PERBANDINGAN DAN EVALUASI EFEKTIVITAS DAN EFISIENSI KINERJA *AUTOMATION TESTING TOOLS* ANTARA KATALON STUDIO DAN SELENIUM IDE DALAM PENGUJIAN PERANGKAT LUNAK PADA APLIKASI BERBASIS WEB.

Studi Kasus: Sistem Informasi Akademik Universitas Lampung dan *Virtual Class* Universitas Lampung

(SKRIPSI)

Oleh

ALYSA ASTRY DJAYANTI

2115061003

PROGRAM STUDI TEKNIK INFORMATIKA

FAKULTAS TEKNIK

UNIVERSITAS LAMPUNG

2025

ABSTRAK

PERBANDINGAN DAN EVALUASI EFEKTIVITAS DAN EFISIENSI KINERJA *AUTOMATION TESTING TOOLS* ANTARA KATALON STUDIO DAN SELENIUM IDE DALAM PENGUJIAN PERANGKAT LUNAK PADA APLIKASI BERBASIS WEB.

Oleh

ALYSA ASTRY DJAYANTI

Pengujian perangkat lunak merupakan tahap penting untuk menjamin kualitas dan keandalan aplikasi, khususnya pada sistem berbasis web yang melibatkan banyak pengguna. Penelitian ini bertujuan membandingkan dua alat uji otomatis, Katalon Studio dan Selenium IDE, dalam hal efektivitas dan efisiensi kinerja selama proses pengujian dengan melakukan pengujian yang mencakup GUI (*Graphical User Interface), end-to-end, cross-browser*, dan kompabilitas. Hasil pengujian menunjukkan bahwa Selenium IDE unggul dalam kecepatan eksekusi dan kemudahan penggunaan sebagai ekstensi browser, namun memiliki keterbatasan dalam menangani skenario kompleks dan pelaporan yang mendalam. Sebaliknya, Katalon Studio lebih efektif untuk pengujian kompleks berkat dukungan *scripting* lanjutan, integrasi lintas platform, dan kemampuan menghasilkan laporan uji yang komprehensif.

Kata Kunci: Automation Testing, Katalon Studio, Selenium IDE, Pengujian Perangkat Lunak, Aplikasi Web.

ABSTRACT

COMPARISON AND EVALUATION OF EFFECTIVENESS AND EFFICIENCY BETWEEN KATALON STUDIO AND SELENIUM IDE AUTOMATION TESTING TOOLS IN WEB-BASED APPLICATION SOFTWARE TESTING

By

ALYSA ASTRY DJAYANTI

Software testing is a crucial phase in ensuring the quality and reliability of an application, especially for web-based systems involving a large number of users. This study aims to compare two automated testing tools, Katalon Studio and Selenium IDE, in terms of their effectiveness and efficiency during the testing process. The testing conducted includes Graphical User Interface (GUI) testing, end-to-end testing, cross-browser testing, and compatibility testing. The results show that Selenium IDE excels in execution speed and ease of use as a browser extension; however, it has limitations in handling complex testing scenarios and generating detailed reports. In contrast, Katalon Studio is more effective for complex testing due to its support for advanced scripting, cross-platform integration, and the ability to produce comprehensive test reports.

Keywords: Automation Testing, Selenium IDE, Katalon Studio, Software Testing, Web-Based Applications.

PERBANDINGAN DAN EVALUASI EFEKTIVITAS DAN EFISIENSI KINERJA *AUTOMATION TESTING TOOLS* ANTARA KATALON STUDIO DAN SELENIUM IDE DALAM PENGUJIAN PERANGKAT LUNAK PADA APLIKASI BERBASIS WEB.

Oleh

ALYSA ASTRY DJAYANTI

Skripsi

Sebagai Salah Satu Syarat untuk Mencapai Gelar

SARJANA TEKNIK

Pada

Jurusan Teknik Elektro

Fakultas Teknik Universitas Lampung

FAKULTAS TEKNIK

UNIVERSITAS LAMPUNG

BANDAR LAMPUNG

2025

Judul Skripsi	Perbandingan Dan Evaluasi Efektivitas Dan Efisiensi Kinerja Automation Testing Tools Antara Katalon Studio Dan Selenium IDE Dalam Pengujian Perangkat Lunak Pada Aplikasi Berbasis Web. (Studi Kasus : Sistem Informasi Akademik Unila & Virtual Class Unila)
Nama Mahasiswa	Alysa Astry Djayanti
Nomor Pokok Mahasiswa	2115061003
Program Studi	S1 Teknik Informatika
Fakultas	Teknik

MENYETUJUI

1. Komisi Pembimbing

Ir. Trisya Septiana, S.T., M.T., IPM.	**Ir. Cigih Forda Nama, S.T., M.T.I., IPM.**
NIP 199009212019032025	NIP 198307122008121003

2. Mengetahui

Ketua Jurusan Teknik Elektro	Ketua Program Studi Teknik Elektro
Herlinawati, S.T., M.T.	**Yessi Mulyani, S.T., M.T.**
NIP 19710314 199903 2 001	NIP 19731226 200012 2001

MENGESAHKAN

1. Tim Penguji

Ketua : Ir. Trisya Septiana, S.T., M.T., IPM. .....

Sekretaris : Ir. Gigih Forda Nama, S.T., M.T.I., IPM. .....

Penguji : Wahyu Eko Sulistiono, S.T., M.T. .....

2. Dekan Fakultas Teknik

Dr. Eng. Helmy Fitriawan, S.T., M.Sc.

NIP 19750928 200112 1 002

Tanggal Lulus Ujian Skripsi:17 Juli 2025

SURAT PERNYATAAN

Dengan ini saya menyatakan bahwa skripsi ini tidak terdapat karya yang pernah dilakukan orang lain dan sepanjang sepengetahuan saya tidak terdapat atau diterbitkan oleh orang lain, kecuali secara tertulis diacu dalam naskah ini sebagaimana yang disebutkan dalam daftar pustaka. Selain itu, saya menyatakan pula bahwa skripsi ini dibuat oleh saya sendiri.

Apabila pernyataan saya tidak benar, maka saya bersedia dikenai sanksi sesuai dengan hukum yang berlaku.

Bandar Lampung, 17 Juli 2025

Alysa Astry Djayanti

NPM. 2115061003

RIWAYAT HIDUP

Penulis lahir di Bandar Lampung pada tanggal 9 Oktober 2003. Penulis merupakan putri bungsu dari tiga bersaudara, anak dari pasangan Ir. Edy Sukosko, S.T., M.T. dan Dharmawati, A.Md.. Penulis mengawali pendidikan di SD Al - Azhar 1 Bandar Lampung dari tahun 2009 hingga 2015, kemudian melanjutkan ke SMP IT AR RAIHAN pada tahun 2015 hingga 2018. Pendidikan menengah atas ditempuh di SMA IT AR RAIHAN, Jurusan Ilmu Pengetahuan Alam (MIPA), dari tahun 2018 hingga 2021.

