

**PENERAPAN METODE EQUIVALENCE PARTITION DALAM
OTOMASI PENGUJIAN PADA WEBSITE E-COMMERCE PRODUK
VIRTUAL
SKRIPSI**



**Oleh:
ADRIAN PASKALIS
72170125**

**PROGRAM STUDI SISTEM INFORMASI
FAKULTAS TEKNOLOGI INFORMASI
UNIVERSITAS KRISTEN DUTA WACANA
TAHUN 2021**

PERNYATAAN KEASLIAN SKRIPSI

Saya menyatakan dengan sesungguhnya bahwa skripsi dengan judul:

PENERAPAN METODE EQUIVALENCE PARTITION DALAM OTOMASI PENGUJIAN WEBSITE ECOMMERCE PRODUK VIRTUAL

yang saya kerjakan untuk melengkapi sebagian persyaratan menjadi Sarjana Komputer pada pendidikan Sarjana Program Studi Sistem Informasi Fakultas Teknologi Informasi Universitas Kristen Duta Wacana, bukan merupakan tiruan atau duplikasi dari skripsi kesarjanaan di lingkungan Universitas Kristen Duta Wacana maupun di Perguruan Tinggi atau instansi manapun, kecuali bagian yang sumber informasinya dicantumkan sebagaimana mestinya.

Jika dikemudian hari didapati bahwa hasil skripsi ini adalah hasil plagiasi atau tiruan dari skripsi lain, saya bersedia dikenai sanksi yakni pencabutan gelar kesarjanaan saya.

Yogyakarta, 29 Juni 2021



Adrian Paskalis
72170125

HALAMAN PERSETUJUAN

Judul Skripsi : PENERAPAN METODE EQUIVALENCE
PARTITION DALAM OTOMASI PENGUJIAN
WEBSITE ECOMMERCE PRODUK VIRTUAL

Nama Mahasiswa : Adrian Paskalis

N I M : 72170125

Matakuliah : Skripsi

Kode : SI4046

Semester : Genap

Tahun Akademik : 2020/2021

Telah diperiksa dan disetujui di Yogyakarta,
Pada tanggal 29 Juni 2021

Dosen Pembimbing I



ARGO WIBOWO, ST., MT.

Dosen Pembimbing II



KATON WIJANA, S.Kom., M.T.

HALAMAN PENGESAHAN

PENERAPAN METODE EQUIVALENCE PARTITION DALAM OTOMASI PENGUJIAN WEBSITE ECOMMERCE PRODUK VIRTUAL

Oleh: Adrian Paskalis / 72170125

Dipertahankan di depan Dewan Penguji Skripsi
Program Studi Sistem Informasi Fakultas Teknologi Informasi
Universitas Kristen Duta Wacana - Yogyakarta
Dan dinyatakan diterima untuk memenuhi salah satu syarat memperoleh gelar
Sarjana Komputer
pada tanggal
10 Juni 2021

Yogyakarta, 29 Juni 2021
Mengesahkan,

Dewan Penguji:

1. ARGO WIBOWO, ST., MT.
2. KATON WIJANA, S.Kom., M.T.
3. UMI PROBOYEKTI, S.Kom., MLIS.
4. YETLI OSLAN, S.Kom., M.T.

Dekan


(RESTYANDITO, S.Kom., MSIS., Ph.D.)

Ketua Program Studi


(Drs. JONGWIK SIANG, M.Sc.)

ABSTRAK

Seiring perkembangan teknologi, dari konvensional hingga *online* banyak perusahaan mengembang website e-commerce untuk menggapai pelanggannya, salah satunya ialah PT. XYZ pada khususnya website e-commerce produk virtual yang melayani pembayaran pulsa, listrik, PBB, BPJS, dsb. Menjaga kualitas dari *website e-commerce* merupakan hal yang penting harus dilakukan oleh perusahaan terkait sehingga, diperlukan pengujian pada *website e-commerce*. Sebelumnya pengujian *website e-commerce* masih dilakukan secara manual, namun masih ditemui beberapa kekurangan yaitu, pengujian sederhana yang dilakukan berulang-ulang, scenario pengujian yang banyak dengan dilakukan oleh beberapa tim *Quality Assurance* (QA) bersamaan, dan kesalahan manusia akibat kejenuhan. Maka penulis akan meneliti pengujian automasi guna mengurangi kekurangan tersebut.

Sistem pengujian automasi yang diteliti menggunakan merupakan pengujian *Black Box* dengan metode *Equivalence Partitions*, yang bermanfaat untuk mengurangi jumlah kasus uji dengan membagi kelas-kelas partisi, yaitu kelas valid dan kelas tidak valid. Sistem pengujian automasi akan menguji *website e-commerce* berdasarkan tampilan antarmuka dan uji fungsionalitas pada fitur-fitur yang ada. Keberhasilan dari sistem pengujian akan ditentukan berdasarkan kalkulasi evaluasi keberhasilan test case yang dieksekusi

Berdasarkan hasil penerapan metode *Equivalence Partitions* pada pengujian automasi *website e-commerce* studi kasus kali ini, tidak ditemukannya *bug/errors* dan tingkat keberhasilan *test case* yang dieksekusi 97,53% dan *test case* metode *Equivalence Partitions* 100%. Sistem pengujian automasi ini, dapat menjawab kekurangan pengujian manual, pengujian yang berulang , jumlah scenario dan penguji yang banyak, telah diatasi dengan satu sitem pengujian serta dapat meminimalkan human errors akibat kejenuhan.

Kata kunci : Pengujian automasi, *Automation Testing*, *Equivalence Partitions*, *Testing*

KATA PENGANTAR

Puji syukur kepada Tuhan Yang Maha Esa atas berkat-Nya yang melimpah, sehingga penulis dapat menyelesaikan laporan akhir skripsi dengan judul “PENERAPAN METODE EQUIVALENCE PARTITION DALAM OTOMASI PENGUJIAN PADA WEBSITE E-COMMERCE PRODUK VIRTUAL PT. XYZ” , sebagai syarat untuk menyelesaikan Program Sarjana (S1) pada Program Studi Sarjana Sistem Informasi Fakultas Teknologi Informasi Universitas Kristen Duta Wacana.

Dalam penyusunan laporan akhir skripsi ini, cukup banyak tantangan serta hambatan yang dihadapi penulis, namun berkat bimbingan, bantuan, dukungan, serta motivasi oleh berbagai pihak sehingga dapat menyelesaikan laporan akhir skripsi ini. Maka dari itu, penulis ingin menyampaikan ucapan terimakasih kepada:

1. Ibunda tercinta, Siska Magdalena dan keluarga besar yang telah memberikan dukungan baik secara moril maupun materil serta doa dan berkat yang tiada henti kepada Penulis.
2. Bapak Argo Wibowo, ST., M.T. selaku dosen pembimbing pertama penulis, yang telah membagi ilmu, saran, dan membimbing penulis sehingga dapat menyelesaikan laporan akhir skripsi ini.
3. Bapak Katon Wijana, S.Kom., M.T. selaku dosen pembimbing kedua penulis, yang juga telah membagi ilmu, saran, dan membimbing penulis sehingga dapat menyelesaikan laporan akhir skripsi ini.
4. Bapak Drs. Jong Jek Siang, M.Sc selaku Kepala Program Studi Sistem Informasi Universitas Kristen Duta Wacana.
5. Seluruh Bapak/Ibu Dosen Program Studi Sistem Informasi yang telah memberikan pengalaman dan pengetahuan yang bermanfaat bagi penulis selama perkuliahan.
6. Teman-teman yang telah berjuang bersama selama kerja praktik dan penyusunan skripsi Putu Abdi Setiawan, Desta Siwi Prabawan, Valeriana Tanesha, Grace Hutabarat, dan Beni Mulia.
7. Teman-teman yang telah berjuang bersama selama perkuliahan, Yos Rafel, Tita Marita, Yashinta Novita, Angkie Octovaldo, Alfadeo Jeremy, Laurentia Cristi, Christian Dorra, Didimus Chandra, Nikolaus

Aryawan, Nana Eka Wulandari, Michael Gerard, Monica Carista, Cynthia Kumalasari, Eva Kristina, dkk.

8. Rekan-rekan anggota Himpunan Mahasiswa Sistem Informasi 2018 dan 2019 karena telah memberikan pengalaman berorganisasi selama perkuliahan
9. Bapak Himawan, Bapak Tulus Wardoyo, dan Squad 2 Virtual yang telah memberikan pengalaman, ilmu, dan pengetahuan baru selama kerja praktik.
10. Kedai *coffeeshop Awor dan LoepaLelah* yang telah memberikan fasilitas tempat yang nyaman dan kualitas wifi yang stabil sehingga penulis nyaman dalam menyelesaikan skripsi.
11. Semua pihak yang tidak dapat disebutkan satu persatu yang telah membantu penulis dalam menyelesaikan laporan akhir skripsi ini.

Penulis meyakini bahwa skripsi yang telah diselesaikan penulis masih jauh dari kata sempurna dikarenakan terbatasnya kemampuan, pengalaman, dan pengetahuan yang dimiliki oleh penulis. Oleh sebab itu, penulis mengharapkan saran, masukan serta kritik yang membangun dari berbagai pihak. Semoga skripsi ini dapat bermanfaat bagi para pembaca dan pihak lainnya yang membutuhkan.

DAFTAR ISI

PERNYATAAN KEASLIAN SKRIPSI.....	ii
HALAMAN PERSETUJUAN.....	iii
HALAMAN PENGESAHAN.....	iv
ABSTRAK.....	v
KATA PENGANTAR	vi
DAFTAR ISI.....	viii
DAFTAR GAMBAR	xi
DAFTAR TABEL.....	xii
DAFTAR KODE PROGRAM.....	xiii
Bab 1 Pendahuluan	1
1.1 Latar Belakang.....	1
1.2 Rumusan Masalah	2
1.3 Batasan Masalah	3
1.4 Tujuan dan Manfaat Penelitian.....	3
1.5 Metodologi Penelitian	3
1.6 Sistematika Penulisan	5
Bab 2 Tinjauan Pustaka	6
2.1 Tinjauan Pustaka	6
2.2 Landasan Teori	7
2.2.1 Pengujian Perangkat Lunak	7
2.2.2 Black Box Testing	7
2.2.3 Metode Equivalence Partitions.....	8
2.2.4 Software Testing Life Cycle (STLC)	9
2.2.5 Dokumentasi Pengujian.....	11
2.2.6 Persentase	12
Bab 3 Analisis dan Perancangan Sistem Pengujian	13
3.1 Data Penelitian	13
3.2 Requirement Analysis.....	14
3.2.1 Requirement System	14
3.2.2 Use Case Diagram	16
3.3 Test Planning	17
3.4 Test Case Development	18

3.4.1	Test Case Development Pengujian Otomasi	19
3.4.2	Test Case Development Metode Equivalence Partitions	19
3.5	<i>Test Environment Setup</i>	20
3.6	Perancangan Evaluasi	21
Bab 4	Penerapan dan Analisis Sistem Pengujian Otomasi	22
4.1	Penerapan Pengujian Automasi Website Studi Kasus	22
4.1.1	Pengujian Automasi pada Halaman Login.....	22
4.1.2	Pengujian Automasi pada Halaman Home.....	23
4.1.3	Pengujian Automasi pada Halaman Produk.....	24
4.1.4	Pengujian Automasi pada Halaman <i>Checkout</i>	25
4.1.5	Pengujian Automasi pada Halaman Pembayaran	26
4.1.6	Pengujian Automasi pada Halaman <i>Thankyou</i>	27
4.1.7	Pengujian Automasi pada Halaman <i>Order History</i>	28
4.2	Penerapan Program Pengujian Automasi	29
4.2.1	Penerapan program Class Page.....	29
4.2.2	Penerapan program Class Test.....	30
4.2.3	Penerapan Program Page Factory	31
4.2.4	Penerapan Program Driver Test	32
4.2.5	Penerapan Program Test . eXtensible Markup Language (XML).....	33
4.2.6	Penerapan <i>Program Master Test eXtensible Markup Language (.XML)</i> ..	34
4.3	<i>Test Execution</i>	35
4.3.1	Test Execution berdasarkan Tabel <i>Test Case</i>	35
4.3.2	<i>Test Exececution</i> berdasarkan <i>Dashboard</i> Hasil Pengujian	37
4.4	<i>Test Cycle Closure</i>	40
4.4.1	Analisa Hasil Pengujian Automasi dan Pengelompokkan Bug	40
4.4.2	Kalkulasi Evaluasi Hasil Pengujian Automasi	41
4.5	Kelebihan dan Kekurangan Penerapan Metode Equivalence Partitions pada Pengujian Otomasi.....	42
4.5.1	Kelebihan Penerapan Metode Equivalence Partitions dan Pengujian Otomasi	42
4.5.2	Kekurangan Penerapan Metode Equivalence Partitions dan Pengujian Otomasi.....	42
5.1	Kesimpulan	43
5.2	Saran	43

DAFTAR PUSTAKA	44
LAMPIRAN.....	46

DAFTAR GAMBAR

Gambar 2.1 contoh Metode Equivalance Partitions.....	8
Gambar 3.1 Use Case Diagram Customer pada website e-commerce studi kasus	16
Gambar 4.1 Pengujian Automasi pada Halaman Login.....	22
Gambar 4.2 Pengujian automasi pada Halaman Home.....	23
Gambar 4.3 Pengujian automasi pada halaman produk	24
Gambar 4.4 Pengujian automasi pada halaman Checkout	25
Gambar 4.5 Pengujian automasi pada halaman pembayaran	26
Gambar 4.6 Pengujian automasi pada halaman Thankyou	27
Gambar 4.7 Pengujian automasi pada halaman riwayat transaksi	28
Gambar 4.8 Dashboard hasil pengujian	37
Gambar 4.9 Dashboard Hasil Pengujian Berhasil.....	38
Gambar 4.10 Dashboard hasil pengujian gagal.....	39

DAFTAR TABEL

Tabel 3.1 Tabel Requirement System	14
Tabel 3..2 Rancangan <i>Test Scenario</i>	15
<i>Tabel 3.3 Test Plan</i>	17
Tabel 3.4 Rancangan Test Case	18
Tabel 4.1 Hasil Penerapan Pengujian Automasi	36

DAFTAR KODE PROGRAM

Kode Program 1 Class Home Page Virtual.....	29
Kode Program 2 Class Home Page test.....	30
Kode Program 3 Class Page Factory.....	31
Kode Program 4 Driver Test.....	32
Kode Program 5 File Test.XML	33
Kode Program 6 Master Test.XML	34

Bab 1

Pendahuluan

1.1 Latar Belakang

Seiring berjalannya waktu dan perkembangan teknologi, perusahaan *retail* waralaba mengembangkan *e-commerce* untuk menjangkau masyarakat yang lebih luas, dapat diakses kapan saja, siapa saja, dan dimana saja melalui *website e-commerce*. Perusahaan terkait, mengembangkan *e-commerce* yang mampu melayani pembelian barang pokok, menu makanan, tiket perjalanan kereta api & pesawat, tiket wahana & konser, serta produk virtual.

Dengan menggunakan website, keuntungan perusahaan dalam mengembang *e-commerce* ialah peningkatan fitur baru untuk pengguna secara otomatis dan akses yang tak terbatas dari semua perangkat yang memiliki jaringan internet. Karena *website e-commerce* menjadi hal yang penting, maka kualitas suatu *website* harus terjaga dan terjamin tidak ada kekurangan serta kesalahan pada *website e-commerce* yang telah dibuat.

Sebagai usaha untuk menjaga dan menjamin kualitas *website e-commerce*, diperlukan pengujian terhadap perangkat lunak sebelum dapat digunakan oleh pengguna. Ada pun tujuan pengujian perangkat lunak menurut Oscar Pastor pada artikel jurnal oleh Komarrudin (2016), ialah untuk menemukan kesalahan yang menyebabkan perangkat lunak yang telah dibangun gagal dan untuk memperoleh produk yang berkualitas yang memberikan produktivitas tinggi.

Pada perusahaan studi kasus ini, terdapat tim *Quality Assurance* (QA) guna melakukan tugas tersebut. Berdasarkan pengalaman penulis, ketika melakukan pengujian perangkat lunak secara manual, ditemukan beberapa kekurangan, yaitu pengujian sederhana yang dilakukan berulang-ulang, *scenario* pengujian yang banyak dengan dilakukan oleh beberapa QA bersamaan, dan kesalahan manusia akibat kejenuhan tekanan pada *deadline* yang menyebabkan beberapa kesalahan

dan kekurangan pada *website e-commerce* terlihat dan dirasakan oleh pengguna *website e-commerce* khususnya pada produk virtual.

Oleh karena itu, penulis mengusulkan untuk menerapkan pengujian automasi dengan Metode *Equivalence Partitions* dapat menemukan kekurangan, kesalahan fungsional, kesalahan antarmuka *website*, kesalahan struktur data, dan kesalahan performa pada *website e-commerce* produk virtual (Adella Rosalina A. G., 2020). Harapannya dengan menerapkan hal tersebut dapat meningkatkan efisiensi waktu, efektivitas, dan meningkatkan hasil akurasi pengujian yang dapat meningkatkan performa perusahaan terhadap *website e-commerce* studi kasus. Hal-hal inilah yang melatarbelakangi penulis untuk menerapkan pengujian automasi pada *website e-commerce* studi kasus kali ini.

1.2 Rumusan Masalah

Berdasarkan latar belakang yang telah disebutkan, maka permasalahan yang dapat diselesaikan, ialah :

Pengujian perangkat lunak secara manual masih memiliki beberapa kekurangan, yaitu :

- i. Pengujian manual dilakukan secara berulang-ulang sesuai dengan *timeline* pekerjaan dan adanya penambahan fitur-fitur baru pada *website e-commerce*.
- ii. Pengujian manual melakukan banyak skenario pengujian dalam satu waktu dan terkadang membutuhkan lebih dari satu orang penguji.
- iii. Kesalahan manusia yang dilakukan oleh penguji yang terkadang masih melewatkan *bug/error*.

