BAB 1
PENDAHULUAN

1.1 Latar Belakang

Selama pengembangan sistem aplikasi, tim pengembangan menyertakan seorang penguji, juga dikenal sebagai *Quality Assurance (QA)*, yang bertujuan untuk memastikan kualitas aplikasi yang sedang dikembangkan. Peran penguji sangat penting dalam memastikan bahwa aplikasi bebas dari kesalahan, memenuhi tujuan pengembangan dan kebutuhan pengguna, serta berfungsi secara optimal saat digunakan. Dengan demikian, penguji menjadi pintu terakhir untuk memeriksa pengembangan aplikasi siap diterapkan di lingkungan produksi.

Tes yang dilakukan oleh penguji dapat dibagi menjadi dua kategori, manual dan otomatis. Dalam pengujian manual, penguji akan langsung membuka aplikasi dan melalui langkah-langkah pengujian, termasuk mengisi data yang diperlukan secara manual. Di sisi lain, dalam pengujian otomatis, penguji membuat skrip pemrograman yang bertanggung jawab untuk menguji aplikasi, mulai dari membuka aplikasi, membuka halaman tertentu, hingga mengisi data data dari pengujian secara otomatis.

Pengujian otomatis, juga dikenal sebagai *automation testing*, melibatkan pembuatan skrip pemrograman untuk menggantikan tugas yang sebelumnya dilakukan oleh manusia. Skrip ini dirancang untuk melakukan serangkaian langkah pengujian yang sebelumnya dilakukan secara manual, sehingga hanya dengan sekali klik tester dapat menunggu hasil akhir pengujian.

Dalam konteks ini, otomatisasi pengujian menawarkan keuntungan yang signifikan dengan efisiensi yang lebih besar dengan mengurangi keterlibatan manusia secara langsung, mempercepat proses pengujian, dan meningkatkan akurasi dalam mengidentifikasi masalah laten. Namun, seiring dengan manfaat tersebut, pengujian manual masih relevan dan penting untuk menguji beberapa aspek yang memerlukan campur tangan manusia dan pemahaman tentang penggunaan aplikasi. Oleh karena itu, penggunaan kombinasi pengujian manual dan otomatis yang tepat akan berkontribusi pada pembuatan aplikasi yang andal dan berkualitas tinggi.

1.2 Rumusan Masalah

Berdasarkan penjelasan latar belakang diatas, yang menjadi rumusan masalah untuk proyek akhir ini adalah *Automation Testing Web UI* menggunakan *Selenium Webdriver* pada web Solusipay fitur Oba Margin milik perusahaan.

1.3 Tujuan

Tujuan dibuatnya *Automation Testing Web UI* ini adalah untuk mengotomatisasi pengujian fungsionalitas, antarmuka pengguna (UI), dan alur kerja situs web. Dengan *Automation*, pengujian web dapat dilakukan lebih efisien dan konsisten, memungkinkan pengujian berulang untuk menjaga kualitas setiap perubahan, dan mengidentifikasi masalah dengan cepat sebelum perilisan.

1.4 Batasan Masalah

Ruang lingkup yang dikerjakan pada Magang saat ini meliputi:

1. Menggunakan Software Eclipse IDEA
2. Melakukan Pengujian pada Web Solusipayweb milik PT. Inovasi Daya Solusi
3. Menggunakan Selenium Webdriver with Cucumber sebagai BDD Framework
4. Menggunakan Bahasa Java sebagai script utama project dan Bahasa Gherkin untuk penulisan langkah-langkah testing
5. Web dibuka memakai browser berupa Google Chrome
6. Melakukan Automation Reporting menggunakan JUnit.

1.5 Definisi Operasional

Berikut merupakan definisi operasional dari beberapa kata kunci Proyek Akhir ini:

