# BAB I. PENDAHULUAN

## 1.1 Latar Belakang

*Optical Distribution Point* (ODP) memiliki fungsi utama sebagai penghubunng kabel fiber optik distribusi dengan kabel drop atau bisa disebut sebagai tempat terminasi jaringan optik mode tunggal. ODP sendiri merupakan tempat terminasi kabel dengan sifat-sifat konstruksi yang kuat, kokoh, tahan korosi, dan tahan di berbagai cuaca sekalipun, mengingat kondisinya yang dipasang di luar ruangan. Meskipun memiliki kondisi yang kuat dan kokoh, hal ini tidak dapat menghindari terjadinya kerusakan maupun gangguan pada ODP. Oleh karena itu, pemantauan terhadap ODP perlu dilakukan (Bayu, dkk., 2020).

Karena kerusakan dan gangguan yang dapat terjadi, maka ODP perlu untuk dilakukan perbaikan algoritma pemrograman Telkom Akses sehingga data dari pelanggan yang tidak berlangganan tidak terdata dalam ODP sistem. Adapun permasalahan lain seperti optikal los yang merupakan keluhan paling sering dilaporkan, perbualnnya rata-rata terdapat 1470 keluhan (Fatuhat, 2023). Aktifitas pemantauan juga kerap dilakukan oleh para teknisi maupun *manager* setempat. Hal yang dilakukan dalam pemantauan ODP ialah dengan memantau lokasi ODP menggunakan aplikasi yang telah disediakan perusahaan (Suprayogi, 2019). Pertama-tama melakukan pemeriksaan terkait daftar lokasi ODP yang harus di periksa, kemudian baru memasukkan informasi tersebut pada aplikasi. Namun proses ini dirasa kurang efektif karena nyatanya para pekerja masih sering mengalami kesulitan dalam mengakses informasi lokasi ODP yang akurat dan lengkap. Kerap dijumpai aplikasi mengalami eror maupun *crash* saat digunakan, maka dari itu seringkali para pekerja harus mencari informasi tambahan lainnya melalui sumber-sumber lain. Proses ini tentu kurang efektif karena memakan cukup banyak waktu.

Upaya yang dilakukan dalam mengatasi permasalahan diatas yaitu membangun sistem monitoring ODP berbasis *website* yakni KYRA. KYRA sendiri merupakan hasil dari projek yang diberikan oleh pihak PT Telkom Akses Tigaraksa. *Website* ini digunakan untuk melakukan monitoring lokasi ODP dengan beberapa fitur yang dapat digunakan seperti "*All* ODP", "Export Excel", dan lain-

lain. Sistem monitoring ODP ini diharapkan dapat memudahkan penggunanya untuk melakukan pemantaun terhadap lokasi ODP sehingga tidak perlu memakan waktu lagi. Didalam sistem ini, terdapat berbagai macam level dan peran akun, sehingga dapat digunakan seluruh pekerja yang berwenang terlepas dari perbedaan jabatan mereka. Pemantauan dapat dilakukan dengan lebih efisien dan efektif karena dilengkapi informasi yang cukup lengkap dan akurat.

Sebelum sistem KYRA dipastikan dapat berjalan dengn baik, hendaknya dilakukan pengujian terhadap sistem dengan *Automation Testing* atau pengujian otomatis. Karena dengan dilakukannya pengujian ini, *developer* dapat mengetahui jika terdapat *bug* maupun kegagalan dalam sistem lainnya. Metode pengujian otomatis yang paling umum ialah *Black Box Testing* terlebih dengan menggunakan teknik *Equivalence Partitioning* yaitu dengan membagi pengujian kedalam beberapa partisi yang membuat pengujian dapat dilakukan dengan lebih cepat.

Tanpa pengujian otomatis, *developer* bisa saja melewatkan suatu kegagalan maupun *bug* yang tidak dapat ditemukan dengan pengujian manual. Pengujian otomatis dinilai lebih efektif dalam menguji sebuat sistem yang cukup kompleks dibandingkan dengan pengujian manual. Bayangkan saja jika menggunakan pengujian secara manual harus dilakukan proses dari awal dan terus berulang-ulang, belum lagi dengan jumlah *test-cases* yang banyak dan bahkan tak terhingga jumlahnya.

