Vol. 8, No. 4, August 2026

ISI Journal of Information Systems and Informatics

Published By Asosiasi Doktor Sistem Informasi Indonesia

Performance and Carbon Footprint Evaluation of a PWA-Based Waste Bank Information System

Muhammad Dzulfiqar[1], Adzanil Rachmadhi Putra[2], Rosyid Abdillah[3]

[1,2,3]Information System Department, Faculty of Industrial Engineering, Telkom University, Surabaya, Indonesia

Received:
February 6, 2026
Revised:
July 21, 2026
Accepted:
July 26, 2026
Published:
August 30, 2026

Corresponding Author:
Author Name*:
Muhammad Dzulfiqar
Email*:
dzulfiqarmuhammad@student.telkomuniversity.ac.id

DOI:
10.63158/journalisi.v8i4.1821

Abstract. The increasing volume of unmanaged waste in Indonesia has encouraged the adoption of digital waste bank management systems. However, the environmental impact of digital transformation, particularly regarding energy consumption and carbon emissions, remains insufficiently explored in community-scale information systems. This study evaluates the performance and environmental efficiency of Sibanksa, a Progressive Web Application (PWA)-based Waste Bank Information System, based on Green IT principles. A quantitative descriptive approach was applied by measuring four indicators: system performance, data transfer efficiency, estimated energy consumption, and digital carbon footprint under cached and non-cached loading scenarios. The results demonstrate that service worker caching significantly reduced data transfer size from an average of 4.84 MB to 20.9 kB, decreased estimated energy consumption from 0.0000594 kWh to 0.0000547 kWh, and reduced estimated carbon emissions from 0.044456 g CO_2 to 0.040886 g CO_2, representing approximately 8% improvements. Nevertheless, the overall page size remained relatively high (8.12 MB without caching and 7.46 MB with caching), while mobile rendering performance indicators, including First Contentful Paint (FCP) and Largest Contentful Paint (LCP), did not satisfy Web Vitals recommendations. This study contributes a practical measurement framework that integrates web performance analysis, energy efficiency assessment, and digital carbon footprint estimation for waste bank management systems. The findings provide empirical insights for developing more sustainable PWA-based information systems aligned with Green IT principles.

Keywords: Waste Bank Information System; Progressive Web Application; Green IT; Digital Carbon Footprint; Energy Efficiency

ISI Journal of Information Systems and Informatics
Published By Asosiasi Doktor Sistem Informasi Indonesia

1. INTRODUCTION

Waste management remains a pressing environmental concern, as the volume of waste continues to rise every year, particularly in developing countries such as Indonesia [1]. According to an article from the Information System of the Ministry of Environment, the percentage of waste that has been managed in Indonesia is 24.69% and is still relatively low. This prompted the government to establish a waste buying and selling based management program called the Waste Bank [1]. The existence of the program was enthusiastically welcomed by the community, especially Sidorukun Indah Gresik Housing which currently has 8 active waste banks. In its operations, the management of each RT is still carried out manually from recording to disbursement. However, these conditions from the results of observations and interviews pose problems related to transparency and time effectiveness so that the presence of a system is urgently needed to support digital transformation.

However, this digital transformation raises concerns regarding energy consumption and carbon emissions. In the digital world, technology used as a means of human activities such as digital interaction, data management, and network use can produce energy in the form of carbon gas that can have a negative impact on the environment [2]. From this phenomenon, the need for technology management, especially in creating technology that has an environmentally friendly architecture in accordance with Green IT Principles. From this principle, Progressive Web Apps-based technology can support Green IT Principles because by utilizing the Service Workers operating in it, the technology can carry out the caching system intelligently, able to run offline to produce a lighter payload [3]. Recent empirical studies conducted in the Indonesian context have shown that PWA caching measurably affects device-level energy/battery consumption [4], while broader comparative analyses indicate that PWAs generally offer competitive energy efficiency relative to other mobile development approaches [5]. This is expected to reduce the volume of data repeatedly transferred over the network, which depending on server efficiency, hosting infrastructure, and grid carbon intensity may contribute to lower estimated energy consumption and carbon emissions. Beyond caching alone, broader sustainable web design practices such as optimized image formats, font subsetting, and reduced unused assets have also been shown to meaningfully affect a website's environmental footprint [6].

ISI Journal of Information Systems and Informatics
Published By Asosiasi Doktor Sistem Informasi Indonesia

A number of researchers have previously conducted research that focused on performance and performance analysis on different domains and website architectures. According to research conducted by Siregar, the analysis was carried out on a study of the business scope with domains focused on E-Commerce that have non-complex requests [7]. In addition, according to research conducted by Arjanu and Suartana, the analysis was carried out on websites with personal-scale domains, namely online notebooks and using architecture for frontends using Next JS and for databases using Firebase [8].

Based on this gap, the present study addresses the following research questions:

1) RQ1: How does service-worker caching affect data transfer size and page-level performance metrics (TBT, FCP, LCP) of the Sibanksa system?
2) RQ2: How does caching affect the system's estimated carbon emissions and energy consumption?
3) RQ3: Which pages of the Sibanksa system require further optimization to align with Green IT Principles?

From the problems that have been formulated, this study specifically focuses on evaluating the environmental efficiency of the Sibanksa PWA implementation rather than PWA technology in general as a representative case for community-scale Green IT adoption in Indonesia's waste-bank digitalization efforts. In addition, this research is expected to provide evaluation related to the management website in the environmental scope study, namely waste banks that support the concept of Green IT Principles as the business goal of the waste bank, which is to support environmental friendliness. From this research, it is hoped that it can develop technology that not only focuses on the needs and problems of users, but also the external scale operations of the technology designed.

2. METHODS

2.1 Research Design

The research is carried out in several stages, namely collecting data with observations in the form of a website as a research object and also a literature review taken from research journals with topics relevant to the research carried out and then identifying

variables and indicators and implementation stages consisting of identifying test scenarios, measurement processes of test scenarios to analysis of test results as shown in Figure 1.

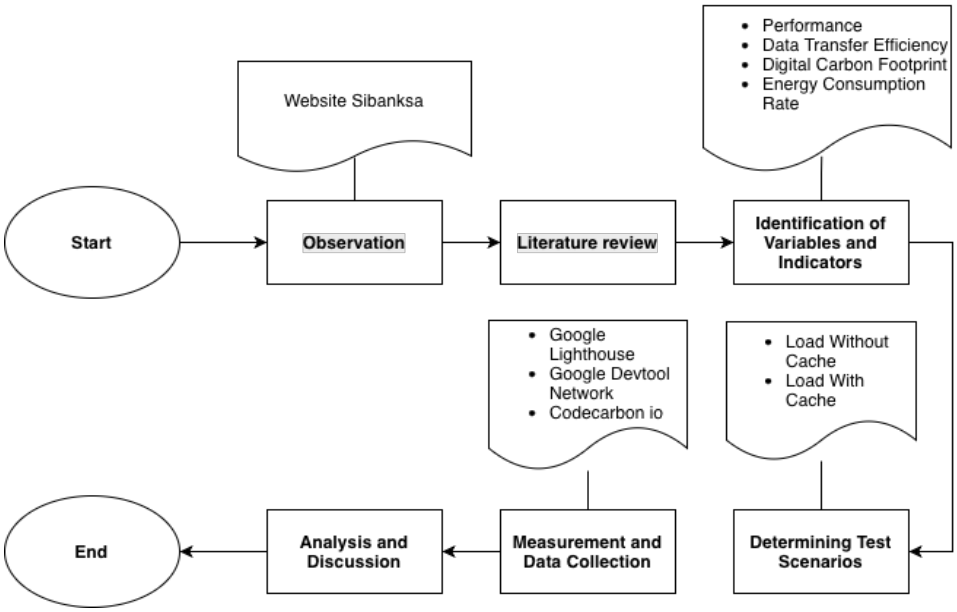

Figure 1. Research Flow Diagram

2.2 Types of Research

This study uses a quantitative approach based on measurement of artifact testing. The research method was chosen because of the analysis related to the System developed with the Progressive Web App in support of Green IT Principles. The system used in this study is Sibanksa (Waste Bank Information System). The analysis carried out on the system is in the form of measurement results related to the level of carbon emissions owned and also measurements related to the performance and performance of the system to be tested. The expected research results in this study are quantitative descriptive in the form of metrics resulting from system testing using tools in the form of Chrome Lighthouse tools, Chrome Devtools Network, and CodeCarbon Io, by analyzing the measurement metrics results of each tool that is focused on the use of carbon emissions owned by the system.

2.3 Research Object and Scope

This research uses a system that has been developed with PWA (Progressive Web Apps), namely Sibanksa as the object of the research. Sibanksa Information System that focuses on managing waste bank operations related to the recording of customer deposits until

disbursement. The system is the result of development based on the needs of 8 Sidorukun Indah Gresik Housing Waste Banks in each RT, which was developed using the Laravel framework with a display using inertia.js and Vue 3. However, as of the testing period (28–30 June 2026), only 5 of these waste banks were actively operating and using the Sibanksa system, while the remaining 3 are targeted for onboarding in subsequent phases. In its implementation, to limit the research to be more focused and directed, this research has a scope as a limitation of research in the form of measurements only being carried out on the pages that are most frequently used by users with the main focus that has the most important role in the business processes of the system and operations.