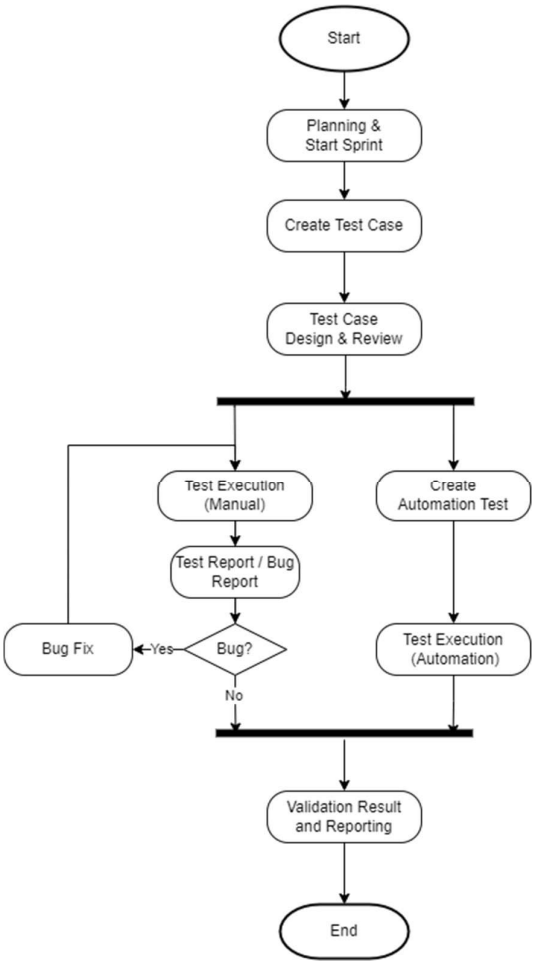
1. *Automation Testing*: Pendekatan pengujian perangkat lunak di mana proses pengujian dilakukan secara otomatis dengan menggunakan alat-alat atau skrip-skrip yang telah dirancang sebelumnya.
2. Web: Mengacu pada lingkungan berbasis internet yang mencakup aplikasi, situs web, dan sumber daya lainnya yang diakses melalui browser web.
3. Selenium WebDriver: Sebuah alat pengujian otomatis untuk aplikasi web yang memungkinkan interaksi langsung dengan elemen-elemen antarmuka pengguna, seperti mengisi formulir dan mengklik tombol.
4. *UI (User Interface)*: Antarmuka yang menghubungkan pengguna dengan perangkat lunak atau sistem. Berfokus pada cara pengguna berinteraksi dengan elemen-elemen visual seperti tombol, kolom, dan menu.

Setelah merinci definisi operasional yang telah dijelaskan, penelitian ini akan fokus pada penerapan teknik Automation Testing menggunakan Selenium WebDriver dalam lingkungan web. Tujuannya

adalah meningkatkan efisiensi dan akurasi pengujian aplikasi dengan mengotomatisasi proses pengujian antarmuka pengguna (UI). Dengan mengidentifikasi elemen kunci dan merancang skenario uji yang sesuai, tujuan utamanya adalah mengurangi potensi kesalahan manusia. Penggunaan alat pendukung juga penting untuk mengembangkan sistem pengujian otomatis ini, dengan harapan memberikan dampak positif pada kualitas keseluruhan aplikasi web yang dikembangkan.

1.6 Metode Pengerjaan

Metodologi yang digunakan dalam pengujian ini adalah *blackbox testing* dengan prinsip kerja scrum. Pengujian *blackbox* berfokus menguji tampilan aplikasi web, fungsi-fungsi aplikasi, dan kesesuaian alur fungsi sesuai dari sudut pandang pengguna, tanpa menguji struktur internal atau source code program secara langsung[1].



Gambar 1.1 Flowchart Metode Pengerjaan

1.6.1 Planning & Start Sprint

Tahap paling awal dimulainya *sprint* adalah *Planning & Start Sprint*. Pada tahap ini *developer, product manager(PM), quality assurance(QA)* akan berdiskusi untuk memilih task yang sekiranya butuh pengujian oleh seorang penguji serta memberikan prioritas dari masing-masing task tersebut dan estimasi waktu yang dibutuhkan.

1.6.2 Create Test Case

Setelah *sprint* dimulai maka *qa* akan mulai membaca *product requirements document (PRD)* yang berisi deskripsi, data-data yang diperlukan, serta persyaratan-persyaratan lainnya terkait task tersebut yang nantinya akan digunakan oleh *qa* sebagai referensi dalam membuat sebuah *test case* atau kasus pengujian.

1.6.3 Test Case Design & Review

Dalam membuat kasus uji, skenario testing dibagi menjadi dua yaitu skenario positif dan negatif. Kasus pengujian yang sudah selesai dibuat oleh *QA* akan dilakukan review terlebih dahulu oleh *product manager (PM)* dan *developer* terkait untuk memastikan bahwa kasus uji tersebut sudah sesuai dengan semua kemungkinan yang mungkin bisa terjadi serta tidak ada kesalahpahaman terkait task tersebut antara *PM, QA*, dan *developer.* Dengan demikian test case pengujian tersebut dapat digunakan sebagai validasi apakah sebuah task sudah sesuai dengan persyaratan yang diharapkan atau belum.