Hal ini dapat diatasi dengan melakukan pengujian otomatis menggunakan metode *Black Box Testing* disertai teknik *Equivalence Partitioning*. Selain itu, dengan dilakukannya pengujian otomatis terhadap *website* KYRA ini dapat membantu *developer* dalam menemukan permasalahan terhadap sistem monitoring tersebut serta para pekerja PT Telkom Akses Tigaraksa yang nantinya akan menggunakan sistem ini agar dapat mengurangi resiko terjadinya kegagalan sistem saat sudah digunakan.

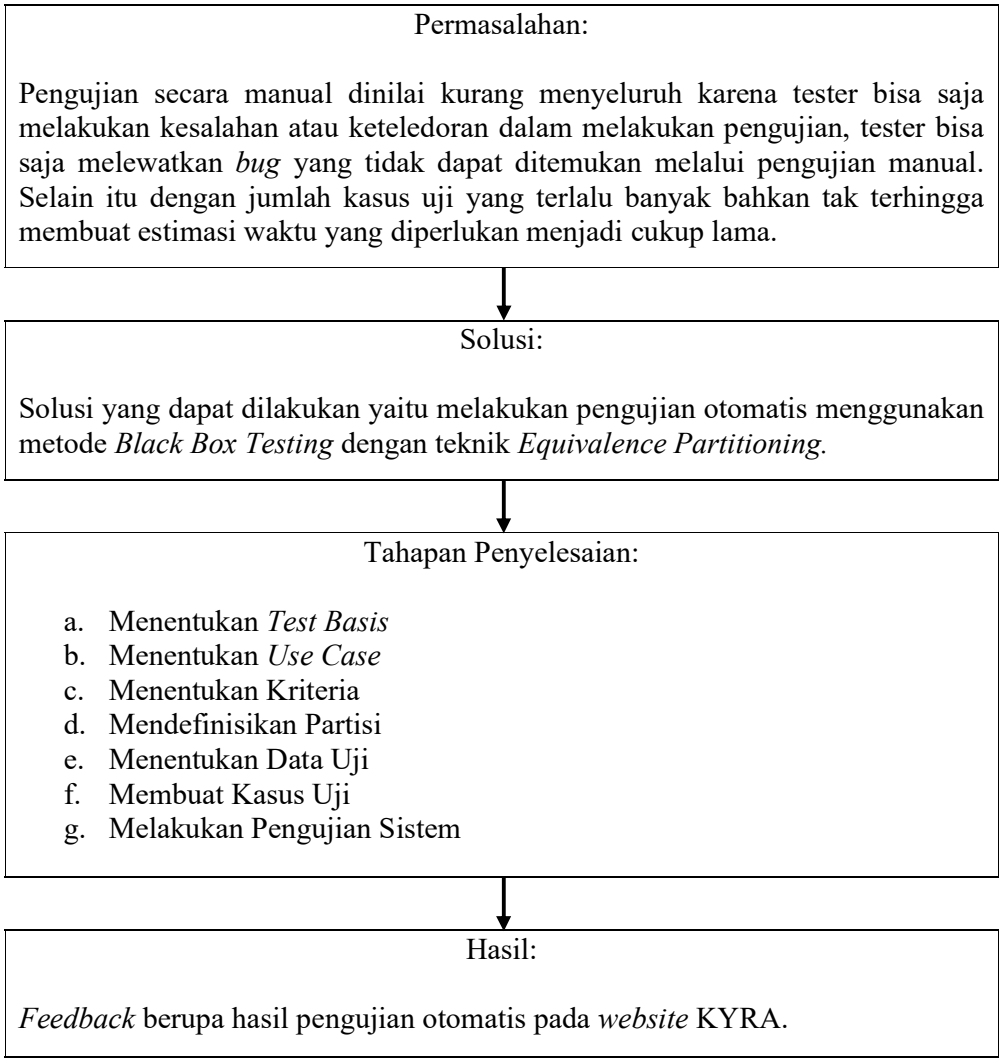
## 1.2 Tujuan

Tujuan penulisan tugas akhir ini agar *website* KYRA dapat berjalan dengan baik tanpa ada *bug* yang dapat mengganggu hingga merusak sistem, sehingga ketika

sistem telah diterapkan, para pengguna terutama teknisi PT Telkom Akses Tigaraksa dapat menggunakannya dengan mudah.