2.4 Test Success Variables and Indicators

Before the measurement was carried out, this study used 4 variables as a reference so that the measurement was more structured with indicators of each variable for reference to the results of the analyzed metrics described in the table 1

Table 1. Research Variables and Indicators

Variable	Indicator	Tools
Performance	TBT, FCP, LCP	Chrome Lighthouse
Data Transfer Efficiency	Total Page size, number of requests, and transfer size	Chrome Devtools Network
Digital Carbon Footprint	Amount of Carbon on each page visit	CodeCarbon io
Energy Consumption Rate	Amount of Energy Consumption Render	CodeCarbon io

Based on the description in the table above, 4 measurement variables are obtained, namely the performance variable to measure the performance of the website, the data transfer efficiency variable to measure the speed of website data transfer, the digital carbon footprint variable to measure the amount of carbon emissions that each page request has, and the energy consumption level variable to measure the amount of CPU energy produced when rendering the page.

2.5 Testing Environment

To ensure the reproducibility of the measurement results, the testing environment is described as follows. All measurements were conducted using a Macbook Pro M3 14-inch laptop with chip Apple M3, 8GB RAM, macOS Tahoe 26.5 as the testing device, using Google Chrome version 150.0.7871.128 as the primary browser for Lighthouse and Chrome DevTools measurements and for carbon and energy footprint measurement using Google Colab. The network condition during testing was a stable WiFi connection with an average speed of 50 Mbps download / 20 Mbps upload, without artificial throttling. The Sibanksa system was hosted on Hostinger's server Asia (Indonesia). All measurements were conducted between 28–30 June 2026.

2.6 Testing Scenarios

In its implementation, measurements are carried out by testing based on each variable with analysis carried out based on indicators of previously identified variables. To measure these four variables, there are 2 scenarios identified to be implemented in the test presented in the table 2

Table 2. Test Measurement Scenarios

Scenario	Conditions	Page
Load without Cache	Service workers on, website cache clean, internet active	Login, Register, Implementation Schedule Management, Waste Management, Record Management
Load with Cache	Service workers are active, website cache is contained, internet is active	

Based on Table 2, 2 test scenarios with different conditions were obtained. The scenario is carried out to test each indicator that has been identified in table 1. Each indicator is tested on the platform, but the performance test indicator is tested on different platforms, namely mobile and desktop. Both mobile and desktop performance metrics were obtained using Google Chrome Lighthouse without CPU or network throttling applied. Mobile metrics were measured using Lighthouse's mobile emulation mode (device viewport and user-agent simulation only, without throttling), while desktop metrics were obtained using Lighthouse's desktop configuration, both reflecting the actual network and hardware conditions of the testing environment.

2.7 Analytical Test and Mold Instruments

Before data management, to test each of these variables, use tools that can help the research run, such as

1) Chrome Lighthouse

   These tools are used to measure website performance and performance with measurement indicators including TBT, LCP, and FCP obtained from page load results which have the highest percentage of contribution weight compared to other metrics [9]. To achieve the success of the performance test, each indicator has a threshold, such as for TBT < 200ms, FCP < 1.8 s, and LCP < 2.5 s [10], [11].

2) Chrome Devtools Network

   These tools are used to measure the efficiency of data transfer. In measuring the data efficiency of a website, it can be seen from the results of website loading in the form of total page size, number of requests, and data transfer that occurs with total page size as the main measurement reference [12]. The threshold for efficient data transfer success from the total page size is < 2mb [13].

3) CodeCarbon IO

   It is a python library that is useful for measuring the content of carbon gas and energy emissions consumed in the operation of a website [14], [15]. In this study, digital carbon footprint and energy consumption were measured using CodeCarbon.io's EmissionsTracker, which was invoked around each HTTP request (tracker.start() before the request and tracker.stop() after the response was received) issued via Python's requests library. This approach captures the energy drawn by the local testing machine's CPU and RAM during the execution of the measurement script itself including request dispatch, response handling, and CodeCarbon's own monitoring overhead — rather than a full browser rendering (JavaScript execution, DOM construction, and asset rendering) or the physical energy consumed by network infrastructure and Sibanksa's hosting server. The resulting energy value (kWh) was then multiplied by the carbon intensity factor (Emissions = Energy × Carbon Intensity) to obtain the estimated carbon footprint in gCO2e. Consequently, the reported values should be interpreted as a proxy estimate of the computational cost of the page-load request, consistent with the exploratory/estimation-based nature of this study

noted in the Conclusion, rather than a direct measurement of end-to-end browser or network energy use. It should be noted that no separate network-energy-intensity coefficient (e.g., kWh per GB of data transferred) is applied in this study; energy is derived solely from CodeCarbon's direct hardware-level measurement around the script execution, using CodeCarbon's world-average carbon-intensity default of 475 gCO2e/kWh replaced with a forced Indonesia-specific value of 748 gCO2e/kWh via the force_carbon_intensity_g_co2e_kwh parameter.

2.8 Data Analysis Techniques

The test results from the research were processed in the form of analysis of engineering techniques as follows:

1) Quantitative Descriptive Analysis

   The technique in this study is carried out by analyzing data taken quantitatively and then analyzing it descriptively with the ultimate goal of interpretation of the results of the descriptive analysis [16].

2) Comparative Study of the Standard value of each indicator

   In the evaluation, the test results in this study are carried out by comparing the results of the studies that have been carried out in the form of reference values of each indicator as a standardization of the comparison of test results carried out with the standard values of each indicator.

3) Test Measurement Replication

   Performance test measurement was carried out by replicating each test scenario carried out 3 times (K=3). This aims to reduce the risk that occurs in the experiment and see the dissemination of data more widely. The risks that occur as a basis for test replication are due to external factors in the website test such as network constraints, slow cache status and server computing load.

3. RESULTS AND DISCUSSION

3.1 Website Identification

The website identification stage was conducted through direct observation and analysis of the system under evaluation to determine the characteristics, architecture, and functional scope of the tested object. The system selected in this research is Sibanksa

(Waste Bank Information System), a web-based platform developed to support waste bank management activities in the Sidorukun Indah Housing Area, Gresik, Indonesia. The system aims to digitalize waste bank operational activities, including user access management, waste collection scheduling, waste data management, and recording of waste deposits.

Sibanksa was developed using a modern web development approach by combining Native Progressive Web Application (PWA) concepts with a full-stack framework architecture. The backend system was implemented using the Laravel framework, while the frontend interface was developed using Inertia.js and Vue.js. This technology combination enables the system to provide responsive user interaction, efficient data processing, and a more application-like experience through PWA capabilities.

The general overview of the research object is presented in Table 3, which summarizes the main characteristics of the Sibanksa website, including the system name, development concept, and technology framework used. Based on the identification results shown in Table 3, Sibanksa can be categorized as a modern web application that integrates backend and frontend technologies to support digital transformation in waste bank management.

Table 3. Overview of Research Objects

Parameter	Description
Test Object	Sibanksa Website
Website Name	Sibanksa
Development Concept	Progressive Web Application (PWA) and Native Web Application
Backend Framework	Laravel
Frontend Framework	Inertia.js and Vue.js

The identification process in this stage is important because understanding the system architecture and functional characteristics provides the foundation for determining the appropriate testing scope. Furthermore, the identified features and workflows become the reference for selecting the application pages and activities evaluated in the subsequent testing process.

3.2 Existing Business Process System

Before conducting website testing, an analysis of the existing business process implemented in the Sibanksa system was performed to understand how each system component supports the operational activities of the waste bank. This analysis was conducted to ensure that the selected test features represent the main functional processes performed by users within the system.

Based on the observation results, the Sibanksa system consists of several interconnected business activities starting from user authentication, activity scheduling, waste management, and waste deposit recording. The authentication process serves as the initial access control mechanism, allowing users to register and log into the system based on their account privileges. After successful authentication, users can access the main waste bank management features.

The waste bank operational process begins with the management of waste collection schedules, where information regarding waste collection activities is organized and communicated to users. Subsequently, the waste management feature is used to maintain information related to waste categories and available waste types. The final core activity is waste deposit recording, which functions as the main transaction process for recording waste submissions from users and managing waste bank activity data.

The overall business process flow of the Sibanksa system is illustrated in Figure 2. As shown in Figure 2, each functional component is connected sequentially, starting from user authentication until the recording of waste deposit transactions. This workflow representation provides a clear understanding of the system boundaries and helps determine the functional areas that require evaluation during the testing phase.

The identification of existing business processes is essential because system testing should not only focus on individual pages but also consider whether the tested features represent important operational activities. Therefore, the business process analysis becomes the basis for selecting representative test scenarios that reflect real user interactions with the Sibanksa platform.

ISI Journal of Information Systems and Informatics
Published By Asosiasi Doktor Sistem Informasi Indonesia

