next JS followed with a lead in several metrics. While Nuxt JS and SvelteKit each excelled in one metric. In the stability and reliability testing using JMeter, Nuxt JS performed very well with excellence in all response time, error rate, and throughput metrics. This was followed by SvelteKit which excelled in several metrics. Astro JS and Next JS showed superiority in a small number of metrics.

REFERENCES

[1] Tjhoernandes, A., Susetyo, Y. A., Kristen, U., & Wacana, S. (2022). Penerapan MAC Address sebagai Autentikasi Aplikasi menggunakan Javascript Bindings Chromium Embedded Framework Python di PT. Jurnal Inovtek Polbeng, 7(1), 26–36.

[2] Nordström, C., & Dixelius, A. (2023). Comparisons of Server-side Rendering and Client-side Rendering for Web Pages.

[3] Meredova, A. (2023). Comparison of Server-Side Rendering Capabilities of React and Vue.

[4] Kroon Celander, E., & Möllestål, A. (2024). A Comparative Analysis of Next. js, SvelteKit, and Astro for E-commerce Web Development.

[5] Siahaan, M., & Kenidy, R. (2023). Rendering performance comparison of react, vue, next, and Nuxt. Jurnal Mantik, 7(3), 1851-1860.

[6] Dinku, Z. (2022). React. js vs. Next. js.

[7] Geovanny, A. (2022). Analisis Rendering Performa Antara Server-Side dan Client-Side Pada Web Application. Jurnal Ilmiah Betrik, 13 (03 Desember), 311-319.

[8] Siahaan, M., & Vianto, V. O. (2022). Comparative Analysis Study of Front-End JavaScript Frameworks Performance Using Lighthouse Tool. Jurnal Mantik, 6(3), 2462-2468.

[9] Sari, N., & Cahyani, D. (2022). Perancangan Sistem Informasi Monitoring Sertifikat Menggunakan Extreme Programming. Jurnal Ilmiah Computer Science, 1(1), 1-6.

[10] Indradewi, I. G. A. A. D., & Wibawa, I. G. P. (2021). Analisis dan Desain Sistem Informasi Pengajuan dan Monitoring Keuangan Kelurahan Berorientasi Obyek pada Kecamatan Denpasar Selatan. Jurnal Krisnadana, 1(1), 1-12.

[11] Chrome. (n.d.). Performance Audits - Chrome for Developers. Chrome Developers. Retrieved November 18, 2023, from https://developer.chrome.com/docs/lighthouse

[12] Ismail, A., Ananta, A. Y., Arief, S. N., & Hamdana, E. N. (2023). Performance testing sistem ujian online menggunakan jmeter pada lingkungan virtual. Jurnal Informatika Polinema, 9(2), 159-164.