Revitalizing CodeQuest Mobile App: Performance Optimization for Immersive OOP Learning

William P. Rey
Mapua University/School of IT,
Makati City, Philippines;
wprey@mapua.edu.ph

2024 9th International Conference on Computer and Communication Systems (ICCCS) | 979-8-3503-5021-0/24/$31.00 ©2024 IEEE | DOI: 10.1109/ICCCS61882.2024.10603304

***Abstract*—In the rapidly evolving landscape of technology and education, the effectiveness of online learning platforms plays a pivotal role in shaping student experiences. This study addresses the need to revitalize CodeQuest (CQ), an online platform dedicated to Object-Oriented Programming (OOP), by focusing on performance optimization. The primary objectives are to identify and rectify performance bottlenecks within CodeQuest and measure the impact of these optimizations on the learning experience of OOP students. Through a mixed-methods approach, combining quantitative and qualitative analyses, the study systematically assesses the platform's performance using tools such as GTmetrix, Google PageSpeed Insights, and others. The pre-optimization analysis reveals challenges in response time, load times, and resource utilization. Strategic optimizations, including addressing client-side workloads and render-blocking resources, are implemented. Post-optimization tests demonstrate a significant improvement, with the GTmetrix grade jumping from 'C' to 'A.' Web vitals metrics, such as Largest Contentful Paint and Total Blocking Time, exhibit remarkable enhancements. These improvements *directly* contribute to a more responsive and visually stable user experience. Positive trends in user engagement metrics affirm the success of the optimization efforts. The study concludes with reflections on the broader implications for educational technology. It highlights CodeQuest as a model for OOP education and a blueprint for enhancing online learning environments through performance optimization.**

***Keywords*—CodeQuest, performance optimization, mobile App**

I. Introduction

In the rapidly evolving landscape of technology and education, the effectiveness of online learning platforms plays a pivotal role in shaping the learning experiences of students. This study aims to address the pressing need to revitalize CodeQuest (CQ), an online platform dedicated to object-oriented programming (OOP), to enhance the overall learning experience. The emphasis is placed on optimizing CodeQuest's performance, thereby contributing to a more immersive and efficient educational journey for learners engaged in OOP studies.

A. *Context and Importance of Performance Optimization*

Educational platforms, especially those focused on programming and complex subjects like OOP, require robust performance to ensure a seamless learning experience. The significance of performance optimization in such platforms cannot be overstated. Swift response times, minimal latency, and efficient resource utilization contribute not only to user satisfaction but also to the effectiveness of the learning process. As we embark on this journey to revitalize CodeQuest, our primary goal is to address these performance challenges, ultimately creating an environment that fosters effective OOP learning.

B. *Objectives and Significance of the Study*

The primary objectives of this study are twofold: firstly, to identify and rectify performance bottlenecks within the CodeQuest platform, and secondly, to measure the impact of these optimizations on the overall learning experience of OOP students. By achieving these objectives, the researchers aim to contribute valuable insights and guidelines for the enhancement of other educational platforms facing similar challenges. The significance of this study lies in its potential to create a positive ripple effect across online learning environments, ultimately benefiting a broader community of students and educators.

C. *Brief Overview of CodeQuest:*

CodeQuest, illustrated in Figure 1, is a mobile internet-based gamified educational platform designed to support the acquisition of Object-Oriented Programming skills. Users undergo secure authentication to gain entry to features such as the Knowledge Bank, Assessment, Leaderboard, and Achievements. The Knowledge Bank provides interactive lessons, and the Assessment module includes time-constrained true/false and multiple-choice examinations. Achievements are unlocked based on user performance, leading to automatic updates on leaderboards. By incorporating gamification elements such as points, leaderboards, and badges, CodeQuest offers an immersive platform for mastering Object-Oriented Programming.

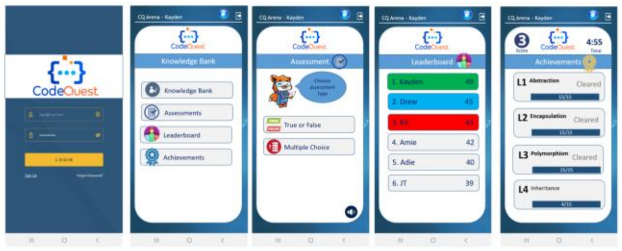

Fig. 1. CodeQuest Sample Screen Output [1].