## 1.3 Kerangka Pemikiran

Teknik yang digunakan dalam metode *Black Box Testing* ini merupakan teknik *Equivalence Partitioning* terdiri dari beberapa tahapan yaitu menentukan *Test Basis*, *Use Case*, dan Kriteria setelah itu dilakukan pendefinisian terhadap partisi, kemudian membuat data serta kasus uji. Setelah semua data terkumpul barulah dilakukan pengujian otomatis terhadap keseluruhan sistem. Dibawah ini telah dibuat kerangka pemikiran sebagai berikut:




Gambar 1. 1 Kerangka Pemikiran

## 1.4 Kontribusi

Setelah dilakukannya *Automation Testing* pada *Website* KYRA dengan metode *Black Box Testing* menggunakan teknik *Equivalence Partitioning,* diharapkan dapat memberikan beberapa kontribusi sebagai berikut:

1. *Web Developer* KYRA
    a) Membantu para pengembang dalam menemukan eror maupun *bug* pada sistem.
    b) Mempermudah pengembang dalam memperbaiki fitur pada sistem ketika terjadi kegagalan sehingga tidak perlu menulis ulang program dari awal, hanya pada bagian yang terdapat *bug* saja.
2. Seluruh pengguna *Website* KYRA
    a) Mempermudah pengguna terutama para teknisi dalam melakukan monitoring ODP yaitu dengan memberikan titik koordinat yang sesuai dengan letak ODP.
    b) Para pengguna dapat memanfaatkan sistem dengan lebih baik dan mudah.
    c) Memperkecil risiko terjadinya kegagalan pada sistem saat dijalankan.

# BAB II. TINJAUAN PUSTAKA

## 2.1 Penelitian Terdahulu

Pada rujukan penelitian pertama yaitu penelitian yang dilakukan oleh Salmadiella (2022), pengujian menggunakan metode *black box equivalence partitioning* karena dinilai lebih detail dalam menentukan data uji ruang berdasarkan pada kriteria dari *field* dan partisi yang telah dibuat pada setiap *field*.

Pada rujukan penelitian kedua yaitu penelitian yang dilakukan oleh Prabowo dkk. (2023), dilakukan 2 teknik pengujian yakni *boundary value analysis* dan *equivalence partitioning.* Dalam penelitian ini disimpulkan bahwa teknik *equivalence partitioning* dapat meninjau lebih banyak kondisi sehingga dihasilkan hasil yang lebih akurat.

Pada rujukan penelitian ketiga yaitu penelitian yang dilakukan oleh Pramudita (2020), dapat dihasilkan data dan kasus uji yang mendetail sesuai dengan *use case* serta kriteria yang telah ditentukan. Selain itu, pengujian juga dapat dilakukan dengan mudah menggunakan teknik *equivalence partitioning*.

Pada rujukan penelitian keempat yaitu penelitian yang dilakukan oleh Shaleh dkk. (2021), pengujian otomatis atau *black box testing* digunakan untuk melihat apakah program sudah berjalan sesuai dan sama dengan fungsi yang diharapkan.

Pada rujukan penelitian kelima yaitu penelitian yang dilakukan oleh Leksanti (2020), menyimpulkan bahwa peengujian otomatis menggunakan *tools* Katalon Studio dinilai lebih efektif dibandingkan pengujian secara manual.

Pada rujukan penelitian keenam yaitu penelitian yang dilakukan oleh Agustian (2020), suatu sistem apabila tidak dilakukan pengujian dapat membuat sistem tersebut berjalan tidak maksimal serta dalam menentukan tingkat integritas dan kerahasiannya. Agustian juga menyimpulkan bahwa pengujian menggunakan teknik *Equivalence Partitioning* adalah metode pengujian paling sesuai karena membagi ruang menjadi beberapa pilihan.