Penulis menjadi mahasiswa di Universitas Lampung, Program Studi Teknik Elektro, sejak tahun 2021 melalui jalur SNMPTN. Selama masa studi, penulis aktif dalam berbagai kegiatan akademik dan organisasi. Penulis merupakan salah satu *Co-Founder* dari *Society of Renewable Enegery* (SRE) Universitas Lampung dan menjabat sebagai *Direcor of Human Resource* selama satu periode. Penulis juga pernah menjabat sebagai *Administration and Legality Manager* AIESEC Universitas Lampung. Selain itu, penulis juga aktif sebagai Anggota Departemen Pengembangan Keteknikan Himpunan Mahasiswa Teknik Elektro dengan menjadi pembicara pada beberapa kegiatan *Electical Goes to School* pada tahun 2022 & 2023 dan menjadi bendahara pada kegiatan *Electrical Engineering in Action* selama dua periode.

Selama masa perkuliahan Penulis juga aktif dalam mengikuti kegiatan pelatihan sertifikasi seperti kelas *UI/UX Design & Research* dari Binar *Academy* pada tahun 2022, kemudian kelas *Frontend Developer* Binar Academy pada tahun 2023 serta mengikuti Studi Independen dari Hactiv8 dengan program *Backend with Golang* pada tahun 2024. Pada tahun 2025 penulis juga mengikuti pelatihan *QA Engineer* dengan sertifikasi BNSP.

PERSEMBAHAN

بِسْمِ اللَّهِ الرَّحْمَٰنِ الرَّحِيمِ

Alhamdulillah, Atas Berkat Rahmat Allah yang Maha Kuasa

KUPERSEMBAHKAN KARYA INI UNTUK

Papa dan Mama Tercinta

Ir. Edy Sukoso, S.T., M.T. dan Dharmawati, A. Md.

Kedua Kakakku Tersayang

Ajeng Rachma Farida, S. Pd. dan dr. Andhika Faisal Fajri

Omaku Terkasih

Hj. Maymunah

Keluarga Besar yang selalu menjadi sumber semangat,
Dosen yang telah membimbing dengan tulus,
serta seluruh teman yang setia membersamai setiap langkah perjalanan ini

MOTTO

حَسْبُنَا اللَّهُ وَنِعْمَ الْوَكِيلُ نِعْمَ الْمَوْلَىٰ وَنِعْمَ النَّصِيرُ

(*Hasbunallāhu wa ni'mal wakīl, ni'mal mawlā wa ni'man naṣīr)*

"Cukuplah Allah sebagai penolong kami, dan Dia sebaik-baik Pelindung. Sebaik-baik Pelindung dan sebaik-baik Penolong."

(Q.S. Ali Imran : 173)

ما أصابك لم يكن ليخطئك، وما أخطأك لم يكن ليصيبك.

(*Mā aṣābaka lam yakun li-yukhṭi'aka, wa mā akhṭa'aka lam yakun li-yuṣībaka)*

"Apa yang sudah menjadi takdirmu tidak akan melewatkanmu, dan apa yang melewatkanmu tidak akan menjadi takdirmu."

(Nabi Muhammad SAW, HR. Abu Dawud no. 4699 dan Tirmidzi no. 2516)

"What doesn't kill you makes you stronger

Stand a little taller

Doesn't mean I'm lonely when I'm alone"

(Kelly Clarkson - Stronger)

SANWACANA

Segala puji hanya milik Allah Subhanahu wa Ta'ala, atas rahmat, hidayah, dan karunia-Nya sehingga penulis dapat menyelesaikan skripsi ini dengan baik. Penyusunan skripsi ini tidak terlepas dari bantuan dan dukungan dari berbagai pihak yang telah memberikan motivasi dan bantuan sepanjang prosesnya

Skripsi dengan judul "**Perbandingan Dan Evaluasi Efektivitas Dan Efisiensi Kinerja *Automation Testing Tools* Antara Katalon Studio Dan Selenium IDE Dalam Pengujian Perangkat Lunak Pada Aplikasi Berbasis Web**." ini merupakan salah satu syarat untuk memperoleh gelar Sarjana Teknik pada Jurusan Teknik Elektro, Fakultas Teknik, Universitas Lampung. Pada kesempatan ini, Penulis ingin menyampaikan ucapan terima kasih yang sebesar-besarnya kepada seluruh pihak yang berperan penting dalam perjalanan penulis untuk menyelesaikan skripsi ini.

1. Kedua orang tuaku, papa dan mama serta kedua kakakku, kak Ajeng dan kak Andhika yang tidak pernah berhenti mendoakan, mendukung, membersamai, memberikan kepercayaan dalam setiap langkah perjalanan hidup, dan selalu memberikan apresiasi dalam tiap pencapaian penulis.
2. Bapak Dr. Eng. Ir. Helmy Fitriawan, S.T., M.Sc selaku Dekan Fakultas Teknik Universitas Lampung.
3. Ibu Herlinawati, S.T., M.T selaku Ketua Jurusan Teknik Elektro Universitas Lampung.
4. Ibu Yessi Mulyani, S.T., M.T selaku Kepala Program Studi Teknik Informatika Universitas Lampung.
5. Ibu Ir. Trisya Septiana, S.T., M.T., IPM. selaku dosen pembimbing utama penelitian ini yang selalu memberikan masukan, arahan, serta menjadi tempat penulis untuk berdiskusi dan bertanya selama proses penyusunan skripsi ini.
6. Bapak Ir. Gigih Forda Nama, S.T., M.T., IPM selaku dosen pembimbing pendamping penelitan yang selalu memberikan saran dalam proses pengerjaan penelitian ini.

7. Bapak Wahyu Eko S., S.T., M. Sc. selaku dosen penguji penelitian yang memberikan saran dan masukan yang membangun dalam penelitian yang dikerjakan penulis.
8. Segenap Dosen di Jurusan Teknik Elektro yang telah memberikan ilmu yang bermanfaat, wawasan, motivasi, dan pengalaman bagi Penulis.
9. Mbak Rika dan Segenap Staf di Jurusan Teknik Elektro dan Fakultas Teknik yang telah sangat membantu penulis baik dalam hal administrasi dan hal-hal lainnya.
10. Chairunisa, Nadia, dan Diah, yang selalu memberikan dukungan, menjadi penghibur serta memberikan pengalaman yang sangat berharga dan membuat kehidupan perkuliahan penulis menjadi sangat amat berarti dan berwarna.
11. Zifa dan Salsa, dan teman SMA lainnya, yang menemani penulis dari masa sekolah dan selalu memberikan tawa, dukungan, kebersamaan dalam setiap perjalanan hidup penulis.
12. Nabila dan teman - teman angkatan 2021 lainnya yang tidak bisa disebutkan namanya satu per satu yang selalu mendukung, menghibur dan memberikan apresiasi serta menjadi teman dalam menyelesaikan penelitian ini.
13. Kak Aurick dan Putri Zahra, rekan satu *functional* di AIESEC yang telah menemani penulis dan selalu memberikan dukungan, apresiasi, serta menjadi tempat untuk berkeluh kesah dalam proses penulisan skripsi ini.
14. Kim, mobil Yaris berwarna abu yang telah menemani perjalanan hidup penulis dari semasa bersekolah hingga menyelesaikan pendidikan S1.

