

IMPLEMENTASI TEKNIK GREY BOX TESTING PADA PENGUJIAN APLIKASI F1MATH

Oleh:

Ang Vincent Colin¹, Syaiful Rahman^{2*}, Hasniati³

^{1,2,3} Teknik Informatika, STMIK Kharisma Makassar

e-mail:¹angvincent_21@kharisma.ac.id, ²syaifulrahman@kharisma.ac.id,

³hasniati@kharisma.ac.id

Abstrak: Aplikasi F1Math dikembangkan untuk membantu siswa dalam mempelajari matematika melalui konsep interaktif yang terinspirasi dari dunia balap Formula 1. Metode testing yang digunakan adalah grey box testing dengan pendekatan Regression Testing, yang menggabungkan analisis terhadap struktur internal aplikasi dan pengamatan terhadap perilaku eksternal. Proses testing dimulai dengan analisis spesifikasi sistem dan kode program, dilanjutkan dengan testing unit, integrasi, dan regresi, serta didukung oleh simulasi pengguna pada berbagai level permainan. Hasil Testing menunjukkan bahwa teknik grey box testing efektif dalam mendeteksi bug pada tahap awal pengembangan. Penerapan teknik ini mempercepat proses perbaikan dan meningkatkan stabilitas aplikasi, khususnya pada fitur permainan dan perhitungan matematika.

Kata kunci: Testing Aplikasi, grey box testing, Firebase, Game Matematika, Game Balapan.

Abstract: The F1Math application was developed to assist students in learning mathematics through an interactive concept inspired by Formula 1 racing. The testing method used is grey box testing with a Regression testing approach, combining an analysis of the application's internal structure and observation of its external behavior. The testing process began with analyzing the system specifications and source code, followed by unit Testing, integration Testing, and regression Testing. User simulations across various game levels were also conducted. The results indicate that the grey box testing technique is effective in detecting bugs during the early stages of development. This approach accelerates the debugging process and enhances application stability, especially in critical areas such as gameplay features and mathematical.

Keywords: Application Testing, grey box testing, Firebase, Math Game, Racing Game

1. PENDAHULUAN

Aplikasi F1Math dirancang untuk meningkatkan minat belajar matematika melalui konsep balap Formula 1. Penelitian ini bertujuan untuk menguji efektivitas teknik grey box testing dalam proses testing aplikasi F1Math. Namun, kompleksitas integrasi antarmuka, logika bisnis, dan *backend* memerlukan metode *testing* yang efektif. *grey box testing* dipilih karena menggabungkan keunggulan *White box* dan *Black box* untuk mengoptimalkan proses *Testing*. Dalam pengembangan dan pemeliharaan aplikasi seperti F1Math, proses *testing* perangkat lunak menjadi langkah yang krusial untuk memastikan bahwa aplikasi tersebut berfungsi sebagaimana mestinya dan memenuhi standar kualitas yang ditetapkan.

* Corresponding author : Syaiful Rahman (syaifulrahman@kharisma.ac.id)

Namun, *testing* perangkat lunak sering kali merupakan tantangan yang kompleks, terutama dengan adanya berbagai macam perangkat keras dan perangkat lunak yang digunakan oleh pengguna.[1] Salah satu pendekatan yang telah diperkenalkan untuk meningkatkan efektivitas *Testing* perangkat lunak adalah teknik *grey box testing*. Sedangkan *testing grey Box* merupakan sebuah kombinasi *testing* antara *White-box* dan *Black-box*. [2] di mana *testing* dilakukan dengan pengetahuan sebagian tentang internal struktur kode sumber aplikasi. [2] Dengan demikian, *grey box testing* dapat memberikan keuntungan dalam mengidentifikasi dan mengatasi potensi cacat atau kelemahan dalam aplikasi dengan lebih efisien.