Pada rujukan penelitian ketujuh yaitu penelitian yang dilakukan oleh Aziz (2020), dilaksankannya pengujian terhadap sistem yang dikembangkan dapat menjamin bahwa sistem tersebut telah bebas dari kesalahan serta menyatakan bahwa sistem telah lengkap dan tepat.

Tabel 1. 1 Penelitian Terdahulu

| No | Peneliti | Judul Penelitian | Metode | Pembahasan | Hasil |
|---|---|---|---|---|---|
| **1** | Alfiora Salmadiella | Pengujian Otomatis *Website* Dengan Metode *Blackbox Testing Equivalence Partitioning* Studi Kasus : *Infotech* (I-Lab) | *Black Box Equivalence Partitioning* | Pada penilitian ini dilakukan pengujian otomatis pada website Infotech (I-Lab) yang merupakan sistem *e-learning* penunjang pelaksanaan praktikum pada perguruan tinggi. Menggunakan Metode *black box tesing* dengan teknik *equivalence partitioning*. *Website Infotech* (I-Lab) sendiri dinilai sangat efektif dalam menajalankan fungsinya (Salmadiella, Alfiora, 2022). | Pengujian ini menghasilkan nilai keefektivitas keberhasilan dari *website Infotect* (I-Lab) sebesar 90% yang berati "sangat efektif" berdasarkan acuan Litbang Depdagri (1991) (Salmadiella, Alfiora, 2022). |
| **2** | Hans Prabowo. | *Product Automation Testing* pada *Kalcare*.com Memanfaatkan Teknik *Boundary Value analysis* dan *Equivalence Partitioning* | *Black Box Equivalence Partitioning* dan *Boundary Value Analysis* | Pada penelitian ini dilakukan pengujian otomatis pada *website Kalcare*.com yang berperan sebagai menia penjualan produk-produk dari perusahaan. Metode yang digunakan yaitu *black box testing* dengan memanfaatkan teknik *boundary value analysis* (BVA) dan *equivalence partitioning* (EP). Pengujian BVA dilakukan dengan 2 kondisi yaitu tampilan peringatan saat *user* salah melakukan dan menampilkan beranda saat *login* berhasil (Prabowo, Hans, dkk., 2023). | Setelah pengujian dilakukan, didapati bahwa teknik terbaik yang direkomendasikan yaitu teknik *equivalence partitioning* karena dinilai dapat melakukan peninjauan lebih banyak kondisi sehingga hasil yang didapat lebih akurat (Prabowo, Hans, dkk., 2023). |
| **3** | Rully Pramudita | Pengujian *Black Box* pada Aplikasi | *Black Box Equivalence Partitioning* | Penelitian ini membahas mengenai pengujian | Pada penggunaan teknik |

| | | | | | |
|---|---|---|---|---|---|
| | | Ecampus Menggunakan Metode *Equivalence Partitioning* | | otomatis pada aplikasi Ecampus yang berfungsi untuk mengelola akademik dan keuangan pada perguruan tinggi. Metode yang digunakan dalam pengujian yaitu *Black Box* dengan teknik *Equivalence Partitioning*. Penelitian kali ini menghasilkan umpan balik yang ditujukan bagi pengelola *Ecampus* (Pramudita, Rully, 2020). | *equivalence partitioning* dinilai dapat digunakan dengan mudah dalam pengujian yang berdasrkan pada data uji serta kasus uji mendetail sesuai dengan *use case* dan kriteri yang telah ditentukan (Pramudita, Rully, 2020). |
| **4** | Ibnu Adha Shaleh | Pengujian *Black Box* pada Sistem Informasi Penjualan Buku Berbasis Web dengan Teknik *Equivalence Partitioning* | *Black Box Equivalence Partitioning* | Penelitian ini membahas mengenai pengujian otomatis pada sistem informasi penjualan buku dengan metode *equivalence partitioning*. Pengujian dengan metode ini bertujuan untuk menilai apakah program sudah sesuai dan sama dengan fungsinya (Shaleh, Ibnu, A., dkk., 2021). | Hal yang dapat disimpulkan dari pengujian tersebut yaitu data dalam *database* sesuai, dapat mengetahui apakah program sudah sesuai yang diharapkan, dan program yang diuji sudah berjalan dengan sesuai yang diharapkan (Shaleh, Ibnu, A., dkk., 2021). |
| **5** | Yokebeth Denna Leksanti | Pengujian *Website ACC Whistle* Menggunakan Metode *Black Box Testing* | *Black Box Testing* | Pada penelitian ini membahas tentang pegujian pada *website* ACC Whistle dengan mengguanaksn metode *black box testing* dan membandingkannya dengan pengujian manual. Perbedaan pengujian otomatis dengan manual yaitu pengujian otomatis membutuhkan *tools* seperti Katalon Studio dan memerlukan *test* | Setelah dilakukan penelitian ini, didapati bahwa pengujian otomatis tidak lebih efektif dibandingkan menggunakan pengujian manual karena adanya kendala-kendala yang lebih menghambat pengujian secara otomatis (Leksanti, |