1.3 Batasan Masalah

- A. Studi kasus pada perusahaan PT.XYZ yang membangun *website e-commerce* xyz Produk Virtual.
- B. Pengujian automasi yang digunakan ialah *Black Box Testing* dengan Metode *Equivalence Partitions*.
- C. Hasil pemograman akan digunakan untuk PT.XYZ, tidak dapat disebarluaskan, serta identitas perusahaan disamarkan karena satu dan lain hal.
- D. Pengujian perangkat lunak yang dilakukan ialah proses yang dilakukan oleh Konsumen dimulai dari *Login*, melihat produk virtual, memasukkan produk ke keranjang belanja, melakukan pembayaran, dan melihat riwayat transaksi.
- E. Pengujian perangkat lunak yang dilakukan menggunakan sampel data 5 kategori produk virtual yang berbeda.

1.4 Tujuan dan Manfaat Penelitian

Merancang dan membangun pengujian automasi untuk membantu melakukan pengujian sehingga mengurangi kerugian bagi perusahaan akibat temuan *errors* oleh Konsumen. Dapat mengurangi kesalahan manusia dalam melakukan pengujian serta mengurangi pengujian manual yang bersifat perulangan dan dapat melakukan pengujian dengan jumlah kasus uji yang membutuhkan waktu dan sumberdaya manusia seminimal mungkin pada *website e-commerce* produk virtual studi kasus kali ini.

1.5 Metodologi Penelitian

Langkah dalam melakukan penelitian adalah sebagai berikut :

A. Studi Pustaka

Dalam rangka melaksanakan penelitian ini, untuk mengumpulkan data hal yang dilakukan penulis adalah melakukan studi pustaka. Studi pustaka dilakukan untuk mempermatang penerapan metode penelitian yang akan dilakukan peneliti. Studi pustaka dilakukan mempelajari jurnal atau artikel-artikel yang sudah ada sehingga dapat dijadikan sebagai referensi. Lalu, penulis juga mempelajari langsung proses bisnis pada *website e-commerce* terkait melalui observasi atau pengamatan langsung dan wawancara dengan narasumber untuk penggunaan data

pengujian yang boleh digunakan dan dokumentasi sistem seperti spesifikasi, *requirement*, dsb.

B. Identifikasi Masalah

Pada tahapan ini bertujuan untuk menggali masalah dan kebutuhan yang diperlukan untuk kepentingan penelitian. Pengujian Sistem akan dikelompokkan berdasarkan *Requirement System*. Dengan berdasarkan *Requirement System* yang akan menjadi objek pengujian. Pendekatan yang akan digunakan peneliti ialah uji kasus dengan menggunakan Metode *Equivalence Partitions*.

C. Pengambilan Data dan Perancangan Pengujian

Pada tahap ini, penulis memperoleh data berdasarkan wawancara dan observasi pada *website* studi kasus seperti yang telah disebutkan pada sub bab sebelumnya. Lalu, penulis merancang sekaligus membuat *test case* atau kasus uji dengan berdasarkan dokumentasi dan requirement sistem yang ada dengan menggunakan data uji yang diperbolehkan.

D. Penerapan pengujian Equivalence Partitions

Pada tahapan ini, penulis membangun dan melakukan pengujian sesuai dengan kasus uji yang telah dibuat sesuai dengan penerapan Metode *Equivalence Partitions* dengan menggunakan *Selenium*, *framework TestNG*. Pengujian yang dilakukan adalah validasi kesalahan fungsional, kesalahan antarmuka *website*, dan kesalahan struktur data.

E. Kalkulasi dan Evaluasi keberhasilan

Tahapan terakhir, penulis melakukan perhitungan hasil pengujian berdasarkan persentase keberhasilan sehingga dapat memberikan jawaban terhadap permasalahan yang ada dan melakukan evaluasi yang berkaitan penerapan pengujian automasi berbasis *website* menggunakan Metode *Equivalence Partitions* sehingga bisa menjadi acuan lebih baik kedepannya.

1.6 Sistematika Penulisan

Pada bab 1 berisi tentang latar belakang, rumusan masalah, batasan masalah, spesifikasi sistem, tujuan dan manfaat penelitian, metodologi penelitian dan sistematika penulisan. Kemudian, pada bab 2 membahas tentang landasan teori yang digunakan dalam melakukan penelitian, berisi dasar-dasar teori yang didapat dari tinjauan pustaka penelitian terkait, dan digunakan untuk menyelesaikan permasalahan dalam merancang sistem pengujian automasi pada *website e-commerce* studi kasus. Kemudian, pada bab 3 berisi tentang pengambilan data, perancangan dan pembangunan sistem pengujian, *requirement system*, rancangan pengujian yang akan dilakukan, dan rancangan kalkulasi dan evaluasi terhadap pengujian automasi.

Pada bab 4, menjelaskan tentang hasil penerapan pengujian automasi dengan menggunakan Metode *Equivalence Partitions* dan hasil kalkulasi dan evaluasi terhadap pengujian automasi lengkap dengan kelemahan dan kelebihan sistem pengujian. Kemudian, bab 5 berisi kesimpulan mengenai penelitian yang dikerjakan, serta usulan yang dapat diimplementasikan untuk pengembangan sistem pengujian otomasi lebih lanjut.

Bab 2

Tinjauan Pustaka

2.1 Tinjauan Pustaka

Pada penelitian yang ditulis oleh M.Nurudin & Windi (2019), mengungkapkan bahwa “kesalahan dan kekurangan pada perangkat lunak dapat ditemukan pada tahap pengembangan sebelum dipasarkan kepada pengguna perangkat lunak, apabila dari *developer* dan *project manager* memberikan waktu khusus untuk melakukan pengujian pada perangkat lunak yang sudah dibangun atau dikembangkan agar kemudian dapat melakukan perbaikan terhadap perangkat lunak sehingga, pengujian menjadi jaminan kualitas perangkat lunak dan menjadi bagian tidak terpisahkan dalam siklus pengembangan perangkat lunak”.

Dikutip dari penelitian yang ditulis oleh Dhega, I Made Sudana, & Noor Hudallah (2020), pengujian menggunakan Metode *Equivalence Partitions* yang merupakan salah satu metode dari *BlackBox Testing* adalah untuk menemukan kesalahan atau kekurangan berupa, fungsi yang hilang atau salah; kesalahan desain antarmuka atau tampilan system; kesalahan struktur data atau akses *database* eksternal; kesalahan performa; dan kesalahan inisialisasi & terminasi. Pada penelitiannya, Febiharsa, dkk (2020) melakukan pengujian terhadap Sistem Informasi Uji Kompetensi Lembaga Sertifikasi Profesi (LSP) Batik. Diungkapkan bahwa pengujian menggunakan *BlackBox Testing*, perangkat lunak tersebut telah dikatakan layak untuk digunakan, namun masih perlu dilakukan pengujian dengan pengguna untuk dapat mengetahui sejauh mana pengguna dapat memahami fungsi dan mengoperasikannya atau dalam istilah lain *user-friendly*.

Berdasarkan penelitian oleh Adelia Rosalina (2020), melakukan pengujian pada *website* menggunakan Metode *Equivalence Partitions* diperlukan beberapa tahap pengujian yaitu, membuat rancangan *test case* berdasarkan fungsi yang ada, membuat Batasan pengujian sesuai dengan Metode *Equivalence Partitions*, membuat model pengujian dari *scenario* pengujian dan hasil yang diharapkan, dan

terakhir, melakukan pengujian berdasarkan rancangan model pengujian. Pada penelitiannya, dkk membagi kelompok 2 nilai yang diuji, yaitu nilai valid dan tidak valid. Setiap kelompok nilai diambil satu contoh untuk data masukkan pada *scenario test*. Hasil dari penelitiannya, bahwa *system* dikatakan layak penggunaan dengan saran bahwa pengujian harus dirancang sebaik mungkin agar dapat menemukan kesalahan secara sistematis dan dapat diperbaiki dengan waktu dan usaha yang minimal.

2.2 Landasan Teori

2.2.1 Pengujian Perangkat Lunak

Dikutip dari penelitian yang ditulis oleh Kommarudin (2016), dengan pengertian yang diungkapkan oleh Perry bahwa pengujian perangkat lunak merupakan suatu proses eksekusi *program* atau *system* untuk mengevaluasi atribut atau kemampuan suatu *program* atau *system* dan menentukan bahwa *program* atau *system* sudah berhasil memenuhi hasil yang dibutuhkan oleh suatu perusahaan.

Adapun pengertian lain, dikutip dari Waskhito & Fajar (2002), ialah proses mencari kesalahan pada setiap komponen perangkat lunak, mencatat hasilnya, dan mengevaluasi setiap aspek pada tiap komponen perangkat lunak dan fasilitas-fasilitas dari perangkat lunak yang akan dikembangkan. Dinyatakan juga bahwa, pengujian perangkat lunak dapat dilakukan dengan 2 cara, yaitu *Black Box Testing* dan *White Box Testing*.

2.2.2 Black Box Testing

Black Box Testing atau yang biasa disebut dengan *data driven testing* atau *input-output testing* merupakan salah satu cara pengujian perangkat lunak yang mengabaikan *internal behaviour* dan struktur *program*. (Handy, 2014). Pada penelitian yang ditulis oleh Komarrudin (2016), *Black Box Testing* melakukan pendekatan pengujian dengan menggunakan data pengujian berasal dari persyaratan fungsional yang telah ditentukan tanpa memperhatikan struktur *program* yang dibuat. Dituliskan oleh Adella (2020), Dalam penerapannya pada pengujian perangkat lunak, *Black Box Testing* sangat efektif digunakan untuk menemukan kesalahan fungsi-fungsi yang hilang atau salah, kesalahan desain antarmuka atau

tampilan, kesalahan dalam struktur data atau akses menuju *database* dan kesalahan pada performa.

2.2.3 Metode Equivalence Partitions

Dikutip dari penelitian yang ditulis oleh Adi Krismadi, dkk (2019) *Equivalence Partitions* adalah metode pengujian *Black Box* yang memecah atau membagi *domain* masukan dari *program* kedalam kelas atau kelompok data sehingga *test case* dapat diuji. Perancangan *test case Equivalence Partitions* berdasarkan evaluasi kelas *Equivalence* untuk kondisi masukan yang menggambarkan kumpulan keadaan yang valid atau tidak. Kondisi masukan dapat berupa nilai *integer*, *numeric*, *range* nilai, kumpulan nilai lainnya yang berhubungan atau *Boolean*.

Pada penelitian yang ditulis oleh Sofiyah (2019), keuntungan menggunakan Metode *Equivalence Partition* pada komparasi dengan Metode *Boundary Value Analysis* ialah dengan menggunakan Metode *Equivalence Partition* pengujian dapat menguji berdasarkan tipe data jangkauan (*range*) dan lebih bebas, serta sangat cocok untuk mengeksplorasi semua kemungkinan berdasarkan kriteria. Namun, dengan menggunakan Metode *Equivalence Partition*, memiliki kesulitan untuk memilih nilai yang *representative* dalam partisi untuk diuji karena jangkauannya yang luas.

Terdapat gambaran lebih jelas untuk penjelasan bagaimana Metode *Equivalence Partitions* tersebut pada sebuah *website* yang disunting oleh Islam (2021).

AGE *Accepts value 18 to 56

EQUIVALENCE PARTITIONING		
Invalid	Valid	Invalid
≤ 17	18-56	≥ 57

Gambar 2.1 contoh Metode Equivalence Partitions

Berdasarkan gambar diatas, asumsinya terdapat kolom “Age” yang menerima nilai dari 18-56 tahun, maka Equivalence Partition akan membagi menjadi kelompok nilai valid dan tidak valid. Nilai valid yang dicontohkan ialah 18-56 tahun dan nilai tidak valid dicontohkan dengan nilai kurang dari ≤ 17 tahun & ≥ 57 tahun. Pada metode ini, peneliti diberikan kebebasan untuk melakukan percobaan berdasarkan kelompok nilai pada tiap *inputan-inputan* yang berbeda. Maka dari itu, metode *Equivalence Partitions* merupakan metode pendekatan yang cocok dalam penelitian pengujian *website e-commerce* studi kasus ini.

2.2.4 Software Testing Life Cycle (STLC)

Dituliskan pada artikel jurnal yang ditulis oleh Mithesh Parihar, Dr. Anu Bharti (2019) *Software Testling Life Cycle* (STLC) ialah tahap-tahap pengujian perangkat lunak yang dilakukan secara sistematis dan terencana. STLC pun juga masih dalam lingkup *Software Development Life Cycle* (SDLC) namun, penerapan STLC terbatas hanya pada tahap pengujian *software* (2019). Aktivitas-aktivitas yang dilakukan pada STLC ini berfokus untuk meningkatkan kualitas dari perangkat lunak. Adapun beberapa aktivitas yang dilakukan pada STLC, yakni :

A. Requirement Analysis

Pada tahap ini, dilakukan analisa terhadap *requirement software* yang diberikan untuk mengetahui detail *software*, fitur, dan fungsi yang akan dibangun dan melakukan validasi jika ada kekurangan.

B. Test Planning

Pada tahap kedua ini, persiapan rencana untuk melakukan pengujian berdasarkan *requirement analysis*, seperti menentukan *tools* yang akan digunakan, estimasi waktu, dan estimasi sumber daya.

C. Test Case Development

Pada tahap ketiga ini, ialah tahap *development* atau tahap pengembangan pengujian. Pada tahap ini berisi acuan dalam pengujian dengan melakukan membuat *test case*, data pengujian, membuat *script automation*.

D. Test Environment Setup

Pada tahap keempat ini, memastikan *environment test* seperti *server/client*, jaringan, dan data pengujian atau *test data* sehingga dapat berjalan sesuai dengan tujuan mereplikasi *environment* pengguna sistem.

E. Test Execution

Pada tahap kelima ini, pengujian dilakukan berdasarkan *planning test* dan *test case* yang telah disepakati pada tahapan sebelumnya. Fitur yang berjalan sesuai dengan *test requirement*, status fitur tersebut adalah berhasil/*passed* dan siap untuk masuk ke tahap *deployment*. Jika fungsi tidak berjalan sesuai requirement maka tergantung dari kategori *Bug/Errors* yang terjadi. Hasil dari pengujian akan dimasukkan kedalam laporan pengujian, jika ditemukan *Bug/Errors* maka akan diberikan kepada *developer* untuk segera diperbaiki dan diuji kembali oleh penguji

F. Test Cycle Closure

Pada tahap terakhir ini, penguji melakukan aktivitas pelaporan penyelesaian pengujian dan hasil pengujian pada studi kasus. Lalu, penguji juga melakukan kalkulasi evaluasi berdasarkan pengujian yang telah berhasil dilakukan. Dari hasil pengujian, temuan *bug* atau *error* akan dikategorikan menjadi seperti dibawah berikut:

- a. *Critical*, apabila ditemukan bug seperti ini mendesak sesegera mungkin untuk diperbaiki, karena dapat menimbulkan kerugian yang besar pada *website e-commerce* studi kasus.
- b. *High*, apabila ditemukan bug seperti ini perlu diperbaiki dengan cepat namun tidak mendesak, karena mempengaruhi fitur-fitur pada *website e-commerce* studi kasus.
- c. *Medium*, apabila ditemukan bug seperti ini perlu diperbaiki dengan segera namun tidak mendesak, karena memberikan dampak minimal pada fitur-fitur pada *website e-commerce* studi kasus.
- d. *Low*, apabila ditemukan bug seperti ini tidak perlu diperbaiki segera karena memiliki dampak yang sangat kecil pada jalannya operasional *website e-commerce* studi kasus.

2.2.5 Dokumentasi Pengujian

Diterjemahkan dari halaman *website* (Ltd, 2020), dokumentasi pengujian ialah suatu artefak dokumen yang dibuat sebelum atau selama pengujian suatu perangkat lunak. Dokumentasi pengujian sangat membantu penguji untuk merancang pengujian yang diperlukan, cakupan pengujian, kemajuan pelaksanaan, dll. Dokumentasi pengujian juga merupakan rangkaian dokumen yang lengkap yang dapat memungkinkan untuk mendeskripsikan perencanaan pengujian, pelaksanaan pengujian, serta hasil pengujian yang diambil dari aktivitas pengujian perangkat lunak.

Dokumentasi pengujian terdiri dari beberapa jenis dokumen, meliputi :

- a. Skenario Pengujian : item atau peristiwa dari system perangkat lunak yang dapat diverifikasi oleh satu atau lebih kasus pengujian
- b. Kasus Uji : sekelompok nilai masukan, prasyarat eksekusi, kondisi dan hasil pasca eksekusi yang diharapkan. Kasus uji dikembangkan dari skenario pengujian.
- c. Data uji : data yang ada sebelum pengujian dilakukan. Data uji digunakan untuk menjalankan kasus uji.
- d. Laporan kerusakan : laporan dokumentasi dari setiap kekurangan dalam system perangkat lunak yang gagal menjalankan fungsi yang diharapkan
- e. Laporan ringkasan pengujian : dokumen tingkat tinggi yang merangkum kegiatan pengujian dan mencatat hasil dari pengujian yang dilakukan.