Bandar Lampung, 17 Juli 2025

Alysa Astry Djayanti

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB I

PENDAHULUAN

1.1 Latar Belakang

Pengembangan perangkat lunak merupakan suatu proses yang kompleks dan melibatkan berbagai tahapan, mulai dari perencanaan, desain, pemrograman, hingga pengujian. Setiap tahapan memiliki peran penting dalam memastikan bahwa perangkat lunak yang dihasilkan dapat berfungsi dengan baik dan memenuhi kebutuhan pengguna. Salah satu tahapan yang paling krusial dalam siklus hidup pengembangan perangkat lunak adalah pengujian. Pengujian perangkat lunak bertujuan untuk mendeteksi kesalahan, memastikan kualitas, dan menjamin bahwa aplikasi dapat digunakan secara optimal oleh pengguna akhir.

Pengujian perangkat lunak dapat dilakukan dengan dua pendekatan utama, yaitu pengujian manual dan pengujian otomatis. Pengujian manual melibatkan penguji yang berperan sebagai pengguna akhir, melakukan serangkaian tes untuk memeriksa apakah fitur-fitur aplikasi berfungsi sesuai spesifikasi [1]. Meskipun metode ini dapat memberikan wawasan yang mendalam tentang pengalaman pengguna, pengujian manual sering kali memakan waktu dan tenaga yang signifikan. Selain itu, pengujian manual tidak selalu mampu mendeteksi semua bug, terutama dalam aplikasi yang kompleks dan sering dianggap tidak efisien [1], [2].

Karena keterbatasan pengujian manual ini, pengujian otomatis dapat menjadi alternatif. Dengan menggunakan alat uji otomatis, penguji dapat membuat skenario uji yang dapat dijalankan berulang kali tanpa memerlukan intervensi manusia. Hal ini tidak hanya menghemat waktu, tetapi juga meningkatkan akurasi dalam mendeteksi kesalahan [3]. Seiring dengan meningkatnya kompleksitas aplikasi yang dikembangkan, penggunaan pengujian otomatis menjadi semakin penting

untuk memastikan bahwa perangkat lunak dapat berfungsi dengan baik di berbagai kondisi.

Selain itu, pengujian otomatis memungkinkan tim pengembang untuk melakukan pengujian lebih sering, yang dapat mempercepat siklus pengembangan dan meningkatkan kualitas produk akhir.

Berbagai alat uji otomatis dibuat untuk memudahkan pengujian perangkat lunak, baik yang bersifat open-source maupun komersial. Alat-alat ini memiliki fitur dan kemampuan yang bervariasi, sehingga penting bagi tim pengembang untuk memilih alat yang paling sesuai dengan kebutuhan mereka. Beberapa alat mungkin hanya mendukung jenis pengujian tertentu, sementara yang lain menawarkan dukungan untuk berbagai jenis pengujian dan bahasa pemrograman. Memahami perbedaan alat pengujian otomatis sangat penting untuk memastikan bahwa alat yang dipilih dapat memberikan hasil yang optimal [4], [5].

Karena kebutuhan tersebut, penelitian ini melakukan analisis komparatif terhadap alat pengujian otomatis yang paling sering digunakan di beberapa tahun terakhir, yaitu Katalon Studio dan Selenium [6]. Dalam melakukan penelitian ini, tentunya dibutuhkan objek penelitian. Peneliti menggunakan *website* yang paling sering digunakan oleh mahasiswa Universitas Lampung, yaitu SIAKADU (Sistem Informasi Akademik Universitas Lampung). *Website* ini memiliki peran yang sangat krusial dalam mendukung kegiatan akademik di Universitas Lampung (Unila). SIAKADU berfungsi sebagai platform untuk mengelola data akademik mahasiswa, mencakup berbagai aspek mulai dari pengisian Kartu Rencana Studi (KRS), akses nilai, hingga proses pembayaran Uang Kuliah Tunggal (UKT).

Untuk memvalidasi hasil perbandingan dari masing-masing alat yang digunakan, penelitian ini juga menggunakan Virtual *Class* (VClass) sebagai objek penelitian tambahan. VClass berfungsi sebagai platform pembelajaran daring yang mengintegrasikan berbagai fitur penting, seperti pengunggahan materi, absensi kelas, dan pengumpulan tugas.

Dengan membandingkan kedua *tool* testing pada objek yang ditentukan, peneliti melakukan pengujian antarmuka pengguna atau *graphical user interface* (GUI),

pengujian *end-to-end*, pengujian lintas browser (*cross-browser*), dan pengujian kompatibilitas. Pengujian GUI memastikan bahwa antarmuka aplikasi intuitif dan mudah digunakan, sehingga mahasiswa dapat dengan cepat memahami cara mengakses berbagai fitur yang tersedia. Pengujian *end-to-end* mengevaluasi alur kerja aplikasi secara keseluruhan. Kemudian, lintas browser memastikan bahwa aplikasi dapat diakses dan berfungsi dengan baik di berbagai jenis browser yang umum digunakan oleh mahasiswa.

Dengan melakukan pengujian-pengujian tersebut, peneliti dapat mengambil kesimpulan untuk membandingkan *automation tools testing* dalam segi efektivitas serta fitur-fitur seperti jenis pengujian yang didukung, lisensi, biaya, dan aspek lainnya, penelitian ini diharapkan dapat memberikan panduan yang berguna dalam memilih alat pengujian otomatis yang paling sesuai.

1.2 Rumusan Masalah

Berdasarkan latar belakang yang telah diuraikan, maka rumusan masalah dalam penelitian ini adalah sebagai berikut:

1. Apa saja kelebihan dan kekurangan dari masing-masing alat berdasarkan parameter yang telah ditentukan?
2. Apa alat *automation testing* yang lebih efektif dan efisien untuk digunakan dalam pengujian perangkat lunak berbasis web berdasarkan pengujian yang dilakukan?
3. Apa rekomendasi perbaikan yang dapat diberikan berdasarkan pengujian yang telah dilakukan?

