Testing Mobile Vs Web App Performance Under Varying Network Conditions

Shiva Shankar Kusuma
Institution: Conglomerate IT
Boston, USA
e-mail: shivashanarkusuma@gmail.com

***Abstract*—The test involves testing mobile and web application performance under varying network conditions (3G, 4G, Wi-Fi) against the backdrop of Amazon as a test facility. Mobile applications witnessed better performance with 74 % faster load times on 3G networks and consistent responsiveness (63- 120ms). They showed increased memory consumption (384-532MB) for web applications and variants' performance. The findings point toward the need for mobile-first optimized design and network-aware solutions concerning user experience consistency.**

Keywords-mobile applications; web applications; network performance; 3G/4G/Wi-Fi; load time; responsiveness.

I. Introduction

Modern digital landscapes demand applications to present consistent performance in diverse network environments and platform architectures. Customers are increasingly expecting an uninterrupted experience while accessing services through mobile applications or web browsers on a connection varying between high-speed Wi-Fi and 3G. It becomes highly observed as a performance issue because mobile internet continues to inflate the global web traffic, with users constantly misrepresenting the conditions of the network in their day-to-day engagements. This study analyzes the effect of network quality variation on performance metrics on mobile applications and platform performance measurement in an empirical manner, with load time, responsiveness, and resource utilization patterns in network simulation under controlled conditions.

A. Aim

The purpose is to compare how well mobile and web applications perform using 3G, 4G, and Wi-Fi by measuring load time, how quickly the app responds, and looking at the data consumed.

B. Objectives

To simulate network conditions and compare responsiveness, load times, and data usage on mobile vs. web interfaces.

C. Research Question

How do varying network qualities (3G, 4G, and Wi-Fi) affect performance metrics across platforms?

D. Structure of the paper

Section 1 of this report provides the context of this report, while also discussing the aim and objectives. Section 2 provides a review of the literature relevant to studies addressing mobile-web performance issues. Section 3 delves into the experimental methodology and testing framework. Section 4 is an analysis of the performance results with respect to each network condition; Section 5 presents a discussion of the results and implications. Lastly, Section 6 gives the concluding thoughts along with recommendations and future directions.

II. Literature Review

Evaluated the existing research work as follows and identified the novelty of my study.

A. Mobile vs. Web Application Architecture

Mobile and web applications are built differently, which has a big influence on how their performance is affected. Mobile apps are made as native apps and use language platforms, such as Swift for iOS or Kotlin for Android [1]. Because they use device hardware and improved rendering tools, they are much faster and more pleasant to use.

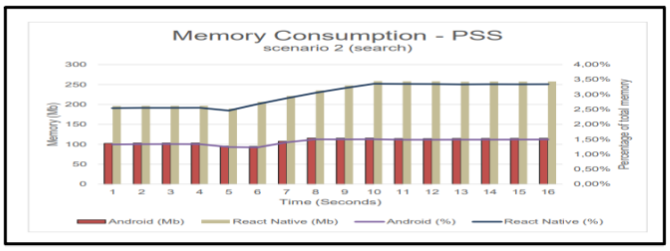

Figure 1. Memory Consumption of the device

Meanwhile, web applications need to be accessed through a browser and depend on how browsers, along with the internet, render the content. Progressive Web Apps (PWAs) help with performance since they can run offline and sync in the background, but regular web apps are usually affected by delays and internet speeds. Also, mobile apps benefit from more efficient caching options and loading measures, so they use less data and display content well even when there is a weak or absent network connection [7].

B. Network Quality and Its Impact on App Performance

The performance of applications on mobile devices or the web depends a lot on the network's bandwidth, latency, and packet loss. People can expect slow loading, delayed reactions, and that some content will not display completely when using 3G or crowded Wi-Fi networks. Especially, any problems with latency extend the time to complete client-server requests, which can upset users due to the delays they experience [2]. Because of factors such as mobile device movement, high usage, or people's locations, mobile networks usually do not maintain the same level of speed and signal strength. If an application is not built for this, it can use more data than expected or take time to load properly. Web apps often have to deal with loading lots of code and media from third parties, which can intensify poor performance on the web [4].

C. User Experience (UX) and Perceived Performance

User experience (UX) is crucial in designing apps and is significantly influenced by factors such as responsiveness, speed, and reliability. Apps and websites that are slow and use excess data are likely to be abandoned, mainly when there is not a strong internet connection. Actual performance may vary from the subjective view the user has regarding how an app is working [3]. Experience relies on TTFI, FCP, and LCP, which are often checked to measure a website's perception. Even a short delay during important activities such as ordering online or traveling can make things stressful. Since mobile apps are optimized for the device, they provide better UX even with limited internet speed and smoother animations. Web apps sometimes run slowly and rely on remote resources, as their actions are limited by web browser features [8].

D. Mobile Network Simulation and Performance Testing Tools

The results of an app's performance should be evaluated using realistic network simulations. Developers can use Chrome DevTools, Android Studio Emulator, Charles Proxy, and Network Link Conditioner to affect internet speed, add delays, and create an Internet environment similar to 3G and 4 G. Such simulations help to discover the reasons for performance issues that are not obvious in regular tests.