| | | | | | |
|---|---|---|---|---|---|
| | | | | *case* serta data *binding,* sedangkan pengujian manual dapat dilakukan secara langsung oleh penguji (Leksanti, Yokebeth, D., 2020). | Yokebeth, D., 2020). |
| **6** | Andrian Agustian | Implementasi Teknik *Equivalence Partitioning* pada Pengujian Aplikasi *E-Learning* Berbasis Web | *Black Box Equivalence partitioning* | Penelitian ini membahas mengenai pengujian otomatis pada web *e-learning* yang merupakan media pembelajaran jarak jauh. Metode yang digunakan merupakan *black box testing* dengan teknik *equivalence partitioning.* Tahpan berdasrkan metode yang diusulkan tersebut ialah mengidentifikasi dan menentukan fungsi sistem berdasrkan pada batas atas dan batas bawah yang diinginkan (Agustian, Adrian, dkk., 2020). | Setelah dilakukan penelitian ini, diambil kesimpulan bahwa metode *black box testing* dengan teknik *equivalence partitioning* merupakan metode yang paling sesuai karena membagi ruang menjadi pilihan-pilihan (Agustian, Adrian, dkk., 2020). |
| **7** | Irfan Abdul Aziz | Pengujian *Black Box* pada Aplikasi Sistem Kasir Berbasis *Website* Menggunakan Teknik *equivalence partitioning* | *Black Box Equivalence partitioning* | Pada penelitian ini mebahas mengenai pengujian pada *website* sistem kasir dengan metode *black box testing* yaitu teknik *equivalnece partitioning*. *Equivalence partitioning* memecah domain *input* dari program ke dalam kelas-kelas data sehingga diperoleh suatu *test case* (Aziz, I., A., dkk., 2020). | Setelah dilakukan penelitian ini, dapat disimpulkan bahwa *software* yang diuji telah bebas dari kesalahan sehingga dapat terhindar dari berbagai kesalahan yang bisa menimbulkan kerugian (Aziz, I., A., dkk., 2020). |

## 2.2 Pengujian Perangkat Lunak

Pengujian perangkat lunak adalah suatu proses menjalankan program maupun sistem yang ingin diuji dengan tujuan melakukan evaluasi kemampuan atau atribut kemudian menentukan apakah program atau sistem tersebut lulus uji serta memenuhi persyaratan yang telah ditentukan oleh perusahaan (M. Z. Komarudin, 2016).

Pengujian ini sangat diperlukan guna memastikan bahwa program maupun sistem berjalan sesuai fungsionalitas yang diharapkan sehingga dapat berfungsi dengan baik. Sudah merupakan kewajiban bagi *developer* atau penguji untuk melakukan pengujian terhadap program atau sistem yang telah dibuat sehingga kegagalan maupun kekurangan sistem dapat ditemukan lebih dini dan dapat segera diperbaiki. Aktifitas pegujian ini merupakan hal yang cukup fatal dan kritis dalam sebuah sistem terutama dalam jaminan kualitas dari perangkat lunak, selain analisis, desain dan pengkodean, pengujian juga termasuk dalam siklus hidup suatu pengembangan perangkat lunak (Mustaqbal, dkk., 2015).

