



JURNAL SISTEM INFORMASI DAN TEKNOLOGI (S I N T E K)

Situs Jurnal

<https://www.sintek.stmikku.ac.id/index.php/home>



OPTIMASI CAKUPAN TEST CASE MENGGUNAKAN METODE EQUIVALENCE PARTITIONING BERBASIS GHERKIN DALAM PENGUJIAN MANUAL APLIKASI KOMPAS.ID

Maya Dwi Latifah^{*1}

¹Sistem Informasi, Fakultas Teknik dan Informatika Universitas Bina Sarana Informatika
¹Jl. Kamal Raya No.18, RT.6/RW.3, Cengkareng Timur, Kecamatan Cengkareng,
Kota Jakarta Barat, DKI Jakarta 11730

¹mdwilatifah@gmail.com

ABSTRAK

Kompas.id merupakan *platform* media digital berbasis langganan milik PT Kompas Media Nusantara yang menyediakan konten berita melalui aplikasi *mobile*. Dalam proses pengujiannya, tim *Quality Assurance* Kompas.id menyusun *Test Case* secara reaktif berbasis *bug* menggunakan pendekatan *exploratory* dan *ad hoc testing*. Pendekatan tersebut menghasilkan skenario yang redundan akibat pengulangan *Test Case* berbasis peran pengguna, serta cakupan kondisi input yang belum optimal. Penelitian ini bertujuan untuk mengoptimalkan *Test Case* pada sebelas fitur aplikasi Kompas.id dengan menerapkan metode *Equivalence Partitioning* (EP) berbasis format *Gherkin*. Metode EP digunakan untuk mengelompokkan kondisi input ke dalam kelas ekuivalen *valid* dan tidak *valid* secara sistematis. Evaluasi dilakukan menggunakan tiga indikator, yaitu jumlah *Test Case*, *Completeness Rate*, dan *Redundansi Rate*. Data dikumpulkan melalui observasi langsung terhadap proses pengujian tim QA dan dokumentasi *test case* eksisting berformat *Gherkin* yang disusun menggunakan *Microsoft Excel*, kemudian dianalisis sebagai *baseline* sebelum penerapan EP. Hasil penelitian menunjukkan bahwa total *Test Case* berkurang dari 101 menjadi 48, sehingga menghasilkan reduksi sebesar 52,48%. *Completeness Rate* meningkat dari 83,85% menjadi 100%, dan *Redundansi Rate* turun dari 39,24% menjadi 0,00%. Dengan demikian, metode EP terbukti efektif dalam mengoptimalkan kualitas *Test Case* pada aplikasi Kompas.id, baik dari sisi kelengkapan cakupan maupun efisiensi skenario pengujian.

Kata Kunci: *Equivalence Partitioning, Format Gherkin, Kompas.id, Manual Testing, Test Case*

1. PENDAHULUAN

Sistem informasi berbasis aplikasi saat ini menjadi elemen penting dalam mendukung efisiensi operasional, pengelolaan informasi, dan pelayanan pada berbagai bidang, termasuk bisnis, pendidikan, dan media digital [4]. Aplikasi tidak hanya berfungsi sebagai sarana penyedia layanan, tetapi juga sebagai media interaksi utama antara sistem dan pengguna [17], sehingga dituntut memberikan kinerja yang optimal dan akurat. Untuk memastikan hal

tersebut, diperlukan proses pengujian perangkat lunak (*software testing*) yang memastikan sistem berjalan sesuai kebutuhan pengguna serta bebas dari kesalahan yang dapat mengganggu kinerja sistem [9], [10]. Pengujian yang tidak dilakukan secara sistematis dapat meningkatkan risiko kegagalan sistem, salah satunya berupa *defect leakage* kondisi di mana cacat tidak terdeteksi pada tahap pengujian dan baru ditemukan setelah sistem diimplementasikan [14], [15].