2.2.6 Persentase

Berdasarkan buku Matrikulasi Matematika Dasar yang ditulis oleh Samuel Rex Riyadi, dkk (2019) Persentase merupakan suatu cara untuk menyatakan pecahan, kata persen berarti per seratus. Pada penelitian kali ini, perhitungan persentas digunakan untuk evaluasi kalkulasi keberhasilan pengujian yang dilakukan, adapun rumus persentase sederhana yang dikemukakan oleh Dra.Zulmiyetri (2019) ialah :

$$P = \frac{a}{N} \times 100\%$$

Dimana :

P = Persentase

a = Jumlah bagian

N = Jumlah Keseluruhan

100% = konstanta

Bab 3

Analisis dan Perancangan Sistem Pengujian

Perancangan sistem pengujian mengacu pada skema *Software Testing Life Cycle* (STLC) yang dituliskan pada artikel jurnal yang berjudul “Role of Software Testing Life Cycle In SDLC” yang dituliskan oleh Mithelseh (2019), artikel jurnal yang berjudul “Software Testing: A Review” dituliskan oleh Ritu (2013), dan laman *website guru99.com* (Guru99, 2021)

3.1 Data Penelitian

Data Data penelitian dibutuhkan untuk menunjang kegiatan penelitian agar mendapatkan hasil yang terbaik. Data-data penelitian yang akan digunakan pada penelitian ini berupa :

- a. Data utama adalah data yang didapatkan dari *webiste e-commerce* studi kasus yang diteliti langsung oleh peneliti. Data ini didapatkan dengan cara melakukan observasi. Bentuk data utama yang digunakan ialah *id element* dan *xpath element* pada tampilan website ecommerce studi kasus penelitian.
- b. Data pendukung adalah data yang didapatkan dari dokumentasi *website e-commerce*, alur proses sistem, dan alur proses bisnis studi kasus. Data ini diperoleh dengan cara melakukan wawancara narasumber tim *internal QA* pada studi kasus terkait. Bentuk data pendukung yang digunakan ialah *requirement system*, dan *data dummy* pengujian.

Agar mendapatkan data utama dan data pendukung seperti yang disebutkan diatas, didapatkan dengan metode berikut ini, yaitu :

- a. Observasi, atau pengamatan langsung. Metode ini merupakan pengumpulan data dengan cara pengamatan langsung pada studi kasus yang diteliti. Mencoba untuk membuat transaksi pada *website e-commerce* yang diteleti, memahami alur proses pada *website e-commerce*, mendapatkan data komponen-komponen atau *element* pada *website e-commerce*.

- b. Wawancara, metode ini merupakan pengumpulan data dengan melakukan dialog dengan narasumber terkait. Pada studi kasus ini, narasumber ialah tim internal QA. Dengan melakukan wawancara, diberikan penjelasan terkait *requirement system* serta data yang boleh digunakan pada pengujian *website e-commerce* studi kasus ini.

3.2 Requirement Analysis

3.2.1 Requirement System

Berdasarkan *Software Testing Life Cycle* (STLC), langkah pertama yang dilakukan ialah *Requirement Analysis*. Dimana pada siklus ini, dilakukan analisa terhadap kebutuhan data dan analisa *Requirement System* yang diberikan dengan cara mempelajari *Requirement System* seperti yang dilampirkan dibawah dan melakukan observasi atau pengamatan langsung pada *website e-commerce* studi kasus.

Sebelum memasuki tahap perencanaan pengujian atau *test planning* diperlukan *requirement system* pada *website e-commerce* studi kasus. *Requirement system* ini berhasil didapatkan sebagai data penelitian melalui wawancara narasumber yang bekerja pada perusahaan studi kasus. *Requirement system* inilah yang akan menjadi acuan analisa untuk melakukan pengujian, berikut adalah dokumen *requirement system* dalam bentuk tabel.

Adapun *requirement system* pada *website* studi kasus kali ini, adalah seperti berikut

Tabel 3.1 Tabel *Requirement System*

No	Requirement ID	Requirement Description
1	RS01	User can Login to Website
2	RS02	User can see the Display Product
3	RS03	User can input product to Checkout Page
4	RS04	User can choose available payment methods
5	RS05	User can see order history

Pada Tabel 3.1 Tabel *Requirement System*. Terdapat kolom “*Requirement ID*” yang merupakan *unique key* atau kunci unik untuk mengidentifikasi *requirement*

system. Lalu, terdapat kolom “*Requirement Description*” yang merupakan penjelasan atau deskripsi dari “*Requirement ID*”. Requirement yang ada akan menjadi obyektifitas pengujian dan diperluas menjadi sebuah *Scenario* Pengujian.

Berdasarkan *Requirement System* yang ada, telah berhasil dianalisa berdasarkan pengamatan langsung atau observasi pada *website e-commerce* studi kasus objek-objek apa saja yang bisa dilakukan pengujian dan hasilnya membagi *test scenario* atau skenario pengujian menjadi beberapa bagian. Berikut hasil rancangan *test scenario* dalam bentuk tabel.

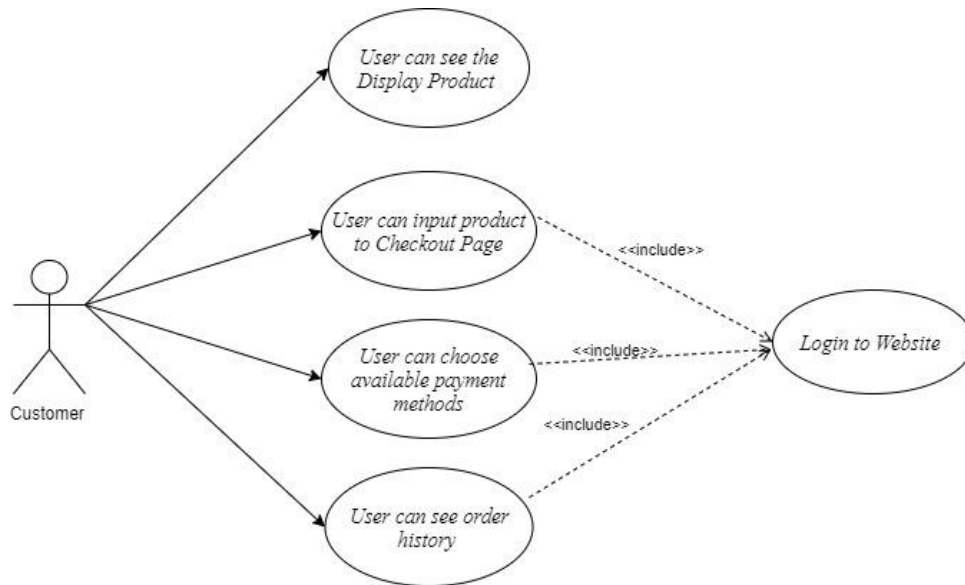
Tabel 3.2 Rancangan *Test Scenario*

No	Requirement ID	Scenario ID	Scenario Desc
1	RS01	TS01	Check the Login Functionality
2	RS02	TS02	Check the Homepage
		TS03	Check the Search Functionality
		TS04	Check the Category and Product Display Page
3	RS03	TS05	Check the Inquiry Customer Number
		TS06	Check the Checkout Page
4	RS04	TS07	Check the Payment Page functionality
		TS08	Check the Thank You Page Functionality
5	RS05	TS09	Check the Order History Functionality

Pada Tabel 3.2 Rancangan *Test Scenario*. Terdapat kolom “*RequirementID*” yang merupakan *unique key* atau kunci unik untuk mengidentifikasi *Requirement System* yang ada. Lalu, terdapat “*ScenarioID*” yang merupakan *unique key* atau kunci untuk mengidentifikasi *test scenario*. Kode “TS” berarti *test scenario* dan diikuti angka berdasarkan urutannya. *Scenario* disini merupakan hasil pengembangan dari *Requiremen System* dan *scenario* yang ada akan dikembangkan lagi menjadi *test case*.

3.2.2 Use Case Diagram

Berdasarkan *requirement system* diatas, maka dapat dirancang *use case diagram* atas apa yang dapat dilakukan oleh *User* atau *Customer* atau Konsumen pada *website e-commerce* studi kasus pada kali ini. Hal ini dapat digambarkan seperti dibawah berikut.



Gambar 3.1 Use Case Diagram *Customer*

Pada use case diagram diatas, *User* atau *Customer* atau Konsumen dapat melakukan melihat produk pada *website* tanpa harus melakukan *login* terlebih dahulu, sedangkan untuk melakukan memasukkan produk ke halaman *checkout*, memilih metode pembayaran, serta melihat riwayat pembelian atau daftar transaksi, *User* atau *Customer* atau Konsumen harus terlebih dahulu melakukan *login* pada *website e-commerce* studi kasus. Dengan berdasarkan *requirement system* dan *use case diagram* ini telah menggambarkan sistem menjadi lebih jelas dan dari sini akan dikembangkan lagi menjadi *test scenario* dan *test case* pada penelitian kali ini.

3.3 Test Planning

Pada tahap kedua ini, persiapan rencana untuk melakukan pengujian dengan berdasarkan hasil analisa terhadap *requirement system* telah menjadi sebuah *test scenario* atau skenario pengujian. Pada tahap perencanaan, akan ditentukan jenis pengujian, metode pengujian, tools yang akan digunakan, waktu dan sumber daya yang dibutuhkan. Hal ini dapat dilihat pada tabel berikut.

Tabel 3.3 Test Plan

no	Perencanaan	Keterangan
1	Jenis Pengujian Otomasi	<i>Black Box Testing</i>
2	Metode Pengujian Otomasi	<i>Equivalence Partitions</i>
3	Cakupan Pengujian Otomasi	<i>Check the Login Functionality (TS01), Check the Homepage (TS02), Check the Search Functionality (TS03), Check the Category and Product Display Page (TS04), Check the Inquiry Customer Number (TS05), Check the Checkout Page (TS06), Check the Payment Page functionality (TS07), Check the Thank You Page Functionality (TS08), Check the Order History Functionality (TS09)</i>
4	Cakupan Pengujian Metode Pengujian Equivalence Partitions	<i>Check the Login Functionality (TS01) & Check the Inquiry Customer Number (TS05)</i>
5	<i>Tools yang digunakan</i>	
	<i>Code Editor</i>	<i>Eclipse IDE For Java Developers 2020</i>
	Framework	<i>TestNG untuk Framework automation testing</i>
		<i>Selenium untuk Framework Pengujian Basis Website</i>
		<i>Extent Reports untuk laporan hasil pengujian berupa diagram serta Log pengujian</i>
6	Aplikasi <i>browser</i>	<i>Windows Desktop Web, Browser Google Chrome versi 90</i>
7	Pengguna Pengujian Otomasi	<i>Tim Internal Quality Assurance (QA)</i>
8	Kapan penggunaan Pengujian Otomasi	<i>1-2 kali dalam satu minggu atau setiap ada pembaharuan fitur pada Website</i>
9	Estimasi Waktu Pengerjaan dan Sumberdaya	<i>60 Hari Kerja dan 1 orang SDM</i>

3.4 Test Case Development

Pada tahap ketiga ini, ialah tahap *development* atau tahap pengembangan pengujian. Pada tahap ini berisi acuan dalam pengujian dengan melakukan membuat *test case* atau kasus uji yang akan dijelaskan pada sub-bab selanjutnya.

Berdasarkan *test scenario* yang telah ada, maka dapat dikembangkan lagi menjadi bagian yang lebih kecil yaitu *test case* atau kasus uji dalam pengujian automasi.

Tabel 3.4 Rancangan *Test Case*

No	ReqID	Req Desc	ScenarioID	Scnenario Desc	TestCase ID	TestCase Desc	Env Test
1	RS01	Login to Website	TS01	Check the Login Functionality	EP01	Login With Invalid Username minimum input and Valid Password	Production
2					EP02	Login With Invalid Username maximum input and Valid Password	
3					EP03	Login With Valid Username and Valid Password	
4	RS02	User can see the Display Product	TS02	Check the Homepage	TC01	Validate Logo is Display	Production
5					TC02	Validate icon Facebook is Display	
6					TC03	Validate icon Instagram is Display	
.							

.
n	n	n	n	n	n	n	n

Pada Tabel 3.4 Rancangan *Test Case*. Terdapat kolom “*TestCaseID*” yang merupakan *unique key* atau kunci unik untuk mengidentifikasi *Test Case*. Isi *ID TestCase* akan dibedakan menjadi 2, yaitu dengan kode “TC” yang berarti *Test Case* untuk menguji komponen serta fungsionalitas *website* dan kode “EP” untuk menguji dengan menggunakan Metode *Equivalence Partitions* pada pengujian automasi. Tabel ini dapat dilihat lebih lengkap pada bagian lampiran.

3.4.1 Test Case Development Otomasi Pengujian

Pada tahapan ini, akan dilakukan pembuatan test case atau kasus uji berdasarkan *Test Scenario* atau skenario pengujian yang telah dirancang. Pada pengujian automasi akan diberikan kode *test case* “TC” sebagai *Unique ID*. *Test case* otomasi pengujian akan melakukan pengujian pada tampilan antarmuka dan fungsionalitas fitur *website* studi kasus. Pada *test case* ini akan diimplementasi pada skenario *Check the Homepage* (TS02), *Check the Search Functionality* (TS03), *Check the Category and Product Display Page* (TS04), *Check the Checkout Page* (TS06), *Check the Payment Page functionality* (TS07), *Check the Thank You Page Functionality* (TS08), dan *Check the Order History Functionality* (TS09).

3.4.2 Test Case Development Metode Equivalence Partitions

Pada tahapan ini, akan dilakukan pembuatan *test case* atau kasus uji berdasarkan Test Scenario atau skenario pengujian yang telah dirancang. Pada pengujian automasi akan diberikan kode *test case* “EP” sebagai *Unique ID*.

Test case pengujian otomasi akan melakukan pengujian pada kolom-kolom masukkan pada *website* studi kasus. Seperti TS01, akan dilakukan pengujian *Equivalence Partitions* karena dapat menguji kolom masukkan berupa *Username* dan *Password*. Pada *test case* ini akan diimplementasi pada skenario pengujian *Check the Login Functionality* (TS01) & *Check the Inquiry Customer Number* (TS05).

Adapun kriteria – kriteria yang telah dipenuhi dalam suatu kasus uji sehingga dapat diterapkannya metode *Equivalence Partitions*, ialah :

- Terdapat kolom masukkan data yang dilakukan oleh *User* atau pengguna.
- Telah diketahui kondisi valid dan tidak valid pada kolom tersebut.
- Kolom masukkan data memiliki tipe data berupa *text* atau angka.

3.5 Test Environment Setup

Pada tahap keempat ini, memastikan *environment test* yang digunakan, pada pengujian kali ini, environment yang digunakan ialah *Production* atau *environment* yang sama yang digunakan untuk Konsumen melakukan transaksi pada *website ecommerce* studi kasus kali ini. Pengujian otomasi akan dilakukan berdasarkan *Test Plan* atau perencanaan pengujian pada sub bab sebelumnya, pada aplikasi *Desktop Web*, dengan *browser Google Chrome* versi 90 dan *Operating Systems Windows* 10 64 bit. Adapun yang perlu dipastikan lagi pada tahap ini ialah, penggunaan *test data*. Hal ini akan dijabarkan penggunaan data pengujian dengan tabel dibawah berikut.

Tabel 3.5 Test Data

Kasus uji atau <i>Test Case</i>	<i>Data Pengujian atau Test Data</i>
<i>Username Valid</i>	087881744704
<i>Username Invalid Min</i>	0
<i>Username Invalid Max</i>	0899999999999999
<i>Password Valid</i>	xxxxxxxxx
<i>Password Invalid Min</i>	x
<i>Password Invalid Max</i>	xxxxxxxxxxxxxxxxxxxxxxxxxxxx
<i>Customer Number BPJS Min</i>	1
<i>Customer Number BPJS Valid</i>	GEF00000
<i>Customer Number BPJS Max</i>	9999999999999999
<i>Customer Number Samsat Min</i>	9
<i>Customer Number Samsat Valid</i>	GEF0000
<i>Customer Number Samsat Max</i>	9999999999999999000000000
<i>Customer Number PDAM Min</i>	9
<i>Customer Number PDAM Valid</i>	GEF0000
<i>Customer Number PDAM Max</i>	9999999999999999000000000

<i>Customer Number PBB Min</i>	999
<i>Customer Number PBB Valid</i>	GEF0000
<i>Customer Number PBB Max</i>	9,999999999999999
<i>Customer Number PLN Min</i>	999
<i>Customer Number PLN Valid</i>	12312312312
<i>Customer Number PLN Max</i>	10000000100

3.6 Perancangan Evaluasi

Evaluasi dari penelitian dilakukan dengan cara seperti berikut ini :

- A. Kalkulasi hasil pengujian automasi, pada evaluasi ini melakukan kalkulasi persentase tingkat keberhasilan uji dari pengujian yang dilakukan. Perhitungan dilakukan dengan rumus seperti dibawah ini

$$\frac{\text{Jumlah Kasus Uji yang Berhasil}}{\text{Jumlah Keseluruhan Kasus Uji}} \times 100\%$$

- B. Kalkulasi hasil pengujian Equivalence Partition, pada evaluasi ini melakukan persentase tingkat keberhasilan pengujina metode equivalence partitions yang akan dilakukan. Perhitungan dilakukan dengan rumus seperti dibawah ini:

$$\frac{\text{Jumlah Kasus Uji Equivalence Partitions yang Berhasil}}{\text{Jumlah Keseluruhan Kasus Uji Equivalence Partitions}} \times 100\%$$

Bab 4

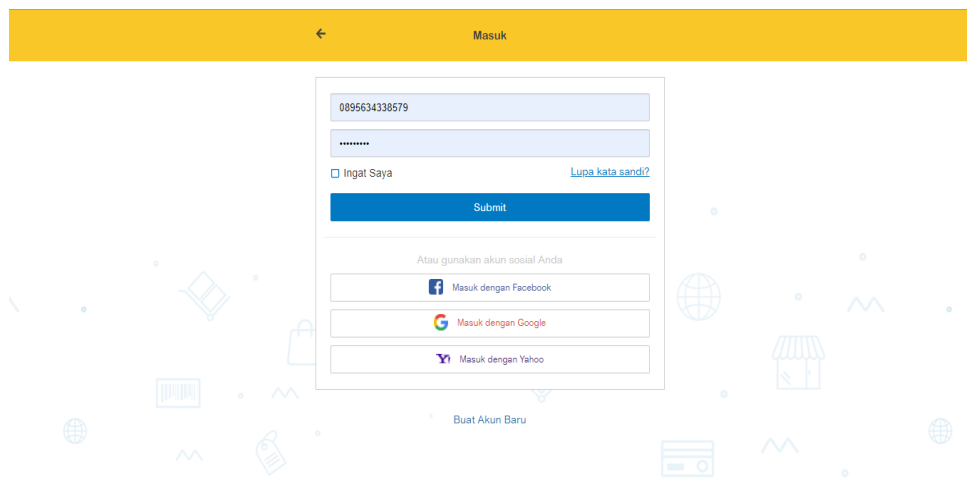
Penerapan dan Analisis Sistem Pengujian Otomasi

4.1 Penerapan Pengujian Automasi Website Studi Kasus

Pada pengujian automasi yang dilakukan oleh penulis seperti yang dituliskan pada bab sebelumnya pada tahap *test planning* (17), bahasa pemrograman yang digunakan ialah Java dengan Framework TestNG dan Selenium. Untuk melakukan pemrograman, penulis menggunakan Eclipse 2020 sebagai *code editor*. Pada penerapan pengujian automasi website studi kasus ini berdasarkan test scenario dan test case yang telah dirancang sebelumnya. Pada sub bab ini, akan dijelaskan halaman apa saja yang dilakukan pengujian.