1.3 Tujuan Penelitian

Adapun tujuan dari penelitian ini adalah:

1. Membandingkan efektivitas Katalon Studio dan Selenium IDE dalam melakukan pengujian otomatis pada aplikasi VCLASS dan SIAKADU Universitas Lampung.
2. Mengidentifikasi kelebihan dan kekurangan dari masing-masing alat pengujian berdasarkan parameter yang ditentukan.
3. Memberikan rekomendasi mengenai alat pengujian otomatis yang paling sesuai untuk pengembangan dan pengujian perangkat lunak.
4. Memberikan rekomendasi perbaikan berdasarkan temuan yang ditemukan selama masa pengujian.

1.4 Batasan Masalah

Batasan dalam penelitian ini adalah:

1. Pengujian yang dilakukan merupakan pengujian fungsional yang mencakup pengujian antarmuka pengguna (GUI), pengujian *end-to-end*, pengujian lintas browser (*cross-browser*), dan pengujian kompatibilitas.
2. Objek pengujian terbatas pada dua aplikasi akademik Universitas Lampung, yaitu SIAKADU dan VCLASS dengan peran mahasiswa.
3. Metode pengujian yang digunakan adalah *black box testing*, yaitu pengujian tanpa memperhatikan struktur internal atau implementasi kode sumber
4. *Automation tools* yang dibandingkan dalam penelitian ini adalah Katalon Studio dan Selenium IDE.
5. Pengujian lintas-browser dilakukan pada Chrome, Safari, dan Firefox

1.5 Manfaat Penelitian

Penelitian ini diharapkan dapat memberi manfaat sebagai berikut:

1. Manfaat Akademis

    a. Memberikan kontribusi bagi pengembangan ilmu pengetahuan di bidang *Software Quality Assurance* (SQA) dan pengujian perangkat lunak.
    b. Menjadi referensi bagi penelitian lanjutan terkait perbandingan dan evaluasi efektivitas alat uji otomatis perangkat lunak.

2. Manfaat Praktis

    a. Membantu perusahaan atau organisasi dalam memilih alat uji otomatis yang paling sesuai dengan kebutuhan mereka berdasarkan kriteria seperti efisiensi, efektivitas, dan biaya.
    b. Memberikan gambaran tentang keunggulan dan kelemahan alat uji Katalon Studio dan Selenium IDE untuk pengujian perangkat lunak sistem informasi akademik dan *virtual class*.
    c. Menyediakan data dan analisis yang dapat membantu pengembang perangkat lunak dalam meningkatkan kualitas aplikasi yang mereka kembangkan.

3. Manfaat Bagi Pengguna Aplikasi

    a. Memberikan saran untuk perbaikan kepada pengelola Sistem Infromasi Akademik Unila.
    b. Mengurangi kemungkinan terjadinya kesalahan atau *bug* yang dapat mengganggu pengalaman pengguna.

1.6 Sistematika Penulisan

Sistematika yang digunakan dalam penulisan penelitian ini adalah sebagai berikut:

BAB I : PENDAHULUAN

Bab ini memuat latar belakang, rumusan masalah, tujuan penelitian, manfaat penelitian, batasan masalah, dan sistematika penulisan.

BAB II : TINJAUAN PUSTAKA

Bab ini berisi teori-teori dasar yang berkaitan dengan penelitian ini, seperti teori mengenai kualitas perangkat lunak, alat *automation testing*, dan parameter evaluasi.

BAB III : METODOLOGI PENELITIAN

Bab ini menjelaskan waktu dan tempat penelitian, metode penelitian, alat penelitian yang digunakan, serta tahapan penelitian yang dilakukan dalam pengujian perangkat lunak.

BAB IV : PEMBAHASAN

Bab ini memaparkan hasil pengujian dengan Katalon Studio dan Selenium, analisis kelebihan dan kekurangan *automation tools*, perbandingan efektivitas dan efisiensi Katalon Studio dan Selenium serta rekomendasi perbaikan berdasarkan hasil pengujian *Virtual Class* dan SIAKADU Universitas Lampung menggunakan metode *automation testing,* serta pembahasan terhadap hasil tersebut.

BAB V : PENUTUP

Bab ini berisi kesimpulan dari hasil penelitian yang telah dilakukan serta saran-saran yang dapat digunakan untuk perbaikan dan pengembangan Sistem Informasi Akademik Universitas Lampung di masa mendatang.

DAFTAR PUSTAKA

LAMPIRAN

BAB II

TINJAUAN PUSTAKA

2.1 Penelitian Terkait

2.1.1 Comparative Review Of The Features Of Automated Software Testing Tools

Penelitian yang dilakukan oleh Heidilyn V. Gamido dan Marlon V. Gamido menyajikan analisis komprehensif terhadap berbagai alat pengujian otomatis, termasuk Selenium, QTP/UFT, TestComplete, Ranorex, Watir, Sahi, dan SoapUI. Studi ini menyoroti fitur dan kapabilitas masing-masing alat untuk membantu pengguna dalam memahami keunggulan dan keterbatasan yang dimiliki oleh setiap alat pengujian. Studi ini mengidentifikasi beberapa aspek utama yang menjadi dasar perbandingan, seperti dukungan lintas platform dan browser, fitur *record* & *playback*, serta bahasa pemrograman yang digunakan. Selain itu, kemudahan dalam mempelajari dan menggunakan alat juga menjadi pertimbangan penting, terutama bagi penguji yang belum memiliki pengalaman dalam pengujian otomatis.

Parameter lainnya mencakup dukungan akses data eksternal, seperti kemampuan alat untuk membaca data dari Excel, CSV, dan XML, yang memungkinkan pengujian dilakukan dengan berbagai skenario data. Studi ini juga membandingkan apakah alat memerlukan keahlian pemrograman khusus, serta kemampuannya dalam menghasilkan laporan pengujian otomatis untuk mempermudah analisis hasil pengujian. Selain itu, biaya penggunaan juga menjadi parameter perbandingan di mana alat *open-source* lebih ekonomis dibandingkan alat berbayar yang menawarkan fitur tambahan dan dukungan lebih baik. Penelitian ini menunjukkan bahwa alat *open-source* seperti Selenium lebih cocok untuk proyek

dengan keterbatasan anggaran, sementara alat berlisensi seperti QTP/UFT direkomendasikan bagi proyek yang memerlukan fitur dukungan yang lebih komprehensif [5].