Kompas.id, platform media digital berbasis langganan milik PT Kompas Media Nusantara, merupakan salah satu aplikasi yang aktif dikembangkan dengan siklus *sprint* dan memerlukan kualitas pengujian yang tinggi. Berdasarkan observasi, proses pengujian manual pada aplikasi ini masih menggunakan pendekatan *exploratory testing* dan *ad hoc testing* yang bersifat fleksibel, tidak terstruktur, dan dilakukan tanpa perencanaan terstruktur [10]. Meskipun penyusunan *test case* telah menggunakan format *Behavior Driven Development* (BDD) dengan *Gherkin*, dalam praktiknya skenario pengujian disusun secara reaktif berdasarkan *bug* yang ditemukan pada setiap *sprint*, bukan dari analisis kelas kondisi input sejak awal. Pola kerja ini menyebabkan *test case* yang dihasilkan cenderung redundan khususnya berupa pengulangan skenario pada setiap peran pengguna tanpa perbedaan perilaku sistem yang signifikan serta belum tersusun berdasarkan representasi kelas input yang sistematis, sehingga berpotensi menimbulkan *defect leakage* pada jalur eksekusi yang belum tercakup [6].

Beberapa penelitian terdahulu telah menunjukkan efektivitas metode *Equivalence Partitioning* (EP) dalam pengujian *black box*. Studi pada [5] membuktikan EP efektif mengidentifikasi kesalahan sistem melalui pengujian kondisi *valid* dan tidak *valid* secara sistematis pada aplikasi Android. Studi pada [17] menunjukkan bahwa pengelompokan data uji ke dalam kelas ekuivalensi membantu memverifikasi kesesuaian fungsionalitas sistem. Studi pada [3] dan [2] masing-masing menunjukkan bahwa EP mampu mengurangi *test case* yang redundan dan meningkatkan efisiensi waktu pengujian. Namun, penelitian-penelitian tersebut umumnya menerapkan EP sebagai teknik pengujian *black box* secara umum, dan belum banyak yang mengintegrasikan EP dengan format *Gherkin* dalam kerangka BDD, khususnya pada konteks pengujian manual tanpa otomatisasi celah inilah yang menjadi fokus penelitian ini.

Penelitian ini mengusulkan penerapan metode *Equivalence Partitioning* yang diintegrasikan ke dalam format *Gherkin* sebagai

solusi untuk mengoptimalkan cakupan dan efisiensi *test case* pada aplikasi Kompas.id. Setiap kondisi input dikelompokkan terlebih dahulu ke dalam kelas ekuivalen *valid* dan tidak *valid* sebelum dituangkan ke dalam skenario *Given When Then*, sehingga setiap skenario yang ditulis benar-benar mewakili satu kondisi berbeda dan tidak tumpang tindih. Kebaruan penelitian ini terletak pada integrasi EP ke dalam format *Gherkin* yang telah digunakan di Kompas.id, sehingga menghasilkan perubahan konkret berupa berkurangnya skenario redundan, meningkatnya cakupan kondisi input yang teruji, dan tersedianya *test case* terstruktur yang dapat dijadikan acuan pada *sprint* berikutnya.

Berdasarkan uraian tersebut, penelitian ini bertujuan untuk:

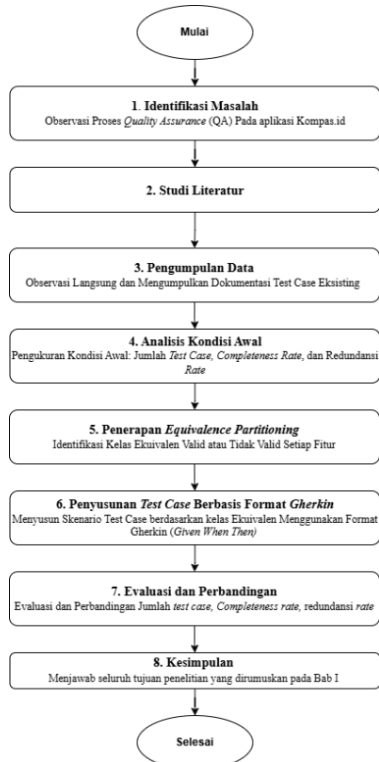
1. Menganalisis dan mengidentifikasi kelemahan proses penyusunan *test case* eksisting pada aplikasi Kompas.id, khususnya redundansi skenario dan rendahnya *completeness rate*
2. Mengoptimalkan dan mengembangkan *test case* berbasis format *Gherkin* menggunakan metode *Equivalence Partitioning*
3. Membandingkan kualitas *test case* sebelum dan sesudah penerapan metode berdasarkan jumlah *test case*, *completeness rate*, dan tingkat redundansi
4. Mengukur efektivitas penerapan metode *Equivalence Partitioning* dalam mengoptimalkan cakupan *test case* pada aplikasi Kompas.id.