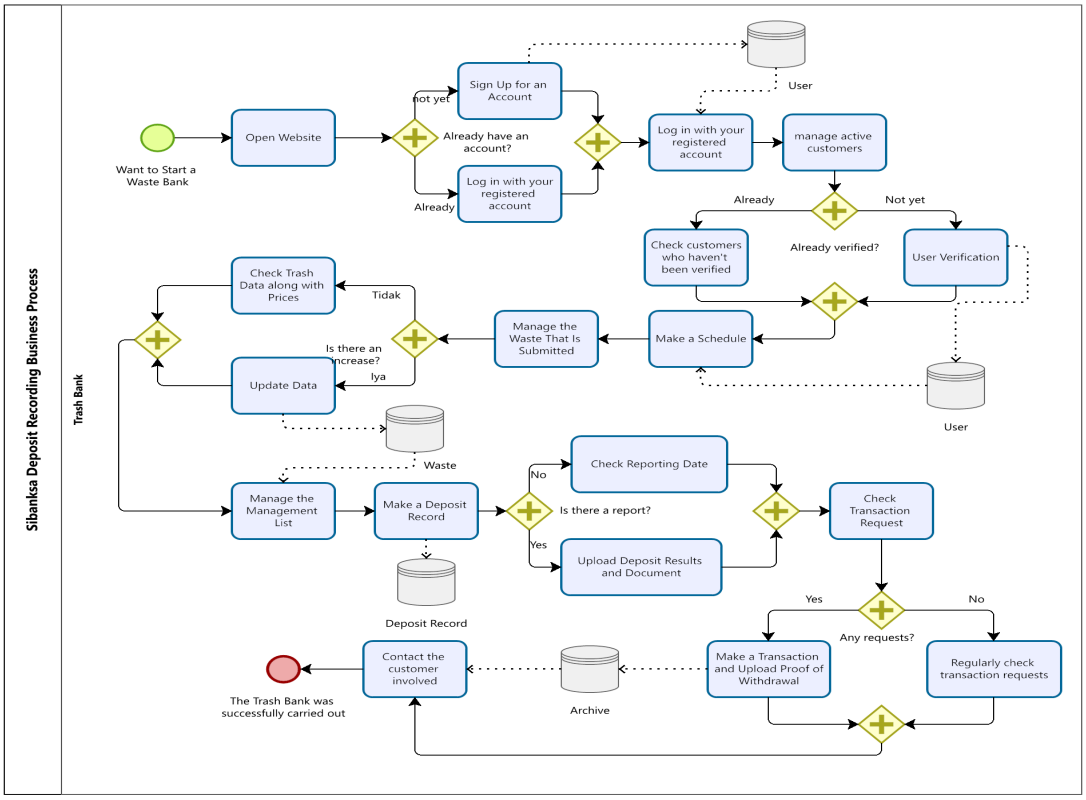

Figure 2. Sibanksa's Existing Business Process

3.3 Identify Test Features

The identification of test features was conducted based on the analysis of the existing business processes and functional components available in the Sibanksa system. The objective of this stage is to determine the specific system activities that will be evaluated during the testing process. The selected features represent the main interactions between users and the system, particularly those related to authentication, information management, and waste bank transactions.

Based on the observation results, five main features were identified as testing objects: login, register, implementation schedule management, waste management, and waste deposit recording. These features were selected because they represent critical processes required for the system to operate properly. The details of the identified test features, including activity codes, page names, and system domains, are presented in Table 4.

Table 4. Identify Test Features

Activity Code	Page	Domain
Act1	Login	/login
Act2	Register	/register
Act4	Implementation Schedule Garbage	/bank-sampah/Jadwal
Act5	Waste	/bank-sampah/Sampah
Act7	Deposit	/bank-sampah/pencatatan

As presented in Table 4, each activity code represents a specific functional module within the Sibanksa website. The login and registration features (Act1 and Act2) were selected to evaluate the accessibility and account management functionality of the system. Meanwhile, the waste collection schedule feature (Act4) represents the information management process that supports coordination between waste bank administrators and users. Furthermore, the waste management feature (Act5) and waste deposit recording feature (Act7) were selected because they represent the primary operational activities of the waste bank system. These features involve data interaction, transaction processing, and information storage, which are important aspects for evaluating the reliability and usability of the application. Therefore, the identified features in Table 4 provide a structured testing scope that covers both basic system access functions and core waste bank management processes. This selection ensures that the evaluation results can represent the overall functional performance of the Sibanksa website rather than only testing isolated components.

3.4 Performance Testing

Performance testing was conducted to evaluate the responsiveness, loading efficiency, and overall user experience quality of the Sibanksa website. The objective of this evaluation is to measure how effectively the system delivers web content to users, particularly in terms of page loading speed and browser execution performance. Website performance is an important quality attribute because slow response times can negatively affect user interaction, accessibility, and overall system usability.

In this study, Google Chrome Lighthouse was used as the performance evaluation tool. Lighthouse is an automated auditing tool that analyzes web applications based on several

performance indicators and provides quantitative measurements related to page loading behavior, rendering processes, and browser execution efficiency. The use of Lighthouse allows the performance characteristics of the Sibanksa website to be evaluated based on standardized metrics commonly applied in modern web application assessment.

The performance evaluation focused on several key indicators, including Total Blocking Time (TBT), First Contentful Paint (FCP), and Largest Contentful Paint (LCP). TBT measures the amount of time during which the browser main thread is blocked and unable to respond effectively to user interactions. A lower TBT value indicates better responsiveness because the website can process user input more efficiently. Meanwhile, FCP represents the time required for the browser to display the first visible content element after the page request is initiated, reflecting the initial loading experience perceived by users. LCP measures the rendering time of the largest visible content element within the viewport, which is commonly associated with the perceived completion of page loading.

The performance testing results generated by Chrome Lighthouse are presented in Figure 3. The figure illustrates the performance measurement results obtained from the Sibanksa website based on the selected indicators. These results provide an overview of the system's capability in delivering content and responding to user requests under the tested environment.

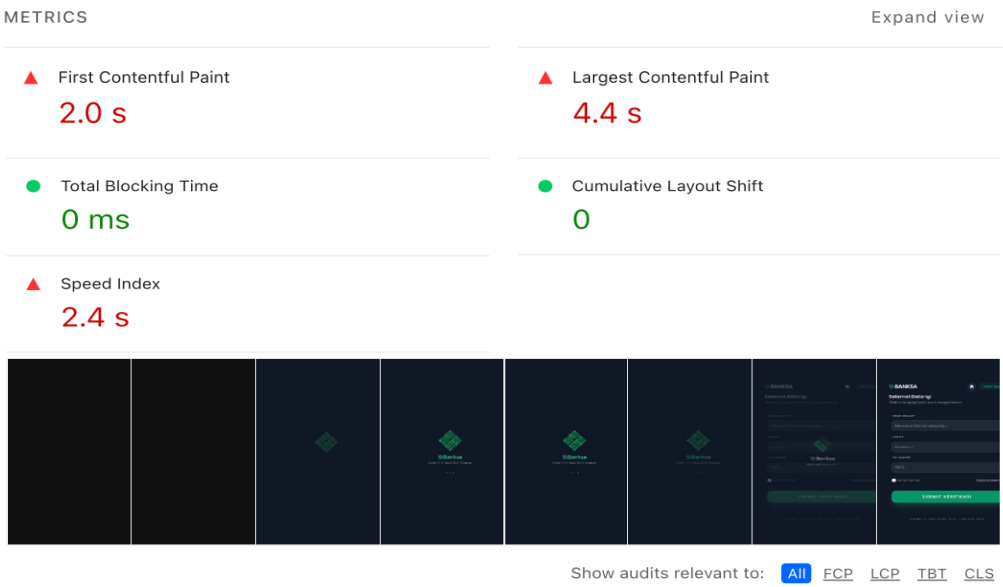

Figure 3. Chrome Lighthouse Testing

Based on the measurements shown in Figure 2, the performance indicators provide an objective assessment of the Sibanksa website's loading characteristics. The combination of TBT, FCP, and LCP metrics allows the evaluation to cover both technical performance aspects, such as browser execution efficiency, and user-centered aspects, such as perceived loading speed. Therefore, the Lighthouse-based evaluation provides a comprehensive understanding of the website performance quality and serves as a reference for identifying potential optimization areas in the Sibanksa system.

1) TBT Measurement

Table 5 presents a test that was carried out by measuring the performance of TBT indicators on Desktop and Mobile platforms based on 2 test scenarios, namely load without cache and load with cache.

Table 5. TBT Measurement (Desktop)

Scenario	Code	TBT (ms) (Desktop)					
		Test 1	**Test 2**	**Test 3**	**Mean**	**Std**	**CV (%)**
	Act1	0	0	20	6,67	11,55	173,21
Load	Act2	0	0	0	0,0	0,0	0.00
Without	Act4	100	80	150	110,00	36,06	32,78
Cache	Act5	30	20	0	16,67	15,28	91,65
	Act7	50	70	90	70,00	20,00	28,57
Load	Act1	0	0	10	3,33	5,77	173,21
With	Act2	0	10	0	3,33	5,77	173,21
Cache	Act4	20	0	0	6,67	11,55	173,21
	Act5	0	0	0	0,00	0,00	0,00
	Act7	0	0	0	0,00	0,00	0,00

Based on the test results in Table 5, it can be seen that the implementation of caching has been shown to be successful in reducing Total Blocking Time (TBT) on desktop devices, where the average TBT in the load with cache scenario is consistently lower than load without cache across all actions tested, with the highest decrease occurring in Act4 (from 110.00 ms to 6.67 ms) and decrease of up to 0 ms in Act5 and Act7. Not only measuring the desktop platform, the test was carried out by measuring the mobile platform presented in Table 6.

Table 6. TBT Measurement (Mobile)

Scenario	Code	TBT (ms) (Mobile)					
		Test 1	**Test 2**	**Test 3**	**Mean**	**Std**	**CV (%)**
	Act1	60	90	60	70	17,32	24,75
Load	Act2	100	80	140	106,67	30,55	28,64
Without	Act4	0	0	70	23,33	40,41	173,21
Cache	Act5	30	30	40	33,33	5,77	17,32
	Act7	0	0	30	10	17,32	173,21
Load	Act1	70	100	90	86,67	15,28	17,63
With	Act2	110	80	90	93,33	15,28	16,37
Cache	Act4	30	0	0	10	17,32	173,21
	Act5	0	0	0	0	0	–
	Act7	0	40	40	26,67	23,09	86,58

In contrast to the results on desktop, the test results in Table 6 show that there is an inconsistent pattern between the load without cache and load with cache scenarios — in some test scenarios such as Act1 and Act2, TBT actually increases after caching is applied (Act1 from 70.00 ms to 86.67 ms; Act2 from 106.67 ms to 93.33 ms, although Act2 remained slightly decreasing), while Act4 and Act5 caching significantly reduced TBT (Act4 from 23.33 ms to 10.00 ms; Act5 from 33.33 ms to 0.00 ms). The increase in TBT in Act1 and Act2 is likely due to the limited CPU processing capacity on mobile devices that are lower than desktop, due to the process of reading and cache validation (cache lookup/matching) by Service Workers.