While it has been instrumental in providing educational resources, the platform currently faces challenges related to performance that hinder its ability to deliver an optimal learning experience. These challenges may include slow load times, latency issues, or suboptimal resource usage. The need for enhancement becomes apparent, signaling the importance of this study in addressing and rectifying these issues for the benefit of CodeQuest's user community.

As the researchers delve into the depths of performance optimization for CodeQuest, this study aims to uncover solutions that not only elevate the platform but also serve as a blueprint for improving educational platforms in a broader context. Through strategic enhancements, we aspire to breathe new life into CodeQuest, fostering an environment where OOP learners can thrive.

II. Literature Review

A. *Performance Optimization in Educational Platforms*

As educational platforms continue to play a critical role in the digital learning landscape, the optimization of platform performance emerges as a pivotal factor influencing the overall learning experience. In this literature review, we explore existing studies related to the performance optimization of educational platforms.

B. *Studies on Performance Optimization in Educational Platforms:*

Several scholars have delved into the realm of performance optimization within educational platforms. Govea et al. (2019) emphasized the importance of responsive and efficient systems in their study on the impact of platform performance on student engagement [2]. Their findings suggested a positive correlation between optimized performance and increased student satisfaction.

Ren (2023) presented challenges in traditional education due to rapid big data development. It suggests innovating education resource allocation using computer vision multimodal learning. The focus is on creating an ideology instructional resource model with an improved algorithm for personalized recommendations. The system architecture aims to promote optimal resource allocation in universities [3].

C. *Impact of Performance Improvements on the Learning Experience:*

Research has consistently shown that improvements in platform performance have a direct impact on the learning experience. Kuosa et al. (2016) present two interactive visualization tools designed to enhance learning and teaching in online courses through learning management systems (LMS). Developed at Tampere University of Technology (TUT) and Unitelma Sapienza University, these tools analyze user log data and discussion forums, offering valuable insights into the learning process. Both tools are adaptable to any LMS, although initially created as plugins for Moodle [4].

Rodriguez et al. (2020) discuss the impact of course duration on learner engagement, retention, and success in Massive Open Online Courses (MOOCs). The study compares two versions of a Study Skills MOOC with the same content but delivered in different lengths – one as a single six-week course and the other as two three-week blocks. The research, involving 617 participants, utilizes learning analytics, surveys, and the Spanish version of the General Self-Efficacy Scale. Both versions increase participants' self-efficacy, but the two three-week block format shows higher learner engagement, retention, and completion rates [5]. The paper explores factors like goal proximity, motivation, interactions, and social modeling as reasons for these differences.

D. *Methodologies and Strategies in Optimization Projects:*

The methodologies applied in past optimization projects provide crucial insights that significantly contribute to our study. Rey et al. conducted an in-depth exploration of various strategies aimed at enhancing performance, encompassing crucial techniques like code refactoring, improvements on the server side, and the integration of content delivery networks (CDNs) [6][7][8]. By comprehensively understanding and synthesizing these methodologies, the researchers gain a solid foundation for devising optimization strategies specifically tailored to address the unique challenges and requirements of CodeQuest. This knowledge allows to draw upon proven approaches, ensuring the effectiveness and relevance of our optimization efforts in the context of the CodeQuest platform.

E. *The Code Quest Mobile Architecture*

Figure 2 illustrates the structure of the CodeQuest mobile app. The backend system is hosted in the cloud and shared with the web app. Users engage with the CQ mobile app interfaces, which are specifically tailored for Android platforms, incorporating the latest release. The app features a secure transport layer that links to the app controller. This controller is intricately linked to authentication, mobile app content, and a REST API, ultimately connecting to a cloud-based web server.

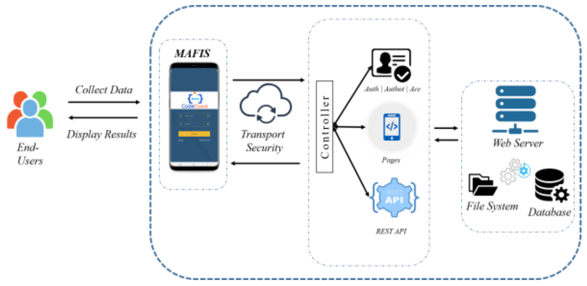

Fig. 2. CodeQuest Mobile Architecture [1].