Artikel ini merupakan hasil orisinal dari penelitian penulis dan belum pernah diterbitkan atau sedang dalam proses penerbitan pada media publikasi lain.

2. METODOLOGI PENELITIAN

Penelitian ini menerapkan metode *Equivalence Partitioning* (EP) yang diintegrasikan ke dalam format *Gherkin* untuk

mengoptimalkan *test case* pada aplikasi Kompas.id. Objek penelitian mencakup sebelas fitur yang diuji terhadap tiga peran pengguna, yaitu *Subscriber* (pengguna dengan langganan aktif), *Regon* (pengguna login tanpa langganan), dan *Guest/Anon* (pengguna yang belum registrasi atau login). Sebelas fitur tersebut dipilih berdasarkan tingkat interaksi pengguna tertinggi pada aplikasi, sehingga prioritas pengujian diarahkan pada area yang paling berpengaruh terhadap pengalaman pengguna sehari-hari. Tahapan penelitian secara keseluruhan disajikan pada Gambar 1.



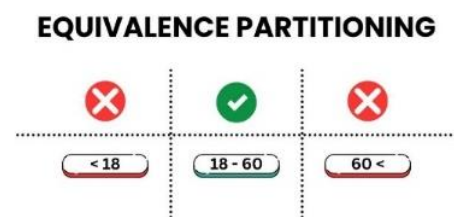
Gambar 1. Kerangka Penelitian

Penelitian dimulai dengan identifikasi masalah melalui observasi langsung terhadap proses pengujian tim *Quality Assurance* (QA) Kompas.id, yang menemukan bahwa penyusunan *test case* masih dilakukan secara reaktif melalui *exploratory testing* dan *ad hoc testing* berdasarkan *bug* yang ditemukan pada setiap *sprint*, sehingga berpotensi menimbulkan *defect leakage* [1]. Studi literatur dilakukan secara paralel terhadap publikasi tahun 2022–2025 yang membahas EP dan *Behavior Driven Development* (BDD).

Pengumpulan data dilakukan melalui dua teknik yang saling melengkapi. Pertama,

observasi langsung terhadap aktivitas tim QA dalam menyusun, melaksanakan, dan mendokumentasikan pengujian pada siklus *sprint* yang sedang berjalan, sehingga diperoleh pemahaman kontekstual atas kondisi pengujian yang sesungguhnya [13]. Kedua, dokumentasi, yaitu pengumpulan *test case* eksisting berformat *Gherkin* yang telah disusun tim QA menggunakan *Microsoft Excel* (*Microsoft 365*). dengan penggunaan perangkat yang spesifik untuk menjaga konsistensi kondisi pengujian selama pengumpulan data berlangsung. Kombinasi kedua teknik tersebut berfungsi sebagai triangulasi metode untuk menjamin validitas dan kredibilitas data yang dikumpulkan [7]. Data tersebut dianalisis sebagai *baseline* untuk mengukur jumlah *test case*, *completeness rate*, dan redundansi awal sebelum penerapan EP.

Penerapan EP dilakukan melalui tiga langkah. Mengacu pada standar ISTQB, kelas ekuivalen didefinisikan sebagai bagian dari domain *input* atau *output* di mana perilaku sistem diasumsikan sama berdasarkan spesifikasinya [18]. Berdasarkan prinsip tersebut, langkah penelitian meliputi: (1) identifikasi parameter *input* pada setiap fitur berdasarkan spesifikasi sistem, (2) pengelompokan parameter ke dalam kelas ekuivalen *valid* dan tidak *valid*, dengan ketentuan bahwa fitur *navigasi* tanpa batas nilai yang dapat didefinisikan diklasifikasikan seluruhnya sebagai kelas *valid* [16], dan (3) penentuan satu nilai representatif untuk mewakili setiap kelas ekuivalen, sehingga jumlah *test case* dapat direduksi tanpa mengurangi cakupan pengujian [6]. Ilustrasi prinsip pengelompokan EP ditunjukkan pada Gambar 2.