Dalam konteks aplikasi F1Math, yang kompleksitasnya terletak pada interaksi antarmuka pengguna, logika bisnis, dan *backend* sistem, implementasi teknik *grey box testing* menjadi relevan. Dengan memahami sebagian struktur internal aplikasi ini, tim *testing* dapat lebih efektif dalam menemukan masalah potensial dan meningkatkan kualitas keseluruhan aplikasi. [3] Oleh karena itu, dalam rangka meningkatkan kualitas dan keandalan aplikasi F1Math, penelitian ini mengusulkan untuk mengimplementasikan teknik *grey box testing* dalam proses *testing* perangkat lunak. Diharapkan bahwa hasil dari penelitian ini akan memberikan wawasan yang berharga dalam pengembangan aplikasi perangkat lunak, khususnya dalam konteks aplikasi pendidikan, serta membantu meningkatkan pengalaman pengguna dalam memahami konsep matematika melalui aplikasi F1Math.

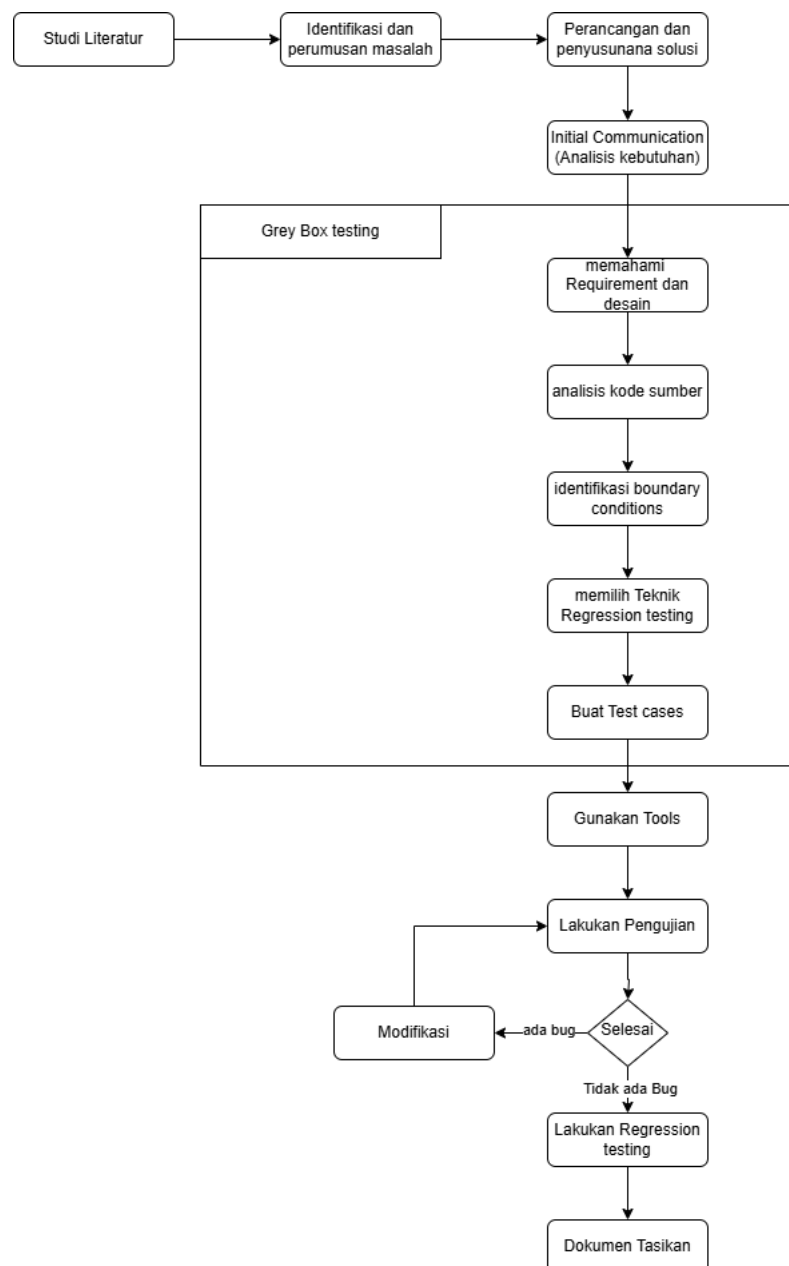
Tujuan dari penelitian ini adalah untuk menguji efektivitas penerapan metode *grey box testing* dalam mengidentifikasi cacat pada aplikasi F1Math, khususnya pada area fungsional yang kritis seperti fitur permainan dan perhitungan matematika. Penelitian ini juga bertujuan untuk menganalisis perbedaan karakteristik *grey box testing* dibandingkan dengan metode pengujian lainnya seperti *black box testing* dan *white box testing*. [4] serta mengevaluasi dampak penerapan *grey box testing* terhadap peningkatan kualitas aplikasi F1Math, yang ditinjau dari efisiensi penggunaan CPU, penggunaan memori, kecepatan start-up, serta pengurangan jumlah bug yang signifikan pada versi terbaru aplikasi.