4.1.1 Pengujian Automasi pada Halaman Login

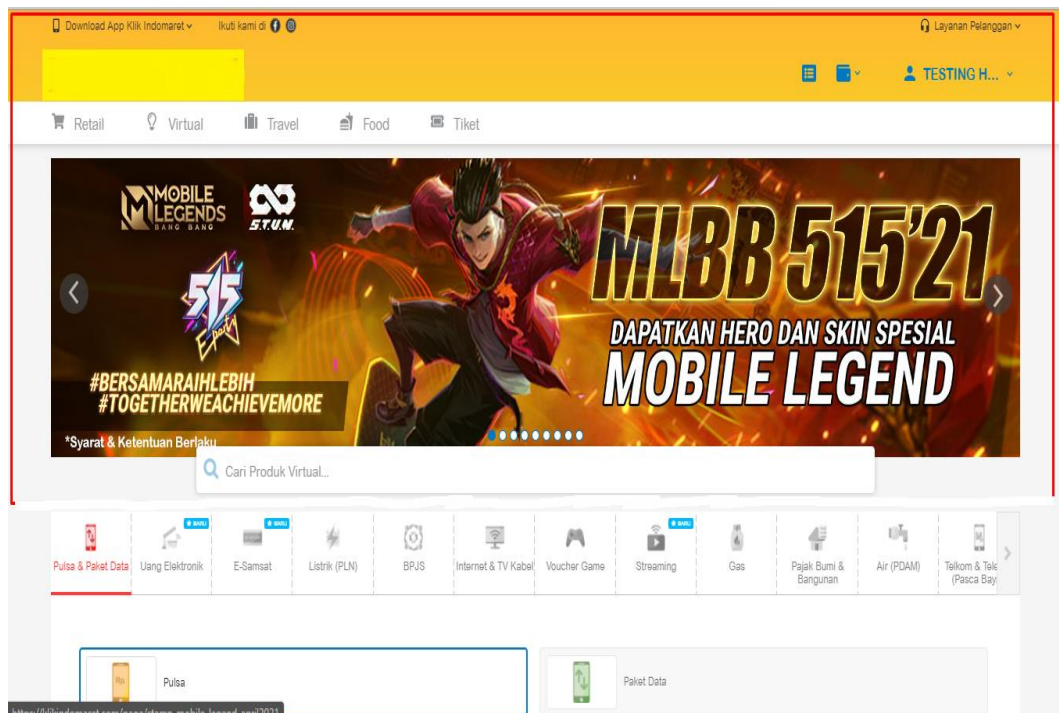
Berdasarkan Tabel 3.2 Rancangan *Test Scenario* pada User Requirement pertama ialah pengujian pada halaman Login. Pada halaman ini akan dilakukan pengujian Fungsionalitas Login dengan memiliki 5 *test case Equivalence Partitions*. Berikut halaman *login website* yang akan diuji.



Gambar 4.1 Pengujian Automasi pada Halaman Login

4.1.2 Pengujian Automasi pada Halaman Home

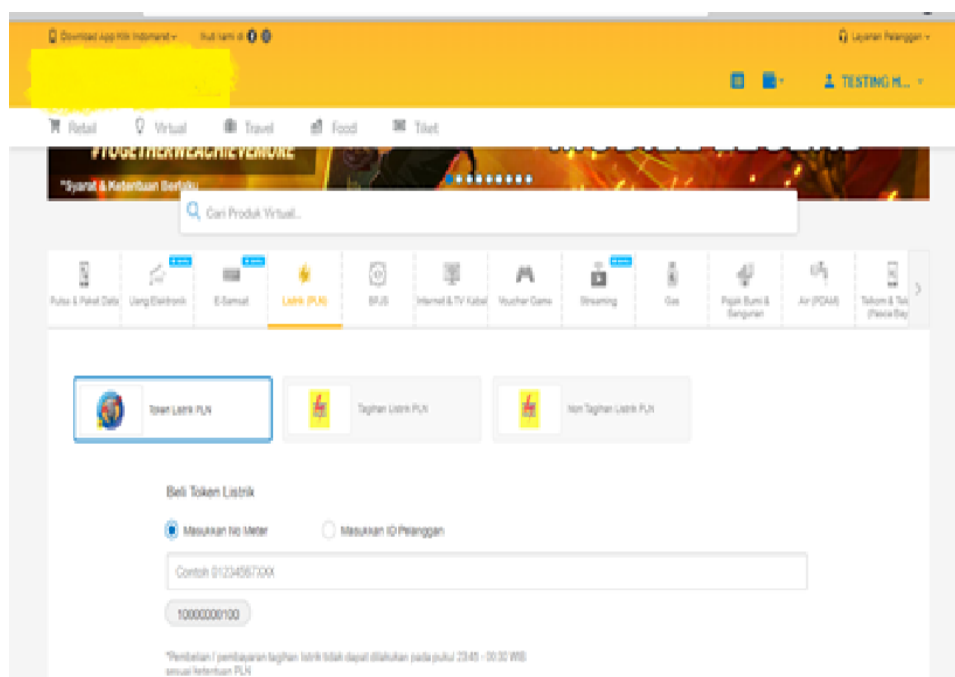
Berdasarkan Tabel 3.2 Rancangan *Test Scenario* pada User Requirement kedua ialah pengujian pada halaman Home pada website studi kasus. Pada halaman ini akan dilakukan validasi berdasarkan komponen-komponen yang ada pada halaman Home. Terdapat 10 kasus uji pada halaman ini. Berikut halaman Home website studi kasus yang akan diuji. Namun, pada halaman ini juga dilakukan pengujian untuk melakukan pengujian fungsionalitas search pada halaman Home dengan 6 kasus uji dilanjutkan validasi pada category yang ada pada halaman Home, dengan 16 kasus uji. Berikut halaman Home yang akan diuji.



Gambar 4.2 Pengujian automasi pada Halaman Home

4.1.3 Pengujian Automasi pada Halaman Produk

Berdasarkan Tabel 3.2 Rancangan *Test Scenario* pada User Requirement ketiga ialah User dapat memasukkan produk ke halaman Checkout. Pada halaman ini akan dilakukan pengujian menggunakan Equivalence Partitions dengan category produk PLN, BPJS, Samsat, PDAM, dan Pajak Bumi Bangunan (PBB) dengan jumlah total 15 kasus uji pada. Berikut adalah contoh halaman category PLN dengan produk Token Listrik.



Gambar 4.3 Pengujian automasi pada halaman produk

4.1.4 Pengujian Automasi pada Halaman *Checkout*

Berdasarkan Tabel 3.2 Rancangan *Test Scenario* pada *User Requirement* ketiga ialah *User* dapat memasukkan produk ke halaman *Checkout*. Pada halaman ini dilakukan pengujian validasi pada komponen-komponen yang ada pada halaman *Checkout* terdapat kurang lebih 8 kasus uji untuk pengujian ini. Berikut contoh halaman *Checkout* yang akan diuji.

Transaksi pada jam 23:30 - 00:30 WIB tidak dapat dilakukan sesuai dengan jadwal pemeliharaan sistem pihak operator.

1. Konfirmasi Pesanan

Produk Virtual

Produk Virtual	Qty	Price
Token Listrik PLN	Qty: 1	Rp 20.000

Total: Rp 20.000

[Lanjut ke Pembayaran](#)

Produk Virtual: Rp 20.000

Biaya Layanan: Rp 2.500

Subtotal: Rp 22.500

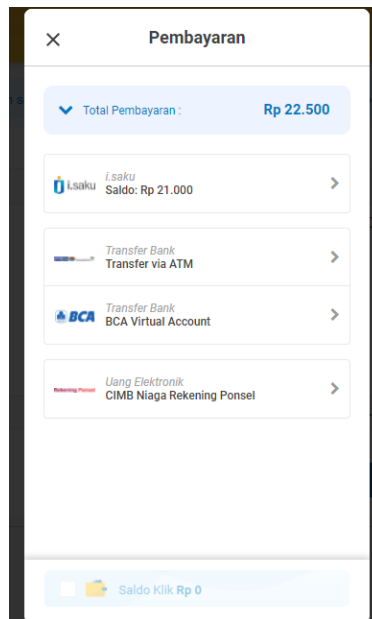
Total Pembayaran: Rp 22.500

Kode i-kupon [Gunakan](#)

Gambar 4.4 Pengujian automasi pada halaman *Checkout*

4.1.5 Pengujian Automasi pada Halaman Pembayaran

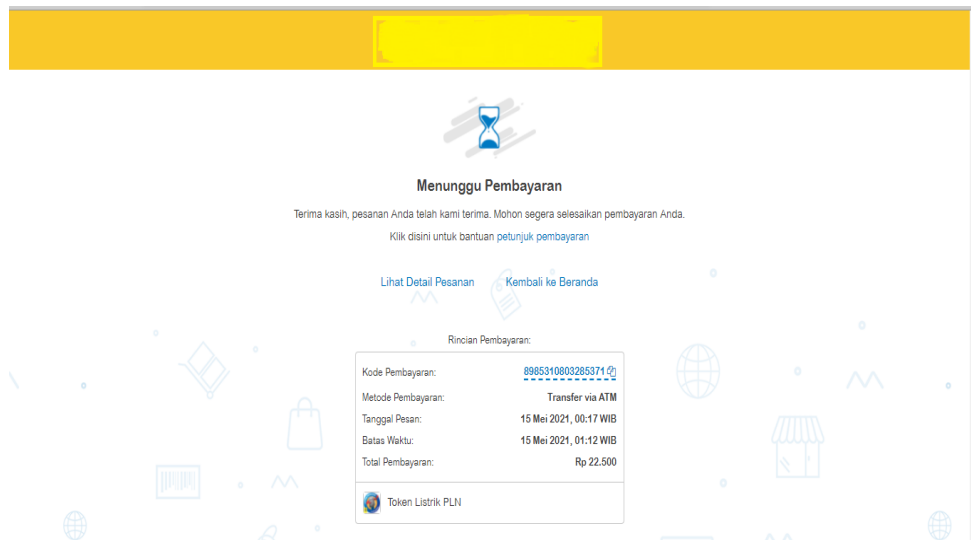
Berdasarkan Tabel 3.2 Rancangan *Test Scenario* pada *User Requirement* keempat ialah *User* dapat memilih pembayaran yang tersedia. Pada halaman ini berisi pilihan metode-metode pembayaran yang disediakan oleh *website* studi kasus. Adapun 6 kasus uji yang dapat dilakukan pada pengujian halaman ini. Berikut contoh halaman pembayaran



Gambar 4.5 Pengujian automasi pada halaman pembayaran

4.1.6 Pengujian Automasi pada Halaman *Thankyou*

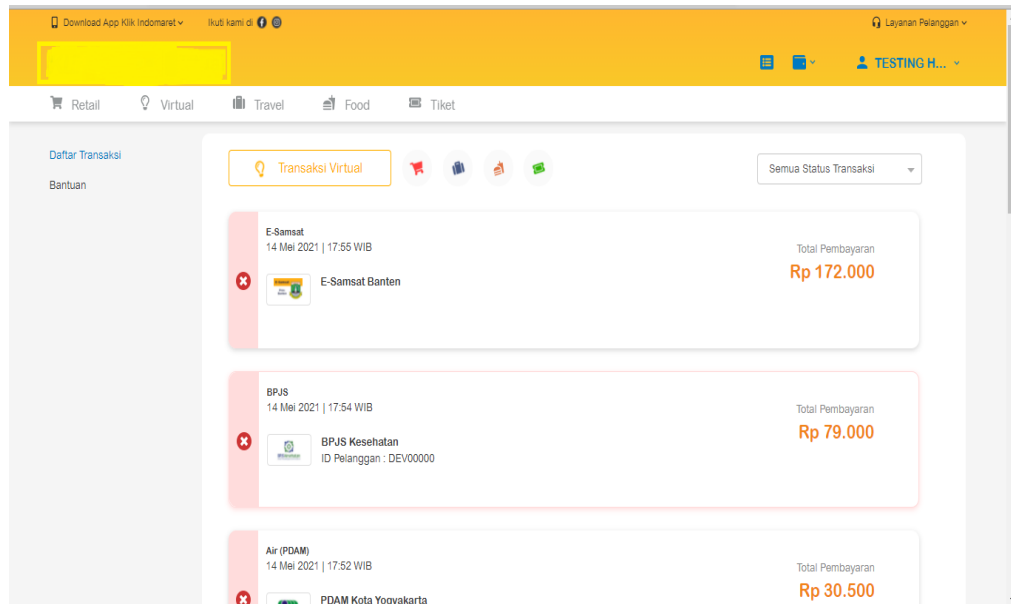
Berdasarkan Tabel 3.2 Rancangan *Test Scenario* pada *User Requirement* keempat ialah *User* dapat memilih pembayaran yang tersedia dan telah melakukan pembayaran maka akan diarahkan ke halaman *Thankyou*, pada halaman ini berisi informasi-informasi pesanan yang dilakukan oleh *User*. Pada halaman *Thankyou* dilakukan pengujian automasi validasi pada komponen-komponen yang ada. Berikut contoh halaman *Thankyou*



Gambar 4.6 Pengujian automasi pada halaman *Thankyou*

4.1.7 Pengujian Automasi pada Halaman *Order History*

Berdasarkan Tabel 3.2 Rancangan *Test Scenario* pada *User Requirement* keempat ialah *User* dapat melihat transaksi atau pesanan yang pernah dilakukan. Pada halaman ini dilakukan pengujian validasi berdasarkan komponen-komponen yang ada. Adapun 23 kasus uji yang dapat dilakukan pada halaman ini. Berikut contoh halaman *order history*



Gambar 4.7 Pengujian automasi pada halaman riwayat transaksi

4.2 Penerapan Program Pengujian Automasi

Penerapan program pengujian automasi menggunakan konsep *Page Object Model* (POM). Dengan menggunakan konsep ini, akan dibagi menjadi *class Page* yang berisi komponen element pada website yang akan diuji beserta metodenya dan *class Test* yang berisi pengujiannya dan *class Page Factory* untuk inisiliasi objek halaman dan *instace* objek halaman itu sendiri.