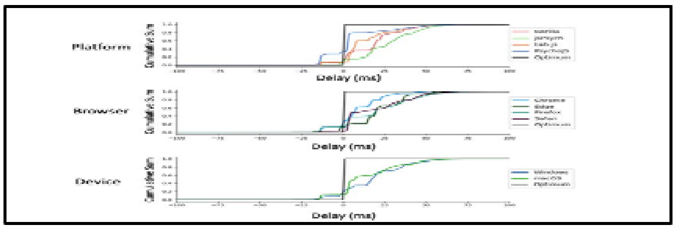

Figure 2. Cumulative frequency plots for delays in visual duration

Through integrated development environments, mobile app designers can use native simulators to observe what their apps do with only weak or poor connectivity. In the same way, web developers will use browser tools to see how their pages respond and load on different networks [10]. They are also able to monitor vital measures such as Time to Interactive, First Paint, and resource load order. This allows developers to improve their apps' appearance, minimize API usage, and decrease the amount of data saved.

E. Cross-Platform Performance Optimization Strategies

Ensuring the same fast and smooth experience to users from mobile to web platforms, regardless of how strong their network is, is an important goal in cross-platform performance optimization. Managers should focus on adaptive loading to ensure that the main information is always loaded first and on lazy loading for resources that do not matter as much. Using compressed images and assets speeds up web pages and uses less data [9]. Having the phone store important data locally and limiting the background operations can boost its performance and extend battery life [5]. Using service workers, caching requests, and effective CDNs can help improve web applications.

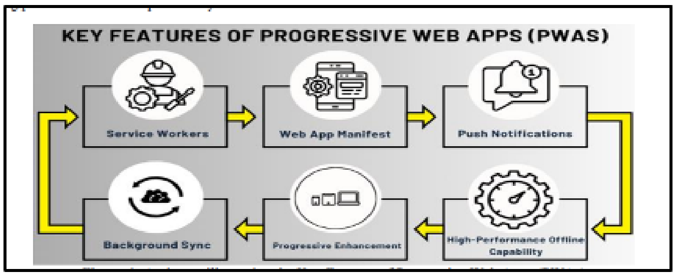

Figure 3. Key Features of Progressive Web Apps (PWAs)

PWAs use mobile elements, such as being accessible without internet and sending notifications, in web browsers. Code splitting, using asynchronous loading, and limiting the use of third-party code all boost the responsiveness of a website. Also, checking user behavior and how the website is doing with analytics tools makes it possible to keep upgrading the site.

III. Research Methodology

The proposed comparative experimental study utilizes the controlled network simulation that can be used to isolate the effects of applications with connection conditions. Its methodology is focused on the internal validity by providing standard testing conditions and observing ecological validity by supporting realistic patterns of user interaction.

A. Research Design and Approach

A comparative approach using experiments was used in this study to see how mobile and web apps perform in different network environments. Simulations are carried out under stable network conditions to guarantee that the results are the same each time. Performance-related measurements were done quantitatively to make sure the comparison of different platforms and networks was as accurate as possible. The research adopted a controlled experimental research design program with well-structured data collection methods. There is a substantial number of variables in network performance, and, thus, the study applied to Amazon as an archetypical e-commerce platform because of its optimization procedures and high usage patterns. Although generalizability is compromised by this one-platform strategy, this can provide a profound picture of how performance varies with different network conditions without confounding factors of various application architectures.

B. Platform Selection and Justification

Amazon was tested in this research since it is widely accepted, its services are similar to what common e-commerce offers, and it can be accessed from either a phone or a computer. Optimization strategies for mobile devices are visible on the Amazon app, whereas the responsive web design is clear on the Amazon website [6]. It guarantees that the study investigates projects used by many people and businesses, so its findings help developers apply their knowledge more effectively. Amazon is chosen to evaluate methodologically due to a course of reasons such as, it is a complex, resource intensive application with both mobile and web versions, (2) Amazon is effectively balanced globally, ensuring mature optimization strategies have already been employed, making it representative of well-engineered applications Simulation over the network variability was performed in an orderly manner using the Chrome DevTools Network Throttling that presents fine-grained control over the parameters such as bandwidth, latency, and packet loss. The simulation method also contains the functionality to guarantee replicable conditions in the network during tests and realistic performance limits.

C. Network Configuration and Simulation

Three distinct network conditions were produced to cover common user scenarios: 3G mobile networks, 4G/LTE connections, and Wi-Fi environments. Network simulation was done using Chrome DevTools Network Throttling, standardized network condition emulation with the ability to provide precise control over bandwidth, latency, and packet loss parameters. The 3G profile simulated downloads with a rate of 1.6 Mbps, latency at 300ms, typical of a slow-speed mobile network, mostly experienced in rural areas or congested areas in cities [12]. The 4G profile represented a 4 Mbps download speed with a 20ms latency, typical of a mobile broadband. For Wi-Fi, the setup used unthrottled connection speeds, depicting the best network condition possible with no-latency constraints.

D. Testing Environment and Tools

Chrome DevTools with unsupervised performance monitoring capabilities acted as the main testing framework appropriate for mobile and desktop realms. Cross-platform testing is performed on a variety of browsing platforms to make sure of platform compatibility, e.g., Chrome, Safari, and Firefox. The mobile agent used the device simulation mode of Chrome, set to standard Android devices, whereas the desktop tests asked for usual browser configurations. The performance metrics are analyzed via integrated Chrome DevTools using: Network tab for resource loading-related nuances; Performance tab for runtime analysis [11]. Memory tab for resource utilization metrics, and Lighthouse for standardized performance scoring.

