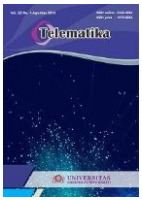

Available online at :
http://ejournal.amikompurwokerto.ac.id/index.php/telematika/

Telematika

Accreditated SINTA "2" Kemenristek/BRIN, No. 85/M/KPT/2020

UNIVERSITAS AMIKOM PURWOKERTO

Functional Evaluation of the Logia Dashboard Using Boundary Value Testing and Cause-Effect Graph Techniques

Muhammad Rizky Aulia Ramadhan[1], Friska Abadi[2], Dodon Turianto Nugrahadi[3], Setyo Wahyu Saputro[4], Rudy Herteno[5]

[1,2,3,4,5]Department of Computer Science, Faculty of Mathematics and Natural Sciences, Universitas Lambung Mangkurat, Banjarbaru, Indonesia

ARTICLE INFO

History of the article:
Received March 15, 2025
Revised April 20, 2025
Accepted August 27, 2025

Keywords:
Black-box Testing
Boundary Value
Cause Effect Graph
Performance Testing

Correspondence:

E-mail: friska.abadi@ulm.ac.id

ABSTRACT

The Logia Dashboard is a web-based information system used to manage rehabilitation plant data on post-mining land. As an alpha-stage system, Logia requires thorough functional and performance evaluation to ensure that all input validations, logical processes, and system responses operate correctly before wider implementation. This study aims to evaluate the functional reliability and performance of the Logia Dashboard by applying a combined approach of Boundary Value Testing (BVT) and Cause-Effect Graph (CEG) techniques, supported by performance testing using Google Lighthouse. The research design adopts a black-box testing approach. BVT is applied to validate input boundaries on critical features, including login, data editing, QR code generation, and account creation. Meanwhile, CEG is used to model logical relationships between input conditions and system outputs to generate systematic test cases. A total of 39 optimized functional test cases were executed in a controlled local environment. Performance testing was conducted using Lighthouse by measuring key metrics such as First Contentful Paint (FCP), Largest Contentful Paint (LCP), Total Blocking Time (TBT), and Cumulative Layout Shift (CLS). The functional testing results show that 37 out of 39 test cases passed, yielding a success rate of 94.87%. Two failed cases were identified in the login feature, indicating weaknesses in input validation feedback. Performance testing produced an average Lighthouse score of 97, demonstrating that the system has excellent load speed and interface stability, although minor layout instability was detected on certain pages. These results indicate that the combined application of BVT and CEG is effective for detecting boundary-related and logical input errors in alpha-stage web systems. The findings also provide concrete recommendations for improving login validation and interface stability, supporting further development of the Logia Dashboard toward a more reliable and robust system for post-mining land management.

1. INTRODUCTION

In recent years, web-based information systems have played an increasingly important role in supporting data management across various sectors, including environmental rehabilitation and post-mining land management (Axza et al., 2023). The reliability of web interfaces and system performance has also been shown to significantly influence user experience and decision-making processes in data-driven systems (Cahyono & Kamarudin, 2024). Errors in data input, validation, or processing can lead to incorrect analysis and inaccurate rehabilitation planning, particularly in environmental monitoring systems (Pradipta et al., 2024). Therefore, ensuring software quality through systematic testing is a fundamental requirement, especially at the early stages of system development where potential defects can still be corrected with minimal impact (Gustinov et al., 2023).

The Logia Dashboard is a web-based administrative information system designed to manage rehabilitation plant data on post-mining land. This system supports several core functions, including user authentication, plant data input and editing, QR code generation for plant identification, data verification, and visualization of rehabilitation progress through tables, maps, and analytical dashboards. However, as

the system is currently in its alpha development stage, it has not yet undergone comprehensive functional and performance testing (Emadi, 2023). This condition raises potential risks related to improper input validation and logical processing errors that may compromise data reliability if not properly addressed (Akinboboye et al., 2021). In addition, unstable system performance may reduce the effectiveness of system usage in real operational environments (Siahaan & Vianto, 2021).