4.2.1 Penerapan program Class Page

Dengan menerapkan konsep *Page Object Model* (POM), pada *class HomePageVirtual* ini menyimpan *element-element* pada komponen *website* yang diambil dari *xpath*, *id*, ataupun *css selector* dari hasil observasi penelitian. Pada kelas ini, terdapat juga metode menggunakan *framework Selenium* yang digunakan untuk melakukan validasi komponen *element website*. Hal ini dapat dilihat pada potongan *program* dibawah berikut.

```
public class HomePageVirtual extends LoginWithPageFactory {
    public WebDriver driver;

    @FindBy(xpath="//*[@id=\"siteHeader\"]/div[1]/div/div[2]/div[1]/div/a[1]")
    public WebElement logoIdm;
    @FindBy(id="downloadApp")
    public WebElement downloadApp ;

    public void logoIsDisplay() {
        try {
            waitElementClickable(logoIdm);
            Boolean status = logoIdm.isDisplayed();
            Assert.assertTrue(status);
        } catch (NoSuchElementException e) {
            e.printStackTrace();
        }
    }

    public void setName(String strUserName) {
        try {
            userNameVirtual.sendKeys(strUserName);
            Thread.sleep(2);
        } catch (InterruptedException e) {
            e.printStackTrace();
        }
    }
}
```

Kode Program 1 *Class Home Page Virtual*

4.2.2 Penerapan program Class Test

Dengan menerapkan konsep *Page Object Model* (POM), pada *class HomePageTest* ini berisi metode-metode pengujian yang dipanggil dari *Page Object Class* yaitu *HomePageVirtual*. Pada *class* ini, menerapkan *framework TestNG* dengan ditandainya *annotations* atau anotasi “@Test” yang menandakan bahwa metode itu ialah metode pengujian . Hal ini dapat dilihat pada potongan program dibawah berikut.

```
public class HomePageTest extends driverTest {
    HomePageVirtual objHomePage;
    LoginPage objLogin;
    public String testUrl;

    @Test
    public void UITestLogo() {
        VirtualTestReports.getTest().log(Status.INFO, "TS02-TC01
        Validate Logo Is Display");
        try {
            objHomePage = new HomePageVirtual(driver);
            objHomePage.logoIsDisplay();
            Thread.sleep(10);
        } catch (InterruptedException e) {
            e.printStackTrace();
        }
    }

    @Test
    public void LoginInvalidUsernameMin() throws InterruptedException{
        VirtualTestReports.getTest().log(Status.INFO, "TS01-EP01 -
        Login Invalid Min Username Valid Pass");
        try {
            objLogin = new LoginPage(driver);
            objLogin.setUserName("");
            objLogin.setPassword("#validpassword");
            objLogin.clikcLogin();
            objLogin.AlertOnLogin();
            Thread.sleep(20);
        } catch (InterruptedException e) {
            e.printStackTrace();
        }
    }
}
```

Kode Program 2 Class Home Page test

4.2.3 Penerapan Program Page Factory

Dengan menerapkan konsep *Page Object Model (POM)*, terdapat *Class Page Factory* yang digunakan untuk ekstensi ke *Object Page* dengan dapat digunakan dalam berbagai cara serta dapat menginisiasi *element web* yang di definisikan pada *Object Page*. Pada penerapan *program page factory* menggunakan *framework Selenium*. Berikut potongan *program* yang ada pada *Page Factory*.

```
public class LoginWithPageFactory {
    public WebDriver driver;

    public LoginWithPageFactory(WebDriver driver){
        this.driver = driver;
        //This initElements method will create all WebElements
        PageFactory.initElements(driver, this);
    }

    public void waitElement(){
        driver.manage().timeouts().implicitlyWait(10,
        TimeUnit.SECONDS);
    }

    public void currentUrl() {
        driver.getCurrentUrl();
    }

    public void waitElementClickable (WebElement element) {
        WebElement firstResult = new WebDriverWait(driver, 60)
        .until(ExpectedConditions.elementToBeClickable(element));
    }

    public void PaymentCenterFrame() {
        driver.switchTo().frame("paymentCenterFrame");
    }

    public void defaultContent() {
        driver.switchTo().defaultContent();
    }
}
```

Kode Program 3 Class Page Factory

4.2.4 Penerapan Program Driver Test

Pada *class Driver Test* berfungsi sama dengan *class Page Factory*, yang dimana pada *class Page Factory* ekstensi ke *Object Page* sedangkan *Driver Test* ekstensi ke *Class Test*. Pada potongan program ini, *driver test* berisi *set property* yang harus dilakukan untuk memulai pengujian, dimana menggunakan *WebDriver* google chrome. Fungsi dari *WebDriver* sebagai kerangka automasi pada suatu Web melalui suatu browser. Pada penerapan program ini, menggunakan *framework TestNG*, hal ini ditandai dengan penggunaan *annotations* atau anotasi “@BeforeSuite” dan “@AfterSuite” yang menandakan apa yang akan dilakukan sebelum pengujian dan apa yang akan dilakukan setelah pengujian. Berikut potongan *program* yang ada pada *Driver Test*.

```
public class driverTest {
    public String testUrl ;
    public WebDriver driver;

    @BeforeSuite
    public void startApps() {
        System.setProperty("webdriver.chrome.driver",
            "D:\\browserdrivers\\chromedriver90.exe");
        testUrl = "https://virtual.klikindomaret.com/";
        driver = new ChromeDriver();
        driver.manage().timeouts().implicitlyWait(10,
            TimeUnit.SECONDS);

        //maximize window
        driver.manage().window().maximize();
        // driver.navigate().to(testUrl);
        driver.get(testUrl);
    }

    @AfterSuite
    public void closeApp() {
        driver.quit();
    }
}
```

Kode Program 4 Driver Test

4.2.5 Penerapan Program Test . eXtensible Markup Language (XML)

Dengan menerapkan konsep *Page Object Model (POM)*, dibutuhkan file *TestNG* berformat *eXtensible Markup Language (XML)*, file *TestNG .XML* ini digunakan untuk membantu dalam mengatur pengujian yang akan dilakukan. File ini berisi nama metode pengujian dari *class Test*.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE suite SYSTEM "https://testng.org/testng-1.0.dtd">
<suite name="Suite">
  <listeners>
    <listener class-name="PageFactory.VirtualListener" />
  </listeners>
  <test thread-count="5" name="Tests" group-by-
instances="true">
    <classes>
      <class name="Tests.HomePageTest">
        <methods>
          <include name="goToLoginPage"> </include>
          <include name="testLogin"> </include>
          <include name="UITestLogo"> </include>
          <include name="verifyPageTitle"> </include>
          <include name="UITestFacebook"> </include>
          <include name="UITestInstagram"> </include>
          <include name="UITestDownload"> </include>
          <include name="UITestLayananPelanggan"> </include>
          <include name="searchSamsat"> </include>
          <include name="searchBPJS"> </include>
          <include name="searchPBB"> </include>
          <include name="searchPDAM"> </include>
          <include name="validateCategoryPulsa"> </include>
          <include name="validateCategoryEmoney"> </include>
          <include name="validateCategoryESasmat"> </include>
          <include name="validateCategoryPLN"> </include>
        </methods>
      </class>
    </classes>
  </test> <!-- Tests -->
</suite> <!-- Suite -->
```

Kode Program 5 File Test.XML

4.2.6 Penerapan *Program Master Test eXtensible Markup Language (.XML)*

Program Master Test.XML bertujuan sama dengan *program Test.XML* yang membantu dalam mengatur pengujian yang akan dilakukan. Namun, pada program ini, berisi kumpulan-kumpulan *Test.XML* yang akan dijalankan berurutan sesuai yang ada di *program* dibawah ini.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE suite SYSTEM "https://testng.org/testng-1.0.dtd">
<suite name="Master Suite">
  <listeners>
    <listener class-name="PageFactory.VirtualListener" />
  </listeners>
  <suite-files>
    <suite-file path="./LoginTest.xml"> </suite-file>
    <suite-file path="./HomepageTest.xml"> </suite-file>
    <suite-file path="./PLNTest.xml"> </suite-file>
    <suite-file path="./PbbTest.xml"> </suite-file>
    <suite-file path="./PDAMTest.xml"> </suite-file>
    <suite-file path="./BpjsTest.xml"> </suite-file>
    <suite-file path="./SamsatTest.xml"> </suite-file>
  </suite-files>
</suite>
```

Kode Program 6 *Master Test.XML*

4.3 Test Execution

Pada tahap kelima STLC ini, pengujian dilakukan berdasarkan tahapan sebelumnya dan ketika pengujian telah berhasil dieksekusi atau dijalankan maka akan mendapatkan laporan atau *reports* hasil pengujian. Pada tahapan *test execution* ini akan dijelaskan hasil realisasi atau implementasi akhir test case yang digunakan serta hasil atau report pengujian.

4.3.1 Test Execution berdasarkan Tabel *Test Case*

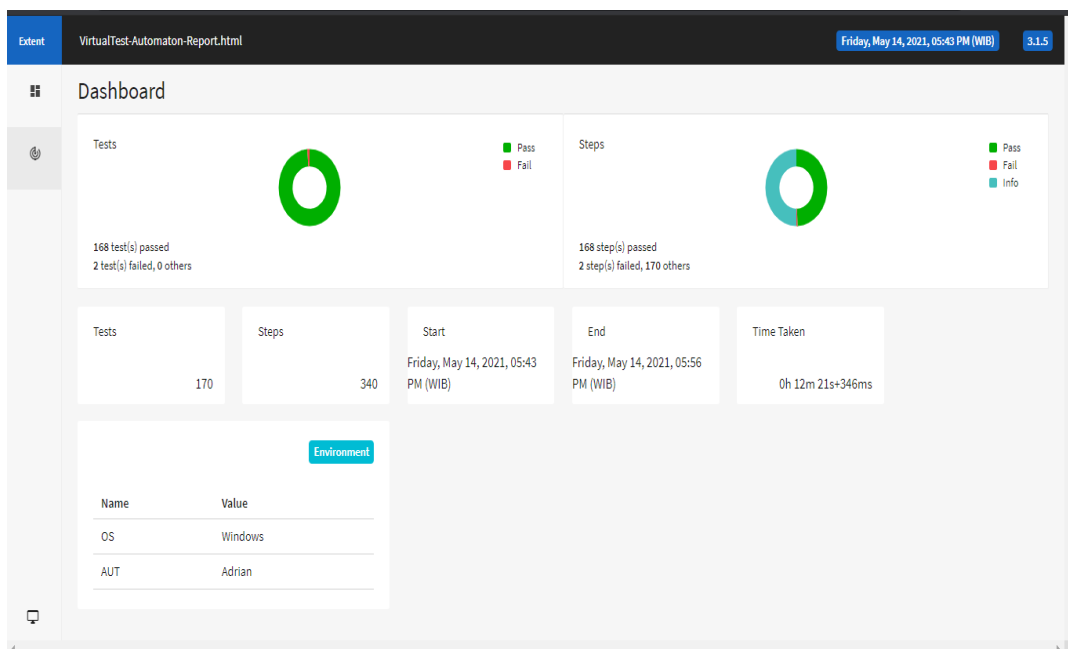
Berdasarkan rancangan test case yang ada pada bab sebelumnya (**Error! Bookmark not defined.**) telah berhasil dilakukan penerapan pemograman pengujian automasi dengan total 170 *tests* yang harus dieksekusi dari program, terdiri dari 81 *test case* dengan kode “TC”, 32 *test case* Metode *Equivalence Partitions* dengan kode “EP”, dan 57 *test case additional* yang berfungsi untuk membantu jalannya pengujian automasi. *Test case additional* ini ditandai dengan *background* berwarna hijau. Test case additional tidak memiliki “*Scenario ID*” dan “*Test Case ID*” sehingga tidak termasuk dalam kalkulasi evaluasi hasil keberhasilan pengujian automasi dan keberhasilan penerapan Metode *Equivalence Partitions*. Berdasarkan tabel hasil penerapan terdapat juga dua *test case* yang ditandai dengan *background* berwarna merah, artinya bahwa *test case* tersebut gagal. Penyebab kegagalan dari *test case* akan dibahas lebih lanjut pada sub-bab analisis pengujian. Berikut adalah contoh tabel hasil penerapan pengujian automasi, tabel lebih lengkap dapat dilihat pada lampiran.

Tabel 4.1 Hasil Penerapan Pengujian Otomasi

No	Scenario ID	Scenario Desc	TC ID	TC Desc	Expected Result	Actual Result
1	TS01	Check the Login Functionality	EP01	Login With Invalid Username minimum input and Valid Password	Passed	Passed
2			EP02	Login With Invalid Username maximum input and Valid Password	Passed	Passed
3	TS03	Check the Search Functionality	TC13	Validate search with keyword "Samsat Jatim"	Passed	Passed
4			TC14	Validate search with keyword "BPJS"	Passed	Failed
5			TC15	Validate search with keyword "PBB Jakarta"	Passed	Failed
6			TC16	Validate search with keyword "PDAM Yogyakarta"	Passed	Passed
7	NULL	Additional Case	NULL	Go To PLN Test for PLN Test	Passed	Passed
8				Go To Token Listrik for PLN Test	Passed	Passed
9				setDenom PLN for PLN Test	Passed	Passed
10				setNominal PLN for PLN Test	Passed	Passed
...			
...			
170				Daftar Trx Test on Order History Page for Samsat Test	Passed	Passed

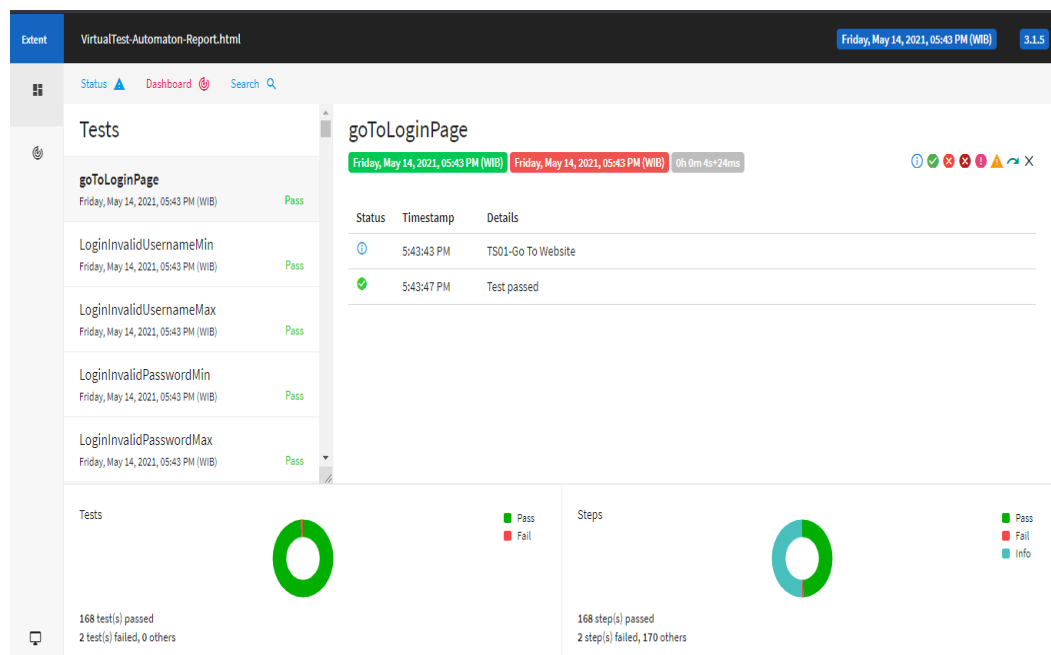
4.3.2 Test Execution berdasarkan Dashboard Hasil Pengujian

Berdasarkan hasil pengujian yang telah dilakukan, berhasil dieksekusi total 170 test case yang terdiri dari test case dengan kode “TC”, *test case* metode *Equivalence Partitions* dengan kode “EP” dan *additonal case* yang digunakan untuk membantu jalannya pengujian automasi. Penerapan dashboard hasil pengujian dengan menggunakan *library* tambahan yaitu, *Extent Reports*. Dengan adanya dashboard, visualisasi hasil pengujian dapat mudah dimengerti dan interaktif dengan adanya diagram *pie charts*. Dapat dilihat dari gambar dashboard hasil dibutuhkan waktu kurang lebih ialah 12 menit pada kolom “Time Taken” dan hasil 168 test berhasil sedangkan 2 test case yang gagal pada kolom “Tests” serta *diagram pie charts*.



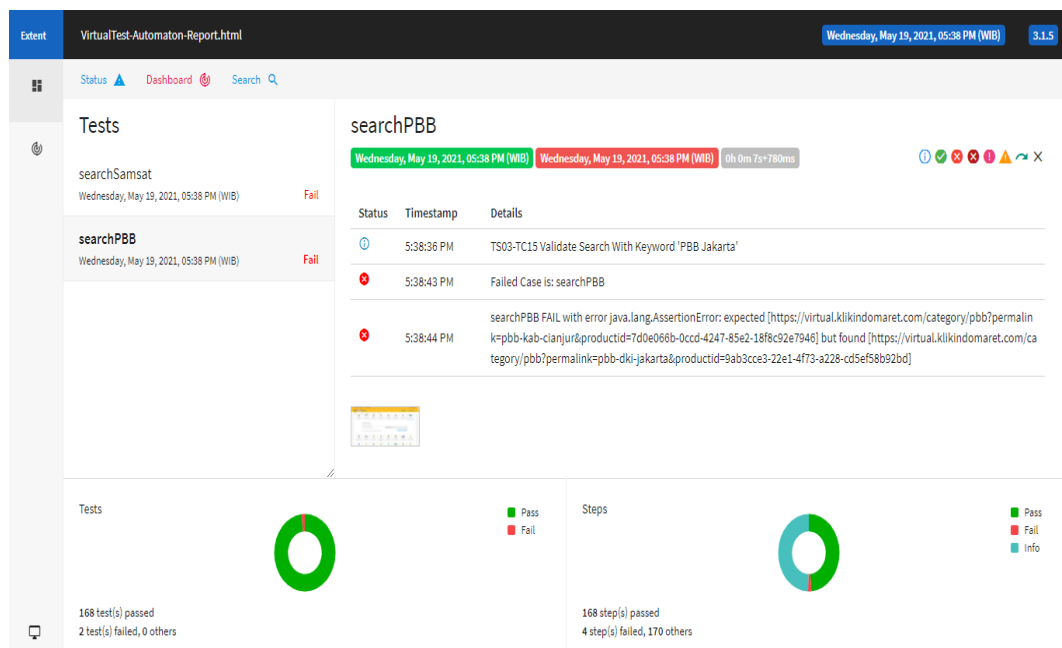
Gambar 4.8 Dashboard hasil pengujian

Pada gambar hasil pengujian dengan menggunakan ExtentReports penguji dapat melihat visualiasi hasil pengujian berurutan sesuai dengan *file Test.XML* dan *Master Test.XML* yang telah dieksekusi, pada hasilnya terdapat 168 *Test Case* atau kasus uji yang berhasil hal ini ditandai dengan keterangan status “*Pass*” atau “*Test Passed*” pada nama metode test yang dieksekusi. Visualisasi didukung dengan adanya diagram *pie charts* dan terdapat kolom “Status”, “Timestamp”, dan “Details” yang berisi info hasil Test yang telah dieksekusi.