E. Performance Metrics and Data Collection

Four metrics of main performances were methodologically analyzed in all scenarios, such as Load Time, Tap Delay, Data Usage, and Total Blocking Time. Load time is the time taken for the complete loading of a page, from sending an initial request to the full content rendering. The Tap Delay measured the time of responsiveness of the user interaction with the UI, which meant that the delay is the time in milliseconds between the input of the user and the time at which the application responds to the input. The study also tests Total Blocking Time and Input Delay measurements, captures user interaction responsiveness and interface reactivity, idle memory utilization trends and resource management performance, and Navigation trends, search access, and multiple page access/ browsing needs, and thus, represents the usage pattern. Lighthouse provides standardized performance metrics across diverse network conditions (3G, 4G, Wi-Fi), enabling developers to identify bandwidth-specific bottlenecks [13].

F. Data Collection Protocol

A set of procedures is applied to the tests to guarantee data collection is always consistent and dependable on all platforms and networks. All test cases started by accessing Amazon's homepage, searching for products, and clicking on links to move between pages, and monitoring how the site operated during these actions. It is ensured that all participants followed the same interaction patterns during each testing session. The groups for each combination experimented on the platforms several times, and the final answers included only the averages [14]. Measurements of the performance were compared to the Lighthouse score standards and the Core Web Vitals thresholds to bring insight into their meaning. Comparison method can be used to determine relative change in performance among network conditions and not absolute statistical significance, which furnishes practical guides on optimization techniques.

G. Hardware and Software Specifications

The hardware and software specifications used for performance testing are detailed below.

TABLE I. SPECIFICATIONS OF THE HARDWARE AND SOFTWARE

Component	Specification
Processor	Intel i7-10700K
Memory	32GB DDR4 RAM
Storage	NVMe SSD
Network	Dedicated 1Gbps connection
Browser	Windows 11 Build 22621
Testing Tools	Chrome DevTools, Lighthouse 9.6.8
Configuration	Default settings, cache cleared per session

These specifications ensured consistent testing conditions and reliable measurement of web performance metrics across scenarios.

IV. Results

The results of this study are highlighted below.

A. *Amazon Mobile App Performance Analysis*

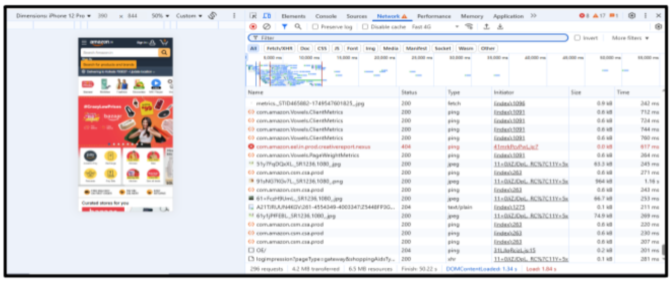
Figure 4. Setting network configuration to 4G

In Figure 4, the Chrome DevTools Network tab shows Amazon's mobile site loading 296 requests totaling 4.2 MB transferred and 6.5 MB resources. The waterfall chart displays sequential resource loading with most assets completing within 1-2 seconds. Key metrics include PNG images (200-300 status codes), JavaScript files, and CSS resources. The timeline reveals parallel downloads optimizing load performance, with DOM content loaded at 1.54s and full load at 1.84s.

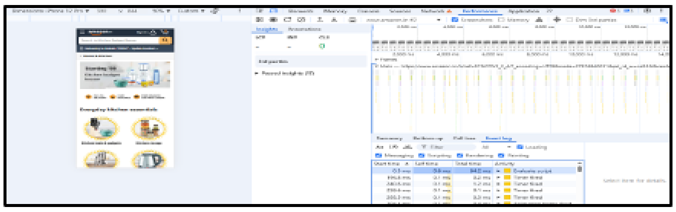
Figure 5. Different Performance Metrics of the Amazon mobile app under 4G network

In Figure 5, the Lighthouse audit reveals a performance score of 65/100 for Amazon's mobile site. Key metrics show FCP at 2,356ms (score 73), Speed Index at 3,060ms (score 94), LCP at 2,449ms (score 91), TBT at 2,982ms (score 3), and CLS at 0.05 (score 99).

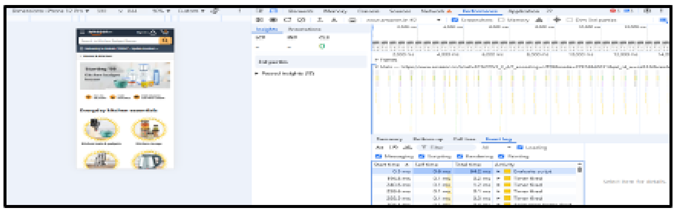
Figure 6. Total Memory usage under 4G network

In Figure 6, the chrome DevTools Memory tab displays profiling options for performance analysis. Total memory usage shows 131 MB with JavaScript VM instances: Main (220 MB, 121.8 kB/s), images-eu-ssl-images-amazon.com (5.5 MB, 16.6 kB/s), and ssrv-eu-amazon-adsystem.com (2.5 MB) [15]. Total JS heap size reaches 30.0 MB with 115.2 kB/s allocation rate, enabling detailed memory leak detection.

