Analisis Komparasi pada Teknik *Black Box Testing* (Studi Kasus: *Website* Lars)

Salmania Jesamine Putri[1], Divi Galih Prasetyo Putri[1,*], Widhy Hayuhardhika Nugraha Putra[2]
[1]Departemen Elektro dan Informatika, Sekolah Vokasi, Universitas Gadjah Mada;
salmania2001@gmail.com
[2]Universitas Brawijaya;
widhy@ub.ac.id
*Korespondensi: divi.galih@ugm.ac.id;

Abstract* – Currently the use of software has dominated almost all fields of knowledge. It is important for developers to ensure the quality of a software so that it is suitable for use by the general public. The quality of the software, one of which can be determined from the output produced, whether it is in accordance with user needs. Software testing is an important activity in the Software Development Life Cycle (SDLC) phase to ensure quality software. There are various methods that may be used in software testing, one of which is black box testing which tests the functionality of a system and does not require the tester to understand program code. In order to achieve optimal testing results, it is necessary to determine the most suitable design of test cases for a software. Equivalence Class Partitioning (ECP), Boundary Value Analysis (BVA), and Decision Table (DT) are commonly used black box testing techniques. This study aims to compare these three technique to determine which technique is most effectively applied to a software. The sample used for testing is the Lars website, an application developed to assist in the hospital accreditation process. The results of testing each technique are measured using standard testing metrics to see which technique is the most optimal. The results obtained by this study are that the ECP outperforms the other technique in catching failures, measured from the calculation of the failed test case metric with a percentage of 51.8% compared to the BVA technique with a result of 33.3% and 46% for DT.

***Keywords* – Black Box Testing, Equivalence Class Partitioning, Boundary Value Analysis, Decision Table, Website Lars.**

Intisari – Saat ini penggunaan perangkat lunak sudah mendominasi hampir seluruh bidang ilmu pengetahuan. Merupakan hal yang penting bagi pengembang untuk memastikan kualitas suatu perangkat lunak sehingga layak digunakan oleh khalayak umum. Kualitas perangkat lunak salah satunya dapat ditentukan dari output yang dihasilkan apakah sudah sesuai dengan kebutuhan pengguna. Pengujian perangkat lunak merupakan salah satu aktivitas yang penting dalam fase Software Development Life Cycle (SDLC) untuk memastikan perangkat lunak yang berkualitas. Terdapat berbagai metode yang mungkin digunakan dalam pengujian perangkat lunak, satu diantaranya adalah black box testing yang menguji fungsionalitas suatu sistem dan tidak mengharuskan penguji untuk memahami kode program. Dalam rangka mencapai hasil pengujian yang optimal, perlu menentukan perancangan kasus uji yang paling tepat digunakan pada suatu perangkat lunak. Equivalence Class Partitioning (ECP), Boundary Value Analysis (BVA), dan Decision Table (DT) merupakan teknik pengujian pada black box yang umum digunakan. Penelitian ini bertujuan untuk membandingkan tiga teknik tersebut, sehingga dapat menentukan teknik mana yang paling efektif diterapkan pada suatu perangkat lunak. Sampel yang digunakan untuk pengujian adalah website Lars yang merupakan aplikasi untuk membantu proses akreditasi rumah sakit. Hasil dari pengujian masing-masing teknik diukur menggunakan standard testing metrics untuk melihat teknik mana yang paling optimal. Hasil yang didapatkan penelitian ini adalah teknik ECP lebih unggul dalam menangkap kegagalan, diukur dari perhitungan metrik test case failed dengan persentase 51.8% dibandingkan teknik BVA dengan hasil 33.3% dan DT 46%.