Gambar 4.9 *Dashboard* Hasil Pengujian Berhasil

Dengan menggunakan *library Extent Reports* pengujian dapat juga melakukan *filter* berdasarkan hasil *test*, yaitu “*Success*” atau “*Failed*”. Pada hasil *report* dibawah ini merupakan hasil pengujian yang tidak berhasil atau *Failed* pada test *searchSamsat*(TS03-TC13) dan *Search PBB*(TS03-TC14) yang merupakan kelompok pengujian kasus uji dengan kode “TC” pada “*TestCaseID*” yang sudah dijelaskan pada sub-bab sebelumnya. Dengan menggunakan dashboard seperti gambar dibawah berikut, pengujian dapat melihat letak kesalahan berupa screenshot halaman website yang terjadi kegagalan saat pengujian dan menampilkan pesan errors pada kolom Details. Berdasarkan hasil pengujian yang diketahui terjadi 2 test case dengan kode “TC” maka akan dilakukan analisis hasil dan akan mempengaruhi tingkat keberhasilan dari penerapan pengujian automasi pada penelitian website studi kasus kali ini. Hal ini akan dijelaskan lebih lanjut pada sub-bab berikutnya.



Gambar 4.10 Dashboard hasil pengujian gagal

4.4 Test Cycle Closure

Pada tahap terakhir pada STLC, penguji melakukan aktivitas pelaporan penyelesaian pengujian dan hasil pengujian pada studi kasus. Lalu, penguji juga melakukan analisa hasil pengujian serta kalkulasi evaluasi berdasarkan pengujian yang telah berhasil dilakukan. Namun, karena pada penelitian kali ini tidak ditemukannya *bug* atau *error* dalam website studi kasus, maka tidak ada klasifikasi bug. Namun ditemukannya kekurangan dari pemograman yang menyebabkan kegagalan pengujian akan dijelaskan pada sub-bab berikut.

4.4.1 Analisa Hasil Otomasi Pengujian dan Pengelompokkan Bug

Berdasarkan tabel Tabel 4.1 Hasil Penerapan Pengujian Otomasi pada sub-bab sebelumnya, terdapat 168 *test case* atau kasus uji yang berhasil dieksekusi pada *website* studi kasus kali ini. Adapun temuan 2 *test case failed* yaitu “TC14” dan “TC15”, dari 6 test case yang dieksekusi pada scenario “*Check the Search Functionality*” atau pada “ScenarioID” “TS03”. Temuan *test case failed* ini bukan merupakan *Bug*, hal ini dibuktikan ketika penguji melakukan pengujian secara manual pada “TC14” dan “TC15” berfungsi dengan baik. Hal ini bisa terjadi karena kesalahan pada pemograman pengujian automasi dan temuan gagal ini biasa disebut *flickering* atau *Flash of Original Content*. Untuk menangani hal ini, bisa dapat melakukan eksekusi ulang ataupun perubahan pada program pengujian automasi. Dengan demikian, bahwa temuan kegagalan pada 2 test case tersebut bukan merupakan bug, maka *website e-commerce* pada studi kasus kali ini bebas dari bug atau errors sehingga aman digunakan untuk Konsumen umum berbelanja.

4.4.2 Kalkulasi Evaluasi Hasil Otomasi Pengujian

Pada evaluasi ini terdapat 2 hal yang akan dievaluasi yaitu keberhasilan kasus uji dan keberhasilan penerapan metode *Equivalence Partitions*. Hal ini akan dijabarkan lebih lengkapnya melalui perhitungan seperti dibawah berikut:

- a. Pada evaluasi ini melakukan kalkulasi persentase tingkat keberhasilan uji dari pengujian yang dilakukan. Evaluasi hasil pengujian dilakukan dengan rumus seperti dibawah ini

$$\frac{\text{Jumlah Kasus Uji yang Berhasil}}{\text{Jumlah Keseluruhan Kasus Uji}} \times 100\%$$

Berdasarkan rumus diatas, maka dapat dilakukan perhitungan seperti:

$$\frac{79 \text{ Kasus Uji yang berhasil}}{81 \text{ Jumlah Kasus Uji}} \times 100\% = 97.53\%$$

- b. Kalkulasi keberhasilan penerapan Metode *Equivalence Partitions*, kalkulasi evaluasi keberhasilan penerapan Metode *Equivalence Partitions* dilakukan dengan rumus seperti dibawah ini

$$\frac{\text{Jumlah Kasus Uji Equivalence Partitions yang Berhasil}}{\text{Jumlah Keseluruhan Kasus Uji Equivalence Partitions}} \times 100\%$$

Berdasarkan rumus diatas, maka dapat dilakukan perhitungan seperti

$$\frac{32 \text{ Kasus Uji Equivalence Partitions yang berhasil}}{32 \text{ Jumlah Kasus Uji Equivalence Partitions}} \times 100\% = 100\%$$

Dengan berdasarkan hasil kalkulasi evaluasi hasil pengujian dari 2 rumus diatas dengan mendapatkan hasil 97,53% dan 100% maka, dapat disimpulkan bahwa penerapan metode *equivalence partitions* pada pengujian automasi website pada penelitian studi kasus ini berhasil dan website ecommerce pada penelitian ini juga bebas dari *bug* atau *errors* dapat digunakan oleh Konsumen umum.

4.5 Kelebihan dan Kekurangan Penerapan Metode Equivalence Partitions pada Pengujian Otomasi

4.5.1 Kelebihan Penerapan Metode Equivalence Partitions dan Pengujian Otomasi

1. Pengujian menggunakan metode Equivalence Partitions telah mewakili kondisi tidak valid dan kondisi valid pada kolom masukkan data pada website studi kasus.
2. Pengujian automasi dapat dieksekusi sesuai keperluan penguji manual ketika adanya pembaharuan fitur pada website atau berkala sesuai kebutuhan.
3. Pengujian automasi dapat memberikan visualisasi laporan atau *report* hasil pengujian.
4. Pengujian terukur dengan kasus uji yang lebih jelas.
5. Pengujian automasi menggunakan sumber daya manusia lebih sedikit.

4.5.2 Kekurangan Penerapan Metode Equivalence Partitions dan Pengujian Otomasi

1. Tidak semua kasus dapat diuji secara otomatis.
2. Data uji valid belum mencerminkan keadaan sesungguhnya, karena menggunakan data pengujian dari hasil wawancara.

Bab 5 **Kesimpulan**

5.1 Kesimpulan

Berdasarkan hasil penerapan dan analisis sistem pengujian automasi yang telah diterapkan yaitu penerapan metode *Equivalence Partitions* dalam pengujian automasi, maka dapat disimpulkan hal-hal sebagai berikut:

- d. Penerapan pengujian automasi pada website studi kasus dapat diterapkan dengan keberhasilan 97.53%.
- e. Penerapan metode *Equivalence Partitions* dalam pengujian automasi dapat diterapkan dengan keberhasilan 100% .
- f. Penerapan pengujian automasi dapat dieksekusi sesuai keperluan penguji ketika adanya pembaharuan fitur pada *website* atau berkala sesuai kebutuhan.
- g. Pengujian automasi mengurangi kebutuhan sumber daya manusia serta dapat mengurangi *human errors* yang dilakukan ketika dilakukan pengujian secara manual.
- h. Mengurangi kerugian perusahaan akibat temuan *bug/errors* oleh Konsumen ketika melakukan transaksi karena dibuktikan berdasarkan evaluasi keberhasilan pengujian automasi website studi kasus bebas dari *bug/errors*.
- i. Pengujian automasi dapat memberikan visualisasi laporan atau *report* hasil pengujian.

5.2 Saran

Dalam penelitian yang telah dilakukan sistem pengujian automasi belum mampu untuk dapat melakukan beberapa hal, yaitu :

- a. *Performance testing API* yang ada pada *website e-commerce* studi kasus.
- b. *Database testing* sehingga dapat memastikan validasi data.
- c. Dapat menerapkan metode *pairwise*, untuk dapat menguji berbagai macam input seperti *checkbox*, *radio button*, *list box*, *text box*, dsb. Namun kekurangannya, diperlukan pertimbangan dan pengembangan *test case* yang tepat.

DAFTAR PUSTAKA

- Adella Rosalina, A. A. (2020). Pengujian Black Box pada Sistem Informasi Penjualan HI Shoe Store Menggunakan Teknik Equivalence Partitions. *Jurnal Informatika Universitas Pamulang*, 26-29.
- Adella Rosalina, A. G. (2020). Pengujian Black Box pada Sistem Informasi Penjualan HI Shoe Store Menggunakan Teknik Equivalence Partitions. *Jurnal Informatika Universitas Pamulang*, 26-29.
- Adi Krismadi, A. F. (2019). Pengujian Black Box berbasis Equivalence Partitions pada Aplikasi Seleksi Promosi Kenaikan Jabatan. *Jurnal Teknologi Sistem Informasi dan Aplikasi*, 155-161.
- Dhega Febiharsa, I. M. (2020). Uji Fungsionalitas (BlackBox Testing) Sistem Informasi Lembaga Sertifikasi Profesi (LSP) Batik dengan Appperfect WebTest dan Uji Pengguna. *Journal of Informatics Education*, 117-126.
- Dra. Zulmiyetri, M. . (2019). *Penulisan Karya Ilmiah*. Jakarta: Kencana.
- Galin, D. (2004). *Software Quality Assurance: From Theory to Implementation* . Pearson Education Inc.
- Handy, J. S. (2014). Aplikasi Pengujian White-Box IBI Online Judge. *Jurnal Informatika dan Bisnis*, 56-69.
- Islam, S. (2021, April 1). *Black Box Testing Technique*. Retrieved from <https://codenboxautomationlab.com/>:
<https://codenboxautomationlab.com/black-box-testing-technique/>
- Ltd, G. T. (2020, 12 01). *Test Documentation in Software Testing*. Retrieved from Guru99: <https://www.guru99.com/testing-documentation.html>
- Mithelesh Parihar, D. A. (2019). Role Of Software Testing Life Cycle(STLC) In Software Development Life. *International Journal of Research*, 649-661.
- Muhamad Nurudin, W. J. (2019). Pengujian Black Box pada Aplikasi Penjualan Berbasis Web Menggunakan Teknik Boundary Value Analysis. *Jurnal Informatika Universitas Pamulang*, 143-148.
- Myers, G. J. (2004). *The Art of Software Testing, Second Edition*. Canada: John Wiley & Sons, Inc.

- MZ, M. K. (2016). Pengujian Perangkat Lunak Metode Black-Box Berbasis Equivalence Partitions Pada Aplikasi Sistem Informasi Sekolah. *Jurnal Mikrotik Edisi Bulan Februari*, volume 06 nomor 03.
- Oscar Pastor, C. (2007). *Model Driven Architecture in Practice, A Software Production Environment Based on Conceptual Modeling*.
- Perry, W. (1990). *A Standard for Testing Application Software*.
- Rizky, D. (2021, April 1). *Mengenal STLC — Software Testing Life Cycle*. Retrieved from [https://medium.com/: https://medium.com/dot-intern/mengenal-stlc-software-testing-life-cycle-d1bc5a938b72](https://medium.com/:https://medium.com/dot-intern/mengenal-stlc-software-testing-life-cycle-d1bc5a938b72)
- Samuel Rex Mulyadi, E. N. (2019). *Buku Matrikulasi Matematika Dasar untuk Tingkat Perguruan Tinggi*. Ponorogo: Uwais Inspirasi Indonesia.
- Shi, M. (2010). Software Functional Testing from the Perspective of Business Practice Computer and Informatics Science. *Computer and Information Science* , Vol 3, No 4.
- Sofiyah, H. P. (2019). Komparasi Dua Teknik Black Box Testing: Equivalence Partitioning dan Boundary Value Analysis. *Prosiding Annual Research Seminar 2019*, 213-220.
- Waskhito Wibisono, F. B. (2002). Pengujian Perangkat Lunak Dengan Menggunakan Model Behaviour UML. *Jurnal Ilmiah Teknologi Informasi*, 43-50.

LAMPIRAN

Lampiran A

Rancangan Lengkap Tabel *Test Case*

No	Req ID	Req Desc	Scenario ID	Scenario Desc	TC ID	TC Desc	Env Test
1	RS01	Login to Website	TS01	Check the Login Functionality	EP01	Login With Invalid Username minimum input and Valid Password	Production
2					EP02	Login With Invalid Username maximum input and Valid Password	
3					EP03	Login With Valid Username and Valid Password	
4					EP04	Login With Valid Username and Invalid Password minimum input	
5					EP05	Login With Valid Username and Invalid Password maximum input	
6	RS02	User can see the Display Product	TS02	Check the Homepage	TC01	Validate Logo is Display	Production
7					TC02	Validate icon Facebook is Display	
8					TC03	Validate icon Instagram is Display	
9					TC04	Validate icon Download App is Display	
10					TC05	Validate icon Customer Care is Display	
11					TC06	Validate icon Order History is Display	
12					TC07	Validate icon itemSaldo is Display	
13					TC08	Validate profile is Display	
14					TC09	Validate Search is Display	

15					TC10	Validate Banner is Display	
16	RS02	User can see the Display Product	TS03	Check the Search Functionality	TC11	Validate search with keyword "Pulsa"	Production
17					TC12	Validate search with keyword "Token Listrik"	
18					TC13	Validate search with keyword "Samsat Jatim"	
19					TC14	Validate search with keyword "BPJS"	
20					TC15	Validate search with keyword "PBB Jakarta"	
21					TC16	Validate search with keyword "PDAM Yogyakarta"	
22	RS02	User can see the Display Product	TS04	Check the Category and Product Display Page	TC17	Validate category "Pulsa & Paket Data"	Production
23					TC18	Validate category "Uang Elektronik"	
24					TC19	Validate category "Esamsat"	
25					TC20	Validate category "Listrik (PLN)"	
26					TC21	Validate category "BPJS"	
27					TC22	Validate category "Internet dan TV Kabel"	
28					TC23	Validate category "Voucher Game"	
29					TC24	Validate category "Streaming"	
30					TC25	Validate category "Gas"	
31					TC26	Validate category "Pajak Bumi & Bangunan"	
32					TC27	Validate category "Air (PDAM)"	
33					TC28	Validate category "Telkom & Telepon (Pasca Bayar)"	

34					TC29	Validate category "Iuran Pemeliharaan Lingkungan"	
35					TC30	Validate category "Asuransi"	
36					TC31	Validate category "Multi Finance"	
37					TC32	Validate category "Pendidikan"	
38	RS03	User can input product to Check out Page	TS05	Check the Inquiry Customer Number	EP06	Input Invalid Customer Number on category BPJS "1"	Production
39					EP07	Input Invalid Customer Number on category BPJS "9999999999999999"	
40					EP08	Input Valid Customer Number on category BPJS "GEF00000"	
41					EP09	Input Invalid Customer Number on category Samsat "9"	
42					EP10	Input Invalid Customer Number on category Samsat "999999999999999900000000"	
43					EP11	Input Valid Customer Number on category Samsat "GEF0000"	
44					EP12	Input Invalid Customer Number on category Perusahaan Daerah Air Minum (PDAM) "9"	
45					EP13	Input Invalid Customer Number on category Perusahaan Daerah Air Minum (PDAM) "999999999999999900000000"	
46					EP14	Input Valid Customer Number on category Perusahaan Daerah Air Minum (PDAM) "GEF0000"	
47					EP15	Input Invalid Customer Number on	

63		nt metho ds			TC43	Validate Payment Methods Transer via ATM	
64					TC44	Validate Payment Methods BCA VA	
65					TC45	Validate Payment Methods CIMB Rekpon	
66					TC46	Validate Choose Available Methods	
67	RS04	User can choose availa ble payme nt metho ds	TS08	Check the Thank You Page Function ality	TC47	Validate Header Thankyou Page	Production
68					TC48	Validate Keterangan Status	
69					TC49	Validate Petunjuk Bayar	
70					TC50	Validate Link to Order History	
71					TC51	Validate link to Home	
72					TC52	Validate rincian pembayaran	
73					TC53	Validate Kode Bayar	
74					TC54	Validate Metode Bayar	
75					TC55	Validate Tanggal Pesanan	
76					TC56	Validate Batas Waktu	
77					TC57	Validate Total Bayar	
78					TC58	Validate Bottom Thankyou Page	
79	RS05	User can see order history	TS09	Check the Order History Function ality	TC59	Validate Logo on Order History	Production
80					TC60	Validate Download Apps on Order History	
81					TC61	Validate icon Facebook on Order History	
82					TC62	Validate icon Instagram on Order History	
83					TC63	Validate Customer Care on Order History	
84					TC64	Validate icon Download App is Display	

85				TC65	Validate icon Order History on Order History
86				TC66	Validate icon itemSaldo on Order History
87				TC67	Validate profile on Order History
88				TC68	Validate iconRetail
89				TC69	Validate iconVirtual
90				TC70	Validate icon Travel
91				TC71	Validate icon Food
92				TC72	Validate iconTliket
93				TC73	Validate iconVirtualOrderHistory
94				TC74	Validate icon Retail Order History
95				TC75	Validate icon Food Order History
96				TC76	Validate icon Tiket Order History
97				TC77	Validate Filter "Konfirmasi"
98				TC78	Validate filter "Proses"
99				TC79	Validate filter "Berhasil"
100				TC80	Validate Filter "Gagal"
101				TC81	Validate Filter "Pengembalian Dana"

Realisasi Tabel Test Case

No	Scenario ID	Scenario Desc	TC ID	TC Desc	Expected Result	Actual Result
1	TS01	Check the Login Functionality	EPO 1	Login With Invalid Username minimum input and Valid Password	Passed	Passed
2			EPO 2	Login With Invalid Username maximum input and Valid Password	Passed	Passed
3			EPO 3	Login With Valid Username and Valid Password	Passed	Passed
4			EPO 4	Login With Valid Username and Invalid Password minimum input	Passed	Passed
5			EPO 5	Login With Valid Username and Invalid Password maximum input	Passed	Passed
6	TS02	Check the Homepage	TC0 1	Validate Logo is Display	Passed	Passed
7			TC0 2	Validate icon Facebook is Display	Passed	Passed
8			TC0 3	Validate icon Instagram is Display	Passed	Passed
9			TC0 4	Validate icon Download App is Display	Passed	Passed
10			TC0 5	Validate icon Customer Care is Display	Passed	Passed
11			TC0 6	Validate icon Order History is Display	Passed	Passed
12			TC0 7	Validate icon itemSaldo is Display	Passed	Passed
13			TC0 8	Validate profile is Display	Passed	Passed
14			TC0 9	Validate Search is Display	Passed	Passed
15			TC1 0	Validate Banner is Display	Passed	Passed
16	TS03	Check the Search Functionality	TC1 1	Validate search with keyword "Pulsa"	Passed	Passed
17			TC1 2	Validate search with keyword "Token Listrik"	Passed	Passed