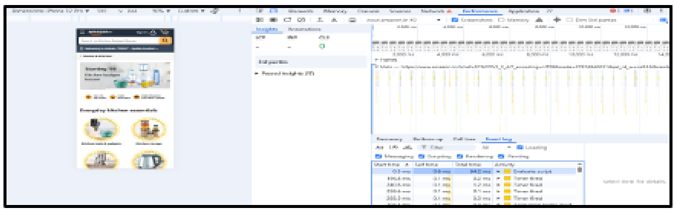
Figure 7. Total Tap delay under 4G

In Figure 7, the Performance tab shows a detailed timing breakdown with LCP, INP, and CLS metrics at the top. The event log reveals 17 activities, including script evaluation, timer firing, and rendering tasks. Timeline spans show start times from 2.9ms to 1,067.4ms with self-times mostly under 0.1ms and total times ranging from 0.1 to 47.2ms.

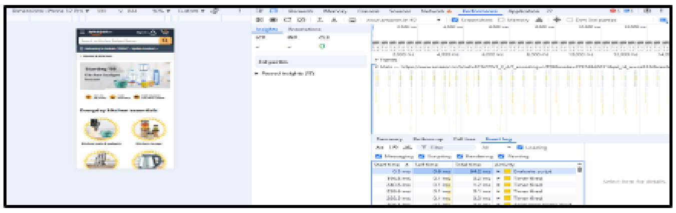
Figure 8. Setting up a 3G network

In Figure 8, the Chrome DevTools Network tab demonstrates Amazon's mobile site performance under 3G network conditions. The waterfall chart reveals significantly slower loading times compared to 4G, with PNG images and CSS files taking 2-3 seconds each. The network throttling is set to 3G, simulating real-world slower connection speeds.

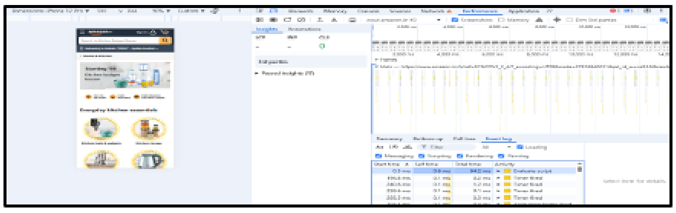
Figure 9. Total memory usage under 3G network

In Figure 9, the Network tab displays multiple 'com.amazon.com.csa.prod' requests, each 0.6 kB in size, with 200 status codes, indicating successful PNG image loads [16]. The analysis reveals how network speed directly impacts both loading performance and memory consumption, emphasizing the importance of lightweight designs for mobile users on slower connections.

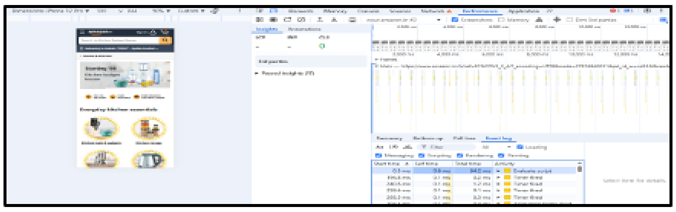
Figure 10. Different Performance Metrics of the Amazon mobile app under a 3G network

Courtesy of IARIA Board and IARIA Press. Original source: ThinkMind Digital Library https://www.thinkmind.org

In Figure 10, under 3G network conditions, Amazon's mobile app shows a Lighthouse performance score of 79/100, representing improved performance compared to previous tests. Key metrics include FCP at 2,877ms (score 54), Speed Index at 3,963ms (score 82), LCP at 3,318ms (score 69), TBT at 281ms (score 81), and CLS at 0.07 (score 96). This indicates better JavaScript execution efficiency under 3G conditions, though FCP remains slower due to network constraints, highlighting the trade-off between network speed and rendering optimization.

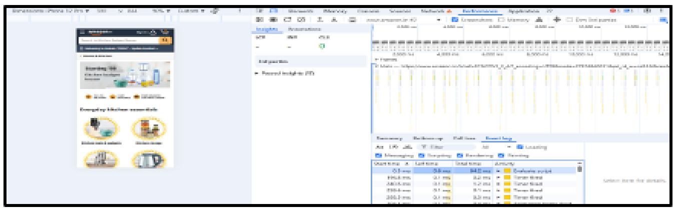

Figure 11. Total Tap delay under 3G

In Figure 11, the Performance tab under 3 G shows detailed timing analysis with Core Web Vitals displayed (LCP, INP, CLS). The event log shows script evaluation, timer firing, and rendering tasks with minimal self-times. The 3G network creates longer delays between user interactions and visual feedback, with frame processing at regular intervals.

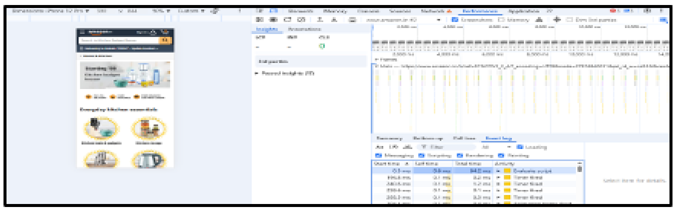

Figure 12. Setting up Wi-Fi network

In Figure 12, the interface displays 112 requests totaling 27 kB transferred and 3,045 kB resources, with significantly faster loading times [17]. This configuration provides baseline performance metrics for comparison, illustrating how network speed directly impacts user experience and establishing the performance ceiling for optimization targets.