III. Methodology

In the swiftly changing realm of mobile applications, emphasizing performance is paramount [9]. Users expect rapid loading, instant responses, and a seamless experience from their apps. This highlights the urgent requirement to optimize mobile application performance, aligning with the ever-growing expectations of users in today's dynamic digital environment. Fulfilling these expectations not only ensures user satisfaction but also positions an app competitively in the saturated mobile market. As technology progresses, app developers consistently confront the challenge of providing high-performance solutions to meet the demands of users who seek smooth and efficient mobile experiences. The methodology depicted in Figure 3 will be employed in this study.

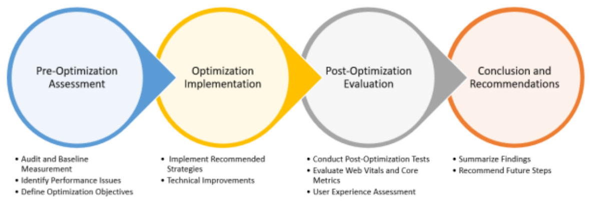
Pre-Optimization Assessment
Optimization Implementation
Post-Optimization Evaluation
Conclusion and Recommendations
• Audit and Baseline Measurement
• Identify Performance Issues
• Define Optimization Objectives
• Implement Recommended Strategies
• Technical Improvements
• Conduct Post-Optimization Tests
• Evaluate Web Vitals and Core Metrics
• User Experience Assessment
• Summarize Findings
• Recommend Future Steps

Fig. 3. Methodology.

A. *Pre-Optimization*

Before optimizing CodeQuest's system performance, it's crucial to assess its current state thoroughly. Using performance analysis tools is essential, and in this study, GTmetrix is employed to evaluate CodeQuest's performance. This tool offers insights into system speed, a critical factor in today's digital landscape with limited user attention spans.

Highlighting the significance of website performance, even a one-second delay can lead to substantial consequences. Studies, including those on major sites like Amazon and Google, show that a mere one-second delay results in a 7% drop in conversions, an 11% decrease in page views, and a 16% decline in customer satisfaction [10]. GTmetrix tests web pages with an Unthrottled Connection using real hardware and specific parameters. Users can adjust these parameters, aligning them with specific testing requirements [11]. This meticulous evaluation ensures optimization efforts target the specific performance aspects crucial for CodeQuest's success in the competitive digital landscape.

B. *Research Design and Approach:*

The research design for optimizing CodeQuest involves a systematic and iterative approach aimed at enhancing the learning experience in Object-Oriented Programming (OOP). The study follows a mixed-methods design, incorporating both quantitative and qualitative analyses to provide a comprehensive understanding of the performance optimization process.

C. *Key Performance Metrics and Rationale:*

To assess the effectiveness of performance optimization, a set of key performance metrics is defined. These metrics include but are not limited to: (1) Response Time: Measured as the time taken for CodeQuest to respond to user actions, (2) Load Time: The time it takes for the platform to load initially and for subsequent interactions and (3) Resource Utilization: Evaluation of server and client-side resource consumption during user interactions.

The rationale behind selecting these metrics lies in their direct impact on user experience and engagement. Improvements in response and load times contribute to a more responsive and user-friendly platform, while efficient resource utilization ensures scalability and stability.

D. *Use of Automated Tools:*

Automated tools play a crucial role in assessing the current system performance and guiding optimization efforts. Tools such as Google PageSpeed Insights, Apache JMeter, and New Relic are employed to conduct comprehensive assessments of CodeQuest's performance. These tools assist in identifying areas of concern, providing actionable insights into performance bottlenecks, and guiding the subsequent optimization process.

E. *Identifying and Addressing Performance Bottlenecks:*

The identification of performance bottlenecks is a critical phase in the optimization process. Through thorough profiling and monitoring using automated tools, specific areas of the CodeQuest platform that contribute to performance degradation are pinpointed. These may include database queries, inefficient algorithms, or rendering bottlenecks in the user interface. Once identified, targeted strategies such as code refactoring, caching mechanisms, and server-side optimizations are employed to address these bottlenecks.