Kata kunci – *Black Box Testing, Equivalence Class Partitioning, Boundary Value Analysis, Decision Table, Website Lars.*

I. PENDAHULUAN

Pengujian perangkat lunak (*software testing*) adalah proses menganalisis perangkat lunak untuk menemukan perbedaan antara kebutuhan perangkat lunak dengan kondisi yang sebenarnya atau proses evaluasi dari suatu perangkat lunak [1]. Pada [2] mengemukakan peran *testing* dalam proses pengembangan perangkat lunak atau *Software Development Life Cycle* (SDLC) sangat penting dan merupakan proses yang cukup menantang. *Software testing* memiliki berbagai macam pendekatan yang bisa digunakan, diantaranya *white box* dan *black box testing*. *White box testing* berfokus pada struktur internal dan implementasi kode program, sedangkan *black box testing* tidak memperhatikan struktur internal namun fokus pada *input* dan *output* program [3]. [4] *Black box testing* adalah pengujian pada fungsionalitas perangkat lunak, biasanya saat melakukan pengujian dengan metode *black box* tester akan berinteraksi dengan antarmuka sistem dengan memberikan *input* dan memeriksa *output* tanpa mengetahui bagaimana cara kerja dan di mana input dikerjakan.

Terdapat berbagai teknik pada *black box testing* yang dapat diterapkan, namun menurut [4] sekalipun memilih salah satu teknik pengujian dari *black box testing*, hasil pengujian dapat bervariasi karena adanya perbedaan teknik dalam mengidentifikasi kebutuhan sistem dan menentukan perancangan kasus uji. Untuk mencegah kemungkinan kesalahan dalam perangkat lunak yang tidak terdeteksi dengan baik, penting untuk

menerapkan teknik *black box testing* yang tepat. Dengan melakukan hal ini, hasil dari pengujian dapat dipercaya dan mengurangi risiko terjadinya kerugian yang signifikan, seperti pemborosan waktu, tenaga, dan biaya. Karena kompleksitas dari setiap aplikasi yang akan diuji sudah pasti berbeda, maka diperlukan analisis komparasi dari berbagai teknik pengujian pada *black box*.

PT Andhara Prima Kreatif (APIK) merupakan perusahaan industri kreatif berbasis teknologi yang menyediakan berbagai layanan kebutuhan produk digital dan penyediaan talenta. PT APIK telah mengembangkan berbagai macam aplikasi yang sudah dapat digunakan secara umum, satu diantaranya yaitu aplikasi Lars. Aplikasi Lars dikembangkan untuk memberikan survei akreditasi rumah sakit berdasarkan peraturan perundang-undangan yang berlaku. Dalam pengembangan aplikasi Lars, proses pengujian diperlukan untuk mengukur kualitas dan kelayakan aplikasi. Oleh karena itu, PT APIK memerlukan analisis komparasi untuk menentukan teknik yang paling efektif digunakan pada proses pengujian aplikasi yang dikembangkan. Analisis komparasi teknik *black box testing* dilakukan dengan menghitung beberapa *software testing metrics.*

II. DASAR TEORI

A. *Black Box Testing*

Black box testing atau biasa disebut *behavioural testing* adalah metode pengujian perangkat lunak berdasarkan perilakunya, bukan menguji struktur atau kode internal dari suatu program dan berfokus pada *input* dan *output* yang telah dicapai [5]. Menurut [6] *black box testing* merupakan pengujian yang dilakukan berdasarkan spesifikasi kebutuhan dan tidak memerlukan pemeriksaan kode pada aplikasi. Proses ini murni dilakukan berdasarkan sudut pandang pengguna. *Black box testing* berperan penting dalam proses pengujian perangkat lunak untuk membantu validasi fungsionalitas dari sistem. Kelebihan dari metode ini adalah penguji tidak perlu memiliki pengetahuan khusus mengenai bahasa pemrograman dan juga pengetahuan tentang implementasi.

B. *Equivalence Class Partitioning*

Equivalence Class Partitioning (ECP) adalah teknik pengujian perangkat lunak yang membagi input nilai menjadi nilai valid dan tidak valid dan memilih representasi dari setiap data uji [7]. Menurut [8] tujuan dari *equivalence partitioning* adalah untuk membagi domain input sistem pengujian menjadi kelas-kelas atau kelompok. Semua input di kelas yang sama memiliki pengaruh serupa pada sistem yang diuji dan mewakili kondisi tertentu. Tabel 1 menggambarkan aturan nilai pengujian pada teknik ECP.