2.1.2 Analisis Perbandingan Performansi *Tool Testing* Antara Appium Dan Katalon Dalam Pengujian Otomatisasi Perangkat Lunak Pada Aplikasi Berbasis *Mobile*

Penelitian ini membahas mengenai perbandingan *performance tools* pengujian pada aplikasi berbasis *website*. Penelitian ini memberikan referensi terkait komparasi perbandingan antara Appium dan Katalon dengan objek penelitian aplikasi Gapura UB yang juga merupakan aplikasi Akademik dari Universitas Brawijaya. Penelitian ini dilakukan dengan mengukur durasi eksekusi *test case* dalam menit menggunakan data yang diambil secara otomatis selama 30 kali percobaan. Skenario pengujian dirancang dengan mencakup fitur-fitur seperti login, logout, visi misi, peta, berita, notifikasi, profil, presensi, layanan perpustakaan, dan jadwal, berdasarkan hasil kuisioner tentang preferensi pengguna aplikasi Gapura UB.

Pengujian mencakup *test case* positif dan negatif untuk mengevaluasi performa aplikasi secara menyeluruh. Hasil analisis menunjukkan bahwa Appium mengungguli Katalon dengan performa eksekusi waktu yang lebih baik sebesar 38.73%, sementara Katalon menunjukkan penggunaan CPU yang lebih rendah sebesar 13.77%. Dengan demikian, penelitian ini memberikan referensi terkait pengujian otomatisasi perangkat lunak *mobile* [6].

2.1.3 Comperative Analysis of Automated Testing Tools on GUI WEB-Based Applications

Penelitian lain meneliti perbandingan *tools automation* dari aspek GUI. Penelitian ini menyajikan analisis komparatif antara Selenium Webdriver dan Katalon Studio yang berfokus pada pemenuhan parameter pengujian, kemudahan penggunaan, kecepatan eksekusi, serta kemampuan pelaporan dan dokumentasi hasil eksekusi. Penelitian tersebut menyatakan bahwa kedua alat ini berhasil memenuhi parameter

pengujian yang telah ditentukan, menunjukkan bahwa keduanya mampu mengeksekusi kasus uji secara efektif.

Dari segi kemudahan penggunaan, Katalon Studio lebih unggul dengan antarmuka yang ramah pengguna dan fitur bawaan yang tidak memerlukan instalasi tambahan, berbeda dengan Selenium yang membutuhkan pustaka tambahan untuk fungsi tertentu. Namun, dalam hal kecepatan eksekusi, Selenium Webdriver lebih unggul karena mampu menyelesaikan pengujian lebih cepat, menjadikannya pilihan yang lebih efisien dalam lingkungan yang menuntut waktu eksekusi singkat. Selain itu, penelitian ini mengevaluasi fitur pengujian khusus seperti pengujian berbasis gambar dan kemampuan bergulir, yang memperkuat pemahaman terhadap kelebihan dan kekurangan masing-masing alat dalam berbagai skenario pengujian.

2.1.4 'Comperative Study of Automated Testing Tools: Selenium, SoapUI, HP Unified Functional Testing and Test Complete'

Penelitian yang berjudul *'Comparative Study of Automated Testing Tools: Selenium, SoapUI, HP Unified Functional Testing and Test Complete'* juga membandingkan empat alat pengujian otomatis, yaitu Selenium, SoapUI, HP UFT, dan TestComplete berdasarkan efektivitas, biaya, dan fitur utama. Hasil penelitian menunjukan bahwa Selenium unggul dalam aspek biaya karena bersifat *open-source*, meskipun memerlukan lebih banyak upaya dalam pembuatan skrip. TestComplete menawarkan antarmuka yang ramah pengguna tetapi kurang dalam keamanan data. HP UFT cocok untuk skenario yang membutuhkan keamanan tinggi, meskipun biayanya lebih mahal. Sementara itu, SoapUI menonjol dalam pengujian API serta mendukung pengujian non-fungsional dengan harga lebih terjangkau dibandingkan HP UFT dan TestComplete .

Evaluasi dilakukan berdasarkan karakteristik alat, kecepatan eksekusi, kemampuan pelaporan, penggunaan kembali skrip, fitur rekam dan pemutaran, serta biaya. Selenium dan SoapUI lebih hemat biaya dibandingkan alat komersial seperti HP UFT dan TestComplete. Studi ini menyimpulkan bahwa SoapUI adalah alat paling unggul karena fitur lengkap dan efektivitas biaya [7]

2.1.5 Analisis GUI Testing pada Aplikasi *E-Commerce* menggunakan Katalon Studio

Penelitian yang dilakukan oleh Annisa Indrayanti, dkk pada tahun 2021 menyoroti pentingnya pengujian GUI dalam memastikan antarmuka yang responsif dan sesuai dengan harapan pengguna. Studi ini menganalisis waktu respons tiga aplikasi e-*commerce*, yaitu Bukalapak, JD.ID, dan Tokopedia dengan hasil bahwa Bukalapak umumnya memiliki waktu respons tercepat, meskipun JD.ID dan Tokopedia unggul pada beberapa halaman tertentu. Faktor yang mempengaruhi waktu respons tidak hanya berasal dari kompleksitas GUI, tetapi juga aspek teknis lainnya yang perlu diperhatikan oleh pengembang.

Penggunaan Katalon sebagai alat pengujian GUI juga disoroti karena kemampuannya dalam merekam kasus uji dan menghasilkan log detail. Studi ini merekomendasikan penelitian lebih lanjut untuk memahami faktor tambahan yang mempengaruhi performa GUI pada aplikasi e-*commerce*. Dengan demikian, makalah ini memberikan wawasan tentang pentingnya pengujian GUI, dampaknya terhadap pengalaman pengguna, dan potensi Katalon sebagai alat pengujian yang efektif [8].

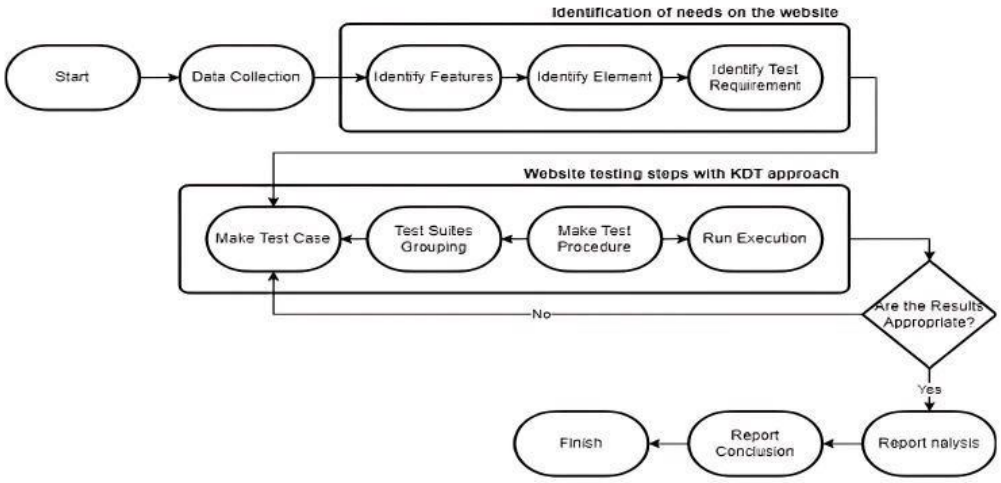
2.1.6 *Efficiency Development Analysis Use of Automation Testing using Katalon on Mobile-Based Applications (Case Insurance Company*)