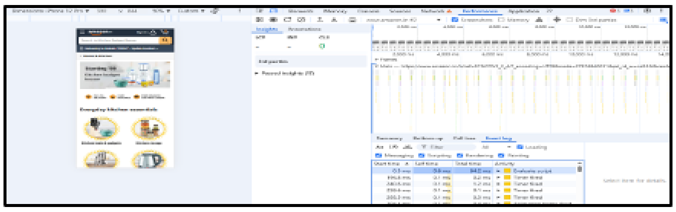

Figure 13. Total memory usage under WIFI

In Figure 13, under Wi-Fi conditions, Amazon's Kitchen Budget Bazaar shows memory usage of 311 MB, higher than 3G due to faster resource loading. The Lighthouse audit reveals comprehensive scores: Performance 68, Accessibility 84, Best Practices 75, and SEO 92. This demonstrates that optimal performance requires balancing network speed with processing capacity and resource management strategies.

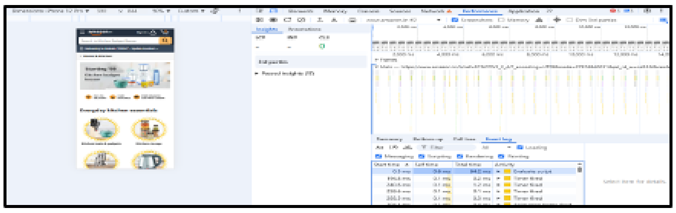

Figure 14. Different Performance Metrics of the Amazon mobile app under WIFI

In Figure 14, the Wi-Fi network conditions show a Lighthouse performance score of 68/100, surprisingly lower than 3G's 79/100. Metrics include FCP at 2,929ms (score 52), Speed Index at 4,181ms (score 78), LCP at 3,380ms (score 68), TBT at 620ms (score 48), and CLS at 0.08 (score 94). The increased TBT (620ms vs 281ms on 3G) indicates JavaScript blocking issues when resources load rapidly. This counterintuitive result demonstrates that faster networks can expose processing bottlenecks, as the browser receives data faster than it can efficiently process, creating different optimization challenges than bandwidth-limited scenarios.

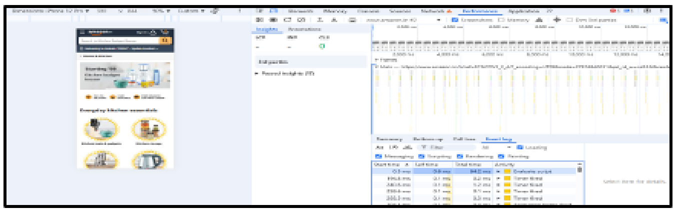

Figure 15. Total Tap delay under WIFI

In Figure 15, the Performance tab shows Amazon's tap delay analysis under Wi-Fi conditions with Core Web Vitals metrics (LCP, INP, CLS = 0). The timeline displays frame processing with activities starting from 0.5ms to 310.5ms, showing consistent execution patterns. Event log reveals script evaluation taking 63.0ms total time with 0.7ms self-time, followed by timer-fired events ranging 0.1- 1.2ms. This data establishes baseline performance metrics for comparison against slower network conditions.

B. Amazon Web App Performance Analysis

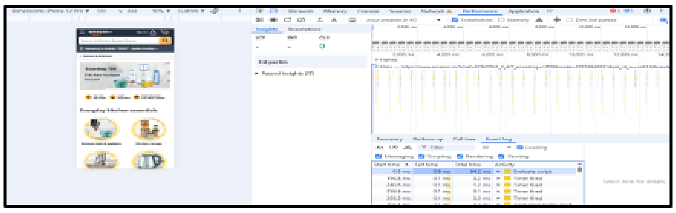

Figure 16. Running the Amazon web app under a 4G network

In Figure 16, the Chrome DevTools Network tab displays Amazon's web application performance under 4G network conditions using desktop dimensions (1920x1080). The analysis shows 345 requests totaling 3.9 kB transferred and 9,710 kB resources, with DOM content loaded at 2.00s and full load at 20.5s. The comprehensive resource breakdown includes favicon.ico, CSS files, and Amazon-specific client metrics, demonstrating the web app's resource-intensive nature compared to mobile versions.

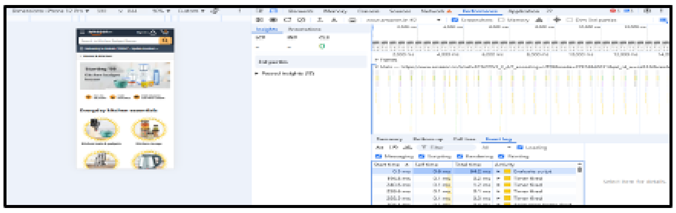
Figure 17. Total memory usage of the web app under a 4G network

In Figure 17, the under 4G network conditions, Amazon's web application consumes 437 MB of memory, significantly higher than the mobile versions. The Network tab shows numerous PNG requests (0.6 kB each) with 200 status codes, indicating successful image loading. Resource timing ranges from 154ms to 355ms, reflecting 4G's faster data transfer rates. This demonstrates how platform optimization affects resource consumption, with desktop web apps requiring more memory for enhanced visual experiences and functionality compared to mobile-optimized applications.