Gambar 2. Contoh Pengelompokan *Equivalence Class*

Setiap kelas ekuivalen yang telah diidentifikasi selanjutnya dikonversi menjadi satu skenario pengujian berformat *Gherkin* dengan struktur *Given When Then*, mengikuti kerangka dokumentasi BDD yang telah digunakan tim QA Kompas.id sebelum penelitian ini [17]. Prinsip "satu kelas ekuivalen satu skenario" diterapkan agar setiap skenario memiliki tujuan pengujian yang jelas dan tidak tumpang tindih dengan skenario lain.

Efektivitas metode EP dievaluasi menggunakan tiga indikator kuantitatif. Pertama, reduksi *test case*, yang mengukur persentase penurunan jumlah *test case* sebelum dan sesudah penerapan EP [3]:

$$\text{Reduksi Test Case} = \left(\frac{\text{Test Case Awal} - \text{Test Case Akhir}}{\text{Test Case Awal}} \right) \times 100\%$$

Kedua, *completeness rate*, yang mengukur proporsi kelas ekuivalen yang terwakili dalam *test case* terhadap total kelas ekuivalen yang seharusnya ada [2], [16]:

$$\text{Completeness Rate} = \left(\frac{\text{Jumlah Kelas Equivalence yang Terwakili}}{\text{Total Kelas Equivalence yang Seharusnya Ada}} \right) \times 100\%$$

Ketiga, *redundansi rate*, yang mengukur proporsi skenario yang menguji kondisi dari kelas ekuivalen yang sama dengan skenario lain, mengacu pada konsep *redundancy score* [12], [2], [3]:

$$\text{Redundansi Rate} = \left(\frac{\text{Jumlah Test Case Redundan}}{\text{Total Test Case}} \right) \times 100\%$$

Penurunan redundansi setelah penerapan EP diukur melalui *redundancy reduction rate*, yaitu

$$\text{Redundansi Reduction Rate} = \left(\frac{\text{Redundansi Sebelum} - \text{Redundansi Sesudah}}{\text{Redundansi Sebelum}} \right) \times 100\%.$$

Hasil nilai indikator *completeness rate*, *redundansi reduction rate* dan *redundansi rate* diinterpretasikan berkisar 0–100%, di mana nilai *completeness rate* dan *Redundansi Reduction Rate* yang mendekati 100% menunjukkan cakupan pengujian yang semakin lengkap, sedangkan nilai redundansi

yang mendekati 0% menunjukkan efisiensi *test case* yang semakin tinggi.

3. HASIL DAN PEMBAHASAN

Analisis kondisi awal terhadap 101 *test case* eksisting pada sebelas fitur aplikasi Kompas.id menunjukkan rata-rata *completeness rate* sebesar 83,85% dan rata-rata *redundansi rate* sebesar 39,24%. Temuan ini mengonfirmasi bahwa pendekatan *exploratory* dan *ad hoc testing* yang selama ini diterapkan menghasilkan dua masalah struktural sekaligus. Sebagian kelas kondisi input, khususnya kondisi tidak *valid*, belum terwakili dalam skenario pengujian, sementara hampir 4 dari 10 *test case* yang ada bersifat duplikatif akibat pengulangan skenario identik lintas peran pengguna (*Anonim*, *Regon*, *Subscriber*) tanpa perbedaan perilaku sistem yang nyata.

Sebagai ilustrasi konkret, fitur *Login* pada kondisi awal memiliki 9 *test case*, dengan 2 parameter input (email/no. ponsel, dan password).

Tabel 1. Test case kondisi awal

No	Gherkin Definition	aktor
1	Given user open Kompas.id app and system auto direct to home page and user klik menu account and user klik button "masuk" when user not input email or No ponsel and user klik button berikutnya then user sees the Mandatory Field Error	Guest
...	...	...
9	Given user open Kompas.id app and system auto direct to home page and user klik menu account and user klik button "masuk" and user input email or No ponsel when user input password passed, and then user delete input email and user klik button masuk then user sees the Mandatory Field Error	Suber

Berdasarkan spesifikasi sistem, teridentifikasi 10 kelas ekuivalen sebagaimana disajikan pada Tabel 2.