Penelitian yang dilakukan oleh Shynta Eza Anggrainy dan Alva Hendi Muhammad dari Universitas Amikom pada tahun 2024 menunjukkan bahwa penggunaan alat seperti Katalon Studio dapat meningkatkan efisiensi pengujian perangkat lunak secara signifikan. Beberapa studi membandingkan pengujian manual dan otomatis, di mana hasilnya menunjukkan bahwa pengujian manual membutuhkan lebih banyak waktu dan tenaga serta memiliki risiko kesalahan manusia yang lebih tinggi. Sementara itu, pengujian otomatis dengan Katalon Studio memungkinkan eksekusi tes yang lebih cepat dan konsisten, terutama dalam pengujian berulang yang sering dilakukan dalam siklus pengembangan perangkat lunak.

Selain itu, penelitian juga mengungkap bahwa Katalon Studio memberikan kemudahan bagi penguji dalam mengotomatisasi proses pengujian tanpa memerlukan keahlian pemrograman yang mendalam. Studi ini menunjukkan bahwa fitur otomatisasi dan dukungan pengujian berkelanjutan yang dimiliki Katalon Studio dapat mempercepat identifikasi serta perbaikan kesalahan dalam aplikasi. Pendekatan pengujian *blackbox* yang digunakan dalam berbagai penelitian juga terbukti efektif dalam menilai fungsionalitas perangkat lunak tanpa bergantung pada struktur internalnya, sehingga lebih sesuai untuk mengevaluasi pengalaman pengguna secara langsung [9].

2.1.7 Analysis Of Quality Assurance Performance In The Application Of Manual Testing And Automation Testing For Software Product Testing

Penelitian yang dilakukan oleh Gesang Ibnu Safaat dan Viany Utami Tjhin pada tahun 2024 mengenai perbandingan pengujian manual dan otomatisasi menunjukkan bahwa otomatisasi memberikan efisiensi yang lebih tinggi dalam berbagai aspek pengujian perangkat lunak. Studi yang dilakukan pada aplikasi jual beli *cryptocurrency* dengan menggunakan Katalon Studio mengonfirmasi bahwa pengujian otomatis mampu mengurangi waktu pengujian secara signifikan dibandingkan dengan pengujian manual. Data menunjukkan bahwa rata-rata waktu eksekusi tes otomatis lebih cepat sekitar 11,04% dibandingkan dengan metode manual, yang membuktikan efektivitas otomatisasi dalam meningkatkan efisiensi pengujian.



Gambar 2.1 Metode penelitian sistematis [10]

Berdasarkan Gambar 2.1 penelitian ini dilakukan melalui tahapan sistematis, dimulai dari perencanaan (*planning*) hingga pelaporan hasil (*bug reporting*). Tahap pertama adalah perencanaan, yaitu peneliti menganalisis kebutuhan pengujian perangkat lunak pada aplikasi jual beli *cryptocurrency*, memahami fitur utama, serta memilih alat uji yang sesuai, yaitu Katalon Studio. Selanjutnya, dilakukan identifikasi fitur dan persyaratan pengujian dengan menyusun daftar skenario uji yang mencakup berbagai interaksi pengguna. Setelah itu, pada tahap pembuatan dan implementasi kasus uji, skenario pengujian dijalankan baik secara manual maupun otomatis. Dalam pengujian manual, penguji mencatat hasil secara langsung, sedangkan pengujian otomatis dilakukan melalui skrip di Katalon Studio yang mencatat hasil dalam bentuk log.

Hasil dari kedua metode ini kemudian dianalisis dan dibandingkan, menunjukkan bahwa pengujian otomatis lebih cepat dengan rata-rata waktu eksekusi 3,177 detik dibandingkan dengan 3,492 detik pada pengujian manual, menghasilkan efisiensi sebesar 11,04%. Tahap akhir adalah pelaporan hasil dan *bug reporting*, di mana temuan selama pengujian didokumentasikan secara sistematis, termasuk deskripsi *bug*, langkah reproduksi, serta bukti pendukung seperti log dan tangkapan layar. Keseluruhan proses ini membuktikan bahwa otomatisasi pengujian menggunakan Katalon Studio lebih efisien dan efektif dibandingkan dengan pengujian manual dalam menguji aplikasi jual beli *cryptocurrency* [10]

2.1.8 Otomatisasi Pengujian Aplikasi Blibli Menggunakan Selenium IDE

Penelitian ini dilakukan dengan mengotomatisasi pengujian aplikasi Blibli menggunakan Selenium IDE untuk meningkatkan efisiensi dan keandalan pengujian perangkat lunak. Tahapan penelitian dimulai dengan perencanaan, di mana fitur utama aplikasi diidentifikasi dan skenario pengujian disusun. Setelah itu, dilakukan implementasi pengujian dengan menjalankan kasus uji yang mencakup pengujian formulir pendaftaran, login dan logout, navigasi menu, serta penambahan produk ke keranjang. Selenium IDE digunakan untuk merekam dan menjalankan kembali interaksi pengguna dengan aplikasi web, memastikan bahwa setiap langkah diuji secara konsisten dan sistematis.

Hasil pengujian menunjukkan bahwa Selenium IDE mampu mengurangi waktu pengujian dibandingkan dengan metode manual, sekaligus memastikan bahwa setiap skenario diuji dengan cakupan yang luas. Indikator keberhasilan pengujian ditunjukkan dengan warna hijau untuk eksekusi yang berhasil dan merah untuk pengujian yang gagal. Dari berbagai pengujian yang dilakukan, seluruh fitur utama aplikasi Blibli berfungsi dengan baik, seperti sistem pendaftaran yang dapat menerima data dengan benar, login yang mengarahkan pengguna ke *dashboard*, serta fitur penambahan produk ke keranjang yang berjalan sesuai harapan.

Kesimpulan dari penelitian ini menegaskan bahwa otomatisasi pengujian menggunakan Selenium IDE tidak hanya mempercepat proses pengujian tetapi juga meningkatkan konsistensi dan akurasi hasil. [11]

Beberapa studi terdahulu ini memberikan dasar yang kuat untuk penelitian yang membandingkan Katalon Studio dan Selenium dalam pengujian perangkat lunak seperti menjadi landasan teori untuk efektivitas pengujian otomatis, menyediakan panduan untuk pengujian aplikasi e-*learning*, panduan untuk membandingkan alat pengujian otomatis, dan sebagai referensi pengujian.