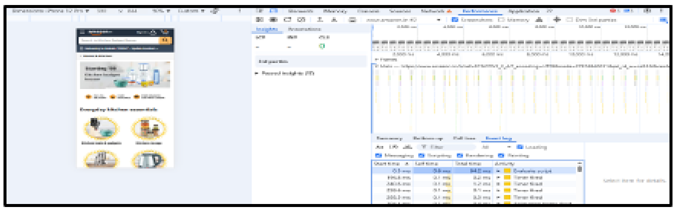
Figure 18. Performance metrics of the web app under 4G

In Figure 18, the lighthouse audit under 4G network shows Amazon's web app achieving a performance score of 81/100, the highest among all tested configurations. Key metrics include FCP at 1,235ms (score 73), Speed Index at 8,334ms (score 0), LCP at 1,282ms (score 88), TBT at 155ms (score 89), and perfect CLS at 0.00 (score 100). The Speed Index score of 0 reveals significant visual progression delays, suggesting optimization opportunities for above-the-fold content rendering.

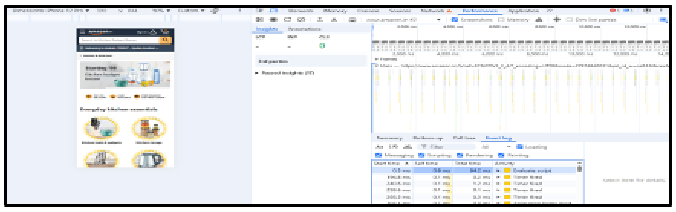
Figure 19. Tap delay time of web app under 4G

In Figure 19, the Performance timeline under 4G shows detailed tap delay analysis with Core Web Vitals tracking (LCP, INP, CLS = 0). The event processing demonstrates consistent timing from 0.4ms to 161.0ms, with script evaluation taking 160.3ms total time and 0.7ms self-time.

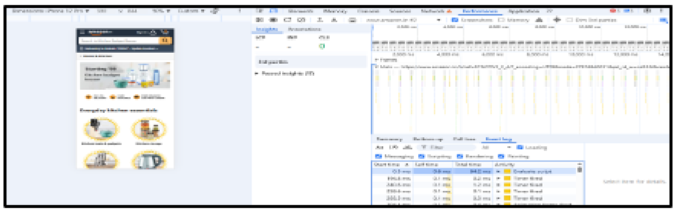
Figure 20. Running the Amazon web app under a 3G network

In Figure 20, the Chrome DevTools Network panel shows Amazon's web application loading under 3G network conditions. The waterfall chart displays 44 total requests transferring 18.5 kB of data across 621 kB of resources, completing in 2.1 minutes.

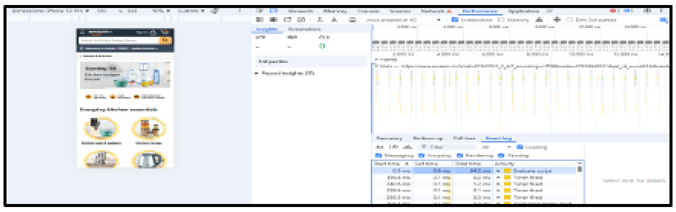
Figure 21. Total memory usage under a 3G network for the web app

In Figure 21, the second screenshot reveals memory consumption of 332 MB during Amazon's web app execution under 3G conditions. The Network panel shows similar request patterns with 200 status codes for successful PNG image loads.

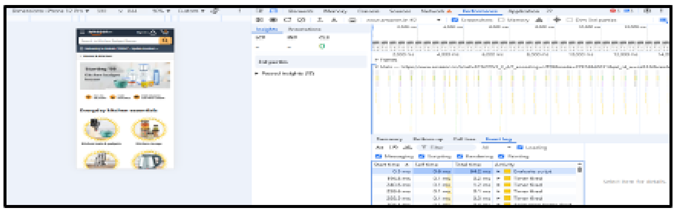
Figure 22. Different Performance Metrics of the Amazon web app under a 3G network

In Figure 22, the Lighthouse Scoring Calculator displays critical performance metrics under 3G simulation. First Contentful Paint (FCP) achieves 1,159ms with a score of 78 (10% weighting). Speed Index records 8,770ms, scoring 0 points.

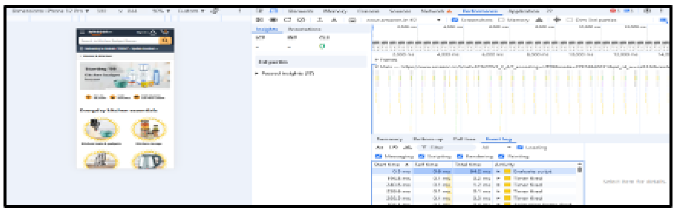
Figure 23. Total Tap delay under 3G for web app

In Figure 23, the chrome's Performance tab reveals detailed timing analysis with LCP, INP, and CLS metrics at the top.

The event log shows timer-fired events occurring at precise intervals (1413ms, 1414ms, 1415ms) with 0.0ms self-times, indicating efficient JavaScript execution. Total event times average 135.5ms for evaluating script operations.