Terdapat beberapa penelitian terdahulu yang menjadi acuan dalam penelitian ini. Berihun, Dongmo, dan Van der Poll [4] dalam penelitiannya yang berjudul "*The Applicability of Automated Testing Frameworks for Mobile Application Testing: A Systematic Literature Review*" membahas pentingnya kualitas aplikasi seluler dan peran pengujian otomatis dalam memastikan aplikasi diterima oleh pengguna melalui tinjauan literatur terkait kerangka kerja pengujian otomatis. Bansal [5] dalam "*A Comparative Study of Software Testing Techniques*" mengkaji keunggulan dan kelemahan *Grey Box Testing* dibandingkan dengan *Black Box Testing* dan *White Box Testing*. Selanjutnya, Yang, Prasad, dan Xie [6] dalam "*A Grey-Box Approach for Automated GUI-Model Generation of Mobile Applications*" membahas pengujian otomatis pada aplikasi seluler dengan pendekatan *grey-box* melalui analisis statis yang mengekstraksi peristiwa GUI dan crawling dinamis yang merekayasa balik model aplikasi. Penelitian Arlt et al. [7] yang berjudul "*Grey-box GUI Testing: Efficient Generation of Event Sequences*" mengembangkan metode pengujian GUI dengan pendekatan *grey-box* untuk

menghasilkan urutan peristiwa yang efisien. Terakhir, Fioraldi, D'Elia, dan Coppa [8] dalam "*WEIZZ: Automatic Grey-Box Fuzzing for Structured Binary Formats*" membahas teknik fuzzing otomatis berbasis grey-box untuk format biner terstruktur dengan mengidentifikasi ketergantungan antara byte input dan instruksi perbandingan guna menghasilkan mutasi input yang lebih terarah.

2. METODE PENELITIAN

Penelitian ini melakukan *testing grey box* terhadap aplikasi F1Math. Aplikasi di *testing* menggunakan Google Firebase TestLab. Gambar 1 menyajikan tahapan yang dilakukan dalam penelitian ini, mulai dari studi literatur, identifikasi masalah, perancangan solusi, hingga pelaksanaan pengujian menggunakan metode Grey Box Testing secara terstruktur.



Gambar 1 Tahapan Penelitian

2.1. Tahapan Penelitian

2.1.1. Studi Literatur

Tahap ini dilakukan untuk memperoleh informasi yang terkait topik penelitian “Implementasi Teknik *grey box testing* dalam *Testing* Aplikasi F1Math” dalam Jurnal , seperti “*Black-box Testing of Grey Box Behavior*”[3], “*grey Box GUI Testing: Efficient Generation of Event Sequences*”[6]

2.1.2. Identifikasi dan perumusan masalah

Pada Tahap menyusun rumusan masalah yang menjadi fokus permasalahan yang akan di kaji dalam penelitian, yaitu bagaimana *Testing* Aplikasi F1Math menggunakan metode *grey box testing*.

2.1.3. Perancangan dan penyusunan Solusi

Pada tahap ini dilakukan perancangan sistem untuk solusi dari masalah yang telah di rumuskan dengan menggunakan metode *grey box testing*.

2.1.4. Initial Communication (Analisis kebutuhan)

Tahap ini perlu menyebarkan kuesioner kepada 30 responden untuk menganalisis kebutuhan dari sistem informasi yang digunakan sebagai acuan dalam pembuatan aplikasi F1Math.

2.1.5. Pahami Persyaratan dan Desain:

Pertama, pahami persyaratan perangkat lunak dan desain sistem secara menyeluruh. Ini membantu developer memahami bagaimana perangkat lunak harus berperilaku dan bagaimana komponen-komponen berinteraksi satu sama lain.