2) FCP Measurement

The following is a test that has been carried out by measuring the performance of FCP indicators on Desktop and Mobile platforms based on 2 test scenarios, namely load without cache and load with cache. Based on Table 7, tests related to FCP indicator on desktop devices showed that the implementation of caching did not consistently provide improvements, even in some actions tended to slow down the rendering time of the first content; Act1 and Act4 were relatively stable with little or no significant change (Act1 from 2.03 s to 1.93 s; Act4 from 2.63 s to 2.60 s), but Act2 and Act5 actually experienced an increase in FCP after caching was applied (Act2 from 1.90 s to 2.10 s with a very high

ISI Journal of Information Systems and Informatics
Published By Asosiasi Doktor Sistem Informasi Indonesia

variability CV of 73.92%; Act5 from 2.70 s to 4.23 s). In addition, high variability results (CV%) for Act2 and Act5 in both scenarios also suggest that the results of FCP measurements were less stable between experiments. Not only measuring the desktop platform, the test was carried out by measuring the mobile platform presented in Table 8.

Table 7. FCP Measurement (Desktop)

Scenario	Code	FCP (s) (Desktop)					
		Test 1	Test 2	Test 3	Mean	Standard Deviation	CV (%)
Load Without Cache	Act1	2,1	2,0	2,0	2,03	0,06	2,84
	Act2	1,9	1,9	1,9	1,9	0	–
	Act4	2,9	2,5	2,5	2,63	0,23	8,77
	Act5	2,6	2,0	3,5	2,7	0,76	27,96
	Act7	2,0	2,6	3,0	2,53	0,5	19,87
Load With Cache	Act1	1,9	1,9	2,0	1,93	0,06	2,99
	Act2	0,5	2,2	3,6	2,1	1,55	73,92
	Act4	2,4	2,8	2,6	2,6	0,2	7,69
	Act5	3,8	5,6	3,3	4,23	1,21	28,58
	Act7	2,0	2,6	2,6	2,4	0,35	14,43

Table 8. FCP Measurement (Mobile)

Scenario	Code	FCP (s) (Mobile)					
		Test 1	Test 2	Test 3	Mean	Std	CV (%)
Load Without Cache	Act1	10,1	10,0	10,1	10,07	0,06	0,57
	Act2	9,9	10,0	10,0	9,97	0,06	0,58
	Act4	13,2	13,2	13,2	13,2	0	0
	Act5	11,8	11,8	11,8	11,8	0	0
	Act7	13,2	13,2	11,9	12,77	0,75	5,88
Load With Cache	Act1	10,0	10,0	10,1	10,03	0,06	0,58
	Act2	9,9	10,0	11,7	10,53	1,01	9,6
	Act4	12,6	13,2	13,2	13	0,35	2,67
	Act5	11,8	11,8	11,7	11,77	0,06	0,49
	Act7	13,2	13,2	13,2	13,2	0	0

Based on the FCP test results in table 8, the mean values in both scenarios ranged from 9.97 s to 13.20 s — well above the standard of good performance. The application of caching on mobile devices also did not show significant improvements, even in some actions such as Act2 there was an increase in FCP (from 9.97 s to 10.53 s) accompanied by a fairly high variability (CV 9.60%), while Act4 was relatively stable with a slight decrease (from 13.20 s to 13.00 s) and Act5 was almost unchanged (11.80 s to 11.77 s). In addition, the relatively low variability results in most actions (CV below 3%) indicate that the measurement results tend to be stable and repeatable.

3) LCP Measurement

Table 9 presents a test that has been carried out by measuring the performance of LCP indicators on Desktop and Mobile platforms based on 2 test scenarios, namely load without cache and load with cache.

Table 9. LCP Measurement (Desktop)

Scenario	Code	LCP (s) (Desktop)					
		Test 1	Test 2	Test 3	Mean	Std	CV (%)
	Act1	4,5	4,4	4,4	4,43	0,06	1,3
Load	Act2	4,3	4,4	4,3	4,33	0,06	1,33
Without	Act4	3,9	3,4	4,3	3,87	0,45	11,66
Cache	Act5	4,5	4,2	4,5	4,4	0,17	3,94
	Act7	3,4	3,6	3,9	3,63	0,25	6,93
Load	Act1	4,3	4,3	4,4	4,33	0,06	1,33
With	Act2	0,5	4,7	5,1	3,43	2,55	74,24
Cache	Act4	3,3	3,7	3,9	3,63	0,31	8,41
	Act5	4,0	5,8	4,5	4,77	0,93	19,5
	Act7	3,3	3,9	3,5	3,57	0,31	8,56

Based on Table 9, the results of the Largest Contentful Paint (LCP) test on desktop devices show that the entire mean value, in both the load without cache and load with cache scenarios, is in the range of 3.43 s to 4.77 s which indicates that the largest content element on the page (usually an image, video, or main text block) requires a relatively long rendering time under both conditions. The caching implementation here also doesn't

show consistent improvements; Act4 and Act7 actually experienced a slight decline (Act4 from 3.87 s to 3.63 s; Act7 from 3.63 s to 3.57 s), but Act5 deteriorated (from 4.40 s to 4.77 s) and Act2 showed quite extreme variability (CV 74.24%). Not only measuring the desktop platform, the test was carried out by measuring the mobile platform presented in Table 10.

Table 10. LCP Measurement (Mobile)

Scenario	Code	LCP (s) (Mobile)					
		Test 1	Test 2	Test 3	Mean	Std	CV (%)
Load Without Cache	Act1	24,5	24,5	24,5	24,5	0	0
	Act2	24,9	24,9	24,9	24,9	0	0
	Act4	21,5	21,5	26,2	23,07	2,71	11,77
	Act5	23,4	23,6	23,6	23,53	0,12	0,49
	Act7	21,5	21,6	21,5	21,53	0,06	0,27
Load With Cache	Act1	24,2	24,9	24,2	24,43	0,4	1,65
	Act2	24,8	24,9	21,6	23,77	1,88	7,9
	Act4	21,5	21,5	21,6	21,53	0,06	0,27
	Act5	21,7	21,8	21,7	21,73	0,06	0,27
	Act7	21,5	26,2	26,2	24,63	2,71	11,02

Based on the tests that have been presented in table 10, LCP test results on mobile devices, the entire mean ranges from 21.53 s to 24.90 s. This value indicates a serious performance issue on the mobile device, most likely caused by a combination of the mobile device's CPU processing limitations, the size of the content element being too large for rendering. The caching implementation in this scenario also did not show that Act4 and Act5 were relatively stable and slightly improved (Act4 from 23.07 s to 21.53 s), but Act7 actually deteriorated (from 21.53 s to 24.63 s), while Act1 and Act2 showed no change. In addition, low variability results (CV%) in most scenarios (below 2%) indicate that measurement results tend to be stable. The results obtained from testing on all indicators are recapitulated related to performance measurement on the Sibanksa website which is described in Table 11.

Table 11. Website Performance Testing Measurement

Scenario	Code	TBT (ms)		FCP (s)		LCP (s)	
		Mobile	Desktop	Mobile	Desktop	Mobile	Desktop
Load Without Cache	Act1	70	6,67	10,07	2,03	24,5	4,43
	Act2	106,67	0	9,97	1,9	24,9	4,33
	Act4	23,33	110	13,2	2,63	23,07	3,87
	Act5	33,33	16,67	11,8	2,7	23,53	4,4
	Act7	10	70	12,77	2,53	21,53	3,63
Average		48,67	40,67	11,56	2,36	23,51	4,13
Load With Cache	Act1	86,67	3,33	10,03	1,93	24,43	4,33
	Act2	93,33	3,33	10,53	2,1	23,77	3,43
	Act4	10	6,67	13	2,6	21,53	3,63
	Act5	0	0	11,77	4,23	21,73	4,77
	Act7	26,67	0	13,2	2,4	24,63	3,57
Average		43,33	2,67	11,71	2,65	23,22	3,95

Based on the tests that have been conducted, each scenario is measured to determine the duration based on indicators on the performance variables tested on different platforms. From the results of these measurements, it was found that the average was quite stable between 2 scenarios but very significant between mobile platforms compared to desktop.

3.5 Data Transfer Efficiency Testing

In the implementation of the test, the tools used namely chrome devtools network to measure performance indicators, such as Total Page size, number of requests, and transfer size, as shown in Figure 4.

1) Total Page Size Measurement

The following is a test that has been carried out by measuring total page size based on 2 test scenarios, namely load without cache and load with cache.