Software testing strategies for web-based systems are commonly implemented using black-box testing methods, which evaluate system functionality based on input-output behavior without requiring access to internal program structures (Huriati et al., 2020). This approach is also effective in identifying security vulnerabilities and injection flaws in web applications (Althunayyan et al., 2022). Among black-box techniques, Boundary Value Testing (BVT) is widely used to detect errors that occur near the minimum and maximum limits of input values, which are statistically more prone to failures (Permatasari et al., 2023). The effectiveness of BVT in detecting input validation errors has also been demonstrated in web-based information systems with numeric and text input constraints (Maulana et al., 2023). Meanwhile, the Cause-Effect Graph (CEG) method is applied to model logical relationships between input conditions and output responses, enabling the systematic derivation of test cases from complex decision logic (Ismiati et al., 2024). The applicability of CEG in improving functional test coverage has been previously demonstrated in service-based and governmental information systems (Selviana et al., 2023).

Although BVT and CEG have been proven effective when applied individually, most previous studies still implement these methods separately and focus only on functional correctness (Widia et al., 2021). Very few works have explored their combined application in a single testing framework, particularly for alpha-stage environmental information systems. In addition, system performance is a critical quality attribute for web applications, as slow response times and unstable layouts can negatively affect user experience (Barus et al., 2022). Performance evaluation using Google Lighthouse has been widely used to assess loading speed, interactivity, and visual stability of modern web applications (Siahaan & Vianto, 2022). The use of Web Vitals indicators such as LCP, TBT, and CLS has also been shown to provide objective measurements of web interface quality (Anggraeni et al., 2024).

Based on these gaps, this study applies a combined approach of Boundary Value Testing (BVT) and Cause-Effect Graph (CEG) to evaluate the functional reliability of the Logia Dashboard and complements it with performance testing using Google Lighthouse. BVT is used to detect potential errors around critical input boundaries, while CEG is employed to analyze logical dependencies between input conditions and system outputs. Performance testing is conducted to ensure that the system not only functions correctly but also provide fast and stable user interactions in real usage scenarios.

This study maintains several limitations to keep the research focused, including: (1) testing is conducted only on the alpha version of the system, (2) Functional testing was limited to system features that contain explicit input restrictions , and (3) performance testing is conducted using Lighthouse under a stable internet connection and only on the user interface.

This study contributes both methodologically and practically. From a methodological perspective, it demonstrates how the integration of BVT and CEG can generate more systematic, efficient, and representative test cases for functional testing. From a practical perspective, the results provide concrete recommendations for improving input validation, login reliability, and interface stability of the Logia Dashboard. In a broader context, this research supports the development of more reliable web-based information systems for post-mining land rehabilitation and environmental data management.

2. RESEARCH METHODS

This study employs a black-box testing approach and automated performance testing to evaluate the quality of the Logia Dashboard system. The object of this study is the alpha version of the Logia Dashboard released in November 2024, a web-based information system designed for managing rehabilitation plant data on post-mining land. The system provides several core features, including user authentication, plant data input and editing, QR code generation, data verification, and visualization through tables, maps, and analytical dashboards. This study does not propose a new testing method, but applies an integrated combination of Boundary Value Testing (BVT) and Cause-Effect Graph (CEG) to evaluate the functional reliability of the system.

Functional testing was conducted manually in a controlled local client–server environment using Visual Studio Code (VS Code) as the development platform (Wintana et al., 2022). The testing environment specifications consisted of an Intel Core i7 processor, 16 GB RAM, Windows 10 operating system, and Google Chrome browser version 120 or later. The local server was configured using Apache through XAMPP, with PHP and MySQL as the backend and database management system. Functional testing was

performed by combining two methods: Boundary Value Testing (BVT) and Cause-Effect Graph (CEG). BVT was used to test input value boundaries, such as character length in input fields and specific numeric limits, and has been proven effective in identifying errors caused by values near domain boundaries (Pratama et al., 2023).

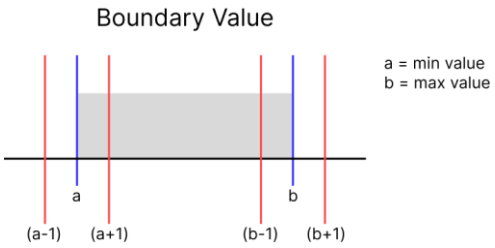

Figure 1. Boundary Value

As illustrated in Figure 1 (Perbawa & Nurohim, 2020), Boundary Value Testing (BVT) was conducted by selecting six critical test values for each input field: below the minimum limit, at the minimum limit, above the minimum limit, below the maximum limit, at the maximum limit, and above the maximum limit (Kartono et al., 2024). For example, for a valid input range of 1–10, the tested values were 0, 1, 2, 9, 10, and 11. In this study, BVT was applied to several critical input fields including username, password, and QR code amount, with boundary limits determined based on the validation rules implemented in the Logia Dashboard.