2.1.6. Analisis Kode Sumber:

Dapatkan akses ke kode sumber, jika memungkinkan. Lihatlah struktur kode, modul-modul penting, dan aliran data. Ini akan membantu developer memahami bagaimana perangkat lunak diimplementasikan.

2.1.7. Identifikasi Boundary Conditions:

Pada tahap ini peneliti harus menemukan titik-titik masuk yang penting dan perubahan status yang memengaruhi aliran logika dalam aplikasi. Identifikasi batasan (boundary conditions) pada input, output, dan proses internal.

2.1.8. Memilih teknik *Testing* Regresi

Tahap ini melibatkan pemilihan teknik *Testing* Regresi untuk memastikan bahwa perubahan kode tidak mengganggu fungsionalitas yang sudah ada. Teknik ini dipilih berdasarkan analisis skenario kritis dan disesuaikan dengan penggunaan tools otomatisasi, sehingga dapat meminimalisir bug baru setelah update.

2.1.9. Buat Test case

Tahap ini melibatkan pembuatan Test case yang menyeluruh untuk menguji setiap fitur aplikasi. Test case disusun berdasarkan skenario *Testing* yang telah dirancang, mencakup Test case positif dan negatif. Penyusunan Test case yang terstruktur bertujuan untuk mendeteksi bug secara dini dan memastikan kualitas aplikasi tetap terjaga setelah setiap perubahan kode.

2.1.10. Gunakan Tools:

Tahap ini melibatkan penggunaan *Firebase Testlab* untuk melakukan *Testing* otomatis pada berbagai perangkat, baik fisik maupun virtual. Dengan tools ini, proses *Testing* dapat dilakukan secara menyeluruh dan efisien, memungkinkan deteksi bug lebih cepat dan memastikan aplikasi berfungsi optimal di berbagai kondisi.

2.1.11. Lakukan *Testing*

Tahap ini melibatkan pelaksanaan *Testing* dengan menerapkan Test case yang telah disusun. *Testing* dilakukan secara manual dan otomatis menggunakan *Firebase Testlab* untuk memastikan setiap fitur berjalan sesuai yang diharapkan. Hasil *Testing* dicatat dan dianalisis untuk mendeteksi bug serta masalah performa, sehingga perbaikan dapat segera dilakukan sebelum aplikasi dirilis.

2.1.12. Lakukan *Testing* Regresi

Pastikan bahwa perubahan yang dilakukan tidak mempengaruhi fungsi yang telah ada sebelumnya dengan melakukan *Testing* Regresi. [12], [13]

2.1.13. Dokumentasikan

Tahap ini mencakup pencatatan seluruh proses dan hasil *Testing*, termasuk Test case, log, screenshot, serta bug yang ditemukan dan perbaikannya. Dokumentasi yang lengkap dan terstruktur sangat penting untuk referensi evaluasi, audit, dan pengembangan aplikasi di masa mendatang.

2.2. Pengumpulan Data

2.2.1. Survei dan Kuesioner

Survei dan Kuesioner dilakukan dengan cara mengumpulkan data hasil Kuisisioner yang dibagikan kepada banyak orang. Dengan menggunakan metode ini penulis dapat mendapatkan data yang dapat digunakan dalam penelitian ini untuk menerapkan nya dalam aplikasi F1Math. Dalam penelitian ini, data yang dikumpulkan dari 30 sampel akan dianalisis dan digunakan sebagai acuan untuk menentukan informasi yang dibutuhkan dalam perancangan sistem. Penelitian ini menggunakan teknik *grey Box* untuk menarik kesimpulan, yang digunakan untuk mendeteksi apakah terdapat error pada aplikasi. Peneliti akan melakukan teknik *grey Box* dengan melakukan tiga kali *Testing* pada aplikasi F1Math.

2.2.2. Penelitian Pustaka

Teknik mengumpulkan data ini digunakan untuk mencari dan mengumpulkan informasi dari literatur terkait seperti Jurnal, artikel atau sumber Informasi yang relevan.

2. 3. Metode *Testing*

Testing dilakukan menggunakan *Firebase Testlab* dan *grey box testing* dengan metode Regresi untuk menemukan bug dengan lebih efisien dalam pengetesan Aplikasi F1Math dan langsung memperbaikinya.