F. *Pre- and Post-Optimization Testing:*

A structured approach is taken to testing, encompassing both pre- and post-optimization phases. In the pre-optimization phase, baseline measurements of key performance metrics are recorded. Subsequently, the identified optimization strategies are implemented. Post-optimization testing involves re-evaluating the platform using the same key metrics. Comparative analysis between pre- and post-optimization results provides insights into the effectiveness of the applied strategies and the overall impact on CodeQuest's performance.

This methodology outlines the systematic approach taken to optimize CodeQuest for enhanced Object-Oriented Programming (OOP) learning. It emphasizes the importance of key performance metrics, the use of automated tools, and the structured testing process to ensure a comprehensive and effective optimization effort.

G. *Test Configuration Settings*

As detailed in Table 1, the test setup parameters encompassed the performance analysis's configuration settings. Given its advantageous geographic proximity to the Philippines, Singapore was selected as the optimal location for the test server. The test URL is set to Singapore and is done via mobile performance through a Samsung S22+ simulated device. The simulated connection is set to LTE (15/10 Mbps, 100ms).

TABLE I. TEST CONFIGURATION SETTINGS

Test Server Location	Singapore
Using	Samsung Galaxy S22+
Connection	LTE (15/10 Mbps, 100ms)

H. *Lighthouse Web Vitals*

Core Web Vitals form a subset of Web Vitals that hold relevance for all web pages. They are mandatory assessments for all website administrators and are readily accessible in all Google tools. Each Core Web Vital represents a distinct facet of the user experience, can be measured under real-world conditions, and offers critical user-centric insight. The scale of measurement employed by Web Vitals, a key consideration in this study, is presented in Table 2.

TABLE II. WEB VITALS SCALE METRIC

Metrics	Parameters	Ideal Value
First Contentful Paint (FCP.)	Load speed	Less than 0.9s
Speed Index (SI.)	Load Time	Less than 1.3s
Largest Contentful Paint (LCP.)	Perceived Loading	Less than 2.5s
Time to Interactive (TTI.)	Interactivity	Less than 2.5s
Total Blocking Time (TBT)	Responsiveness	Less than 150ms
Cumulative Layout Shift (CLS.)	Visual Stability	Less than 0.1

Core Web Vitals encompass several essential metrics that directly impact user experience: (1) Largest Contentful Paint (LCP): LCP measures loading speed and aims for a loading time of less than 2.5 seconds from the page's initial load to ensure a pleasant user experience; (2) Total Blocking Time (TBT): TBT is a metric that assesses interaction. For a decent user experience, web pages should have a TBT of fewer than 150 milliseconds, ensuring smooth interactions; (3) Cumulative Layout Shift (CLS): CLS is a visual stability metric. To provide an excellent user experience, web pages should aim for a CLS of less than 0.1, minimizing unexpected layout shifts; (4) First Contentful Paint (FCP): FCP measures the time it takes for a browser to render the first piece of DOM content after a user lands on a page. It considers images, non-white canvas elements, and SVGs as DOM content. Content within iframes is excluded from this metric; (5) Speed Index (SI): SI gauges how quickly content is visually displayed during the page load, providing insights into perceived loading speed; (6) Time to Interactive (TTI): TTI determines how long it takes for a page to become fully interactive. This occurs when meaningful content is displayed, event handlers are registered for the most visible page elements, and the page responds to user interactions within 2.5 seconds, marking it as fully interactive [12].

These Core Web Vitals are essential benchmarks for web developers and administrators, guiding efforts to optimize websites for a superior user experience.

IV. RESULT AND DISCUSSION

A. *CodeQuest Pre-Optimization Tests*

After specifying the test setup parameters, CQ underwent an initial test to assess its performance to identify areas for optimization. The evaluation report generated by GTmetrix is summarized in Table 3. The assessment resulted in an assigned "C" letter grade, equivalent to a performance rating of 70%. The total page performance, as evaluated by GTmetrix, is a comprehensive measure that considers the speed of page loading, its interactivity, and its visual stability. This grade serves as an indicator of the overall user experience provided by CQ.

TABLE III. GTMETRIX PRE-OPTIMIZATION TEST RESULTS

GTmetrix Grade		
Letter Rating	**Performance (70%)**	**Structure (30%)**
C	**70%**	**83%**

The Structure score of 83% demonstrated how CQ was designed for maximum performance.