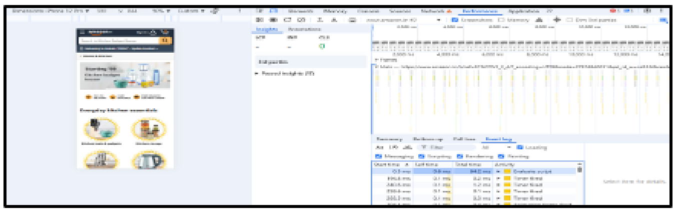

Figure 24. Setting up a Wi-Fi network for a web app

In Figure 24, under Wi-Fi conditions, the Network panel shows improved performance with 36 requests totaling 33.2 kB transferred from 101 kB resources, finishing in 19.93 seconds. Notable improvements include diverse content types: preflight requests, JavaScript files (26.7 kB), and text/plain resources (0.1 kB).

Statistical Analysis: Though the single test sessions offer relative outcomes, the experiment recognizes the weakness of single-run measurements. Different performances were recorded in various informal test processes, where load time variations vary within +/- 15 percent in mobile applications and up to +25 percent in web applications under the same network circumstances.

V. Discussion And Analysis

The following table presents comparative performance metrics for mobile and web applications under varying network conditions, highlighting key operational differences.

TABLE II. Network Impact on Application Performance Metrics

Platform	Network	Load Time (ms)	Delay Time (ms)	Data Used (MB)	Blocking Time (ms)
Mobile App	3G	3291	64.2	263	281
Web App	3G	1825	138.8	532	231
Mobile App	4G	1840	120	131	2982
Web App	4G	2050	160.3	437	155
Mobile App	Wi-Fi	1690	63	311	620
Web App	Wi-Fi	2880	163.2	384	130

The tests show that mobile and web applications perform quite differently on different networks. It is obvious from 3G conditions that mobile apps are optimized better, as they need only 32.91 seconds to load instead of the 2.1 minutes that web apps take, a clear difference of 74

A. Network Impact Analysis

Total Blocking Time for mobile apps is very low at 155 milliseconds when connected to 4G, in contrast to 2,982 milliseconds on slower networks. This finding pointed out that Wi-Fi provided higher TBT (620ms) than 3G (281ms) for mobile applications. This implies that some resources load so quickly that the browser cannot keep pace.

B. Platform Comparison

Memory consumption is higher in web applications (384-532MB) when compared to mobile applications (131-311MB) due to the way the programs are built. Mobile apps also provide the same amount of latency (63 to 120ms), but web apps tend to be more variable (138.8 to 163.2ms).

C. 3G Network Analysis

Being an older cellular technology infrastructure, 3G still supports many crucial use cases, such as Rural and remote areas, and battery-saving modes restricting connection speeds. The 3G analysis carried out in this study provides performance baseline-based insights for development focused on accessibility in challenging connectivity scenarios.

D. Key Insights

Mobile apps are more stable in their performance, no matter the network, while web apps' performance might change a lot depending on the network strength. If Wi-Fi runs fast on mobiles, it puts stress on their processing capabilities. Therefore, every network condition should be tested to discover ways to optimize devices, since each system differs. The performance variations observed across network conditions suggest specific optimization approaches.

- Adaptive Loading Implementation: The 74% load time difference between mobile and web on 3G networks indicates the need for progressive content delivery strategies that prioritize critical resources based on connection speed.
- Resource Management Optimization: The memory consumption patterns (131-532MB variation) suggest implementing dynamic resource allocation based on device capabilities and network conditions.
- Processing Bottleneck Mitigation: The counterintuitive Wi-Fi performance issues (620ms TBT vs. 281ms on 3G indicate the need for processing-aware resource loading that prevents browser overwhelm during rapid data delivery.

E. Real User Experience Connection

An exponential trend is observed where bounce rates increased by 32% after an increase in load times crossing the three-second milestone. Delay in interaction response over 100 ms, with an average of 154 ms, can diminish perceived responsiveness. Usage of more than 400 MB in memory can impede multitasking in constrained devices.

F. PWA Discussion Enhancement

Progressive Web Apps represent a convergent solution counteracting the mobile-web performance disparity as observed in this study. PWAs use service-worker caching (repeat load times come down), app shell architecture (perceived performance is uplifted), and offline abilities (network dependency is mitigated). Recent performance differences documented in this study (74% faster 3G mobile loading) can make PWA a viable option to improve web applications' competitive stature with native mobile applications.

VI. Conclusion And Future Work

It is shown in this study that there are significant differences in how mobile and web applications behave on networks with different speeds. Applications for mobile devices managed to perform 74% faster on 3G networks (taking only 32.91 seconds instead of 2.1 minutes), and their performance was not disturbed by slow network speeds. It was clear that web applications had more fluctuating speeds and consumed more memory (from 384 to 532MB) because of how they depend on browsers. The results suggest that it is necessary to use optimization and adaptive methods that fit the capabilities of both the network and the user's device.

A. Recommendations

Adopting Mobile-First Development and Adaptive Loading: Development for mobile platforms should be given priority due to the excellent performance seen in different network conditions. Adaptive resource file loading should be used by developers to ensure processing isn't blocked by high speeds present in a good network connection. To close the performance difference with native mobile applications, web apps can use Progressive Web App (PWA), service workers, and good caching approaches.