3. HASIL DAN PEMBAHASAN

Bab ini membahas hasil yang diperoleh dari proses pengujian serta analisis terhadap temuan yang didapatkan selama pelaksanaan penelitian.

3.1. Deskripsi Data

Hasil penelitian dalam Tabel 1 menyajikan perbandingan kinerja antara versi lama dan versi baru dari aplikasi yang diuji. Beberapa aspek yang dibandingkan meliputi durasi eksekusi, jumlah error, penggunaan CPU dan memori, frame rate, serta waktu respons. Selain itu, disajikan pula hasil analisis regresi linear yang menggambarkan hubungan antara penggunaan CPU dan waktu start-up.[9][10] Perubahan signifikan ditunjukkan pada efisiensi pemrosesan, pengurangan error, peningkatan performa visual, serta peralihan mode jaringan dari server-based menjadi peer-to-peer yang meningkatkan keandalan dan mengurangi penggunaan bandwidth. Secara keseluruhan, Tabel 1 menunjukkan bahwa versi baru aplikasi memberikan peningkatan kinerja yang lebih baik dan pengalaman pengguna yang lebih optimal.

Tabel 1. Tabel Perbandingan Kinerja Aplikasi: Versi Lama vs Versi Baru

Kategori	Versi Lama	Versi Baru	Perubahan / Catatan
Durasi Eksekusi	307 detik (5:07)	100 detik (1:40)	Pengurangan 67% waktu
Jumlah Masalah / Error	2 masalah	0 masalah	Tidak ada error di versi baru
Penggunaan CPU			
- Minimum	22% (Test 9)	9% (Test 21)	13% – Efisiensi meningkat
- Maksimum	35% (Test 35)	28% (Test 5 & 24)	7% – Stabilitas CPU lebih baik
- Rata-rata	28.5%	17.5%	11% – Optimisasi menyeluruh
Penggunaan Memori (KiB)			
- Minimum	190.000 KiB	172.000 KiB	9,47% – Efisiensi meningkat
- Maksimum	225.000 KiB	230.000 KiB	2,22% – Peningkatan beban

			maksimum
- Rata-rata	202.000 KiB	198.000 KiB	1,98% – Lebih hemat memori
Frame Rate	50 FPS	60 FPS	20% – Lebih mulus
Waktu Respons (Start-up)	318.2 ms (rata-rata)	222.3 ms (rata-rata)	Lebih cepat dan responsif
Analisis Regresi Linear			Perubahan / Catatan
- Persamaan Regresi	$Y = -11.44 + 1156.52X$	$Y = 3.42 + 1251.85X$	Intercept versi baru lebih realistis
- R^2	0.15	0.19	Hubungan CPU-startup sedikit lebih kuat
- Slope	1156.52 ms/%CPU	1251.85 ms/%CPU	Sensitivitas CPU di versi baru
Outlier & Anomali	Test 35 (CPU tinggi, waktu tinggi)	Test 5 & 24 (CPU tinggi, waktu rendah); Test 21 (CPU rendah, waktu tinggi)	Data versi baru lebih bervariasi, namun konsisten
Network Multiplayer	Versi Lama	Versi Baru	Perubahan / Catatan
- Mode	Online (Server-based)	Peer-to-Peer	Versi Baru Tidak butuh server pusat
- Kinerja	Tergantung koneksi internet	Langsung antar perangkat	Latency berkurang
- Ketersediaan	Terganggu jika server down	Tetap bisa berjalan offline	Lebih andal
- Keamanan	Data di server pusat	Data langsung antar perangkat	Privasi meningkat
- Penggunaan Bandwidth	Tinggi	Lebih rendah	Hemat bandwidth

- Pengalaman Pengguna	Bisa lag	Lebih stabil & cepat	Lebih baik
-----------------------	----------	----------------------	------------

3.2. Pembahasan

3.2.1. Durasi Eksekusi

Versi baru menunjukkan efisiensi luar biasa dengan durasi eksekusi hanya **100 detik**, turun drastis dari **307 detik** pada versi lama. Penurunan sebesar **67%** ini mencerminkan peningkatan performa yang signifikan, baik dari sisi optimalisasi kode maupun manajemen proses internal aplikasi.