2.2 Pengujian Perangkat Lunak

Dari beberapa literatur, dapat disimpulkan definisi terkait pengujian, yaitu pengujian merupakan proses penting dalam pengembangan perangkat lunak yang mengevaluasi apakah suatu sistem memenuhi persyaratan yang ditentukan. Ini melibatkan validasi dan verifikasi untuk mengidentifikasi *bug*, kesalahan, atau persyaratan yang hilang dalam perangkat lunak, sehingga memberikan wawasan kepada pemangku kepentingan tentang kualitas produk [12]. Pengujian perangkat lunak juga proses yang krusial dalam siklus hidup pengembangan perangkat lunak. Tujuan utama dari pengujian adalah untuk memastikan bahwa perangkat lunak berfungsi sesuai dengan spesifikasi yang telah ditetapkan dan memenuhi kebutuhan pengguna [1], [13], [14], [15].

Dalam melakukan pengujian, ada beberapa tahapan yang perlu dilakukan. Langkah pertama adalah perencanaan uji, di mana ruang lingkup, tujuan, sumber daya, dan jadwal pengujian ditentukan. Tahap ini penting untuk memastikan bahwa pengujian dilakukan secara sistematis. Setelah perencanaan selesai, langkah berikutnya adalah pengembangan uji, yaitu pembuatan kasus uji yang mencakup berbagai skenario agar pengujian dapat mencakup semua aspek penting dari perangkat lunak [10], [12].

Selanjutnya, eksekusi uji dilakukan dengan menjalankan kasus uji yang telah dibuat. Pada tahap ini, setiap hasil diamati untuk melihat apakah ada *bug* atau masalah yang muncul. Jika ditemukan kesalahan, maka akan dicatat dalam pelaporan uji, yang berisi informasi mengenai *bug* atau kendala yang ditemukan agar tim pengembang dapat memperbaikinya. Setelah semua pengujian selesai, dilakukan analisis hasil uji untuk mengevaluasi temuan yang ada. Dari hasil analisis ini, dapat diputuskan bagian mana yang perlu diperbaiki atau ditingkatkan. Kemudian, dalam pengujian perangkat lunak, terdapat beberapa jenis pengujian yang digunakan, seperti pengujian unit, pengujian integrasi, dan pengujian sistem, yang masing-masing bertujuan untuk mengidentifikasi berbagai masalah dalam perangkat lunak.

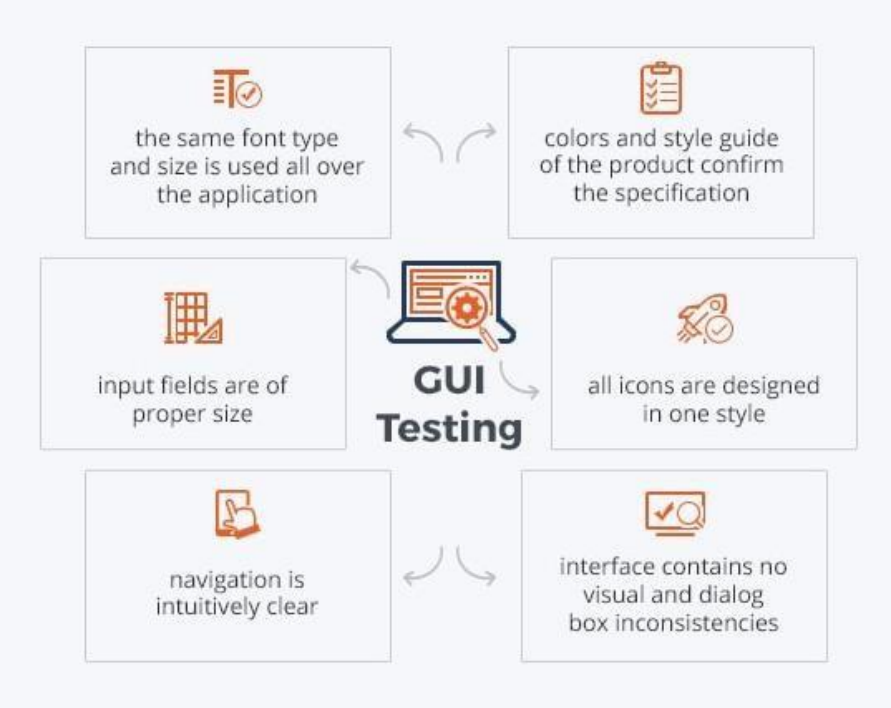
Selain jenis pengujian, terdapat juga teknik yang digunakan untuk menguji perangkat lunak, seperti pengujian kotak putih (*whitebox-testing*) yang berfokus

pada struktur internal aplikasi, serta pengujian kotak hitam (*blackbox-testing*) yang lebih menekankan pada fungsionalitas tanpa melihat bagaimana sistem bekerja di dalamnya. Pengujian perangkat lunak dapat dilakukan secara manual atau otomatis. Pengujian manual melibatkan interaksi langsung dengan aplikasi oleh tester untuk mengevaluasi fungsionalitasnya, sedangkan pengujian otomatis menggunakan alat khusus untuk menjalankan skrip pengujian tanpa campur tangan manusia [1], [14].

2.2.1 *Graphical User Interface* (GUI)

Antarmuka Pengguna Grafis atau *Graphical User Interface* (GUI) adalah tampilan visual yang memungkinkan pengguna berinteraksi dengan aplikasi perangkat lunak melalui elemen-elemen seperti jendela, ikon, tombol, dan menu. GUI bertindak sebagai perantara antara pengguna dan perangkat lunak, membuat navigasi serta penggunaan aplikasi menjadi lebih mudah dan intuitif.

Pengujian GUI sangat berperan dalam meningkatkan pengalaman pengguna. Dengan GUI, pengguna tidak perlu memahami instruksi baris perintah yang kompleks untuk menjalankan suatu aplikasi, sehingga teknologi menjadi lebih ramah bagi pengguna non-teknis. Selain itu, desain GUI yang baik dapat meningkatkan efisiensi dan kenyamanan dalam menggunakan aplikasi, sementara desain yang buruk dapat menyebabkan kebingungan dan menurunkan produktivitas.



Gambar 2. 2 GUI Testing [16]

Salah satu aspek penting dalam GUI adalah waktu respons, yaitu seberapa cepat aplikasi merespons *input* dari pengguna. Kompleksitas desain GUI dapat mempengaruhi performa aplikasi yang dapat memengaruhi kepuasan dan keterlibatan pengguna. Untuk memastikan GUI berfungsi dengan baik, pengujian diperlukan. Pengujian GUI mencakup evaluasi semua elemen antarmuka, seperti menu, tombol, teks, dan *widget* lainnya, untuk memastikan bahwa semuanya beroperasi sesuai spesifikasinya. Selain itu, analisis kinerja GUI secara berkala juga penting agar aplikasi tetap dapat ditingkatkan dan memberikan pengalaman yang lebih baik bagi pengguna[8] .