The Cause-Effect Graph (CEG) method was applied to model the logical relationships between input conditions (causes) and system responses (effects). The procedure began by identifying relevant causes and effects for each tested feature. These relationships were then represented using Boolean operators (AND, OR, and NOT) and transformed into a decision table containing all possible input combinations. From this decision table, representative and valid test cases were selected and executed manually to verify whether the system output matched the expected logical behavior. This procedure enables systematic detection of logical and rule-based processing errors (see figure 2).

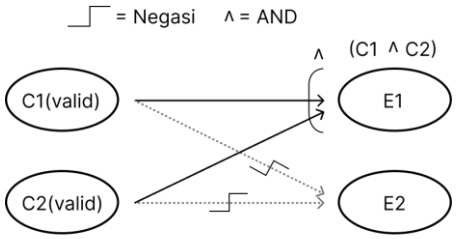

Figure 2. Cause Effect Graph

The test cases designed from both methods were then combined and optimized by selecting one representative test case when both methods addressed the same functional objective in order to eliminate redundancy and improve testing efficiency. The combination of BVT and CEG is methodologically justified because BVT is effective in detecting input validation errors at boundary limits, while CEG is effective in identifying logical and rule-based processing errors caused by specific combinations of input conditions. By integrating both methods, this study is able to detect both numerical/textual boundary failures and logical decision errors that cannot be fully captured by a single method alone. Once the final test cases were determined, functional testing was executed manually on the system in a local environment. The success rate of the functional testing was calculated using the following formula:

$$Success\ Rate\ (\%) = \left(\frac{P}{T}\right) \times\ 100 \quad (1)$$

P represents the number of passed test cases and T represents the total number of executed test cases.

A test case was classified as passed if the system output fully matched the expected result defined in the test scenario, including correct input validation, successful data processing, and appropriate system responses. Conversely, a test case was classified as failed if any deviation occurred, such as incorrect validation behavior, system error messages, incorrect data processing, or unexpected output results.

In addition to functional testing, performance testing was conducted on the deployed Logia Dashboard hosted on Google Cloud Platform (GCP) using App Engine and accessed via a public HTTPS endpoint. Performance evaluation was performed using Google Lighthouse in Chrome DevTools, measuring First Contentful Paint (FCP), Speed Index, Total Blocking Time (TBT), Cumulative Layout Shift (CLS), and the overall performance score as indicators of loading speed, interactivity, and visual stability (Developers, 2024).

Lighthouse testing was executed in desktop mode with a simulated fast 4G network, CPU throttling disabled, device emulation disabled, and browser cache cleared before each test. Each test was repeated several times, and the average values were used as the final results to ensure consistency and reduce environmental bias (Siahaan & Vianto, 2022). By integrating BVT and CEG for functional testing with Lighthouse-based performance evaluation, this study provides a comprehensive assessment of both the functional reliability and performance efficiency of the Logia Dashboard in the alpha stage.

3. RESULTS AND DISCUSSION

Functional testing was conducted on features within the Logia Admin Dashboard system that were identified as having input constraints, namely: login, edit data, generate QR code, and add account. For each feature to be tested, test cases were designed using two black-box methods: Boundary Value Testing (BVT) and Cause-Effect Graph (CEG).

One of the examples is the generate QR code feature, where valid input scenarios were identified as detailed in Table 1. BVT was used to evaluate the numeric boundary of the input for the number of QR codes that can be generated, which ranges from 1 to 99,999. Based on this range, six test values were determined: below the lower bound (BLB), lower bound (LB), above the lower bound (ALB), below the upper bound (BUB), upper bound (UB), and above the upper bound (AUB). The BVT test values are detailed in Table 2.