Tabel 1. Aturan Input ECP

Rule	Keterangan
Min	Kondisi ketika nilai input berada di bawah/kurang dari *rule* yang ada
Middle	Kondisi ketika nilai input sesuai dengan *rule* yang ada
Max	Kondisi ketika nilai input berada di atas/lebih dari *rule* yang ada

C. *Boundary Value Analysis*

Boundary Value Analysis (BVA) adalah teknik pada *black box testing* yang menguji batas atas dan batas bawah nilai yang dimasukkan ke dalam sistem [9]. Pada intinya, teknik ini memilih data uji yang dekat dengan batas domain data untuk menghasilkan *input* uji di dekat batas dan menemukan *failure* yang disebabkan kesalahan implementasi pada nilai batas. Dalam proses pengembangan program, *developer* cenderung mengabaikan kondisi batas, sehingga *defect* cenderung terkonsentrasi di dekat batas. Oleh karena itu, tes data yang dipilih berada dalam batas atau dekat dengan batas. Dalam pengertian itu, teknik BVA adalah perluasan dan penyempurnaan dari teknik *Equivalence Class.* Tabel 2 menggambarkan aturan nilai pengujian pada teknik BVA.

Tabel 2. Aturan Input BVA

Rule	**Keterangan**
Min(-)	Tepat di bawah batas bawah
Min(+)	Tepat di atas batas bawah
Middle	Berada pada *range* pada aturan pengujian
Max(-)	Tepat di bawah batas atas
Max(+)	Tepat di atas batas atas

D. *Decision Table*

Decision Table (DT) merupakan teknik pada *black box testing* yang digunakan untuk menguji perangkat lunak pada input yang berbeda kombinasi dengan menggabungkan input dan output dan merangkumnya ke dalam tabel. *Decision Table* juga sering disebut sebagai tabel *cause-and-effect* karena tabel pada *Decision Table* berisikan beberapa sebab dan akibat [10]. Pada teknik DT, digunakan sejumlah kombinasi input untuk dapat menentukan output yang harus muncul sesuai dengan kebutuhan sistem [11]. Tabel 3 menggambarkan aturan nilai pengujian pada teknik ECP.

Tabel 3. Aturan Input DT

Kondisi	**R1**	**R2**	**R3**	**R4**
Username (T/F)	F	T	F	T
Password (T/F)	F	F	T	T
Output (E/S)	E	E	E	S

Keterangan:
$T = True$ $E = Error$
$F = False$ $S = Success$

$R = Rule\ Pengujian$
$R1 = Rule\ Pengujian\ 1$
$R2 = Rule\ Pengujian\ 2, dst.\ \ Total\ number\ of\ combinations = 2^n$

E. Selenium IDE

Selenium adalah *software testing framework* yang digunakan untuk mengotomatisasi aplikasi *web* yang banyak digunakan untuk pengujian fungsional, pengujian regresi, dan pengujian performa. Selenium IDE merupakan salah satu *tools* Selenium berbasis *web* yang pada awalnya diimplementasikan sebagai *add-on* pada *Firefox*, namun saat ini sudah dapat digunakan di setiap *browser web*. Beberapa kelebihan Selenium IDE, yaitu memberikan dukungan *multi-browser*, tidak memerlukan pemahaman mendalam terhadap bahasa pemrograman, dapat mengatur *breakpoint* dan *debug*, dapat melakukan *record-and-playback*, serta penggunaanya yang mudah [12].

F. *Software Testing Metrics*

Software Testing Metrics berperan penting dalam proses pengujian dan evaluasi produk perangkat lunak dengan menyediakan standar dan pengukuran yang objektif. Dengan menggunakan *software testing metrics*, pengembang juga dapat mengukur dan memantau sejauh mana perkembangan perangkat lunak [13] [14] [15]. Dalam penelitian ini *metrics* yang digunakan dapat dilihat di persamaan (1), (2), (3) dan (4).

$$Test\ Case\ Executed\ \% = \left(\frac{Test\ Case\ Executed}{Test\ Case\ Written}\right) \times\ 100\% \quad (1)$$