ISI Journal of Information Systems and Informatics
Published By Asosiasi Doktor Sistem Informasi Indonesia

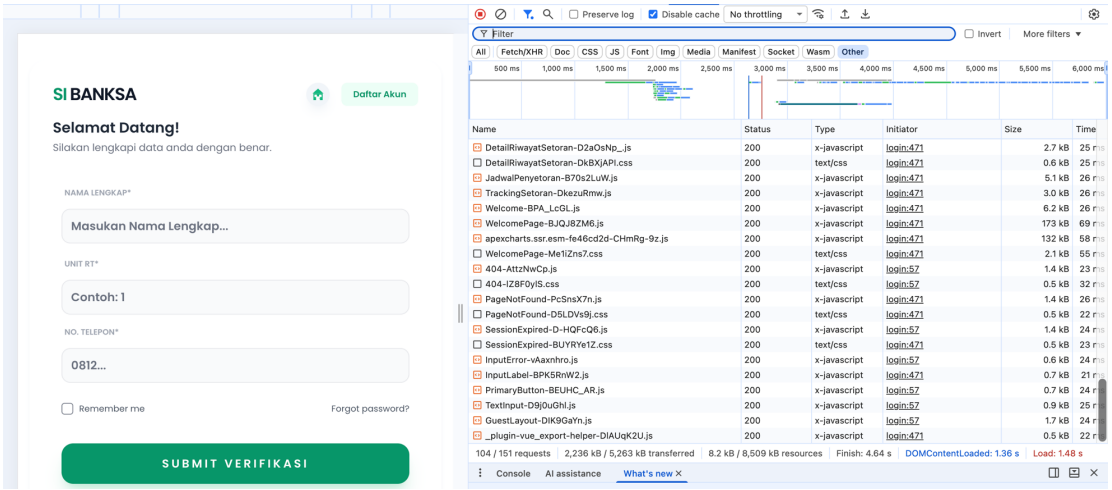

Figure 4. Chrome Devtools Network Testing

Table 12. Total Page Size Testing Measurements

Scenario	Code	Total Page Size (Mb)					
		Test 1	Test 2	Test 3	Mean	Std	CV (%)
Load Without Cache	Act1	8,4	8,4	8,4	8,4	0	0
	Act2	8,8	8,8	8,8	8,8	0	0
	Act4	8,2	8,2	8,2	8,2	0	0
	Act5	8,2	8,2	8,2	8,2	0	0
	Act7	7	7	7	7	0	0
Load With Cache	Act1	6,8	6,8	6,8	6,8	0	0
	Act2	7,1	7,1	7,1	7,1	1,087	1,53
	Act4	8,2	8,2	8,2	8,2	0	0
	Act5	8,2	8,2	8,2	8,2	0	0
	Act7	7	7	7	7	0	0

According to the measurements in the Table 12, the total page size varies between the tested environments. In the Load Without Cache setup, average values range from 7 Mb to 8.8 Mb. Introducing cache (Load With Cache) successfully decreases the page size for certain activities, such as Act1 (from 8.4 Mb down to 6.8 Mb) and Act2 (from 8.8 Mb down to 7.1 Mb), while Act4, Act5, and Act7 stay identical at 8.2 Mb, 8.2 Mb, and 7 Mb. Furthermore, the metrics reflect high precision, as almost all scenarios achieved a Standard Deviation of 0 and a CV of 0%, except for Act2 under cached conditions (Standard Deviation = 1.087, CV = 1.53%). Overall, the implementation of cache proves useful in streamlining page size.

2) Number Of Requests

Table 13 presents a test that has been carried out by measuring the number of requests based on 2 test scenarios, namely load without cache and load with cache. According to the test data outlined in Table 13, In the Load Without Cache scenario, mean request values range between 147 and 160. Upon activating cache (Load With Cache), request numbers consistently drop across test cases, reaching a lower average range of 143 to 158. This drop shows that cached resources successfully eliminate the need to re-request certain components from the server. Additionally, both test conditions achieved absolute consistency, displaying a Standard Deviation of 0 and a CV of 0% throughout all activities (Act1 to Act7). In conclusion, applying cache proves effective in reducing server request overhead while maintaining consistent delivery.

Table 13. Number Of Requests Testing Measurements

Scenario	Code	Number of Requests					
		Test 1	Test 2	Test 3	Mean	Std	CV (%)
Load Without Cache	Act1	151	151	151	151	0	0
	Act2	147	147	147	147	0	0
	Act4	160	160	160	160	0	0
	Act5	160	160	160	160	0	0
	Act7	157	157	157	157	0	0
Load With Cache	Act1	143	143	143	143	0	0
	Act2	143	143	143	143	0	0
	Act4	158	158	158	158	0	0
	Act5	158	158	158	158	0	0
	Act7	156	156	156	156	0	0

3) Transfer Size

Table 14 presents a test that has been carried out by measuring transfer size based on 2 test scenarios, namely load without cache and load with cache. Based on the tests presented in Table 14, the transfer size test results show a significant difference between the scenarios. In the Load Without Cache scenario, the mean transfer size ranges from 4200 Kb to 5300 Kb or equivalence to 4,2 mb to 5,3 Mb. On the other hand, implementing cache in the Load With Cache scenario results in a drastic reduction in data transfer size,

with the mean dropping significantly to a range of 18.7 Kb to 22.3 Kb. This substantial decrease indicates that the caching mechanism successfully optimizes network efficiency by serving stored resources locally rather than downloading full assets repeatedly. Furthermore, the Load With Cache scenario exhibits a Standard Deviation of 0 and a Coefficient of Variation (CV) of 0% across all test cases (Act1 to Act7), indicating absolute stability and high consistency in data delivery during cached conditions. Overall, these results demonstrate that applying cache significantly improves resource loading performance while maintaining perfect stability across repeated tests. The results obtained from testing on all indicators are recapitulated related to Data Transfer measurement on the Sibanksa website which is described in Table 15.

Table 14. Transfer Size Testing Measurements

Scenario	Code	Transfer Size (Kb)					
		Test 1	Test 2	Test 3	Mean	Std	CV (%)
Load Without Cache	Act1	5100	5100	5100	5100	0	0
	Act2	5300	5300	5300	5300	0	0
	Act4	4800	4800	4800	4800	0	0
	Act5	4800	4800	4800	4800	0	0
	Act7	4200	4200	4200	4200	0	0
Load With Cache	Act1	19	19	19	19	0	0
	Act2	18,7	18,7	18,7	18,7	0	0
	Act4	22,2	22,2	22,2	22,2	0	0
	Act5	22,3	22,3	22,3	22,3	0	0
	Act7	22,3	22,3	22,3	22,3	0	0

Table 15. Data Transfer Efficiency Testing Measurements

Scenario	Code	Total Page Size (Mb)	Number of Requests	Transfer Size (Kb)
Load Without Cache	Act1	8,4	151	5100
	Act2	8,8	147	5300
	Act4	8,2	160	4800
	Act5	8,2	160	4800
	Act7	7,0	157	4200

ISI Journal of Information Systems and Informatics
Published By Asosiasi Doktor Sistem Informasi Indonesia

Scenario	Code	Total Page Size (Mb)	Number of Requests	Transfer Size (Kb)
Average		8,12	155	4840
Load With Cache	Act1	6,8	143	19.0
	Act2	7,1	143	18.7
	Act4	8,2	158	22.2
	Act5	8,2	158	22.3
	Act7	7,0	156	22.3
Average		7,46	151,6	20,9

In the efficient testing of data transfer, it is carried out by testing based on 2 scenarios, namely load without cache and with cache load with the main parameter as a reference, namely the total page size of each activity code. The test results obtained an average of 8.12 mb for without cached loads, 7.46 mb for cached loads.

4) Breakdown Asset Value

Based on the asset breakdown presented in Table 16, JavaScript emerges as the dominant contributor to total page size, accounting for an average of 70.26% (6.082 MB) across the five tested pages. Among the tested pages, Act4 (Implementation Schedule) recorded the highest total asset size at 11.88 MB, driven primarily by its JavaScript payload of 7.99 MB, while Act7 (Deposit) recorded the lowest at 6.66 MB. This result indicates that the JavaScript bundle size is largely fixed across the application rather than page-specific.

Table 16. Breakdown Type Asset Matrix

Asset Type / Code	Images (Mb)	JavaScript (Mb)	CSS (Mb)	Fonts (Mb)	Other /Documents (Mb)	Total Asset (Mb)
Act1	4,5	4,8	0,571	0,0236	0,0082	9,9028
Act2	1,7	4,7	0,566	0,0236	0,0082	6,9978
Act4	3,09	7,99	0,614	0,18	0,0082	11,8822
Act5	0,002	7,05	0,606	0,18	0,0082	7,8462
Act7	0,002	5,87	0,606	0,17	0,0082	6,6562
Average	1,8588	6,082	0,5926	0,11544	0,0082	8,65704

ISI Journal of Information Systems and Informatics
Published By Asosiasi Doktor Sistem Informasi Indonesia

Asset Type / Code	Images (Mb)	JavaScript (Mb)	CSS (Mb)	Fonts (Mb)	Other /Documents (Mb)	Total Asset (Mb)
Percentage of Contribution	21,47%	70,26%	6,85%	1,33%	0,09%	100%

Based on the asset breakdown in Table 16, where JavaScript accounts for 70.26% of total page size, the following optimization strategies are recommended for future development: (1) code splitting and tree-shaking to load only the JavaScript modules required per route, reducing framework overhead from Vue 3 and Inertia.js; (2) lazy loading of non-critical components and images below the fold; (3) image compression and modern format adoption (e.g., WebP) for the Login and Register pages, which recorded the largest image payloads (4.5 MB and 1.7 MB respectively); (4) font subsetting to reduce the 0.115 MB average font footprint; (5) HTTP compression (Gzip) and appropriate cache-control headers to further reduce transfer overhead; and (6) removal of unused CSS/JavaScript through build-time analysis (e.g., Webpack Bundle Analyzer).

From these recommendations, it given that waste bank administrators are volunteer community members who predominantly access the system via mobile devices during informal, on-site data collection, an LCP of over 20 seconds under emulated mobile conditions raises a practical usability concern: administrators may perceive the system as unresponsive during real-world use, potentially discouraging adoption. This suggests that mobile-first optimization should be prioritized over further network-layer caching improvements, since the current bottleneck lies in asset size and rendering rather than data transfer.

3.6. Digital Carbon Footprint Testing

In the implementation of the test, the tools used, namely codecarbon io to measure digital carbon footprint and energy indicators, such as Amount of Carbon on each page visit and energy of rendering as shown in figure 5.