Table 1. Test Values for the Generate QR Code Feature

Atribut	**Values**
Scenario Name	Generate QR
Objective	Performing QR Code generation
Initial Condition	On the Table page
Final Condition	QR Code generated successfully
Input Field	QR Amount
Valid Input Requirements	1. Input field is not empty 2. The number of QR Codes is not less than 1 and not more than 99,999 3. Input is numeric

Table 2. Test Values for Generate QR Code Feature

Data Input	**Values**	**Details**
QR Amount	0	BLB
QR Amount	1	LB
QR Amount	2	ALB
QR Amount	99,998	BUB
QR Amount	99,999	UB

QR Amount	100,000	AUB

Based on Table 3 and Table 4, the total number of combinations of causes is $2^3 = 8$ combinations. These combinations include several logically invalid scenarios due to interdependencies between the causes. For instance, if C1 is false (the input field is empty), then C2 and C3 must also be false, since there is no input to validate. Similarly, if C2 is false (the input is not numeric), then C3 must be false as well, as the input cannot be evaluated against a numeric range. These logical rules and dependencies can be visualized using a diagram, as shown in Figure 3.

Table 3. Generate QR Code Cause

Cause	Details
C1	Input field is not empty
C2	Input is numeric
C3	The number of QR Codes is not less than 1 and not more than 99,999

Table 4. Generate QR Code Effect

Effect	Details
E1	QR Code generated successfully
E2	Displays an error message

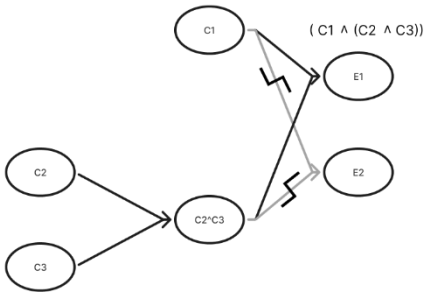

Figure 3. Cause Effect Graph

After identifying the relationships between causes and effects, the possible input combinations were organized into a decision table for systematic analysis. The decision table was then optimized using the don't-care condition method, resulting in four principal combinations as shown in Table 5. The don't-care notation ("–") is used to represent input conditions that do not influence the output, allowing multiple columns with equivalent logical outcomes to be merged. This technique effectively reduces the number of test-case combinations without compromising coverage, as recommended in structured decision-table simplification practices (Bialy et al., 2015). The logic behind this reduction is consistent with Boolean minimization approaches that leverage don't-care terms to simplify logical expressions (Nugroho, 2021) and reflects widely adopted decision-modeling principles in DMN-based decision tables (Leoni et al., 2021).

Table 5. Optimized Decision Table

C1	C2	C3	E1	E2
True	True	True	Yes	No
True	True	False	No	Yes
True	False	-	No	Yes

False	-	-	No	Yes

From the results of the BVT and CEG analysis, a total of 10 initial test cases were generated. However, upon evaluation, redundancies were identified. For example, test case TC001 (BVT – input 0) had the same objective as TC008 (CEG – input 0, does not meet the minimum requirement). Therefore, one of them can be eliminated without losing test coverage. A similar approach was applied to TC002 (BVT – input 1) and TC007 (CEG – valid input). The combined results are shown in Table 6, and the final outcome after optimization resulted in eight test cases listed in Table 7.

Table 6. Combined Test Cases

ID	Methods	Values	Details
TC001	BVT	0	BLB
TC002	BVT	1	LB
TC003	BVT	2	ALB
TC004	BVT	99,998	BUB
TC005	BVT	99,999	UB
TC006	BVT	100,000	AUB
TC007	CEG	1	Condition fulfilled
TC008	CEG	0	The quantity of QR codes does not comply with the defined limits.
TC009	CEG	a	Not a number
TC010	CEG	-	No input

Table 7. Optimized Test Cases

ID	Data Input	Details
GQR_001	0	BLB
GQR_002	1	LB
GQR_003	2	ALB
GQR_004	99,998	BUB
GQR_005	99,999	UB
GQR_006	100,000	AUB
GQR_007	a	Not a number
GQR_008	(empty)	No input

By reducing two test cases from the original total of ten, testing efficiency increased by 20% without compromising the validity or coverage of the tests. This strategy demonstrates that combining and optimizing two black-box methods can produce a more systematic and resource-efficient test case design. It also helps minimize excessive testing and focuses on the most representative scenarios.