3.2.2. Jumlah Masalah / Error

Versi lama masih mengandung **2 masalah/error**, sedangkan versi baru berhasil menghilangkan seluruh error yang terdeteksi dalam *Testing*. Ini menunjukkan efektivitas *Testing* regresi dan *grey box testing* dalam mengidentifikasi dan memperbaiki bug.[11][12]

3.2.3. Penggunaan CPU

Minimum penggunaan CPU turun dari 22% ke 9%, menunjukkan peningkatan efisiensi dalam idle state atau skenario ringan. Maksimum penggunaan CPU menurun dari 35% ke 28%, menandakan stabilitas yang lebih baik dalam kondisi beban tinggi. Rata-rata pemakaian CPU juga turun signifikan dari 28.5% ke 17.5%, menunjukkan optimisasi menyeluruh yang berdampak positif pada konsumsi daya dan kinerja perangkat.

3.2.4 Penggunaan Memori

Terdapat penurunan penggunaan memori minimum dan rata-rata, menandakan efisiensi memori yang lebih baik. Namun, memori maksimum sedikit meningkat (2.22%), kemungkinan karena adanya fitur tambahan atau peningkatan proses latar belakang yang lebih kompleks.

3.2.5. Frame Rate

Peningkatan dari 50 FPS ke 60 FPS menghasilkan pengalaman visual yang lebih mulus dan responsif, sangat penting dalam game edukatif yang berbasis interaksi real-time seperti F1Math.

3.2.6. Waktu Respons (Start-up)

Versi baru memperlihatkan waktu start-up rata-rata sebesar 222.3 ms, jauh lebih cepat dibandingkan versi lama (318.2 ms), yang berarti pengguna dapat langsung mulai bermain tanpa delay signifikan.

3.2.7. Analisis Regresi Linear

Persamaan regresi pada versi baru menunjukkan intercept yang lebih realistis (3.42 vs -11.44), serta slope yang meningkat, menunjukkan sensitivitas waktu terhadap penggunaan CPU yang sedikit lebih tinggi, namun masih dalam batas wajar. Nilai R^2 meningkat dari 0.15 ke 0.19, mengindikasikan hubungan yang sedikit lebih kuat antara penggunaan CPU dan waktu startup.

3.2.8. Outlier & Anomali

Versi lama menunjukkan anomali pada Test 35, dengan CPU dan waktu yang tinggi. Sementara itu, versi baru menunjukkan outlier yang lebih seimbang (contohnya Test 5, 24, dan 21), menandakan variasi data yang lebih stabil dan konsisten meskipun kompleks.

3.2.9. Network Multiplayer

Perubahan dari sistem server-based ke peer-to-peer sangat signifikan. Latency menurun dan koneksi menjadi lebih stabil. Aplikasi tetap dapat berjalan saat offline, meningkatkan ketersediaan. Privasi dan efisiensi bandwidth membaik karena data dikirim langsung antar perangkat tanpa perantara server. Pengalaman pengguna juga meningkat berkat gameplay yang lebih lancar dan andal.

4. KESIMPULAN

Berdasarkan hasil penelitian dan pembahasan yang telah dilakukan, dapat disimpulkan bahwa:

1. Efektivitas *grey box testing* dalam *Testing* Aplikasi F1Math: Implementasi teknik *grey box testing* terbukti efektif dalam mengidentifikasi cacat pada aplikasi F1Math, terutama pada area kritis seperti fitur permainan dan perhitungan matematika. Kombinasi pengetahuan struktur internal aplikasi dan fokus pada perilaku eksternal memungkinkan *Testing* yang lebih efisien, mempercepat proses perbaikan bug, serta meningkatkan stabilitas aplikasi. Hal ini tercermin dari penurunan rata-rata penggunaan CPU (dari 28.5% ke 17.5%) dan peningkatan kecepatan *start-up* (dari 318.2 ms ke 222.3 ms) pada versi baru setelah perbaikan.
2. Perbedaan Karakteristik *grey box testing* dengan Teknik Lain: *grey box testing* menggabungkan keunggulan *Black Box Testing* (fokus pada fungsionalitas tanpa akses kode) dan *White Box Testing* (analisis jalur eksekusi kode).[13] Pada aplikasi F1Math, pendekatan ini memungkinkan *Testing* integrasi antarmuka pengguna Misalnya,
3. Dampak pada Kualitas Aplikasi: Versi baru F1Math menunjukkan peningkatan signifikan dalam efisiensi (CPU dan memori lebih rendah) serta responsivitas (waktu *start-up* lebih cepat). Analisis Regresi Linear juga mengonfirmasi bahwa optimisasi kode dan penerapan *grey box testing* berkontribusi pada pengurangan *bug* kritis (dari 2 masalah pada versi lama menjadi 0 pada versi baru).