## 2.3 UML

Metode pemodelan secara visual yang berfungsi untuk menggambarkan, mengkonstruksi, dan mendeskripsikan serta mendokumentasikan sistem merupakan pengertian dari *Undefied Modeling Language* (UML). Dapat juga dijadikan standar dalam pembuatan *blueprint* sebuah sistem, yang bersifat konseptual seperti fungsi sistem dan proses bisnis serta mencakup hal-hal yang konkrit seperti skema basis data, komponen dalam suatu sistem, dan pernyatan bahasa pemrograman. Salah satu dari 14 jenis diagram UML yakni *use case diagram*. Berikut ini merupakan elemen-elemen yang terdapat dalam *use case diagram* (Fu'adi & Prianggono, 2022):

| SIMBOL | NAMA | KETERANGAN |
|---|---|---|
|  | *Actor* | Menspesifikasikan himpuan peran yang pengguna mainkan ketika berinteraksi dengan *use case*. |
|  | *Dependency* | Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri *(independent)* akan mempengaruhi elemen yang bergantung padanya elemen yang tidak mandiri. |
|  | *Generalization* | Hubungan dimana objek anak (*descendent*) berbagi perilaku dan struktur data dari objek yang ada di atasnya objek induk (*ancestor*). |
|  | *Include* | Menspesifikasikan bahwa *use case* sumber secara *eksplisit*. |
|  | *Extend* | Menspesifikasikan bahwa *use case* target memperluas perilaku dari *use case* sumber pada suatu titik yang diberikan. |
|  | *Association* | Apa yang menghubungkan antara objek satu dengan objek lainnya. |
|  | *System* | Menspesifikasikan paket yang menampilkan sistem secara terbatas. |
|  | *Use Case* | Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu aktor. |
|  | *Collaboration* | Interaksi aturan-aturan dan elemen lain yang bekerja sama untuk menyediakan prilaku yang lebih besar dari jumlah dan elemen-elemennya (sinergi). |
|  | *Note* | Elemen fisik yang eksis saat aplikasi dijalankan dan mencerminkan suatu sumber daya komputasi. |

Sumber: dumetschool_use_case_diagram

Gambar 2. 1 Elemen Use Case Diagram

## 2.4 *Black Box Testing*

Didalam pengujian perangkat lunak terdapat beragam metode yang dapat digunakan, seperti metode *Black Box.* Metode ini merupakan metode pengujian yanng dilakukan tanpa melihat bagian kinerja internal atau kode program yang ada pada sistem. Hal ini membuat penguji beranggapan bahwa perangkat lunak tersebut adalah sebuah "kotak hitam" yang dirasa cukup dikenali dengan melakukan proses testing bagian luar dan tidak pernting jika dilihat isi dalamnya (Salamah, dkk., 2017).

Pengujian *Black Box* ini tidak dibutuhkan keterampilan khusus dalam memahami isi dari kode program yang ada karena itulah metode ini tidak

dibutuhkan akses kedalam kode program. Oleh karenanya, dapat diambil kesimpulan bahwa metode *Black Box Testing* adalah pengujian fungsional karena penguji hanya berkepentingan dengan fungsionalitasnya berdasarkan pada *requirements spesification* yang ada (Priyaungga, B., A., dkk., 2020).

## 2.5 *Equivalence Partitioning*

*Equivalence Partitioning* merupakan salah satu teknik dalam metode *Black Box Testing*. Teknik ini adalah pengujian yang didasarkan pada data pada setiap *form*, pengujian yang memecah domain masukkan kedalam kelas-kelas untuk menentukan *Test Case* dari program yang ingin diuji. *Test Case* dari *Equivalence Partitioning* sendiri dirancang untuk menggambarkan berbagai kondisi dari masukkan yang ada yakni dalam kondisi *valid* dan *non-valid*.Teknik ini didasrkan pada premis dari masing-masing *input* dan output dari suatu komponen partisi didalam kelas-kelas yang ada, berdasarkan pada spesifikasi komponen dan akan diperlakukan oleh komponen terseut secara ekivalen atau sama (Aziz, 2020).