Indeed, the GTmetrix rating, which considers both the front-end architecture and the real user performance, provides a more comprehensive assessment of the total webpage experience. In this context, the Structure score of 83% reflects the well-designed and optimized architecture of ePondo, indicating that it has been crafted for maximum performance and efficiency. This underscores the importance of the speed of page loading and the underlying design and structure in delivering an optimal user experience.

This concise set of statistics holds paramount significance in optimizing web performance, as it condenses the multitude of timings and audits that scrutinize page performance. These three key aspects of the online user experience—loading speed, interactivity, and visual stability—are each addressed by a distinct measure, making them crucial metrics for enhancing overall web performance and user satisfaction, as shown in Table 4.

TABLE IV. WEB VITALS PRE-OPTIMIZATION TEST RESULTS

Web Vitals			
Measurement	**Parameters**	**Ideal Value**	**Measured Value**
Largest Contentful Paint	Loading	Less than 2.5s	**3.8s**
Total Blocking Time	Interactivity	Less than 150ms	**0ms**
Cumulative Layout Shift	Visual Stability	Less than 0.1	**0**

Performance indicators derived from Lighthouse Performance data are summarized in Table 5.

TABLE V. RE-OPTIMIZATION PERFORMANCE METRICS TEST RESULTS

Ideal Value	Parameters	Measured Time	Remarks
First Contentful Paint	Load speed	**3.8s**	Much longer than recommended

(FCP.)			
Speed Index (SI.)	Load Time	**2.7s**	Much longer than recommended
Largest Contentful Paint (LCP.)	Perceived Loading	**3.8s**	Much longer than recommended
Time to Interactive (TTI.)	Interactivity	**3.8s**	Longer than recommended
Total Blocking Time (TBT)	Responsiveness	**0ms**	Good. Nothing to do here
Cumulative Layout Shift (CLS.)	Visual Stability	**0**	Good. Nothing to do here

These indicators play a vital role in assessing and optimizing web performance: (1) First Contentful Paint (FCP): FCP gauges how swiftly end users can access the page's content. To provide a satisfactory user experience, FCP should be less than 0.9 seconds; (2) Speed Index (SI): SI measures how quickly the above-the-fold portion of the page becomes visually complete. An SI value of less than 1.3 seconds is ideal, but the observed SI value is 2.7 seconds, indicating room for improvement. (3) Largest Contentful Paint (LCP): LCP assesses how users perceive the loading process. The measured LCP value of 3.8 seconds exceeds the desired threshold of less than 1.2 seconds, suggesting a need for optimization; (4) Time to Interactive (TTI): TTI evaluates how rapidly a page loads and identifies instances where a site may appear interactive but is not. The observed TTI value is 3.8 seconds, which should be reduced to less than 2.5 seconds for an improved user experience; (5) Total Blocking Time (TBT): TBT measures the page's responsiveness to user interactions by calculating the duration the website was unresponsive. Reducing TBT enhances user interaction and engagement; and (6) Cumulative Layout Shift (CLS): CLS assesses the perceived visual stability of a page load by measuring unexpected movements of page elements during rendering. The goal is to achieve a CLS value of 0 to prevent disruptive layout shifts.

These performance indicators provide valuable insights into the user experience and offer clear guidance on areas requiring optimization to enhance CQ's performance.

B. *CQ Performance Optimization*

This section delves into the findings from GTmetrix, highlighting key performance issues that need to be addressed. By tackling these concerns, CQ's Performance can be significantly enhanced, leading to a better user experience for its visitors.

1) *Top Issues Impacting Performance*

Several issues were identified impacting the Performance of CQ, as shown in Table 6.

TABLE VI. GTMETRIX AUDIT ON CODEQUEST

Impact	Audit	Remarks
Med-Low	Serve static assets with an efficient cache policy	Potential savings of 237KB
Med-Low	Eliminate render-blocking resources	Potential savings of 271ms
Low	Reduce unused CSS	Potential savings of 95.2KB
Low	Ensure text remains visible during web front load	Two fonts found
Low	Reduce Javascript execution time	705ms spend executing JavaScript

Addressing performance issues highlighted by GTmetrix is essential for improving CQ's overall performance. Here are the key findings and recommended actions:

1. Client-Side Workload: CQ generates a substantial number of client-side workloads, which makes it unsuitable for caching numerous datasets on the client side [12]. This suggests a need to optimize client-side data handling to reduce the computational burden.