Keterangan:
$Test\ Case\ Executed$ = total *test case* yang dapat dieksekusi
$Test\ Case\ Written$ = total *test case* yang dibuat

$$Test\ Case\ Passed\ \% = \left(\frac{Test\ Case\ Passed}{Total\ Test\ Case}\right) \times\ 100\% \quad (2)$$

Keterangan:
$Test\ Case\ Passed$ = $total\ test\ case\ yang\ berhasil$
$Total\ Test\ Case$ = $total\ test\ case\ yang\ diuji$

$$Test\ Case\ Failed\ \% = \left(\frac{Test\ Case\ Failed}{Total\ Test\ Case}\right) \times\ 100\% \quad (3)$$

Keterangan:
$Test\ Case\ Failed = total\ test\ case\ yang\ gagal$
$Total\ Test\ Case = total\ test\ case\ yang\ diuji$

$$Total\ Test\ Case = jumlah\ test\ case\ yang\ dibuat \quad (4)$$

Keterangan:
Total test case dapat memperkirakan lamanya waktu pengujian. Semakin banyak test case, maka waktu yang dibutuhkan untuk pengujian semakin lama.

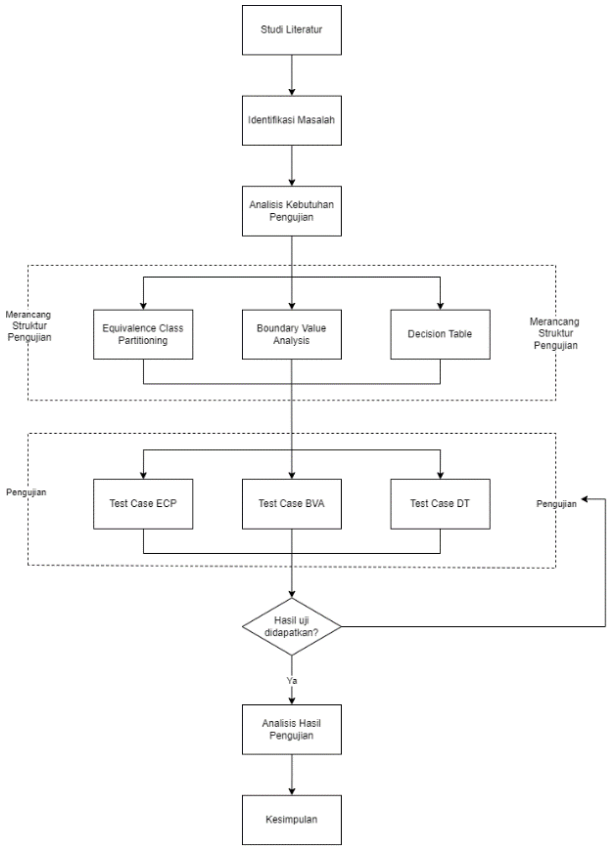
III. METODOLOGI

A. Objek Penelitian

Objek pada penelitian ini adalah *website* Lars, yaitu software yang dikembangkan oleh PT Andhara Prima Kreatif (PT APIK) yang digunakan untuk membantu proses akreditasi rumah sakit.

B. Tahapan Penelitian

Tahapan penelitian diilustrasikan pada Gambar 1.



Gambar 1. Alur Penelitian

C. Rancangan Pengujian

Pengujian *website* Lars dilakukan secara automasi dengan menggunakan bantuan *tools* Selenium IDE. Untuk melakukan pengujian, diperlukan *test case* dan data yang akan diinputkan pada pengujian.

Contoh pada Tabel 4 dan 5 adalah data input jika suatu elemen fitur memiliki rentang nilai valid dari angka 1 sampai 9999. Tabel 4 menunjukkan data input yang akan digunakan dalam metode ECP.

Tabel 4. Contoh Data Input ECP

Kondisi	Contoh Data
Invalid	-1
Valid	1000
Invalid	100000

Keterangan:
Nilai invalid = di bawah rentang nilai terendah
Nilai valid = di antara rentang nilai yang sudah ditentukan
Nilai invalid = di atas rentang nilai tertinggi

Tabel 5 menunjukkan data input yang akan digunakan dalam metode BVA.