ISI Journal of Information Systems and Informatics
Published By Asosiasi Doktor Sistem Informasi Indonesia

```
def loginWithCacheTest():

    pages = 'login'
    url_path = "/login"

    os.makedirs("./hasil_codecarbon", exist_ok=True)

    tracker = EmissionsTracker(
        project_name="SiBanksa_GreenIT",
        output_dir="./hasil_codecarbon",
        output_file="loginWithCache_results.csv",
        log_level="warning",
        force_carbon_intensity_g_co2e_kwh=748.0,  # intensitas karbon Indonesia
        save_to_file=True,
    )

    tracker.start()

    print(f"Mengukur halaman: {pages}")
    for i in range(3):
        response = requests.get(BASE_URL + url_path)
        print(f"  [{i+1}] Status: {response.status_code} | Size: {len(response.content)} bytes")
        time.sleep(1)

    emissions = tracker.stop()
    energy_Kwh = tracker._total_energy.kWh
    return (emissions, energy_Kwh)

emissions, energy_Kwh = loginWithCacheTest()
emissions_gram, rating, persentase_rating, emisi_per_tahun_kg, jumlah_pohon  = tampilkan_hasil("Login Dengan Cache", emissions, energy_Kwh, kunjungan_per_t
data_list = [
    ("Load Dengan Cache", "Act1",rating, persentase_rating, emissions_gram, jumlah_pohon)
```

Figure 5. Codecarbon Io Testing

3.7. Carbon Measurement

Table 17 presents a test that has been carried out by measuring Amount of Carbon on each page visit indicators based on 2 test scenarios, namely load without cache and load with cache.

Table 17. Carbon Footprint Testing Measurements

Scenario	Code	Carbon Footprint (g CO_2)					
		Test 1	Test 2	Test 3	Mean	Standard Deviation	CV (%)
Load Without Cache	Act1	0,036131	0,036036	0,03765	0,036606	0,000906	2,47
	Act2	0,035253	0,037276	0,034061	0,03553	0,001625	4,57
	Act4	0,055571	0,051741	0,060472	0,055928	0,004376	7,83
	Act5	0,051543	0,049724	0,050338	0,050535	0,000925	1,83
	Act7	0,042925	0,044088	0,044036	0,043683	0,000657	1,5
Load With Cache	Act1	0,037571	0,040656	0,032179	0,036802	0,004291	11,66
	Act2	0,039886	0,038428	0,036085	0,038133	0,001918	5,03
	Act4	0,041581	0,040985	0,046959	0,043175	0,003291	7,62
	Act5	0,043752	0,046507	0,039086	0,043115	0,003751	8,7
	Act7	0,043348	0,042837	0,043428	0,043204	0,000321	0,74

Based on the tests presented in Table 17, CO2 emission test results on the Sibanksa website indicate that the overall mean ranges from 0.0355 g CO2eq (Act2, Load Without Cache) to 0.0559 g CO2eq (Act4, Load Without Cache). This value suggests that the

activities with heavier content or additional server-side processing (Act4, Act5) generate a noticeably higher carbon footprint per visit than lighter activities such as Act1 and Act2. In addition, the variability results (CV%) across scenarios are considerably higher than those observed in the LCP test, ranging from 0.74% (Act7, Load With Cache) to 11.66% (Act1, Load With Cache), suggesting that carbon emission measurements are inherently more sensitive to environmental noise — such as background processes and momentary CPU/RAM fluctuations on the testing machine — and therefore less stable than rendering-based metrics like LCP. Notably, Act1 and Act2 show slightly higher mean carbon values under the cached scenario compared to uncached (0.036802 g vs 0.036606 g for Act1; 0.038133 g vs 0.035530 g for Act2), unlike the clearer reduction observed in Act4, Act5, and Act7. Given the high CV for Act1 under cache (11.66%), this is most plausibly explained by replication-level noise — such as background CPU load or service-worker cache-validation overhead — rather than caching genuinely increasing emissions, since these two lightweights, mostly static-asset pages have less to gain from caching in the first place. The results obtained from testing on all indicators are recapitulated related to Digital Carbon Footprint measurement on the Sibanksa website, which is described in Table 18.

Table 18. Digital Carbon Footprint Measurement

Scenario	Code	Co2 Quantity (g CO_2)
Load Without Cache	Act1	0,036606
	Act2	0,03553
	Act4	0,055928
	Act5	0,050535
	Act7	0,043683
Average		0,044456
Load With Cache	Act1	0,036802
	Act2	0,038133
	Act4	0,043175
	Act5	0,043115
	Act7	0,043204
Average		0,040886

In the measurement of the digital carbon footprint test, it is carried out by measuring based on 2 test scenarios, namely Load with cache and without cache. From the test

results, an average of each scenario was produced, namely for uncached loads, carbon emissions of 0.044456 g CO2 were obtained, and for cached loads, an average carbon emission of 0.040886 g CO2 was obtained, which is an ~8.0% reduction. In addition, based on the test results, it can also be seen that Act4, the implementation schedule management page, has the highest (worst) emission value for the load-without-cache scenario, while Act7 has the highest (worst) emission value for the load-with-cache scenario.

3.8. Energy Consumption Testing

1) Energy Consumption Measurement

Table 19 presents a test that has been carried out by measuring Amount of Carbon on each page visit indicators based on 2 test scenarios, namely load without cache and load with cache.

Table 19. Energy Rendering Testing Measurements

Scenario	Code	Carbon Footprint (g CO_2)					
		Test 1	Test 2	Test 3	Mean	Standard Deviation	CV (%)
Load Without Cache	Act1	0,0000483	0,00004817	0,00005033	0,00004893	0,00000121	2,48
	Act2	0,00004713	0,00004983	0,00004554	0,0000475	0,00000217	4,57
	Act4	0,00007429	0,00006917	0,00008084	0,00007477	0,00000585	7,82
	Act5	0,00006891	0,00006648	0,0000673	0,00006756	0,00000124	1,83
	Act7	0,00005739	0,00005894	0,00005887	0,0000584	0,00000088	1,5
Load With Cache	Act1	0,00005023	0,00005435	0,00004302	0,0000492	0,00000573	11,66
	Act2	0,00005332	0,00005137	0,00004824	0,00005098	0,00000256	5,03
	Act4	0,00005559	0,00005479	0,00006278	0,00005772	0,0000044	7,62
	Act5	0,00005849	0,00006217	0,00005225	0,00005764	0,00000501	8,7
	Act7	0,00005795	0,00005727	0,00005806	0,00005776	0,00000043	0,74

Based on the tests presented in Table 19, energy consumption test results on the Sibanksa website show that the overall mean ranges from 0.0000475 kWh (Act2, Load Without Cache) to 0.0000748 kWh (Act4, Load Without Cache). This value indicates that activities with heavier content or additional server-side processing (Act4, Act5) consume noticeably more energy per visit than lighter activities such as Act1 and Act2. The caching

implementation in this scenario shows a clear benefit for the heavier activities: Act4 and Act5 improved substantially with caching enabled (Act4 from 0.0000748 kWh to 0.0000577 kWh, and Act5 from 0.0000676 kWh to 0.0000576 kWh), while Act7 remained relatively stable (0.0000584 kWh to 0.0000578 kWh). It indicating that caching does not uniformly reduce energy consumption across all activities. In addition, the variability results (CV%) across scenarios range from 0.74% (Act7, Load With Cache) to 11.66% (Act1, Load With Cache), showing the same pattern of variability observed in the CO2 emission. The results obtained from testing on all indicators are recapitulated related to Energy Consumption measurement on the Sibanksa website, which is described in Table 20.

Table 20. Energy Consumption Measurement

Scenario	Code	Energy consumption (kWH)
Load Without Cache	Act1	0,00004893
	Act2	0,00004750
	Act4	0,00007477
	Act5	0,00006756
	Act7	0,00005840
	Average	0,00005943
Load With Cache	Act1	0,00004920
	Act2	0,00005098
	Act4	0,00005772
	Act5	0,00005764
	Act7	0,00005776
	Average	0,00005466

Based on Table 20, Energy consumption testing is carried out with 2 test scenarios, namely load without cache and with cache. From the measurement results, it was obtained that on average each scenario, namely for a load without a cache produces an energy of 0.00005943 kWH while for a load with a cache produces energy of 0.00005466 kWH.

3.9. Discussion

The results of the research conducted related to the test of system performance measurement tested on both platforms, namely mobile and desktop, as shown in figure

6, it can be seen that the desktop platform has better performance compared to mobile, this can be seen from the value of the indicator, which for TBT, FCP and LCP on the desktop platform while on the mobile platform only the TBT< 1.18s indicator is in the threshold. This is due to the effect that occurs from the FCP (>1.8 s) and LCP (2.5 s) indicators regarding the loading render of each large asset (image, font) that is forced to render with a fast time, thus affecting a fairly long processing time for medium-scale platforms such as mobile. This is happened because no artificial network or CPU throttling was applied during testing, the elevated FCP and LCP values on mobile emulation reflect genuine payload and rendering overhead rather than simulated network constraints, underscoring the need for asset optimization.