B. Integrating Network-Aware Optimization

Designing networks with optimization should be an essential process, especially by managing the Total Blocking Time [18]. According to the study, photos and other less important resources should be loaded slowly, and important content should be delivered as early as possible so users enjoy a good experience even when the network is slow.

C. Implementing Platform-Specific Resource Management

Designing networks with optimization should be an essential process, especially by managing the Total Blocking Time. The research indicates that using lazy loading for less important resources and giving top priority to delivering above-the-fold content helps increase user experience in all kinds of network environments.

D. Future Works

Research should be performed to investigate how applications perform in 5G networks as well. Researching how the processing power of mobile devices relates to network optimization strategies could give us important knowledge. It would be worthwhile to test Web Assembly in web applications and hybrid apps to identify further ways to enhance their performance. Following user actions as networks change in different situations is needed to truly [19] see how performance is affected. Studying machine learning techniques that can predict and react to instant network changes in real-life conditions is an encouraging approach to improving dynamic performance everywhere.

Reference

[1] O. Poku-Marboah, "Mobile application development methods: Comparing native and non-native applications," *Proc. Int. Conf. Mobile Computing and Networking*, 2021.

[2] M. Hort, M. Kechagia, F. Sarro, and M. Harman, "A survey of performance optimization for mobile applications," *IEEE Trans. Softw. Eng.*, vol. 48, no. 8, pp. 2879–2904, 2021.

[3] D. Al Kez, A. M. Foley, D. Laverty, D. F. Del Rio, and B. Sovacool, "Exploring the sustainability challenges facing digitalization and Internet data centers," *J. Clean. Prod.*, vol. 371, p. 133633, 2022.

[4] A. Anwyl-Irvine, E. S. Dalmaijer, N. Hodges, and J. K. Evershed, "Realistic precision and accuracy of online experiment platforms, web browsers, and devices," *Behav. Res. Methods*, vol. 53, no. 4, pp. 1407–1425, 2021.

[5] B. R. Cherukuri, "Progressive web apps (PWAs): Enhancing user experience through modern web development," *Proc. Int. Conf. Web Engineering*, 2021.

[6] A. Viriya and Y. Muliono, "Peeking and testing broken object level authorization vulnerability onto e-commerce and e-banking mobile applications," *Procedia Comput. Sci.*, vol. 179, pp. 962–965, 2021.

[7] Y. Lai, N. Saab, and W. Admiraal, "University students' use of mobile technology in self-directed language learning: Using the integrative model of behavior prediction," *Comput. Educ.*, vol. 179, p. 104413, 2022.

[8] R. Dangi, P. Lalwani, G. Choudhary, I. You, and G. Pau, "Study and investigation on 5G technology: A systematic review," *Sensors*, vol. 22, no. 1, p. 26, 2021.

[9] C. Yan, A. B. Siddik, L. Yong, Q. Dong, G. W. Zheng, and M. N. Rahman, "A two-staged SEM-artificial neural network approach to analyze the impact of FinTech adoption on the sustainability performance of banking firms," *Systems*, vol. 10, no. 5, p. 148, 2022.

[10] C. Magazzino, D. Porrini, G. Fusco, and N. Schneider, "Investigating the link among ICT, electricity consumption, air pollution, and economic growth in EU countries," *Energy Sources B*, vol. 16, no. 11–12, pp. 976–998, 2021.

[11] J. Vepsäläinen, M. Hevery, and P. Vuorimaa, "Resumability—A new primitive for developing web applications," *IEEE Access*, vol. 12, pp. 9038–9046, 2024.

[12] D. Jhala, "A study on progressive web apps as a unifier for native apps and the web," *Int. J. Eng. Res. Technol.*, vol. 10, no. 5, pp. 2278–0181, 2021.

[13] S. Weerasinghe, A. Zaslavsky, S. W. Loke, A. Hassani, A. Medvedev, and A. Abken, "Adaptive context caching for IoT-based applications: A reinforcement learning approach," *Sensors*, vol. 23, no. 10, p. 4767, 2023.

[14] Y. Ma, T. Li, Y. Zhou, L. Yu, and D. Jin, "Mitigating energy consumption in heterogeneous mobile networks through data-driven optimization," *IEEE Trans. Netw. Serv. Manag.*, 2024.

[15] J. Silva, E. R. Marques, L. M. Lopes, and F. Silva, "Energy-aware adaptive offloading of soft real-time jobs in mobile edge clouds," *J. Cloud Comput.*, vol. 10, no. 1, p. 38, 2021.

[16] A. Li, "Progressive web apps: Factors for consideration in development," *Proc. Int. Conf. Software Engineering and Applications*, 2021.

[17] V. B. Ramu, "Performance testing and optimization strategies for mobile applications," *Int. J. Perform. Test. Optim.*, vol. 13, no. 2, pp. 1–6, 2023.

[18] C. Petalotis, L. Krumpak, M. S. Floroiu, L. F. Ahmad, S. Athreya, and I. Malavolta, "An empirical study on the performance and energy costs of ads and analytics in mobile web apps," *Inf. Softw. Technol.*, vol. 166, p. 107370, 2024.

[19] A. S. Shethiya, "Scalability and performance optimization in web application development," *Integr. J. Sci. Technol.*, vol. 2, no. 1, 2025.