2. Eliminate Render-Blocking Resources: Optimizing the elimination of render-blocking resources can potentially save 271ms. Certain resources hinder the first painting of the page. Consider delaying non-essential JavaScript and CSS resources and inline important ones. Two approaches are under consideration: (1) moving tags to the bottom of the page to allow them to be parsed last, eliminating early blocking, and (2) adding async or defer attributes, which require a thorough understanding of site functionality.

3. Optimize CSS by Reducing Unused CSS: There's an anticipated saving of up to 95.2KB by optimizing CSS. Reducing unnecessary stylesheet rules and deferring CSS not required for above-the-fold content can reduce network activity and the bytes used in loading.

4. Ensure Text Visibility During Web Font Load: Projected savings of up to 248 million seconds can be achieved by using the font-display CSS function to maintain text visibility while web fonts are being loaded. This attribute specifies how the font is displayed during page load, considering whether it has been downloaded and is ready for use.

5. Reduce JavaScript Execution Time: JavaScript execution took 705 milliseconds. Minimizing client-server queries is crucial for displaying website information to optimize website performance. This can be achieved through code optimization and minification, code-splitting to load only necessary code during the initial page load, and removing unused or dead code. Minification involves removing

comments, unnecessary code, and white spaces to reduce JavaScript file size.

Addressing these performance issues through optimization strategies will contribute to a faster and more efficient CQ application, ultimately providing an improved user experience.

2) Other techniques considered for optimization

To further optimize CQ's Performance, the following strategies can be implemented:

1. Optimizing Images: High-resolution images can significantly impact bandwidth usage during loading. Large image sizes are a common cause of slow page load times. Compressing images without compromising quality is essential for faster website loading. The paper by Gizas et al. outlines various technologies and approaches that can be synergized to enhance performance, emphasizing design patterns, coding practices, and image-loading procedures [13].

2. Minimizing HTTP Requests: HTTP requests are generated when a browser requests a page, file, or image from a web server. These requests can constitute over 80% of the time it takes for a web page to load, as demonstrated in recent research [14]. Understanding how different user agents handle HTTP/2 priority can provide insights into optimizing these requests, as different vendors may have varying implementations [15].

3. Enabling Compression: Compression is a crucial strategy for conserving bandwidth and improving ePondo's performance. Compression technologies, such as "zipping," can significantly reduce load times, potentially decreasing download times by up to 70% [14].

4. Utilizing CDN (Content Delivery Network): CDN is a network of computers distributed across various data centers designed to provide high availability to end users. It enhances page load times by bringing content closer to users and reducing content download times. Hosting ePondo's media files on a CDN is an effective strategy for speeding up the system, saving up to 60% of bandwidth, and reducing the number of requests by half [16].

By implementing these optimization strategies, CQ can significantly enhance its performance, resulting in faster load times and an improved user experience.

C. CodeQuest Post-Optimization Performance Test

After meticulously implementing the optimization approaches for CodeQuest, a post-optimization system performance test was conducted. The optimization efforts were expected to yield an improvement in CQ's performance.

1) Post-Optimization Testing

As shown in Table 7, GTmetrix upgraded CQ's Performance from a C grade before optimization to an A after optimization. The post-optimization performance rating of 99% represents a significant improvement, marking a 29% increase compared to the initial test result of 70%. This improvement underscores the effectiveness of the optimization efforts in enhancing CQ's overall performance.

TABLE VII. GTMETRIX POST-PERFORMANCE REPORT TEST

GTmetrix Grade		
Letter Rating	**Performance (70%)**	**Structure (30%)**
A	**99%**	**98%**

The performance of web vitals was tested at 694ms, which is significantly faster than the 3.8 seconds recorded before the optimization process, as indicated in Table 8. While interactivity may slightly increase the estimated duration, it remains well below the excellent value of 300ms. Visual stability, as measured by cumulative layout shift, remains excellent at 0. These results demonstrate a substantial improvement in web vitals performance, reflecting the positive impact of the optimization process on CQ's user experience.