Tabel 5. Contoh Data Input BVA

Kondisi	Contoh Data
Batas bawah	0
Batas atas	2
Nilai tengah	2000
Batas bawah 2	9998
Batas atas 2	10000

Keterangan:
Nilai batas bawah= nilai tepat di bawah rentang terendah
Nilai batas atas = nilai tepat di atas rentang terendah
Nilai tengah = nilai di antara rentang yang sudah ditentukan
Nilai batas bawah 2= nilai tepat di bawah rentang tertinggi
Nilai batas atas 2 = nilai tepat di atas rentang tertinggi

Tabel 6 menunjukkan data input yang akan digunakan dalam metode DT.

Tabel 6. Contoh Data Input DT

Kondisi	Contoh Data
True	55167
	77777
False	11234567890
	1234!@
	F1235
	FFFFF
	1

Tabel 6 adalah contoh data untuk elemen fitur kode pos yang terdiri dari 5 angka.

IV. HASIL DAN PEMBAHASAN

A. Hasil Pengujian ECP

Tabel 7 menunjukkan hasil pengujian metode ECP.

Tabel 7. Hasil Pengujian ECP

Elemen Fitur	*Pass*	*Fail*	*Total Test Case*
Nama Rumah Sakit	2	1	3
Nomor Telepon	2	1	3
Email	3	2	5
Password	2	1	3
Konfirmasi *Password*	2	1	3
Email (*login*)	2	1	3
Password (l*ogin*)	2	1	3
Email (*forget password)*	2	1	3
Induk RS	1	2	3
Kapasitas	1	2	3
NIB	1	2	3
Izin Operasional	1	2	3
Alamat Lengkap	1	2	3
Kode Pos	1	2	3
Nama Direktur	1	2	3
Email Direktur	1	2	3
Nomor Direktur	1	2	3
Divisi	1	2	3
Total	**27**	**29**	**56**

B. Hasil Pengujian BVA

Tabel 8 menunjukkan hasil pengujian metode BVA.

Tabel 8. Hasil Pengujian BVA

Elemen Fitur	*Pass*	*Fail*	*Total Test Case*
Nama Rumah Sakit	4	1	5
Nomor Telepon	4	1	5
Email	5	2	7
Password	4	1	5
Konfirmasi *Password*	4	1	5
Email (*login*)	5	1	6
Password (*login*)	4	1	5
Email (*forget password*)	4	1	5
Induk RS	3	2	5
Kapasitas	3	2	5
NIB	3	2	5
Izin Operasional	3	2	5
Alamat Lengkap	3	2	5
Kode Pos	1	4	5
Nama Direktur	3	2	5
Email Direktur	3	2	5
Nomor Direktur	3	2	5
Divisi	3	2	5
Total	**62**	**31**	**93**

C. Hasil Pengujian DT

Tabel 9 menunjukkan hasil pengujian metode DT.

Tabel 9. Hasil Pengujian DT

Teknik	*Pass*	*Fail*	*Total Test Case*
ECP	27	29	56
BVA	62	31	93
DT	39	45	75

D. Perbandingan Hasil Pengujian

Tabel 10 menunjukkan perbandingan hasil pengujian.

Tabel 10. Perbandingan Hasil

Fitur	*Pass*	*Fail*	*Total Test Case*
Registrasi	31	1	32
Login	4	0	4
Forget Password	1	1	2
Profil	3	33	36

E. Perhitungan *Standard Testing Metrics*

1. *Test Case Executed*

$$Test\ Case\ Executed\ ECP\ \% = \left(\frac{56}{56}\right) \times 100\% = 100\%$$

$$Test\ Case\ Executed\ BVA\ \% = \left(\frac{93}{93}\right) \times 100\% = 100\%$$

$$Test\ Case\ Executed\ DT\ \% = \left(\frac{74}{74}\right) \times 100\% = 100\%$$