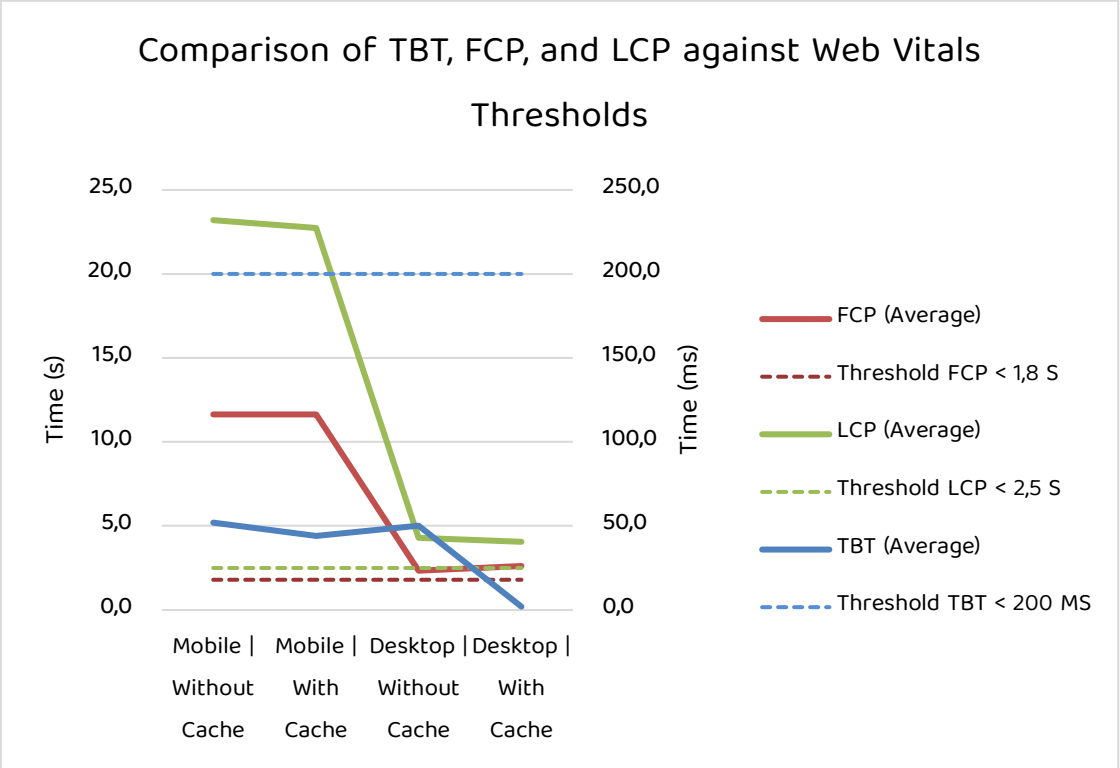

Figure 6. Comparison TBT, FCP, and LCP between Platform for Each Scenario

The results of the research conducted related to the measurement of data transfer efficiency can be seen in figure 7 that the Sibanksa system has a very efficient data transfer which was reduced from an average of 4,840 kB to 20.9 kB (a reduction of over 99% in average transfer size) when the system is in a render condition with a cache of Service Workers execution registered in it. However, the results of this measurement also produce a very large website page size of around 7400 – 8300 kb far from the ideal

threshold of 2000 kb which affects the render time for the page load of each asset which is related to the performance measurement indicators, namely FCP and LCP. This makes the size of the page affect the performance of the system, especially on medium-scale platforms such as mobile. Despite the substantial reduction in transfer size, the number of HTTP requests remained consistently high (143–160 requests) across both scenarios, indicating that caching reduces the volume of data transferred per request but does not reduce the number of individual requests issued. This suggests an opportunity for request bundling — combining multiple small JavaScript or CSS files into fewer bundles — and further leveraging of the Cache API to serve grouped assets from a single service-worker-managed request, rather than relying on caching alone to offset a high request count.

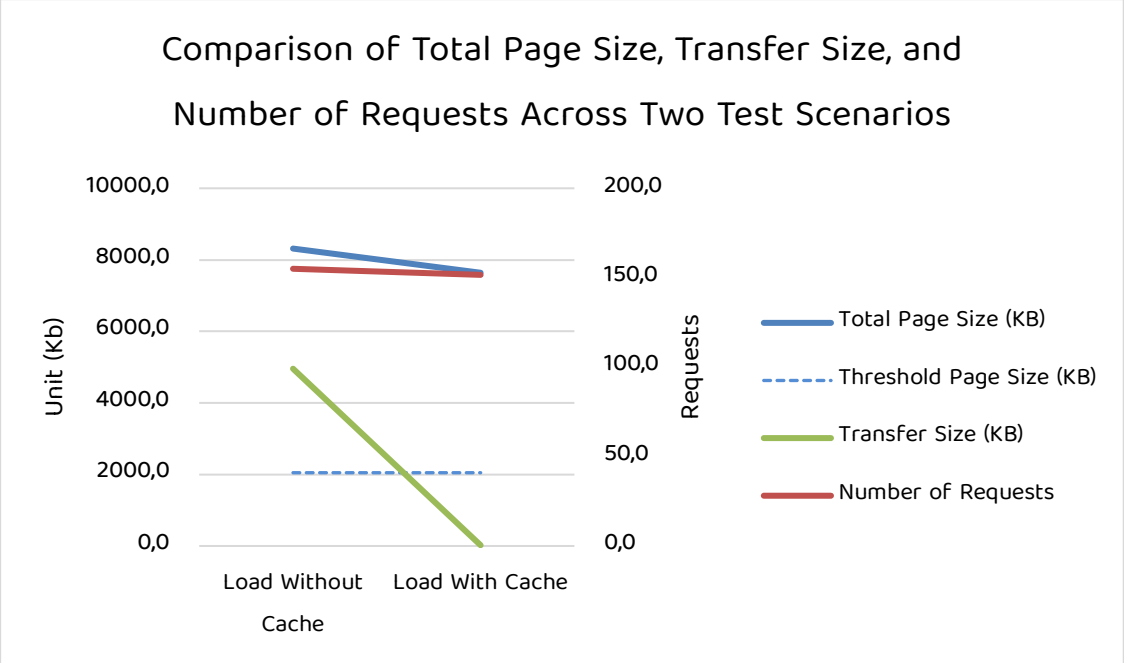

Figure 7. Comparison of Total Page Size, Transfer Size and Number of Requests Each Test Scenario

The results of the study related to the measurement of carbon emission footprint shown in figure 8 can be seen that the Sibanksa system produces a very low carbon footprint of around 0,044456 – 0,040886 grams, especially in the render condition using cache. These results indicate that the Sibanksa system demonstrates strong efficiency in network-level data transfer, supported by a ~8% reduction in estimated carbon emissions when caching is enabled. While an 8% reduction in absolute terms may appear modest, it

is consistent with the 8% reduction observed in energy consumption, reinforcing the internal validity of the emissions estimation model used. At the scale of a single community waste-bank system with limited concurrent users, this reduction is unlikely to be operationally significant on its own; however, if service-worker caching were adopted as a standard practice across similar community-scale information systems in Indonesia, the cumulative effect across many deployments could represent a more meaningful contribution to reducing the aggregate digital carbon footprint of public-sector and community digital services. However, this network-level efficiency does not extend to rendering performance, particularly on mobile platforms, where FCP and LCP values remain well above the recommended Web Vitals thresholds due to large uncompressed page assets. This study did not include a direct comparison with a non-PWA or conventional (non-cached) equivalent of the system, as the Sibanksa system was developed exclusively as a PWA from the outset. Consequently, the reported improvements reflect the effect of enabling versus disabling service-worker caching within the same PWA implementation, rather than an isolated comparison against a fundamentally different architecture.

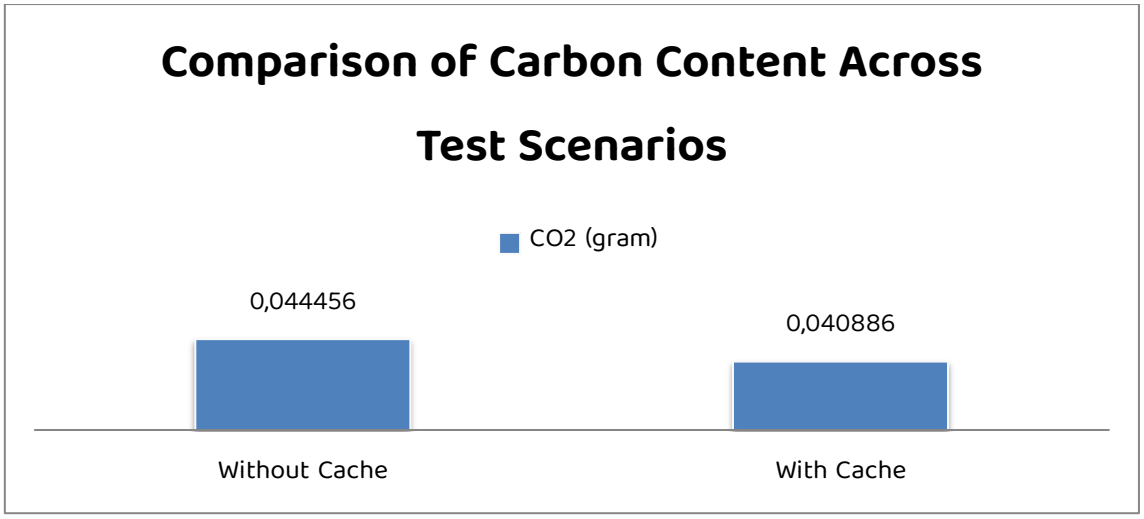

Figure 8. Comparison of Carbon Content Each Scenario

The results of the study related to the measurement of energy consumption in the system in figure 9 can be seen that the system consumes very little energy in both scenarios, especially in loading with a cache which decreases by 0.00000477 kWh or as much as 8% of the energy consumed compared to without a cache due to the presence of service workers working in it.