TABLE VIII. WEB VITALS POST-PERFORMANCE REPORT TEST

Web Vitals			
Measurement	**Parameters**	**Ideal Value**	**Measured Value**
Largest Contentful Paint	Loading	Less than 2.5s	694ms
Total Blocking Time	Interactivity	Less than 150ms	89ms
Cumulative Layout Shift	Visual Stability	Less than 0.1	0

Table 9 provides an overview of the post-performance metric testing conducted, showcasing the improvements resulting from optimization:

• F.C.P. (First Contentful Paint) was assessed 457ms faster than the initial test, measuring 3.8s.

• S.I. (Speed Index) improved significantly, measuring 571ms, faster than the 2.7s recorded in the initial test.

• LCP (Largest Contentful Paint) also demonstrated improvement, measuring at 694ms, compared to the initial 3.8s.

• T.T.I. (Time to Interactive) showed a substantial boost with a measured value of 661ms, far faster than the pre-test value of 3.8s.

• TBT (Total Blocking Time), indicating responsiveness, had a measured value of 89ms, still better than the outstanding value of 150ms.

• Visual stability, represented by CLS (Cumulative Layout Shift), remained at its best score of 0, indicating excellent visual stability.

These metrics collectively reflect the positive impact of the optimization efforts on CQ's Performance, resulting in faster loading times, improved interactivity, and enhanced user experience.

TABLE IX. POST-PERFORMANCE METRICS REPORT TEST

Metrics	Parameters	Measured Time	Remarks
First Contentful Paint (FCP.)	Performance	457ms	Good. Nothing to do here
Speed Index (SI.)	Load Time	571ms	Good. Nothing to do here
Largest Contentful Paint (LCP.)	Loading	694ms	Good. Nothing to do here
Time to Interactive (TTI.)	Interactivity	661ms	Good. Nothing to do here
Total Blocking Time (TBT)	Responsiveness	89ms	Good. Nothing to do here
Cumulative Layout Shift (CLS.)	Visual Stability	0	Good. Nothing to do here

D. *Findings of Performance Analysis:*

The performance analysis of CodeQuest, both in its initial state and after optimization, provides insightful observations. Initial assessments revealed notable challenges in response time, load times, and resource utilization. Post-optimization, these metrics were re-evaluated to gauge the impact of the applied strategies.

E. *Impact on Key Performance Metrics:*

The optimizations implemented in CodeQuest resulted in significant improvements across key performance metrics. Response times saw a noteworthy reduction, ensuring a more responsive user experience. Load times, both initial and subsequent, were substantially reduced, contributing to quicker interactions for learners. Resource utilization exhibited more efficient usage of server and client-side resources, promoting scalability and stability.

F. *Impact on Overall Learning Experience:*

The improvements in performance metrics directly translated into an enhanced learning experience for CodeQuest users. Learners experienced a more fluid and seamless interaction with the platform, allowing them to focus on the core principles of Object-Oriented Programming. The optimized CodeQuest contributed to a positive perception of the platform's reliability and usability, fostering an environment conducive to effective learning.

G. *Changes in User Engagement:*

Analysis of user engagement metrics indicated a positive shift following performance optimization. User engagement metrics, including session duration, page views, and interaction frequency, exhibited an upward trend. Learners were more likely to spend extended periods on the platform, explore a greater variety of content, and engage more actively in learning activities. These changes underscored the importance of performance optimization in not only attracting but also retaining learner interest.

H. *Addressing Challenges Encountered:*

While the optimization process yielded substantial benefits, certain challenges were encountered. These included intricacies in code refactoring, compatibility issues with existing features, and the need for careful consideration in implementing server-side optimizations. Addressing these challenges required a collaborative effort among the development team, leading to iterative refinements in the optimization strategies.

V. CONCLUSION

In conclusion, our study successfully achieved its objectives of identifying and rectifying performance bottlenecks in the CodeQuest platform dedicated to Object-Oriented Programming (OOP). Through strategic optimizations targeting client-side workloads, render-blocking resources, and other key areas, we significantly improved CodeQuest's performance metrics. The post-optimization results, including a notable increase in the GTmetrix grade from 'C' to 'A' and enhanced web vitals metrics, reflect a transformative impact on the learning experience. Learners now benefit from faster load times, heightened interactivity, and improved overall usability. While challenges were encountered, collaborative efforts led to successful code refinements. CodeQuest stands revitalized as a model for OOP education and a blueprint for enhancing online learning platforms, emphasizing the crucial role of performance optimization in shaping effective and enjoyable learning experiences.