Pada teknik ini terdapat 7 tahapan yaitu menentukan *use case*, menentukan kriteria, mendefinisikan partisi, membuat data uji, membuat kasus uji, melakukan pengujian, dan yang terakhir evaluasi (Pramudita, Rully, 2020). Berikut merupakan 7 tahapan dari teknik *Equivalence Partitioning*:

### 2.5.1 Penetuan Use Case yang Diuji

Pada tahapan pertama merupakan dilakukannya penentuan terhadap *Use Case* yang akan digunakan untuk pengujian. *Use Case* ini akan dipilih dari total keseluruhan *Use Case* yang ada.

### 2.5.2 Penentuan Kriteria

Tahapan kedua yaitu dengan menentukan kriteria dari *Use Case* sebelumnya yang telah ditentukan.

### 2.5.3 Pendefinisian Partisi

Tahapan selanjutnya setelah mementukan kriteria yaitu mendefinisikan partisi dengan memecah kriteria menjadi beberapa partisi berdasarkan karakteristrik dari kriteria tersebut.

### 2.5.4 Pembuatan Data Uji

Seteleh tahapan-tahapan sebelumnya diselesaikan, maka dilanjutkan dalam pembuatan data uji yang disajikan dalam bentuk tabel yang berisi daftar data uji. Hal ini akan menjadi bahan untuk menentukan kasus uji.

### 2.5.5 Pembuatan Kasus Uji

Pada tahapan kelima adalah membuat kasus uji berdasarkan tabel daftar data uji yang telah diselesaikan pada tahapan sebelumnya. Nantinya kasus uji disajikan dalam bentuk tabel agar lebih mudah saat dilakukan pengujian oleh penguji.

### 2.5.6 Pengujian

Pada tahapan ini merupakan tahapan dimana dilakukan pengujian menyeluruh terhadap sistem sesuai dengan yang ada pada tabel kasus uji. Data yang dihasilkan pada tahapan ini nantinya akan disajikan dalam tabel hasil uji.

### 2.5.7 Evaluasi Hasil Pengujian

Setelah melakukan pengujian terhadap sistem, maka dilanjutkan pada tahapan terakhir yaitu tahapan evaluasi hasil dari pengujian. Tahapan ini memaparkan temuan apa saja selama dilakukannya pengujian seperti menjelaskan kekurangan maupun kelebihan dari sistem juga menambahkan saran agar sistem dapat berjalan lebih baik.

## 2.6 *Automation Testing*

Automation Testing atau pengujian otomatis berbeda dengan pengujian manual yang di mana manusia yang bertanggung jawab dalam melakukan uji fungsionalitas terhadap perangkat lunak dengan cara yang sama seperti dilakukan oleh pengguna atau *user.* Jika dibandingkan dengan pengujian manual, pengujian otomatis lebih memakan sedikit waktu dalam melakukan pengujian menyeluruh terhadap sistem, namun lebih banyak waktu yang digunakan untuk mempertahankan kode program uji sembari meningkatkan cakupan pengujian yang menyeluruh karena pengujian otomatis dilakukan menggunakan suatu alat atau *tools* uji otomatis (Li, & Wu, 2019).

Meningkatkan efektivitas, efisiensi, dan *coverage* dalam *testing* merupakan tujuan utama *automation testing* dalam dunia SQA (*Software Quality Assurance*) yang sebenarnya. Hal ini bisa saja tidak terasa manfaatnya bila sistem yang dikembangkan memiliki skala yang cukup kecil. *Test automation* dapat menjadi sebuah terobosan dalam menghemat waktu dengan mengotomatiskan proses pengujian yang bersifat repetitif atau *smoke testing* (tes guna memastikan fungsionalitas yang penting dan *critical* berjalan baik) yang wajib dilakukan secara berkala setelah terdapat rilis/perubahan (Saifudin, Aries, 2023).