DAFTAR PUSTAKA

- [1] Z. A. K. Hamza and M. Hammad, "Testing approaches for Web and mobile applications: An overview," *Int. J. Comput. Digit. Syst.*, vol. 90, no. 4, pp. 657–664, Jul. 2020, doi: 10.12785/ijcds/090413.
- [2] I. R. Dhaifullah, M. Muttanifudin, A. A. Salsabila, and M. A. Yakin, "Survei Teknik Pengujian Software," 2022.

- [3] B. Tyler and N. Soundarajan, "Black-box Testing of Grey-box Behavior."
- [4] N. G. Berihun, C. Dongmo, and J. A. Van der Poll, "The Applicability of Automated Testing Frameworks for Mobile Application Testing: A Systematic Literature Review," *Computers*, vol. 12, no. 5, May 2023, doi: 10.3390/computers12050097.
- [5] A. Bansal, "International Journal of Computer Science and Mobile Computing A Comparative Study of Software Testing Techniques," 2014. [Online]. Available: www.ijcsmc.com
- [6] W. Yang, R. Prasad, and T. Xie, "LNCS 7793 - A Grey-Box Approach for Automated GUI-Model Generation of Mobile Applications."
- [7] S. Arlt, I. Banerjee, C. Bertolini, A. M. Memon, and M. Schäf, "Grey-box GUI Testing: Efficient Generation of Event Sequences," May 2012, [Online]. Available: <http://arxiv.org/abs/1205.4928>
- [8] A. Fioraldi, D. C. D'Elia, and E. Coppa, "WEIZZ: Automatic grey-box fuzzing for structured binary formats," in *ISSTA 2020 - Proceedings of the 29th ACM SIGSOFT International Symposium on Software Testing and Analysis*, Association for Computing Machinery, Inc, Jul. 2020, pp. 1–13. doi: 10.1145/3395363.3397372.
- [9] Margaret Teacher Banjarnahor and L. S. Istiyowati, "Smoke Automation and Regression Testing on a peer-to-peer lending Website with the Data-DrivenTesting Method," *J. RESTI (Rekayasa Sist. dan Teknol. Informasi)*, vol. 6, no. 4, pp. 684–691, Aug. 2022, doi: 10.29207/resti.v6i4.4220.
- [10] T. Nurhayati, D. A. Alfiani, and D. Setiani, "The Effect of Crossword Puzzle Application on The Students' Learning Motivation in Science Learning," *Al Ibtida J. Pendidik. Guru MI*, vol. 6, no. 1, p. 124, Jun. 2019, doi: 10.24235/al.ibtida.snj.v6i1.4186.
- [11] S. E. Prasetyo and N. Hassanah, "Analisis Keamanan Website Universitas Internasional Batam Menggunakan Metode ISSAF."
- [12] M. Rizky, A. Agustin, T. Cahya Herdiyani, A. Waluyo, and J. Riyanto, "Pengujian Black Box menggunakan Teknik State Transition Testing pada Aplikasi BMI Calculator Berbasis Android," 2021. [Online]. Available: <http://pijarpemikiran.com/index.php/Scientia>
- [13] S. Acharya and V. Pandya, "Bridge between Black Box and White Box-Gray Box Testing Technique." [Online]. Available: www.ijecse.org

Lampiran

Link Hasil Penelitian

https://drive.google.com/drive/folders/1Yj16UFeakkSXkalaUjky_mYNMHprvIEw?usp=sharing