Tabel 2. Contoh Kelas Ekuivalen Fitur Login

Kode	Field	Deskripsi	Validitas
EP1	Email atau No. Ponsel	Format email valid	Valid
EP2	Email atau No. Ponsel	Format email tidak valid	Tidak Valid
EP3	Email atau No. Ponsel	Email kosong	Tidak Valid
EP4	Password	Password valid (≥ 8 karakter)	Valid
EP5	Password	Password tidak valid (< 8 karakter)	Tidak Valid
EP6	Password	Password kosong	Tidak Valid
EP7	Email atau No. Ponsel	No. telepon valid	Valid
EP8	Email atau No. Ponsel	No. telepon tidak valid	Tidak Valid
EP9	Email atau No. Ponsel	No. telepon kosong	Tidak Valid
UR1	User Role	Subscriber / Regon	Valid

Dari 10 kelas ekuivalen tersebut, *test case* eksisting hanya merepresentasikan 6 kelas sehingga perhitungan *completeness rate* sebagai berikut:

$$\text{Completeness Rate} = \left(\frac{6}{10} \right) \times 100\% = 60,00\%$$

sementara 6 dari 9 *test case* yang ada bersifat redundan karena mengulang skenario identik untuk *role Regon* dan *Subscriber* sehingga perhitungan redundansi sebagai berikut:

$$\text{Redundansi Rate} = \left(\frac{6}{9} \right) \times 100\% = 66,67\%$$

Berdasarkan sepuluh kelas ekuivalen yang telah diidentifikasi, ditentukan nilai representatif untuk masing-masing kelas sebagai data uji dalam penyusunan skenario pengujian. Berdasarkan nilai representatif tersebut disusun enam tabel kombinasi kelas *Equivalence Partitioning* pada fitur *login* sebagai berikut.

Tabel 3 kelas *Equivalence Partitioning* pada fitur *login*

Skenario	Deskripsi Skenario	Kelas EP yang Digunakan	Jenis
SC1	Semua input valid	UR1, EP1, EP4	Positive
SC2	Login dengan no. telepon valid	UR1, EP7, EP4	Positive
SC3	Email tidak valid	UR1, EP2, EP4	Negative
SC4	No. telepon tidak valid	UR1, EP8, EP4	Negative
SC5	Password tidak valid	UR1, EP1, EP5	Negative
SC6	Semua field kosong	UR1, EP3, EP6, EP9	Negative

kelas ekuivalen dikombinasikan menjadi 6 skenario unik dalam format *Gherkin*, salah satunya *test case* SC1 sebagai berikut:

Scenario: Failed login due to password length less than minimum requirement

Given the user is on the Kompas.id login page

When the user inputs a valid email

And inputs a password shorter than the required length "Pass1"

And submits the login form

Then the system displays an error message on the password field

And the user cannot proceed with the login process

Skenario di atas merepresentasikan kombinasi kelas EP1 dan EP5 yang sebelumnya sama sekali tidak terwakili dalam *test case* eksisting. Melalui pendekatan ini, fitur *Login* mengalami perubahan dari 9 *test case* dengan *completeness rate* 60% dan *redundansi rate* 66,67%, menjadi 6 *test case* sehingga perhitungan *completeness rate* sebagai berikut:

$$\text{Completeness Rate} = \left(\frac{10}{10} \right) \times 100\% = 100\%$$

redundansi *rate*

$$\text{Redundansi Rate} = \left(\frac{0}{6}\right) \times 100\% = 0\%$$

Perhitungan redundansi *reduction rate* setelah metode adalah sebagai berikut:

$$\text{Redundansi Reduction Rate} = \left(\frac{66,67\% - 0}{66,67\%}\right) \times 100\% = 100\%$$

sekaligus menambahkan cakupan pada kondisi tidak *valid* yang sebelumnya luput dari pengujian (EP2, EP6, EP7, EP8).

Pola serupa terjadi secara konsisten pada sepuluh fitur lainnya. Secara agregat, jumlah *test case* berkurang dari 101 menjadi 48 skenario perhitungannya sebagai berikut:

$$\text{Reduksi Test Case} = \left(\frac{101 - 48}{101}\right) \times 100\% = 52,48\%$$

rata-rata *completeness rate* meningkat menjadi 100%, dan rata-rata *redundansi rate* turun menjadi 0% di seluruh fitur. Ringkasan perbandingan disajikan pada Tabel 3.