Table 8. Functional Testing Results

Features	Test Case	Passed	Failed
Login	5	3	2
Edit Data	11	11	0
Generate QR	8	8	0
Add Account	15	15	0
total	39	37	2

As shown in Table 8, a total of 39 test cases were executed across the Login, Edit Data, Generate QR Code, and Add Account features, of which 37 passed and 2 failed. Using the calculation defined in Equation (1), the system achieved a functional success rate of 94.87%. Both failed cases were found in the Login feature—one involving an invalid email format and the other an incorrect password. In these scenarios, the system refreshed the page without displaying any error message, indicating inadequate frontend error-handling and missing user feedback mechanisms (Inal, 2024). In contrast, all test cases for Edit Data, Generate QR Code, and Add Account passed successfully, demonstrating stable functional behavior.

These results indicate that the system is generally reliable; however, the Login feature requires improvement. The error-handling logic should be updated so that invalid credentials produce clear feedback messages rather than a silent page reload. Enhancing both client-side and server-side validation would significantly improve usability and overall system robustness (Alomari et al., 2020).

For performance testing, Lighthouse was used to evaluate several key metrics, including First Contentful Paint (FCP), Largest Contentful Paint (LCP), Total Blocking Time (TBT), Cumulative Layout Shift (CLS), and Speed Index (SI). According to Google Chrome Developers, the recommended thresholds for good performance are: FCP < 1.8 s, LCP < 2.5 s, TBT < 200 ms, CLS < 0.1, and SI < 4.3 s.

Table 9. Performance Testing Results

Pages	**FCP (s)**	**LCP (s)**	**TBT (ms)**	**CLS**	**SI (s)**	**Performance**
Login	0.7	0.7	0	0.002	1.2	99
Dashboard	0.7	1.4	110	0.008	1.1	94
Table	0.6	0.9	40	0.044	0.8	98
Map	0.9	1.3	30	0.003	2.1	92
Analytics	0.8	1	40	0.048	0.9	98
Data Verification	0.6	0.8	40	0.052	0.9	99
Comparison	0.7	0.7	0	0.034	0.7	99
Manage Option	0.7	1	50	0.001	0.7	98
manage User	0.6	0.9	20	0.036	0.6	99
History	0.8	1	40	0.119	0.8	94
Average						97

As detailed in Table 9, all system pages achieved performance scores above 90, with an average score of 97. Although all Lighthouse metrics fall within acceptable limits, several areas still require optimization. For example, the map page recorded the lowest score (92), primarily due to slightly higher Speed Index (SI) and Cumulative Layout Shift (CLS) values, which indicate minor rendering delays and

layout instability. The dashboard page shows a Total Blocking Time (TBT) of 110 ms and a Largest Contentful Paint (LCP) of 1.4 seconds—both still categorized as "good," but suggesting that JavaScript execution and resource loading could be further optimized to enhance responsiveness. Additionally, the history page has the highest CLS value (0.119), slightly exceeding the ideal threshold of 0.1, which may cause subtle visual disruptions during loading. This issue can be improved by defining fixed sizes for visual elements, such as images or map components, to prevent layout shifts and ensure a more stable rendering experience across all pages (Sumakul & Mailoa, 2023).

Overall, the combination of functional testing methods and performance analysis provides a comprehensive overview of the system's reliability. The integration of BVT and CEG proved consistent with previous studies such as Maulana et al. (2023) and Selviana et al. (2023), which also reported that boundary-value issues are more frequent than logical decision errors. In this study, boundary-related failures in the Login feature aligned with similar patterns observed in earlier black-box evaluations, emphasizing the importance of strict server-side validation. The relationship between functional and performance results also indicates that although functional errors did not directly impact performance scores, unresolved validation weaknesses have the potential to trigger additional processing load and layout instability, particularly on pages with more dynamic rendering such as the map and history pages.

4. CONCLUSIONS AND RECOMMENDATIONS

This study confirms that the combined use of Boundary Value Testing (BVT) and Cause-Effect Graph (CEG) is effective for assessing the functional reliability of the Logia Dashboard. With a functional success rate of 94.87%, the system exhibited only boundary-related input issues, aligning with findings from previous black-box testing studies. Performance evaluation using Lighthouse also showed strong results, with all pages scoring above 90 and meeting Web Vitals thresholds, despite minor layout instability on specific pages.