2. *Test Case Passed*

$$Test\ Case\ Passed\ ECP\ \% = \left(\frac{27}{56}\right) \times 100\% = 48.2\%$$

$$Test\ Case\ Passed\ BVA\ \% = \left(\frac{62}{93}\right) \times 100\% = 66.7\%$$

$$Test\ Case\ Passed\ DT\ \% = \left(\frac{40}{74}\right) \times 100\% = 54\%$$

3. *Test Case Failed*

$$Test\ Case\ Failed\ ECP\ \% = \left(\frac{29}{56}\right) \times 100\% = 51.8\%$$

$$Test\ Case\ Failed\ BVA\ \% = \left(\frac{62}{93}\right) \times 100\% = 33.3\%$$

$$Test\ Case\ Failed\ DT\ \% = \left(\frac{34}{74}\right) \times 100\% = 46\%$$

4. *Total Test Case*

Tabel 11 menunjukkan hasil perhitungan *total test case*.

Tabel 11. *Total Test Case*

Teknik	*Total Test Case*
ECP	56
BVA	93
DT	74

F. Perbandingan Metriks

Tabel 12 menunjukkan hasil perbandingan metriks.

Tabel 12. Perbandingan Metriks

Metrics	ECP	BVA	DT
Executed	100%	100%	100%
Pass	48.2%	66.7%	54%
Fail	51.8%	33.3%	46%
Total Test Case	*Low*	*High*	*Medium*

Dari hasil yang ditunjukkan pada Tabel 12, teknik ECP mampu menemukan banyak kegagalan, dibuktikan dengan persentase *test case fail* yang tinggi jika dibandingkan dua teknik lain. Hasil ini sesuai dengan hasil penelitian terdahulu yang dilakukan oleh [16] yang menunjukkan teknik ECP mampu menangkap paling banyak kegagalan dengan waktu yang paling singkat dibandingkan teknik BVA, PWT, dan DT.

Berdasarkan pengujian yang telah dilakukan pada *website* Lars, teknik ECP dapat menangkap lebih banyak kegagalan karena memiliki perbandingan tes negatif yang lebih tinggi dibandingkan BVA, sedangkan DT lebih memfokuskan pada logika kombinasi setiap fitur. Pada ECP, jumlah minimal tes untuk setiap fitur adalah tiga dengan perbandingan negatif tes dibanding positif tes adalah 2:1, sedangkan minimal tes pada BVA adalah lima dengan perbandingan tes negatif dibanding tes positif adalah 2:3, sedangkan DT mencoba segala kemungkinan yang dapat terjadi.

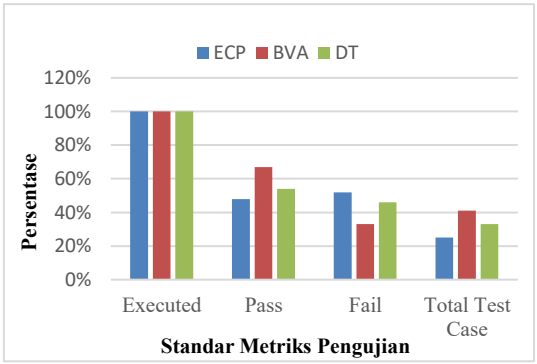
Teknik ECP menjadi teknik yang paling efektif untuk diterapkan pada pengujian *website* Lars karena ECP mampu menguji seluruh *test case* yang telah dibuat dan berhasil menangkap lebih banyak kegagalan dengan jumlah kasus pengujian yang paling sedikit. Selain itu, jumlah kasus uji pada teknik ECP paling sedikit dan jenis input data tidak harus spesifik seperti BVA.

Teknik BVA mampu menemukan kesalahan yang lebih spesifik dibandingkan ECP dan DT. Namun berdasarkan perhitungan metriks pada penelitian ini, teknik BVA menjadi teknik yang paling tidak disarankan untuk pengujian *website* Lars karena membutuhkan banyak kasus uji dan tidak mendeteksi banyak kegagalan, serta membutuhkan waktu pengujian yang lebih lama. Data yang digunakan pada teknik BVA juga harus spesifik pada batas-batas nilai yang relevan dengan perubahan, sehingga memerlukan pemahaman dan ketelitian dalam menentukan nilai input.