1.6.4 Parallel Test

Proses testing yang berjalan bersamaan yaitu *Manual Testing* dan *Automation Testing.* Proses ini dilakukan secara beriringan, saat melakukan pembuatan *Script* untuk *Automation Testing* dilakukan *Manual Testing* untuk memeriksa proses yang dilakukan *Automate* itu merupakan proses yang benar.

1.6.4.1 Manual Testing

Proses pengujian manual melibatkan pengujian perangkat lunak secara langsung oleh *QA* tanpa menggunakan alat otomatis. Prosesnya dimulai dengan memahami persyaratan perangkat lunak dan merencanakan pengujian, kemudian membuat skrip kasus yang menjelaskan proses pengujian. Reviewer akan menjalankan skrip ini, mencatat hasil review, dan mengidentifikasi masalah. Setelah *Developer* menyelesaikan masalah, pengujian lebih lanjut dilakukan untuk memastikan bahwa masalah tersebut teratasi

dengan benar. Selain itu, kajian penelitian juga dilakukan untuk memastikan tidak adanya dampak negatif dari perubahan yang dilakukan. Meskipun pengujian manual memerlukan waktu dan sumber daya, metode ini penting untuk memastikan bahwa perangkat lunak aman sebelum dirilis dan untuk mengidentifikasi masalah yang mungkin terlewatkan oleh alat otomatis.

1.6.4.2 Automation Testing

Proses bekerja dengan pengujian otomatis melibatkan penggunaan alat dan skrip otomatis untuk melakukan pengujian perangkat lunak. Proses ini dimulai dengan persiapan peralatan yang tepat untuk kelompok pengujian dan perencanaan. Skrip *Automation testing* juga dibuat berdasarkan skenario ini. Pengujian dilakukan secara otomatis oleh alat, sehingga menghasilkan hasil pengujian yang cepat dan konsisten. Dengan mengulangi pengujian secara rutin, *Developer* dapat lebih fokus pada pengujian manual untuk area yang kompleks atau memerlukan penilaian manusia. Meskipun pengujian otomatis menghemat waktu dan tenaga, perlu diingat bahwa tidak semua aspek pengujian dapat diotomatisasi dalam desain skrip dan alat.

1.6.5 Validation Result and Reporting

Validation Result & Reporting mencakup proses untuk memvalidasi hasil pengujian dan menyusun laporan mengenai hasil tersebut. Setelah pengujian manual atau otomatis, hasil pengujian dikumpulkan dan dianalisis untuk memastikan bahwa perangkat lunak memenuhi persyaratan. Proses validasi adalah membandingkan hasil pengujian dengan kriteria yang telah ditentukan dan memastikan bahwa setiap masalah atau bug diidentifikasi dan diperbaiki sebelum perangkat lunak dirilis. Setelah persetujuan selesai, laporan hasil pengujian dikeluarkan, termasuk ringkasan hasil pengujian, masalah yang teridentifikasi, metode pengujian yang digunakan, dan rekomendasi untuk langkah selanjutnya. Laporan ini merupakan sumber informasi penting bagi tim pengembangan dan manajemen untuk membuat keputusan tentang kualitas dan kesiapan perangkat lunak sebelum dirilis.

1.7 Jadwal Pengerjaan

Proses pengerjaan proyek akhir ini dilakukan selama 4 bulan mulai dari bulan Januari 2023 hingga April 2023. Tahapan pengerjaan dibagi menjadi 6, yaitu *Planning & Start Sprint, Create Test Case, Test Design & Review, Manual Testing, Automation Testing, Validation Result and Reporting*. Tabel pengerjaan dapat dilihat dibawah ini:

Tabel 1.1 Jadwal Pengerjaan

No	Kegiatan	Waktu Pelaksanaan															
		Januari 2023				Februari 2023				Maret 2023				April 2023			
		M 1	M 2	M 3	M 4	M 1	M 2	M 3	M 4	M 1	M 2	M 3	M 4	M 1	M 2	M 3	M 4
1	Planning & Start Sprint																
2	Create Test Case																
3	Test Case Design & Review																
4	Manual Testing																
5	Automation Testing																
6	Validation Result and Reporting																