Overall, the system demonstrates reliable behavior in its alpha stage yet requires improvements in login validation and interface stability. Future work should extend this combined testing approach to more complex modules, incorporate security testing, and evaluate performance across varied platforms to ensure broader operational robustness.

REFERENCES

Akinboboye , O., Afrihyia, E., Frempong , D., Appoh, M., Omolayo, O., Umar, M. O., . . . Okoli, I. (2021). A risk management framework for early defect detection and resolution in technology development projects. *International Journal of Multidisciplinary Research and Growth Evaluation, 2*(4), 958-974. https://doi.org/10.54660/.IJMRGE.2021.2.4.958-974

Alomari, H. W., Ramasamy, V., Kiper, J. D., & Potvin , G. (2020). A user interface (UI) and user experience (UX) evaluation framework for cyberlearning environments in computer science and software engineering education. *Heliyon, 6*, 1-18. https://doi.org/10.1016/j.heliyon.2020.e03917

Althunayyan, M., Saxena, N., Li, S., & Gope, P. (2022). Evaluation of black-box web application security scanners in detecting injection vulnerabilities. *Electronics, 11*(13), 2049. https://doi.org/10.3390/electronics11132049

Anggraeni, O. S., Sugiarto, L., & Agustin, T. (2024). Studi komparatif performa framework JavaScript modern dalam pengembangan aplikasi web. *Modem: Jurnal Informatika dan Sains Teknologi, 2*(4), 162–177. https://doi.org/10.62951/modem.v2i4.239

Axza, F., Sofi'ie, F., & Qoiriah, A. (2023). Analisis perbandingan framework front-end JavaScript React dan Vue pada pengembangan website. *Journal of Informatics and Computer Science (JINACS), 5*(2), 157–164. https://doi.org/10.26740/jinacs.v5n02.p157-164

Barus, A. C., Sinambela, E. S., Purba, I., Simatupang, h., Marpaung, M., & Pandjaitan, N. (2022). Performance testing and optimization of DiTenun website. *Journal of Applied Science, Engineering, Technology, and Education, 4*(1), 45-54. https://doi.org/10.35877/454RI.asci841

Bialy, M., Lawford, M., Pantelic, V., & Wassyng, A. (2015). A methodology for the simplification of tabular designs in model-based development. *2015 IEEE/ACM 3rd FME Workshop on Formal Methods in Software Engineering* (pp. 47-53). Florence, Italy: IEEE. https://doi.org/10.1109/FormaliSE.2015.15

Cahyono, N., & Kamarudin. (2024). Perbandingan GTMetrix, Lighthouse, Pingdom dan PageSpeed Insight dalam evaluasi performa website. *Jurnal Ilmiah Media Sisfo, 18*(2), 201–210. https://doi.org/10.33998/mediasisfo.2024.18.2.1901

Developers, G. (2024). *Performance scoring in Lighthouse*. Retrieved 2024, from Google Chrome Developer Documentation: https://developer.chrome.com/docs/lighthouse/performance/performance-scoring?hl=id

Emadi, J. (2023). The development of a design theory for web-based information systems. *Journal of Robotics Spectrum, 1*, 13-23. https://doi.org/10.53759/9852/JRS202301002

Gustinov, M. D., Azani, N. W., Ghani, R. A., Auliani , S. N., Maharani, S., Hamzah, M. L., & Rizki, M. (2023). Analysis of web-based e-commerce testing using black box and white box methods. *International Journal of Information System and Innovation Management, 1*(1), 20-31. https://doi.org/10.55583/ijisim.v1i1.687

Huriati, P., Azmi , H., Wati, Y., Meidelfi, D., & Lestari, T. (2020). Black box testing on the online quiz application using the equivalence partitions method. *International Journal of Advanced Science Computing and Engineering, 2*(2), 51-56. https://doi.org/10.62527/ijasce.2.2.48

Inal, Y. (2024). Usability and optimization of online apps in user's context. *PeerJ. Computer science, 10*, 1-27. https://doi.org/10.7717/peerj-cs.2561

Ismiati, S., Aditiawan, F. P., & Nurlaili, A. L. (2024). Black box testing with the equivalence partitioning and cause effect graph method in archive information system. *Jurnal Teknik Informatika (JUTIF), 5*(4), 981–990. https://doi.org/10.52436/1.jutif.2024.5.4.1944