Tabel 4. Rekapitulasi Hasil Evaluasi

Metrik	Sebelum EP	Sesudah EP
Jumlah <i>Test Case</i>	101	48
<i>Completeness Rate</i> (rata-rata)	83,85%	100%
Redundansi <i>Rate</i> (rata-rata)	39,24%	0%

Penurunan jumlah *test case* dari 101 menjadi 48 secara logis berimplikasi pada efisiensi waktu pengujian, karena tim QA tidak lagi perlu menyusun dan mengeksekusi skenario yang berulang untuk kondisi input yang secara perilaku sistem setara. Berkurangnya beban penyusunan *test case* yang redundan ini memungkinkan alokasi waktu pengujian dialihkan pada eksplorasi kondisi lain yang lebih kritis. Selain itu, tercapainya *completeness rate* 100% menjadikan kumpulan *test case* hasil penelitian ini berpotensi digunakan kembali

(*reusable*) pada siklus *sprint* berikutnya tanpa perlu disusun ulang dari awal, sehingga cakupan pengujian yang telah lengkap dapat langsung menjadi acuan baku pengujian regresi pada iterasi selanjutnya.

Reduksi jumlah *test case* yang diperoleh sejalan dengan temuan [3] yang menunjukkan bahwa pengelompokan data input ke dalam kelas ekuivalen secara sistematis mampu menurunkan jumlah *test case* tanpa menurunkan efektivitas pengujian, karena setiap kelas kondisi tetap terwakili oleh minimal satu skenario. Peningkatan *completeness rate* hingga 100% juga konsisten dengan [5], yang membuktikan bahwa EP efektif mengidentifikasi kondisi *valid* dan tidak *valid* secara menyeluruh pada pengujian. Namun demikian, kedua penelitian tersebut menerapkan EP sebagai teknik *black box testing* yang berdiri sendiri, sedangkan penelitian ini menunjukkan bahwa manfaat serupa dapat dicapai ketika EP diintegrasikan langsung ke dalam kerangka dokumentasi *Gherkin* yang sudah berjalan pada tim QA, tanpa mengubah format pengujian yang telah menjadi praktik organisasi.

Peningkatan *completeness rate* paling signifikan di luar contoh *Login* terjadi pada fitur *Login* pada Mode Tampilan, dari 58,33% menjadi 100%. Temuan penting lainnya adalah munculnya skenario negatif baru pada fitur *Search Filter* Pencarian dan *Filter Search* Buku, dua fitur yang sebelumnya sama sekali tidak memiliki representasi kondisi *input* gagal. Kondisi ini memperkuat argumen [6] bahwa kondisi *input* yang tidak teruji berpotensi menimbulkan *defect leakage*, sekaligus membuktikan secara empiris bahwa pendekatan reaktif berbasis *bug* cenderung mengabaikan jalur pengujian negatif yang justru krusial untuk mendeteksi kegagalan sistem sejak dini.

Dari sisi redundansi, seluruh fitur mencapai *redundansi reduction rate* sebesar 100%, kecuali fitur *login* pada mode tampilan yang sejak kondisi awal tidak memiliki redundansi sehingga indikator tersebut tidak berlaku (N/A). Pola redundansi yang paling dominan pada kondisi awal adalah pengulangan skenario identik lintas peran pengguna tanpa perbedaan

perilaku sistem, sebagaimana terlihat paling ekstrem pada fitur *Side Navigation* dengan 20 dari 30 *test case* bersifat redundan. Hasil ini selaras dengan [2] dan [16], yang menegaskan bahwa pengelompokan kondisi input ke dalam kelas ekuivalen memastikan setiap kondisi dengan perilaku sistem yang sama cukup diwakili satu nilai representatif. Sebaliknya, pada fitur yang bersifat navigasi murni tanpa parameter input dengan batas nilai yang dapat didefinisikan, seluruh kondisi tetap tergolong kelas *valid* dan tidak menghasilkan skenario negatif setelah EP diterapkan konsisten dengan prinsip EP yang menyatakan pengelompokan kelas bergantung pada perilaku sistem, bukan pada jumlah peran pengguna semata [11].