ISI Journal of Information Systems and Informatics
Published By Asosiasi Doktor Sistem Informasi Indonesia

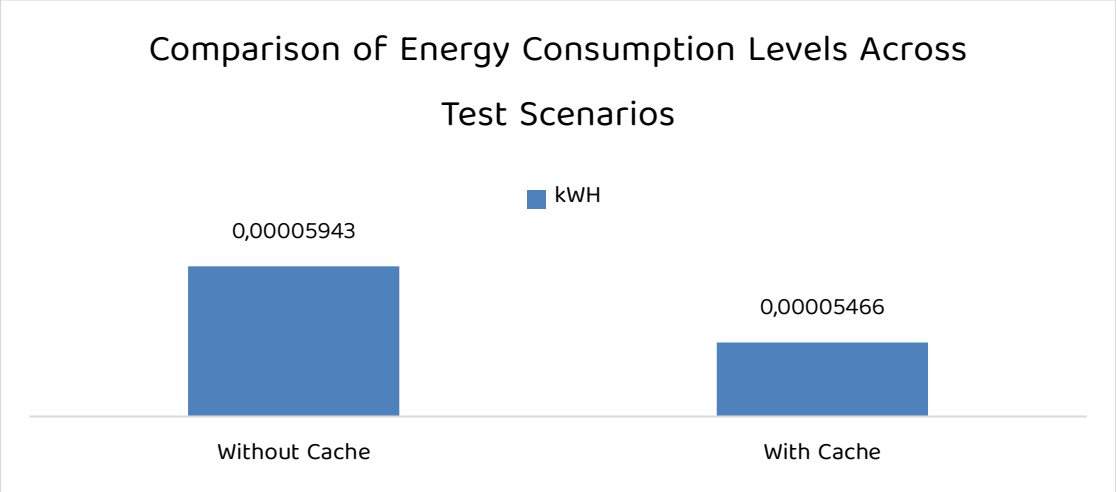

Figure 9. Comparisons of Consumption Energy Each Scenario

Based on the report presented in Table 21, it can be seen that the Sibanksa system has a very efficient use of the network, as can be that can cut the amount of data transfer which has an impact on decreasing the estimated amount of carbon and energy produced in the system. However, even though the system has an efficient network, it does not affect the performance of the system which Sibanksa has performance that needs to be reviewed and improved for the entire platform because it is still very slow and far from the threshold in rendering assets such as images, JS, and fonts.

Table 21. Comparison of Measure Testing Report between Standard Thresholds

Variable	Metric	Threshold	Result (Cached)	Status
Performance	TBT (Mobile)	< 200 ms	43,33 ms	Pass
	FCP (Mobile)	< 1,8 s	11,71 s	Fail
	LCP (Mobile)	< 2,5 s	23,22 s	Fail
	TBT (Desktop)	< 200 ms	2,67 ms	Pass
	FCP (Desktop)	< 1,8 s	2,65 s	Fail
	LCP (Desktop)	< 2,5 s	3,95 s	Fail
Data Transfer	Total Page Size	< 2 MB	7,46 MB	Fail
	Transfer Size	—	20,9 kB	Pass (vs uncached)
Carbon Footprint	Reduction vs uncached	—	~8% reduction	Pass (improvement)

ISI Journal of Information Systems and Informatics
Published By Asosiasi Doktor Sistem Informasi Indonesia

Variable	Metric	Threshold	Result (Cached)	Status
Energy Consumption	Reduction vs uncached	—	~8% reduction	Pass (improvement)

4. CONCLUSION

Based on the results of the research conducted, it can be concluded that the Sibanksa waste bank information system demonstrates fairly good network-transfer efficiency, along with relatively low estimated carbon emissions and energy consumption when service-worker caching is enabled, indicating partial support for Green IT Principles. However, rendering performance particularly on mobile platforms—still requires optimization, primarily by reducing unnecessary framework-reactivity assets that affect page load on medium-scale platforms such as mobile. In addition, total page size remains substantially above the recommended 2 MB threshold (7.46 MB even with caching enabled), indicating that transfer-size efficiency alone is insufficient without broader asset optimization. In terms of page-level optimization priority, Act4 (Implementation Schedule Management) and Act7 exhibited the highest carbon footprint under the non-cached and cached scenarios respectively, while the Login and Register pages recorded the largest image payloads — making these pages the primary targets for asset optimization. Future work should include: (1) real-device energy measurement using physical power meters or browser-based energy APIs to validate the tool-based estimates used in this study; (2) server-side profiling and database-query optimization to assess energy consumption beyond the client-side scope measured here; (3) comparison with a non-cached or non-PWA equivalent of the system to isolate the specific contribution of service-worker caching; (4) implementation of the asset optimization strategies, followed by re-measurement of mobile rendering performance; and (5) real-user monitoring and field testing with actual waste bank administrators under real network and device conditions. This study acknowledges that carbon and energy values were obtained through the CodeCarbon io estimation tool rather than direct physical power measurement, and results may vary depending on network conditions and server load.

ACKNOWLEDGMENT

Thank you to God Almighty for giving me the time and convenience to complete this research. Not forgetting also Telkom University Surabaya who has provided the opportunity to carry out this research and also the Information Systems Study Program along with research colleagues Mr. Adzanil Rachmadhi Putra and also Mr. Rosyid Abdillah who have helped and provided full support in the implementation of this research.

REFERENCES

[1] A. Nugroho, "Waste Bank Concept: Having Savings and Income from Waste," *Akad. J. Mhs. Humanis*, vol. 2, no. 2, pp. 46–54, 2022, doi: 10.37481/jmh.v2i2.468.

[2] P. T. Chountalas, S. K. Chrysikopoulos, K. K. Agoraki, and N. Chatzifoti, "Modeling Critical Success Factors for Green Energy Integration in Data Centers," *Sustainability*, vol. 3543, no. 17, pp. 1–22, 2025, doi: 10.3390/su17083543.

[3] C. Kusuma, M. Bimanatara, F. A. Akbar, and E. Y. Puspaningrum, "Implementasi Progressive Web Application (PWA) Dalam Pengembangan Sistem Pesan - Antar Makanan (Studi Kasus : Wirawiri Bojonegoro)," *JITET (Jurnal Inform. dan Tek. Elektro Ter.*, vol. 13, no. 2, pp. 185–196, 2025, doi: 10.23960/jitet.v13i2.6132.

[4] W. Kurniawan, N. T. Romadloni, R. Ayatulloh, and K. Noor, "Analisis Dampak Cache Progressive Web Apps Terhadap Konsumsi Baterai Android," *JITET (Jurnal Inform. dan Tek. Elektro Ter.*, vol. 13, no. 2, 2025, doi: 10.23960/jitet.v13i2.6221.

[5] S. Huber, L. Demetz, and M. Felderer, "A comparative study on the energy consumption of Progressive Web Apps," *Inf. Syst.*, vol. 108, 2022, doi: 10.1016/j.is.2022.102017.

[6] O. Hulleberg, H. L. Granum, S. G. Hansen, M. Moen, C. V. M. B, and Y. Inal, *The Awareness and Practices of Web Developers Toward Sustainable Web Design*. Springer Nature Switzerland, 2023. doi: 10.1007/978-3-031-35699-5_11.

[7] M. E. Siregar, "Evaluasi Implementasi Progressive Web App terhadap Performa Website dengan Google Lighthouse dan Chrome DevTools," *J. Algoritm.*, vol. 6, no. 2, pp. 382–394, 2025, doi: 10.35957/algoritme.v6i2.15712.

[8] R. S. Arjanu and I. M. Suartana, "Analisis Performa Teknologi Progressive Web App pada Aplikasi Online Notebook," *JINACS J. Informatics Comput. Sci.*, vol. 06, no. 04, pp. 1190–1195, 2025, doi: 10.26740/jinacs.v6n04.p1190-1195.

ISI Journal of Information Systems and Informatics
Published By Asosiasi Doktor Sistem Informasi Indonesia

[9] V. Vasilijević, N. Kojić, and N. Vugdelija, "A new approach in quantifying user experience in web-oriented applications," in *Proc. 4th Int. Sci. Conf. on Recent Advances in Information Technology, Tourism, Economics, Management and Agriculture (ITEMA)*, Oct. 2020, pp. 9–16.

[10] A. Prambayun *et al.*, "Analisis Performa Website Karya Usaha Menggunakan Google Lighthouse dan Gtmetrix," *JIK J. Ilmu Komput.*, vol. 10, no. 1, pp. 48–53, 2025, doi: 10.47007/komp.v10i01.9668.

[11] R. R. Gaddam, "Optimizing Core Web Vitals: A comprehensive framework for enhanced digital performance," *Journal of Engineering and Computer Sciences*, vol. 4, no. 7, pp. 704–711, 2025.

[12] R. Kanyal and S. R. Sarangi, "StealthDev: Side-Channel-Resistant Forensic Framework for Investigating Websites with Anti-Debugging," in *Proc. ACM Asia Conf. on Computer and Communications Security*, Jun. 2026, pp. 1816–1831.

[13] F. A. Setyowidodo, A. Pinandito, and M. A. Akbar, "Studi Perbandingan Lazy Loading Native dan IntersectionObserver JavaScript Dalam Pengelolaan Konten Gambar Pada Pengembangan Website," *J. Pengemb. Teknol. Inf. dan Ilmu Komput.*, vol. 9, no. 2, pp. 1–9, 2025.

[14] L. Lannelongue, J. Grealey, and M. Inouye, "Green Algorithms : Quantifying the Carbon Footprint of Computation," *Adv. Sci.*, vol. 2100707, no. 8, pp. 1–10, 2021, doi: 10.1002/advs.202100707.

[15] A. Tapp, H. R. Roth, Z. Xu, A. Parida, H. Nisar, and M. G. Linguraru, "Standardized Methods and Recommendations for Green Federated Learning," *arXiv preprint arXiv:2602.00343*, 2026.

[16] D. Saif, C. Lung, and A. Matrawy, "An Early Benchmark of Quality of Experience Between HTTP / 2 and HTTP / 3 using Lighthouse," in *IEEE*, 2021. doi: 10.1109/ICC42927.2021.9500258.