Kartono, F. K., Nursaadah, S., Nugroho, M. N., Tama, D. A., Mashudi, F. A., Wicaksono, A., & Nasir, M. (2024). Pengujian black box testing pada sistem website Osha Snack: Pendekatan teknik boundary value. *Jurnal Kridatama Sains Dan Teknologi, 6*(2), 754-766. https://doi.org/10.53863/kst.v6i02.1407

Leoni, M. d., Felli, P., & Montali, M. (2021). Integrating BPMN and DMN: Modeling and analysis. *J Data Semant 10, 10*, 165–188. https://doi.org/10.1007/s13740-021-00132-z

Maulana, B. A., Mawarni, E., Hidayattuloh, M. Y., Suryawijaya, V., & Saifudin, A. (2023). Pengujian black box pada sistem informasi barang berbasis web menggunakan metode boundary value analysis. *OKTAL : Jurnal Ilmu Komputer dan Science, 2*(6), 1747-1753. Retrieved 2024, from https://journal.mediapublikasi.id/index.php/oktal/article/view/3094

Nugroho, E. D. (2021). Development of applications for simplification of Boolean functions using Quine–McCluskey method. *Telematika: Jurnal Telematika Dan Teknologi Informasi, 18*(1), 27-36. https://doi.org/10.31315/telematika.v18i1.3195

Perbawa, D. S., & Nurohim, G. S. (2020). Pengujian aplikasi berbasis website dengan black box testing metode boundary value analysis dan responsive testing. *Journal Speed–Sentra Penelitian Engineering dan Edukasi, 12*(4), 1-5.

Permatasari, I., Adhania, F., Putri, S. A., & Nursari, S. R. (2023). Pengujian black box menggunakan metode analisis nilai batas pada aplikasi DANA. *Konstelasi: Konvergensi Teknologi dan Sistem Informasi, 3*(2), 373-387. https://doi.org/10.24002/konstelasi.v3i2.8289

Pradipta, I. G., Kusuma, I. D., & Yuhana, U. L. (2024). Black box testing in the Access by KAI application using boundary value analysis and graph-based testing. *INFOTECH Journal, 10*(1), 122-127. https://doi.org/10.31949/infotech.v10i1.9577

Pratama, S. D., Lasimin, L., & Dadaprawira, M. N. (2023). Pengujian black box testing pada aplikasi Edu Digital berbasis website menggunakan metode equivalence dan boundary value. *Jurnal Teknologi Sistem Informasi dan Sistem Komputer TGD, 6*(2), 560-569. https://doi.org/10.53513/jsk.v6i2.8166

Selviana, E. Z., Wahanani, H. E., & Aditiawan, F. P. (2023). Black box testing in community service systems using boundary value analysis and cause effect graph methods (Case study of Jombang District Community and Village Empowerment Service). *East Asian Journal of Multidisciplinary Research, 2*(10), 4161-4184. https://doi.org/10.55927/eajmr.v2i10.6435

Siahaan, M., & Vianto, V. O. (2021). Comparative analysis study of front-end JavaScript frameworks performance using Lighthouse tool. *Jurnal Mantik, 6*(3), 3463-3468. https://doi.org/10.35335/mantik.v6i3.3131

Sumakul, J. R., & Mailoa, E. (2023). Analisa performa website kabupaten kota di Provinsi Sulawesi Utara menggunakan website performance testing tools. *Jurnal Minfo Polgan, 12*(1), 1262-1271. https://doi.org/10.33395/jmp.v12i1.12701

Widia, D. M., Rosalin, S., Asriningtias, S. R., & Sonalita, E. (2021). Black box testing menggunakan boundary value analysis dan equivalence partitioning pada aplikasi pengadaan bahan baku batik dengan pendekatan use case. *JIMP : Jurnal Informatika Merdeka Pasuruan, 7*(2), 15-21. https://doi.org/10.37438/jimp.v6i1.300

Wintana, D., Pribadi, D., & Nurhadi, M. Y. (2022). Analisis perbandingan efektivitas white-box testing dan black-box testing. *Jurnal Ladang Artikel Ilmu, 2*(1), 8-16. https://doi.org/10.31294/larik.v2i1.1382