Teknik DT memiliki jenis perancangan kasus uji yang berbeda dengan ECP dan BVA karena tidak berfokus pada satu elemen dan pengujian karakter, tetapi menguji berbagai kombinasi input dan kondisi yang berbeda. Pengujian ini membantu memahami bagaimana perubahan dalam satu kondisi dapat mempengaruhi hasil pengujian.

G. Grafik

Gambar 2 menunjukkan grafik hasil perhitungan metriks *software testing.*



Gambar 2. Grafik Hasil Perhitungan Metriks

V. SIMPULAN

Hasil pengujian dengan teknik ECP menunjukkan *test case failed* sebesar 51.8%, pada teknik BVA sebesar 33.3%, dan teknik DT 46%. Berdasarkan hasil tersebut, *website* Lars perlu melakukan perbaikan fungsionalitas karena masih banyak ditemukan *bug.*

Teknik yang paling efektif untuk digunakan pada *website* Lars adalah ECP yang memiliki tingkat keberhasilan dalam menemukan *bug* sebesar 51.8% dan jumlah kasus pengujian yang paling sedikit dibandingkan dengan BVA dan DT.

Bagi peneliti selanjutnya, dapat melakukan penelitian dengan mengimplementasikan lebih banyak teknik. Selain itu, dapat melakukan penelitian lebih lanjut mengenai penggabungan teknik ECP dan DT yang dapat diimplementasikan pada aplikasi lain.

VI. REFERENSI

[1] M. E. Khan, "Different Approaches to Black Box Testi Technique for Finding Errors," vol. 2, p. 10, 2011.

[2] R. V. Binder, Testing Object-Oriented Systems: Mode Patterns, and Tools, Addison-Wesley Professional, 1999.

[3] Z. A. Hamza dan M. Hammad, "Web and Mobile Applicatio Testing using Black and White Box approaches," *IEEE,* p. 2019.

[4] K. I. Seo dan E. M. Choi, "Comparison of Five Black-box Testi Methods for Object-Oriented Software," *IEEE,* p. 8, 2006.

[5] U. K. Tiwari dan S. Kumar, "Components Integration-Effe Graph: A Black Box Testing and Test Case Generation Techniq for Component-Based Software," *Springer Link,* p. 15, 2017.

[6] S. Nidhra dan J. Dondeti, "Black Box and White Box Testi Techniques - A Literature Review," *International journal embedded systems and applications,* vol. 2, no. 2, pp. 29-5 2012.

[7] M. Sholeh, I. Gisfas, Cahiman dan M. A. Fauzi, "Black B Testing on ukmbantul.com Page with Boundary Value Analys and Equivalence Partitioning Methods," *IOP Publishing,* p. 2021.

[8] K. Naik dan P. Tripathy, Software Testing and Quality Assuran Theory and Practice, A JOHN WILEY & SONS, INC PUBLICATION, 2008.

[9] I. P. A. Prayudha, R. S. Hartati dan Y. Divayana, "Bounda Value Analysis Testing Techniques on Learning Manageme System Applications," *International Journal of Engineering a Emerging Technology,* vol. 4, p. 4, 2019.

[10] J. Joosten, A. E. Permanasari dan T. B. Adji, "The Use Decision Table for Reducing Complex Rules in Softwa Testing," *IOP Science,* p. 8, 2020.

[11] Meenu dan Navita, "Study and Analysis of Software Testing *IJRITCC,* vol. 3, no. 12, p. 6674–6678, 2015.

[12] 2023. [Online]. Available: https://www.selenium.dev. [Diaks 15 April 2023].

[13] P. M. Jacob dan D. M. Prasanna, "A Comparative Analysis Black Box Testing Strategies," *IEEE,* no. 16, p. 6, 2016.

[14] I. G. S. Aryandana, Pengukuran Performa Metode Softwa Testing Equivalence Class Partitioning dan Boundary Val Analysis, Yogyakarta, 2019.

[15] Hamilton. [Online]. Available: ps://www.guru99.com/software-testing-metrics-complete- orial.html. [Diakses June 2023].

[16] Xu, L. Chen, C. Wang dan O. Rud, "A Comparative Study Black-Box Testing with Open Source Applications," *IEEE,* 6, 2016.