Secara keseluruhan, hasil ini menunjukkan bahwa optimalisasi *test case* tidak selalu berarti menambah jumlah pengujian, melainkan menyusun ulang skenario yang ada agar setiap kelas kondisi input terwakili secara unik. Signifikansi temuan ini terletak pada pembuktian bahwa integrasi EP ke dalam format *Gherkin* dapat diterapkan pada konteks pengujian manual tanpa otomatisasi sebuah celah yang belum banyak dibahas pada penelitian terdahulu yang umumnya berfokus pada penerapan EP sebagai teknik *black box testing* independen [17], [5], [3], [2].

4. KESIMPULAN

Penerapan metode *Equivalence Partitioning* berbasis format *Gherkin* pada pengujian manual aplikasi Kompas.id terbukti mampu menjawab seluruh tujuan penelitian. Kondisi awal menunjukkan proses penyusunan *test case* yang reaktif menghasilkan redundansi skenario berbasis peran pengguna serta cakupan kelas ekuivalen yang belum optimal. Setelah metode diterapkan, jumlah *test case* berhasil direduksi secara signifikan tanpa mengorbankan kelengkapan pengujian, cakupan kelas ekuivalen meningkat hingga terwakili secara menyeluruh, dan seluruh redundansi skenario berhasil dieliminasi. Temuan tambahan berupa skenario negatif yang sebelumnya tidak terwakili pada fitur pencarian dan filter turut mengonfirmasi bahwa pendekatan reaktif

berpotensi melewati *defect* pada jalur pengujian yang jarang tersentuh. Hasil ini menegaskan bahwa akar penyebab inefisiensi pengujian bukan terletak pada volume *test case*, melainkan pada tidak adanya kerangka pengelompokan input yang sistematis sejak awal, sekaligus melengkapi penelitian terdahulu yang umumnya menerapkan *Equivalence Partitioning* sebagai teknik *black box testing* berdiri sendiri tanpa dikaitkan langsung dengan format *Gherkin*.

Penelitian ini masih terbatas pada penerapan *Equivalence Partitioning* secara manual pada satu studi kasus aplikasi, sehingga penelitian lanjutan disarankan untuk mengombinasikan metode ini dengan *boundary value analysis* guna memperkuat deteksi *defect* pada nilai batas, serta melakukan evaluasi berkelanjutan pada siklus sprint berikutnya untuk mengukur konsistensi *completeness rate* dan redundansi *rate* dalam jangka panjang.

DAFTAR PUSTAKA

- [1] Abbas, S., Masood, S., Tahir, S., Azhar, M., Sajid, A., & Banu, S. (2025). Hybrid Automated-Manual Testing for Enhanced DevOps Pipelines. *Journal for Future Society and Education*, 2(4), 66–77. <https://doi.org/10.3991/jfse.v2i4.59235>.
- [2] Ansfridus, M., Dwiandiyanta, B. Y., & Wahyu Rahardjo Emanuel, A. (2024). Black Box Testing dengan Metode Equivalence Partitioning pada Techno Expertise Academy (TEA) Astra Credit Companies (Vol. 4, Number 2).
- [3] Apriliandra, A. R., & Nuryasin, I. (2024). Pengujian Blackbox pada Website Sistem Pemesanan Travel Online Gemilang Travel Berbasis Teknik Equivalence Partitions. *Jurnal Teknologi Sistem Informasi Dan Aplikasi*, 7(2), 859–867. <https://doi.org/10.32493/jtsi.v7i2.39049>.
- [4] Luqman Nuryana, M., Ibrahim, T., & Arifudin, O. (2024). IMPLEMENTASI DAN TRANSFORMASI SISTEM INFORMASI MANAJEMEN DI ERA DIGITAL. In *Jurnal Tahsinia* (Vol. 5, Number 9).
- [5] Maulana, M. R., Susanto, E. B., & Christianto, P. A. (2024). Pengujian Black Box dengan