18			TC1 3	Validate search with keyword "Samsat Jatim"	Passed	Passed
19			TC1 4	Validate search with keyword "BPJS"	Passed	Failed
20			TC1 5	Validate search with keyword "PBB Jakarta"	Passed	Failed
21			TC1 6	Validate search with keyword "PDAM Yogyakarta"	Passed	Passed
22	TS04	Check the Category and Product Display Page	TC1 7	Validate category "Pulsa & Paket Data"	Passed	Passed
23			TC1 8	Validate category "Uang Elektronik"	Passed	Passed
24			TC1 9	Validate category "Esamsat"	Passed	Passed
25			TC2 0	Validate category "Listrik (PLN)"	Passed	Passed
26			TC2 1	Validate category "BPJS"	Passed	Passed
27			TC2 2	Validate category "Internet dan TV Kabel"	Passed	Passed
28			TC2 3	Validate category "Voucher Game"	Passed	Passed
29			TC2 4	Validate category "Streaming"	Passed	Passed
30		Check the Category and Product Display Page	TC2 5	Validate category "Gas"	Passed	Passed
31			TC2 6	Validate category "Pajak Bumi & Bangunan"	Passed	Passed
32			TC2 7	Validate category "Air (PDAM)"	Passed	Passed
33			TC2 8	Validate category "Telkom & Telepon (Pasca Bayar)"	Passed	Passed
34			TC2 9	Validate category "Iuran Pemeliharaan Lingkungan"	Passed	Passed
35			TC3 0	Validate category "Asuransi"	Passed	Passed
36			TC3 1	Validate category "Multi Finance"	Passed	Passed
37			TC3 2	Validate category "Pendidikan"	Passed	Passed

38	TS05	Check the Inquiry Customer Number	EP0 6	Input Invalid Minimal Custome Number on category BPJS "1"	Passed	Passe d
39			EP0 7	Validate Invalid Minimal Customer Number on BPJS	Passed	Passe d
40			EP0 8	Input Invalid Customer Number on category BPJS "999999999999"	Passed	Passe d
41			EP0 9	Validate Invalid Maximal Customer Number on BPJS	Passed	Passe d
42			EP1 0	Input Valid Customer Number on category BPJS "GEF00000"	Passed	Passe d
43			EP1 1	Validate Valid Maximal Customer Number on BPJS	Passed	Passe d
44			EP1 2	Input Invalid Minimal Customer Number on category Samsat "9"	Passed	Passe d
45			EP1 3	Validate Invalid Minimal Customer Number on Samsat	Passed	Passe d
46			EP1 4	Input Invalid Customer Number on category Samsat "999999999999999900000000"	Passed	Passe d
47			EP1 5	Validate Invalid Maximal Customer Number on Samsat	Passed	Passe d
48			EP1 6	Input Valid Customer Number on category Samsat "GEF00000"	Passed	Passe d
49			EP1 7	Validate Valid Customer Number on Samsat	Passed	Passe d
50			EP1 8	Input Invalid Customer Number on category Perusahaan Daerah Air Minum (PDAM) "9"	Passed	Passe d
51			EP1 9	Validate Invalid Minimal Customer Number on PDAM	Passed	Passe d

66		Check the Checkout Page	TC3 4	Validate tab Konfirmasi Pesanan on	Passed	Passed
67			TC3 5	Validate image product checkout page	Passed	Passed
68			TC3 6	validate delete product on checkout page	Passed	Passed
69			TC3 7	Validate Item Product on Checkout Page	Passed	Passed
70			TC3 8	Validate item Subtotal on Checkout Page	Passed	Passed
71			TC3 9	Validate item Total on Checkout Page	Passed	Passed
72			TC4 0	Validate inputCoupon on Checkout Page	Passed	Passed
73	TS07	Check the Payment Page functionality	TC4 1	Validate Total Pembayaran	Passed	Passed
74			TC4 2	Validate Payment Methods Isaku	Passed	Passed
75			TC4 3	Validate Payment Methods Transer via ATM	Passed	Passed
76			TC4 4	Validate Payment Methods BCA VA	Passed	Passed
77			TC4 5	Validate Payment Methods CIMB Rekpon	Passed	Passed
78			TC4 6	Validate Choose Available Methods	Passed	Passed
79	TS08	Check the Thank You Page Functionality	TC4 7	Validate Header Thankyou Page	Passed	Passed
80			TC4 8	Validate Keterangan Status	Passed	Passed
81			TC4 9	Validate Petunjuk Bayar	Passed	Passed
82			TC5 0	Validate Link to Order History	Passed	Passed
83			TC5 1	Validate link to Home	Passed	Passed
84			TC5 2	Validate rincian pembayaran	Passed	Passed
85			TC5 3	Validate Kode Bayar	Passed	Passed
86			TC5 4	Validate Metode Bayar	Passed	Passed
87			TC5 5	Validate Tanggal Pesan	Passed	Passed

88			TC5 6	Validate Batas Waktu	Passed	Passe d
89			TC5 7	Validate Total Bayar	Passed	Passe d
90			TC5 8	Validate Bottom Thankyou Page	Passed	Passe d
91	TS09	Check the Order History Functionality	TC5 9	Validate Logo on Order History	Passed	Passe d
92			TC6 0	Validate Download Apps on Order History	Passed	Passe d
93			TC6 1	Validate icon Facebook on Order History	Passed	Passe d
94			TC6 2	Validate icon Instagram on Order History	Passed	Passe d
95			TC6 3	Validate Customer Care on Order History	Passed	Passe d
96			TC6 4	Validate icon Download App is Display	Passed	Passe d
97			TC6 5	Validate icon Order History on Order History	Passed	Passe d
98			TC6 6	Validate icon itemSaldo on Order History	Passed	Passe d
99			TC6 7	Validate profile on Order History	Passed	Passe d
10 0			TC6 8	Validate iconRetail	Passed	Passe d
10 1			TC6 9	Validate iconVirtual	Passed	Passe d
10 2		Check the Order History Functionality	TC7 0	Validate icon Travel	Passed	Passe d
10 3			TC7 1	Validate icon Food	Passed	Passe d
10 4			TC7 2	Validate iconTliket	Passed	Passe d
10 5			TC7 3	Validate iconVirtualOrderHistory	Passed	Passe d
10 6			TC7 4	Validate icon Retail Order History	Passed	Passe d
10 7			TC7 5	Validate icon Food Order History	Passed	Passe d
10 8			TC7 6	Validate icon Tiket Order History	Passed	Passe d

109			TC77	Validate Filter "Konfirmasi"	Passed	Passed
110			TC78	Validate filter "Proses"	Passed	Passed
111			TC79	Validate filter "Berhasil"	Passed	Passed
112			TC80	Validate Filter "Gagal"	Passed	Passed
113			TC81	Validate Filter "Pengembalian Dana"	Passed	Passed
114	NULL	ADDITIONAL CASE		Go To Login Page for Login Test	Passed	Passed
115				Go To Login Page for HomePage Test	Passed	Passed
116				Go To Login Page for PLN Test	Passed	Passed
117				Go To Login Page for PBB Test	Passed	Passed
118				Go To Login Page for PDAM Test	Passed	Passed
119				Go To Login Page for BPJS Test	Passed	Passed
120		ADDITIONAL CASE		Go To Login Page for Samsat Test	Passed	Passed
121				Login Test for HomePage Test	Passed	Passed
122				Login Test for PLN Test	Passed	Passed
123				Login Test for PBB Test	Passed	Passed
124				Login Test for PDAMTest	Passed	Passed
125				Login Test for BPJS Test	Passed	Passed
126				Login Test for Samsat Test	Passed	Passed
127		ADDITIONAL CASE		Verify HomePage Title on HomePage Test	Passed	Passed
128				Go To PLN Test for PLN Test	Passed	Passed
129				Go To Token Listrik for PLN Test	Passed	Passed
130				setDenom PLN for PLN Test	Passed	Passed
131				setNominal PLN for PLN Test	Passed	Passed

13 2			Go To Checkout for PLN Test	Passed	Passed
13 3			Go To Payment Center Page For PLN Test	Passed	Passed
13 4			Pilih Alat Bayar for PLN Test On Payment Center Page	Passed	Passed
13 5	ADDITIONAL CASE		ClickBayar for PLN Test On Payment Center Page	Passed	Passed
13 6			Daftar Trx Test on Order History Page	Passed	Passed
13 7			Go To PBB Test	Passed	Passed
13 8			Go To PBB Jakarta	Passed	Passed
13 9			pilih Tahun Pajak 1	Passed	Passed
14 0			pilih Tahun Pajak 2	Passed	Passed
14 1	ADDITIONAL CASE		pilih Tahun Pajak 3	Passed	Passed
14 2			CheckOut Test for PBB test	Passed	Passed
14 3			go To Payment Center PBB	Passed	Passed
14 4			Pilih Alat Bayar PBB	Passed	Passed
14 5			Clickbayar PBB	Passed	Passed
14 6			Daftar Trx Test on Order History Page for PBB Test	Passed	Passed
14 7	ADDITIONAL CASE		go To PDAM Test	Passed	Passed
14 8			pilih Nama PDAM	Passed	Passed
14 9			input Text PDAM	Passed	Passed
15 0			select Input Text PDAM	Passed	Passed
15 1			Checkout Test for PDAM Test	Passed	Passed
15 2			go To Payment Center PDAM	Passed	Passed
15 3	ADDITIONAL CASE		Pilih Alat Bayar PDAM	Passed	Passed
15 4			ClickBayar PDAM	Passed	Passed

155			ThankyouPage Test for PDAM	Passed	Passed
156			Daftar Trx Test on Order History Page for PDAM Test	Passed	Passed
157			Go To BPJS Test	Passed	Passed
158			Go To BPJS Kesehatan	Passed	Passed
159		ADDITIONAL CASE	Go To Paymen Center Page for BPJS Test	Passed	Passed
160			Pilih Alat Bayar for BPJS Test	Passed	Passed
161			Click Bayar for BPJS Test	Passed	Passed
162			ThankyouPage Test for BPJS Test	Passed	Passed
163			Daftar Trx Test on BPJS Test	Passed	Passed
164			Go To Samsat Test	Passed	Passed
165			Go To Samsat Banten	Passed	Passed
166			Checkout Test for Samsat Test	Passed	Passed
167		ADDITIONAL CASE	Go To Payment Center on Samsat Test	Passed	Passed
168			Click Bayar For Samsat Test	Passed	Passed
169			Thankyou Page Test for Samsat Test	Passed	Passed
170			Daftar Trx Test on Order History Page for Samsat Test	Passed	Passed

Lampiran B

Listing Program Class Page Login

```
public class LoginPage extends LoginWithPageFactory{

    @FindBy(xpath="//*[@id=\"siteHeader\"]/div[1]/div/div[2]/div[2]/div/div[2]/div/a")
    public WebElement LoginHomeVirtual;
    @FindBy(name="Email")
    public WebElement userNameVirtual;
    @FindBy(name="Password")
    public WebElement passwordVirtual;
    @FindBy(xpath = "//*[@id=\"site-content\"]/div/div/div[1]/div/form/div[2]/button")
    public WebElement login;

    public LoginPage (WebDriver driver) {
        super(driver); //constructor
    }

    public void clickButtonLoginHome() throws NoSuchElementException{
        try {
            LoginHomeVirtual.click();
        } catch (NoSuchElementException e) {
            e.printStackTrace();
        }
    }

    public void setUserName(String strUserName) {
        try {
            userNameVirtual.sendKeys(strUserName);
            Thread.sleep(2);
        } catch (InterruptedException e) {
            e.printStackTrace();
        }
    }

    public void setPassword(String strPassword) throws
    NoSuchElementException
    {
        try {
            waitElement();
            passwordVirtual.sendKeys(strPassword);
            Thread.sleep(2);
        } catch ( InterruptedException e) {
            e.printStackTrace();
        }
    }

    public void clikcLogin() throws NoSuchElementException{
        try {
            waitElement();
            login.click();
            Thread.sleep(2);
        } catch (InterruptedException e) {
            e.printStackTrace();
        }
    }
}
```

Listing Program Class LoginTest

[illegible]

Listing Program Program Class HomePage

```
public class HomePageVirtual extends LoginWithPageFactory {
    public WebDriver driver;

    @FindBy(xpath="//*[@id=\"siteHeader\"]/div[1]/div/div[2]/div[1]/div/a[1]")
        public WebElement logoIdm;
    @FindBy(css = "<img
src=\"/Assets/image/virtual_logo.png\">")
        public WebElement urlImageLogo;
    @FindBy(id="downloadApp")
    @FindBy(name="search")
        public WebElement search;
    @FindBy(xpath="//*[@id=\"site-content\"]/div[1]")
        public WebElement bannerVirtual;
    @FindBy(xpath="//*[@id=\"site-content\"]/div[3]/div[1]/div/div[1]/div[1]/div")
        public WebElement categoryPulsa;
    @FindBy(xpath="//*[@id=\"site-content\"]/div[3]/div[1]/div/div[1]/div[2]/div")
        public WebElement categoryEmoney;
        public WebElement resultSearchPulsa;
    @FindBy (xpath="//*[@id=\"listprod0\"]")
        public WebElement resultSearchPLN;
    @FindBy (xpath="//*[@id=\"listprod0\"]")
        public WebElement resultSearchBPJS;
    @FindBy (xpath="//*[@id=\"listprod0\"]")
        public WebElement resultSearchPBB;
    @FindBy (xpath="//*[@id=\"listprod0\"]")
        public WebElement resultSearchPDAM;

    public HomePageVirtual(WebDriver driver) {
        super(driver);
        this.driver =driver;
    }

    public void bannerIsDisplay() throws NoSuchElementException {
        try {
            //waitElement();
            waitElementClickable(bannerVirtual);
            Boolean status =
bannerVirtual.isDisplayed();
            Assert.assertTrue(status);
        } catch (NoSuchElementException e) {
            e.printStackTrace();
        }
    }
}
```

```

public void categoryPulsaIsDisplay() throws
NoSuchElementException {
    try {
        waitElement();
        Boolean status = categoryPulsa.isDisplayed();
        Assert.assertTrue(status);
    } catch (NoSuchElementException e) {
        e.printStackTrace();
    }
}

public void validateCategoryPulsa() throws
NoSuchElementException {
    try {
        waitElementClickable(categoryPulsa);
        categoryPulsa.click();
        String
        expectedUrl="https://virtual.klikindomaret.com/"
        ;
        String actualUrl = driver.getCurrentUrl();
        Assert.assertEquals(actualUrl, expectedUrl);

        String expectedTitle = "Beli Pulsa Online,
        Mudah & Aman | KlikIndomaret" ;
        String actualTitle = driver.getTitle();
        Assert.assertEquals(actualTitle,
        expectedTitle);

    } catch (NoSuchElementException e) {
        e.printStackTrace();
    }
}

public void searchSamsatKeyword(String strKeywordSamsat) {
    try {
        waitElementClickable(search);
        search.sendKeys(strKeywordSamsat);
        Thread.sleep(10);
    } catch (InterruptedException e) {
        e.printStackTrace();
    }
}

```

Listing Program Class HomePage Test

```
public class HomePageTest extends driverTest {

    HomePageVirtual objHomePage;
    LoginPage objLogin;
    public String testUrl;

    @Test
    public void goToLoginPage() {
        VirtualTestReports.getTest().log(Status.INFO, "TS01-Go To Website");
        try {
            objLogin = new LoginPage(driver);
            objLogin.clickButtonLoginHome();
            Thread.sleep(20);
        } catch (InterruptedException e) {
            e.printStackTrace();
        }
    }

    @Test
    public void UITestLogo() {
        VirtualTestReports.getTest().log(Status.INFO, "TS02-TC01 Validate
        Logo Is Display");
        try {
            objHomePage = new HomePageVirtual(driver);
            objHomePage.logoIsDisplay();
            Thread.sleep(10);
        } catch (InterruptedException e) {
            e.printStackTrace();
        }
    }

    @Test
    public void UITestSearch() throws InterruptedException{
        VirtualTestReports.getTest().log(Status.INFO, "TS02-TC09 Validate
        Search Is Display");
        try {
            objHomePage = new HomePageVirtual(driver);
            objHomePage.searchIsDisplay();
            Thread.sleep(10);
        } catch (InterruptedException e) {
            e.printStackTrace();
        }
    }
}
```

```

@Test
public void searchSamsat() throws InterruptedException{
VirtualTestReports.getTest().log(Status.INFO,"TS03-TC13 Validate
Search With keyword 'Samsat Jawa Timur' ");
    try {
        objHomePage = new HomePageVirtual(driver);

        objHomePage.searchSamsatKeyword("Samsat Jawa
timur");

        objHomePage.clickSearchSamsatKeyword();
        Thread.sleep(10);

    } catch (InterruptedException e) {
        e.printStackTrace();
    }
}

@Test
public void validateCategoryPulsa() throws InterruptedException
{
VirtualTestReports.getTest().log(Status.INFO,"TS03-TC17 Validate
Category Pulsa & Paket Data");
    try {
        objHomePage = new HomePageVirtual(driver);
        objHomePage.categoryPulsaIsDisplay();
        objHomePage.validateCategoryPulsa();
        Thread.sleep(10);
    } catch (InterruptedException e) {
        e.printStackTrace();
    }
}