REFERENCES

[1] William P. Rey (2024). CodeQuest App: A Gamified OOP Education for Dynamic Online Learning Engagement. 2024 The 6th International Conference on Computer Science and Technologies in Education (CSTE 2024), Xi'an, China.

[2] Jaime Govea, Ernesto Edye, Solange Revelo-Tapia, & William Villegas (2023). Optimization and Scalability of Educational Platforms: Integration of Artificial Intelligence and Cloud Computing. Computers. 12. 223. 10.3390/computers12110223.

[3] Z. Ren. (2023) Optimization of Innovative Education Resource Allocation in Colleges and Universities Based on Cloud Computing and User Privacy Security. Wireless Pers Commun. https://doi.org/10.1007/s11277-023-10572-4

[4] K. Kuosa, D. Distante, A. Tervakari, L. Cerulo, A. Fernández, J. Koro, & M. Kailanto (2016). Interactive Visualization Tools to Improve Learning and Teaching in Online Learning Environments. Int. J. Distance Educ. Technol., 14, 1-21.

[5] Brenda Cecilia Padilla Rodriguez, Alejandro Armellini & Ma Concepción Rodriguez Nieto (2020) Learner engagement, retention and

success: why size matters in massive open online courses (MOOCs), Open Learning: The Journal of Open, Distance and e-Learning, 35:1, 46-62, DOI: 10.1080/02680513.2019.1665503

[6] William Penaflor Rey and Ronaldo Juanatas (2022) Towards a Performance Optimization of Mobile Automated Fingerprint Identification System (MAFIS) for the Philippine National Police. ICCAI '22: Proceedings of the 8th International Conference on Computing and Artificial Intelligence. March 2022. Pages 380–386. https://doi.org/10.1145/3532213.3532270.

[7] William Penaflor Rey, Patricia Mae S. Gacusan, Ma. Nicole Dominique S. Leynes, and Rey Martin A. Destacamento (2022). ePondo: A Web-Based Rewards Crowdfunding Platform. ICEBI 2022: Proceedings of the 2022 6th International Conference on E-Business and Internet. October 2022. Pages 96–105. https://doi.org/10.1145/3572647.3572662

[8] William Penaflor Rey (2024). Performance Optimization of Mobile Automated Biometric Identification System (MABIS) for Philippine Law Enforcement Agency. 2024 10th International Conference on Computing and Data Engineering (ICCDE 2024). January 15-17, 2024, Bangkok, Thailand.

[9] GTMetrix, "Optimization Explained." April 22, 2021. https://gtmetrix.com/

[10] Editorial Staff, "How to Use GTMetrix Plugin to Improve WordPress Site Performance," December 12, 2020. https://www.wpbeginner.com/plugins/how-to-use-gtmetrix-plugin-to-improve-wordpress-site-performance/

[11] Addy Osmani, "Optimizing Web Vitals using Lighthouse," web. dev. May 11, 2021. https://web.dev/optimize-vitals-lighthouse/

[12] Ah Zau Marang, "Analysis of web performance optimization and its impact on user experience," KTH Royal Institute of Technology, Stockholm, Sweden. 2018.

[13] A. B. Gizas & S. P. Christodoulou, "Performance-optimized pages' architecture, navigation and images techniques for JQuery mobile sites." PCI '15: Proceedings of the 19th Panhellenic Conference on Informatics. Athens, Greece, 2015. https://doi.org/10.1145/2801948.2801995

[14] Atman Rathod, "How to improve GTmetrix score of your website" CMARIX Technolabs Pvt. Ltd., October 30, 2017

[15] https://www.cmarix.com/blog/how-to-improve-gtmetrix-score-of-your-website/

[16] M. Wijnants, R. Marx, P. Quax, & W. Lamotte, "HTTP/2 Prioritization and its Impact on Web Performance". WWW '18: Proceedings of the 2018 World Wide Web Conference. April 2018. Lyon France, 2018. https://doi.org/10.1145/3178876.3186181