- Teknik Equivalence Partitioning pada Aplikasi Monitoring Pemberian Obat Filariasis Berbasis Android. *KLIK: Kajian Ilmiah Informatika Dan Komputer*, 4(4), 2179–2187. <https://doi.org/10.30865/klik.v4i4.1603>.
- [6] Mintarsih, M. (2023). Pengujian Black Box Dengan Teknik Transition Pada Sistem Informasi Perpustakaan Berbasis Web Dengan Metode Waterfall Pada SMC Foundation. *Jurnal Teknologi Dan Sistem Informasi Bisnis*, 5(1), 33–35. <https://doi.org/10.47233/jteksis.v5i1.727>.
- [7] Putri, H. J., & Murhayati, S. (2025). Metode Pengumpulan Data Kualitatif.
- [8] Qodo. (2026). What is Equivalence Partitioning? Best Practices & Benefits. <https://www.qodo.ai/glossary/equivalence-partitioning>.
- [9] Sahyudi, S., & Voutama, A. (2025). Pengujian Fungsional Black Box Siska UNSIKA dengan Equivalence Partitioning untuk Validasi Input dan Output Sistem. *Jurnal Teknologi Sistem Informasi Dan Sistem Komputer TGD*, 8(2), 148–155. <https://ojs.trigunadharma.ac.id/index.php/jsk/index>.
- [10] Salamudin, S., & Sutabri, T. (2025). MIFORTEKH (Jurnal Manajemen Informatika & Teknologi) Implementasi Pengujian Black Box Menggunakan Metode Equivalence Partition Pada Aplikasi Siska Presensi Mobile. 5(1). <https://journal.stiestekom.ac.id/index.php/mifortek>.
- [11] Samdono, A., Sari, A. P., & Aditiawan, F. P. (2024). PENGUJIAN BLACK BOX PADA SISTEM INFORMASI STOK DAN PENJUALAN BERBASIS WEBSITE MENGGUNAKAN METODE EQUIVALENCE PARTITIONING (STUDI KASUS: CV. ALGANI KARYA MANDIRI). In *Jurnal Mahasiswa Teknik Informatika* (Vol. 8, Number 1).
- [12] Saputra, M. C., Katayama, T., Kita, Y., Yamaba, H., Aburada, K., & Okazaki, N. (2022). Measuring redundancy score for test suite evaluation by using test cases matching approach. *Journal of Robotics, Networking and Artificial Life*, 8(1), 1–5. <https://doi.org/10.2991/jrnal.k.210521.001>.
- [13] Sekar Sari, A., Aprisilia, N., & Fitriani, Y. (2025). Teknik Pengumpulan Data dalam Penelitian Kualitatif: Observasi, Wawancara, dan Triangulasi. In *Indonesian Research Journal on Education* (Vol. 5). <https://irje.org/index.php/irje>.
- [14] Suderajat, A., Leo Contantinus Meze Laki, A., Anggito Abimanyu, A., Hadikristanto, W., & Halim Anshor, A. (2025). Penerapan Manajemen Risiko dalam Software Quality Assurance: Studi Kasus Pengujian Black Box pada Aplikasi Inventori Maintenance. In *Jurnal Indonesia: Manajemen Informatika dan Komunikasi (JIMIK)* (Vol. 6, Number 1). <https://doi.org/10.35870/jimik.v6i1.1214>.
- [15] Syihab, L. N., Janniethia, & Sugiarti, Y. (2024). Pengujian Strategi Perangkat Lunak: Tinjauan Literatur Sistematis. *Jurnal Ilmiah Informatika Dan Komputer*, 1(2), 112–118.
- [16] Wahyudin, A., & Wahyudin, D. (2025). Blackbox Testing: Teknik Equivalence Partitions untuk Menguji Aplikasi Monitoring Project-Based Learning
- [17] Yulianti, Y., Khaidar, A. Al, Fazriansyah, R., Ramadhan, S. G., Putra, W. A., & Sitio, S. L. M. (2023). Pengujian Black Box pada Website MyUnpam menggunakan Teknik Equivalence Partitioning. *Jurnal Teknologi Sistem Informasi Dan Aplikasi*, 6(2), 154–161. <https://doi.org/10.32493/jtsi.v6i2.25501>
- [18] Erik van Veenendaal. (2012). *Standard glossary of terms used in Software Testing*.