```

Listing Program Class Page Product (PLN)

```
public class PlnPage extends LoginWithPageFactory {
    @FindBy (xpath="//*[@id=\"3f74698b-6cc6-4f84-b507-bb02eb2f9567\"]")
    public WebElement TokenListrikPLN;
    @FindBy (xpath="//*[@id=\"nomor_prabayar_NoMeter\"]")
    public WebElement InputToken;
    @FindBy (xpath="//*[@id=\"3f74698b-6cc6-4f84-b507-bb02eb2f9567\"]/div[2]/div[2]/div[2]/div[2]/div[1]/div/div/div/div")
    public WebElement pilihDenom;
    @FindBy (xpath="//*[@id=\"3f74698b-6cc6-4f84-b507-bb02eb2f9567\"]/div[2]/div[2]/div[2]/div[2]/div[1]/div/div/div/ul/li[1]")
    public WebElement pilih20k;
    @FindBy (xpath="//*[@id=\"3f74698b-6cc6-4f84-b507-bb02eb2f9567\"]/div[3]/div/div[2]/button")
    public WebElement buttonBayar;
    @FindBy (xpath="//*[@id=\"3f74698b-6cc6-4f84-b507-bb02eb2f9567\"]/div[1]/div/div[2]")
    public WebElement ErrorMessagePLN;

    public PlnPage(WebDriver driver) {
        super(driver);
    }

    public void goToPLN() throws NoSuchElementException{
        try {
            waitElementClickable(IconPLN);
            IconPLN.click();
        } catch (NoSuchElementException e) {
            e.printStackTrace();
        }
    }

    public void setInquiryPLN(String strInquiryPln) throws NoSuchElementException{
        try {
            waitElementClickable(InputToken);
            InputToken.clear();
            InputToken.sendKeys(strInquiryPln);
        } catch (NoSuchElementException e) {
            e.printStackTrace();
        }
    }

    public String ErrorMessagePLNIsDisplay() {
        try {
            waitElementClickable(ErrorMessagePLN);
            String actualText = ErrorMessagePLN.getText();
            String expectedText = "NOMOR METER YANG ANDA MASUKKAN SALAH, MOHON TELITI KEMBALI";
            Assert.assertEquals(actualText, expectedText);
        } catch (NoSuchElementException e) {
            e.printStackTrace();
        }
        return ErrorMessagePLN.getText();
    }
}
```


Listing Program PLN Test

```
public class plnTest extends driverTest {
    LoginPage objLogin;
    PlnPage objPlnPage;
    CheckoutPage objCheckOutPage;
    PaymentCenterPage objPCFrame;
    ThankYouPage objTQPage;
    DaftarTrxPage objDaftarTrx;

    @Test
    public void inputInvalidMinTokenPLN() throws InterruptedException{
        VirtualTestReports.getTest().log(Status.INFO,"TS05-EP30 Input
        Invalid Min Customer Number PLN");
        try {
            objPlnPage = new PlnPage(driver);
            objPlnPage.setInquiryPLN("00000000000");
            Thread.sleep(1000);
            objPlnPage.ErrorMessagePLNIsDisplay();
            Thread.sleep(100);
        } catch (InterruptedException e) {
            e.printStackTrace();
        }
    }

    @Test
    public void inputValidTokenPLN() throws InterruptedException{
        VirtualTestReports.getTest().log(Status.INFO,"TS05-EP32 Input Valid
        Customer Number PLN");
        try {
            objPlnPage = new PlnPage(driver);
            objPlnPage.setInquiryPLN("10000000100");
            Thread.sleep(1000);
        } catch (InterruptedException e) {
            e.printStackTrace();
        }
    }

    @Test
    public void setNominalPLN() throws InterruptedException{
        VirtualTestReports.getTest().log(Status.INFO,"Choose
        Denominal for PLN");
        try {
            objPlnPage = new PlnPage(driver);
            objPlnPage.chooseDenomPLN();
            Thread.sleep(10);

        } catch (InterruptedException e) {
            e.printStackTrace();
        }
    }
}
```

Listing Program Report

```
public class VirtualReports {
    private static ExtentReports extent;
    private static String reportFileName = "VirtualTest-Automaton-Report".html";
    private static String fileSeperator =
        System.getProperty("file.separator");
    private static String reportFilepath =
        System.getProperty("user.dir") +fileSeperator+ "TestReport";
    private static String reportFileLocation = reportFilepath
        +fileSeperator+ reportFileName;
    public static ExtentReports getInstance() {
        if (extent == null)
            createInstance();
        return extent;
    }
    public static ExtentReports createInstance() {
        tring fileName = getReportPath(reportFilepath);
        ExtentHtmlReporter htmlReporter = new ExtentHtmlReporter(fileName);
        htmlReporter.config().setTestViewChartLocation(ChartLocation.BOTTOM)
        ;
        htmlReporter.config().setChartVisibilityOnOpen(true);
        htmlReporter.config().setTheme(Theme.STANDARD);
        htmlReporter.config().setDocumentTitle(reportFileName);
        htmlReporter.config().setEncoding("utf-8");
        htmlReporter.config().setReportName(reportFileName);
        htmlReporter.config().setTimeStampFormat("EEEE, MMMM dd, yyyy, hh:mm
a '('zzz')");
        extent = new ExtentReports();
        extent.attachReporter(htmlReporter);
        extent.setSystemInfo("OS", "Windows");
        extent.setSystemInfo("AUT", "Adrian");
        return extent;
    }
    private static String getReportPath (String path) {
        File testDirectory = new File(path);
        if (!testDirectory.exists()) {
            if (testDirectory.mkdir()) {
                System.out.println("Directory: " + path + " is created!" );
                return reportFileLocation;
            } else {
                System.out.println("Failed to create directory: " +
path);
                return System.getProperty("user.dir");
            }
        } else {
            System.out.println("Directory already exists: " +
path);
        }
        return reportFileLocation;
    }
}
```

Listing Program Report Listener

```
public class VirtualListener extends driverTest implements
ITestListener{
public void onStart(ITestContext context) {
System.out.println("*** Test Suite " + context.getName() + " started
***");
}
@Override
public void onFinish(ITestContext context) {
System.out.println(("*** Test Suite " + context.getName() + " ending
***"));
VirtualTestReports.endTest();
VirtualReports.getInstance().flush();
}
@Override
public void onTestStart(ITestResult result) {
System.out.println(("*** Running test method " +
result.getMethod().getMethodName() + "..."));
VirtualTestReports.startTest(result.getMethod().getMethodName());
}
@Override
public void onTestSuccess(ITestResult result) {
System.out.println("*** Executed " +
result.getMethod().getMethodName() + " test successfully...");
VirtualTestReports.getTest().log(Status.PASS, "Test passed");
}
@Override
public void onTestFailure(ITestResult result) {
try {
VirtualTestReports.getTest().log(Status.FAIL, "Failed Case is: " +
result.getMethod().getMethodName());
VirtualTestReports.getTest().addScreenCaptureFromPath(TestUtilities.
takeScreenshot(result.getMethod().getMethodName()));
VirtualTestReports.getTest().log(Status.FAIL,
result.getMethod().getMethodName()+" FAIL with error " +
result.getThrowable());

} catch (IOException e) {
// TODO Auto-generated catch block
e.printStackTrace();
}
}
```

Listing Program Driver Test

```
public class driverTest {
    public String testUrl ;
    public static WebDriver driver;

    @BeforeSuite
    public void startApps() {
        System.setProperty("webdriver.chrome.driver",
"D:\\browserdrivers\\chromedriver90.exe");
        testUrl = "https://virtual.klikindomaret.com/";
        driver = new ChromeDriver();
        driver.manage().timeouts().implicitlyWait(10,
TimeUnit.SECONDS);

        //maximize window
        driver.manage().window().maximize();
        // driver.navigate().to(testUrl);
        driver.get(testUrl);
    }

    @AfterSuite
    public void closeApp() {
        driver.quit();
    }
}
```

Listing Program Page Factory

```
public class LoginWithPageFactory {
    public WebDriver driver;
    public LoginWithPageFactory(WebDriver driver){
        this.driver = driver;
        //This initElements method will create all WebElements
        PageFactory.initElements(driver, this);
    }
    public void waitElement(){
        driver.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS);
    }
    public void waitElementLong(){
        driver.manage().timeouts().implicitlyWait(20, TimeUnit.SECONDS);
    }
    public void currentUrl() {
        driver.getCurrentUrl();
    }
    public void alertLogin() throws NoAlertPresentException {
        try {
            new WebDriverWait(driver, 60)
                .until(ExpectedConditions.alertIsPresent());
            Alert alertklik = driver.switchTo().alert();
            alertklik.accept();
            Thread.sleep(100);
        } catch (InterruptedException e) {
            e.printStackTrace();
        }
    }
    public void waitElementClickable (WebElement element) {
        WebElement firstResult = new WebDriverWait(driver, 60)
            .until(ExpectedConditions.elementToBeClickable(element));
    }
    public void PaymentCenterFrame() {
        driver.switchTo().frame("paymentCenterFrame");
    }
    public void defaultContent() {
        driver.switchTo().defaultContent();
    }
}
```

LAMPIRAN B



Universitas Kristen Duta Wacana
Fakultas Teknologi Informasi Program Studi Sistem Informasi
Jl. Dr. Wahidin Sudirahusada 6-28 Yogyakarta 55224
Telp.: (0274)563020 Faks.: (0274)613235



F23.09.001

BERITA ACARA UJIAN SKRIPSI

(Dibaca oleh Ketua Tim Penguji)

Pada hari ini : Kamis, 10 Juni 2021

Telah dilakukan Ujian Skripsi untuk mahasiswa tersebut dibawah ini :

Nama Mahasiswa : **Adrian Paskalla**
No. Induk Mahasiswa : **72170125**
Judul Skripsi : **PENERAPAN METODE EQUIVALENCE PARTITION DALAM PENGUJIAN AUTOMASI PADA WEBSITE ECOMMERCE PRODUK VIRTUAL PT. XYZ**

Dosen Pembimbing I : **ARGO WIBOWO, ST., MT.**

Dosen Pembimbing II : **KATON WIJANA, S.Kom., M.T.**

NILAI

(Lingkari yang dipilih)

A	A-	B+	B	B-	C+	C	D
----------	-----------	-----------	----------	-----------	-----------	----------	----------

Keterangan : LULUS / TIDAK LULUS

(Coret salah satu)

No.	CATATAN PERBAIKAN
1.	Judul menjadi penerapan metode equivalence partition dalam otomatisasi pengujian website ecommerce produk virtual
2.	rumusan masalah cukup bagian A saja
3.	STLC dipelarkan sejak langkahnya dan menjadi kerangka bab 2 dan 4. Proses pengembangan test case masih menjadi subbab
4.	ubah urutan kesimpulan

Perubahan di atas harus sudah diselesaikan paling lambat tanggal Jumat, 9 Juli 2021

Dewan Penguji Skripsi :

1. **ARGO WIBOWO, ST., MT.**
2. **KATON WIJANA, S.Kom., M.T.**
3. **UMI PROBOYEKTI, S.Kom., MLIS.**
4. **YETLI OSLAN, S.Kom., M.T.**

1.

2.

3.

4.

Mahasiswa yang diuji,

(Adrian Paskalla)

Yogyakarta, 10 Juni 2021
Ketua Tim Penguji

(ARGO WIBOWO, ST., MT.)

Catatan: 1 (satu) lembar untuk mahasiswa
1 (satu) lembar untuk arsip























UNIVERSITAS KRISTEN DUTA WACANA
FAKULTAS TEKNOLOGI INFORMASI
PROGRAM STUDI SISTEM INFORMASI

KARTU KONSULTASI SKRIPSI

Mulai Sem. Gnp/2020

NIM : 72170125 Nama Mahasiswa : Adrian Paskalis
Judul Skripsi : PENERAPAN METODE EQUIVALENCE PARTITION DALAM PENGUJIAN AUTOMASI PADA WEBSITE ECOMMERCE PRODUK VIRTUAL PT. XYZ
Pembimbing I : ARGO WIBOWO, ST., MT. Pembimbing II : KATON WIJANA, S.Kom., M.T.

Pembimbing I		Pembimbing II	
1	Tanggal Konsultasi : 3 Maret 2021	1	Tanggal Konsultasi : 4 Maret 2021
Catatan Perkembangan/Revisi Skripsi: - Revisi penulisan bab 1 & 2	Paraf Dosen:  Paraf Mahasiswa: 	Catatan Perkembangan/Revisi Skripsi: - Revisi penulisan dan format pada bab 1	Paraf Dosen:  Paraf Mahasiswa: 
2	Tanggal Konsultasi : 12 April 2021	2	Tanggal Konsultasi : 9 Maret 2021
Catatan Perkembangan/Revisi Skripsi: - Acc bab 1 dan 2	Paraf Dosen:  Paraf Mahasiswa: 	Catatan Perkembangan/Revisi Skripsi: - Revisi penulisan dan sitasi, studi pustaka bab 2 - Acc bab 1	Paraf Dosen:  Paraf Mahasiswa: 
3	Tanggal Konsultasi : 21 April 2021	3	Tanggal Konsultasi : 16 April 2021
Catatan Perkembangan/Revisi Skripsi: - Revisi bab 3 tambahkan use case	Paraf Dosen:  Paraf Mahasiswa: 	Catatan Perkembangan/Revisi Skripsi: - Revisi penulisan bab 2 dan 3	Paraf Dosen:  Paraf Mahasiswa: 
4	Tanggal Konsultasi : 29 April	4	Tanggal Konsultasi : 23 April 2021
Catatan Perkembangan/Revisi Skripsi: - Acc bab 3	Paraf Dosen:  Paraf Mahasiswa: 	Catatan Perkembangan/Revisi Skripsi: - Acc bab 2 - Revisi tabel bab 3	Paraf Dosen:  Paraf Mahasiswa: 
5	Tanggal Konsultasi : 17 May 2021	5	Tanggal Konsultasi : 18 May 2021
Catatan Perkembangan/Revisi Skripsi: - Revisi bab 4	Paraf Dosen:  Paraf Mahasiswa: 	Catatan Perkembangan/Revisi Skripsi: - Acc bab 3 - Demo program bab 4	Paraf Dosen:  Paraf Mahasiswa: 

6	Tanggal Konsultasi : 21 May 2021	
Catatan Perkembangan/Revisi Skripsi: - ACC bab 4 ACC bab 5	Paraf Dosen:	
	Paraf Mahasiswa:	
7	Tanggal Konsultasi : 25 Mei 2021	
Catatan Perkembangan/Revisi Skripsi: Acc ujian Pendadaran	Paraf Dosen:	
	Paraf Mahasiswa:	
8	Tanggal Konsultasi :	
Catatan Perkembangan/Revisi Skripsi:	Paraf Dosen:	
	Paraf Mahasiswa:	
9	Tanggal Konsultasi :	
Catatan Perkembangan/Revisi Skripsi:	Paraf Dosen:	
	Paraf Mahasiswa:	
10	Tanggal Konsultasi :	
Catatan Perkembangan/Revisi Skripsi:	Paraf Dosen:	
	Paraf Mahasiswa:	
11	Tanggal Konsultasi :	
Catatan Perkembangan/Revisi Skripsi:	Paraf Dosen:	
	Paraf Mahasiswa:	

Mengetahui,
Koordinator Skripsi SI

(Drs. Wimmie Handiwidjojo, M.T.)

6	Tanggal Konsultasi : 21 May 2021	
Catatan Perkembangan/Revisi Skripsi: - ACC bab 4 dan ACC bab 5	Paraf Dosen:	
	Paraf Mahasiswa:	
7	Tanggal Konsultasi :	
Catatan Perkembangan/Revisi Skripsi: Acc Pendadaran	Paraf Dosen:	
	Paraf Mahasiswa:	
8	Tanggal Konsultasi :	
Catatan Perkembangan/Revisi Skripsi:	Paraf Dosen:	
	Paraf Mahasiswa:	
9	Tanggal Konsultasi :	
Catatan Perkembangan/Revisi Skripsi:	Paraf Dosen:	
	Paraf Mahasiswa:	
10	Tanggal Konsultasi :	
Catatan Perkembangan/Revisi Skripsi:	Paraf Dosen:	
	Paraf Mahasiswa:	
11	Tanggal Konsultasi :	
Catatan Perkembangan/Revisi Skripsi:	Paraf Dosen:	
	Paraf Mahasiswa:	



Universitas Kristen Duta Wacana
Fakultas Teknologi Informasi Program Studi Sistem Informasi
Jl. Dr. Wahidin Sudirahusada 5-25 Yogyakarta 55224
Telp.: (0274)563929 Faks.: (0274)513235



FORMULIR PERBAIKAN (REVISI) SKRIPSI

Dicetak tanggal: 01-07-2021 10:48:56

Yang bertanda tangan di bawah ini:

Nama : Adrian Paskalis
N I M : 72170125
Judul Skripsi : PENERAPAN METODE EQUIVALENCE PARTITION DALAM OTOMASI
PENGUJIAN WEBSITE ECOMMERCE PRODUK VIRTUAL
Tanggal Pendadaran : Kamis, 10 Juni 2021 pukul 15:00 WIB

Telah melakukan perbaikan tugas akhir dengan lengkap.

Demikian pernyataan kami agar dapat dipergunakan sebagaimana mestinya.

Yogyakarta, Kamis, 1 Juli 2021

Dosen Pembimbing I

ARGO WIBOWO, ST., MT.

Dosen Pembimbing II

KATON WILANA, S.Kom., M.T.