## 2.7 *Selenium*

*Selenium* merupakan salah satu *tools* atau alat yang digunakan dalam melakukan pengujian secara otomatis pada suatu perangkat lunak. Kerangka pengujian *software* yang digunakan untuk aplikasi berbasis web yang menyediakan terpadu antarmuka serta dapat berfungsi pada hampir seluruh *browser* dan juga tes dapat dilakukan dalam beberapa bahasa pemrograman. *Selenium* memiliki beberapa macam komponen seperti *Selenium* IDE, *Selenium WebDriver*, API *Selenium Client*, *Selenium Grid* dan *Selenium* RC (*Selenium* RC sudah ditinggalkan dan saat ini digantikan dengan *Selenium WebDriver*). *Selenium* IDE adalah ekstensi yang ada pada *browser* Firefox dan memungkinkan penguji untuk merekam, mengubah, serta menguji *debug*. *Selenium* inti berada dalam *Selenium* IDE yang memungkinkan untuk merekam, memutar ulang, dan menyimpan tes dengan mudah dan cepat di lingkungan nyata saat sistem sedang berjalan. *Selenium* adalah perangkat lunak *open source* rilisan di bawah Lisensi *Apache* 2.0. Perangkat lunak ini berjalan di berbagai platform seperti *Linux* / UNIX, *Platform Windows* dan OS X. Ini dianggap sebagai salah satu *Open Source* yang difasilitasi dengan pengujian berbasis *browser* bagi *tester* perangkat lunak maupun pengembang (Saifudin, Aries, 2023). Berikut merupakan kelebihan dan kekurangan *Selenium*:

## 2.8 *Lean Canvas*

*Lean Canvas* merupakan suatu dokumen yang berfungsi untuk melakukan evaluasi dari sebuah model bisnis. Dalam rangka meminimalisir resiko pada sistem, *Lean Canvas* dapat digunakan guna membangun pengujian yang sesuai. Selain itu, tujuan penggunaan *Lean Canvas* juga sebagai wadah pendokumentasian untuk mengatasi masalah manajemen pengujian agar lebih mudah. Maka dari itu, *Lean Canvas* digunakan supaya menjadi bagian dari laporan pengujian serta hasil tersebut dipaparkan berupa beberapa poin yang sederhana (Nidagundi, Padmaraj, 2016).

Dalam hal ini, *Lean Canvas* digunakan sebagai media penjelas terkait pelaporan pengujian yang telah dilakukan oleh *tester*. *Lean Canvas* menjadi media dokumentasi laporan pengujian agar managemen dari hasil pengujian dapat tertata dengan baik yang berupa bagan-bagan.

## 2.9 *Test Basis*

Bagi seorang *Tester*, wajar baginya jika ia mendapati kesuliatan dalam menentukan spesifikasi dari sistem yang akan diuji. Seorang *Tester* berusaha untuk menguji secara tepat agar dapat mengidentifikasi kesalahan pada sistem secara tepat. Oleh karenanya *Tester* perlu melakukan analisis terhadap sistem terlebih dahulu sebelum dilakukan pengujian, dengan cara membuat sebuah *Test Basis*. *Test* Basis dapat berupa *requirements* dari suatu sistem, spesifikasi, skrip program, maupun sebuah bisnis proses. Pada suatu kondisi bahkan pengujian dapat dilakukan berdasarkan pada pengalam dari para pengguna dalam menggunakan sistem tersebut tanppa harus didokumentasikan sebelumnya. Dengan kata lain, pengujian dapat dilakukan meskipun tidak dilakukan pendokumentasian pada sistem saat dijalankan. Namun, alangkah baiknya bagi seorang penguji ketika ia ingin melakukan pengujian, hendaklah ia melakukan analisa terlebih dahulu terhadap sistem yang ingin diuji supaya diperoleh informasi yang berguna untuk pengujiannya (Salmadiella, Alfiora, 2022).

Seperti apa yang dinyatakan oleh Hamlet Dick, dalam karyanya "*Foundations of Softwate Testing",* pada tahun 1994, "*Test analysis is the process of looking at something tahat can be used to derive test information. This basis for the tests is*

*called ‘test basis’. It could be a system requirement, a technical spesification, the code itself (for structural testing), or a business process”* .