Proses pengujian GUI dapat dilakukan secara manual maupun otomatis. Namun, pengujian manual memiliki sejumlah kelemahan, salah satunya adalah cakupan pengujian yang terbatas, karena penguji cenderung mengulangi langkah yang sama, sehingga antarmuka lainnya mungkin tidak teruji [17]. Selain itu, kesulitan dalam mereproduksi kegagalan juga menjadi masalah, karena urutan perintah yang digunakan dalam pengujian tidak dicatat. Di samping itu, pengujian manual tidak secara otomatis merekam waktu respons [18] .

2.2.2 Pengujian *End-to-End*

End-to-end testing (E2E *testing*) adalah metode pengujian perangkat lunak yang bertujuan untuk memastikan bahwa seluruh alur kerja aplikasi berfungsi dengan baik dari awal hingga akhir [2]. *End-to-end testing*, yang juga disebut sebagai system *testing* atau *functional* testing, dapat disebut proses pengujian perangkat lunak secara menyeluruh dari awal hingga akhir, mencakup aspek integrasi, dependensi, serta ketersediaan [19] .

Metode ini melibatkan simulasi skenario pengguna nyata untuk memverifikasi bahwa semua komponen sistem, termasuk antarmuka pengguna, basis data, dan integrasi dengan sistem eksternal, berinteraksi dengan benar dan memberikan hasil yang diharapkan [2]. *End-to-end testing* berpotensi memerlukan waktu lama untuk diselesaikan, untuk itu penting untuk terus mengupayakan pemilihan metode pengujian yang efisien, baik secara manual maupun menggunakan otomatisasi [2]. Dengan melakukan pengujian secara menyeluruh, E2E *testing* dapat mengidentifikasi masalah integrasi yang mungkin tidak terdeteksi dalam pengujian unit atau integrasi, serta membantu menemukan *bug* sebelum aplikasi diluncurkan ke publik.

Proses E2E testing biasanya meliputi beberapa tahapan penting, yaitu perencanaan pengujian, pembuatan skenario pengujian, eksekusi pengujian, dan verifikasi hasil. Pada tahap perencanaan, penguji menentukan skenario yang mencakup seluruh alur aplikasi. Selanjutnya, skenario tersebut diuji baik secara manual maupun otomatis. Hasil dari pengujian ini kemudian dianalisis untuk memastikan bahwa aplikasi memenuhi spesifikasi dan harapan pengguna akhir [2], [19].

2.2.3 Pengujian Antar Browser (*Cross-Browser*)

Pengujian antar-browser adalah proses untuk memastikan bahwa sebuah situs web atau aplikasi web berfungsi dengan baik di berbagai browser dan versinya. Setiap browser memiliki cara berbeda dalam merender halaman web, sehingga tanpa pengujian yang tepat, pengalaman pengguna bisa menjadi tidak konsisten.

Gambar 2. 3 Cross-browser Testing [20]

Pengujian lintas browser sangat penting karena pengguna internet tidak terbatas pada satu jenis browser saja. *User* dapat mengakses situs web melalui Chrome, Firefox, Safari, Edge, atau bahkan peramban bawaan dari sistem operasi mereka. Jika sebuah situs web tidak dioptimalkan untuk berbagai browser, pengguna dapat mengalami masalah seperti tata letak yang berantakan, elemen interaktif yang tidak berfungsi, atau kesalahan dalam menampilkan konten. Tujuan utama pengujian ini adalah memberikan pengalaman pengguna yang konsisten, terlepas dari browser yang digunakan
[21].

Pengujian lintas browser dimulai dengan identifikasi browser yang menjadi prioritas, berdasarkan popularitas dan audiens yang ditargetkan. Browser seperti Chrome, Firefox, Safari, dan Edge biasanya menjadi pilihan utama untuk pengujian. Pembuatan kasus uji adalah tahapan berikutnya yang krusial. Kasus uji yang dibuat harus mencakup berbagai elemen penting dalam situs web, seperti tata letak, navigasi, formulir, dan elemen interaktif lainnya. Kasus uji ini berfungsi untuk memastikan bahwa setiap elemen situs web berjalan dengan benar di browser yang diuji. Pengujian dapat dilakukan secara manual atau otomatis menggunakan alat yang telah dipilih, tergantung pada kebutuhan dan kompleksitas situs.

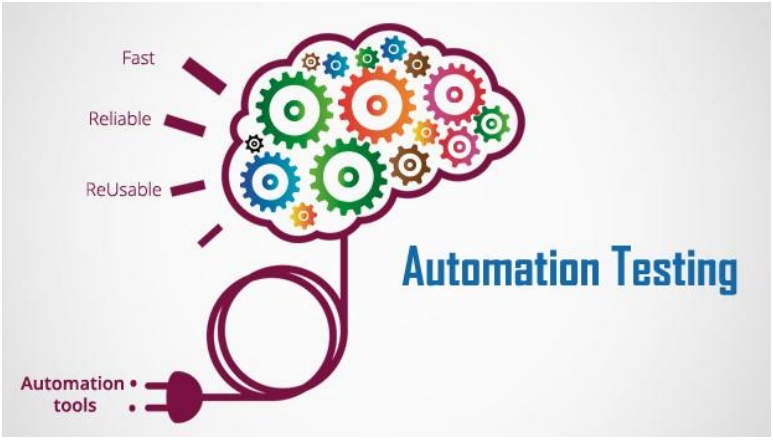
Setelah pengujian selesai, hasil yang diperoleh perlu dianalisis untuk mengidentifikasi masalah yang muncul, seperti inkonsistensi dalam tampilan atau masalah fungsional di browser yang berbeda. Jika ditemukan masalah, pengembang akan melakukan perbaikan, yang bisa melibatkan penyesuaian kode CSS, JavaScript, atau HTML agar situs web dapat tampil dan berfungsi dengan baik di semua browser yang diuji. Setelah perbaikan dilakukan, penting untuk melakukan pengujian ulang untuk memastikan bahwa masalah telah terselesaikan dan tidak ada masalah baru yang muncul, memastikan kualitas dan konsistensi pengalaman pengguna di berbagai browser.

2.2.4 Pengujian Kompatibilitas

Pengujian Kompatibilitas bertujuan untuk memastikan bahwa aplikasi dapat berjalan dengan baik pada berbagai perangkat keras, sistem operasi, dan konfigurasi perangkat lunak lainnya. Ini termasuk memeriksa apakah aplikasi berfungsi secara optimal di berbagai versi sistem operasi, serta pada perangkat *mobile* dengan berbagai ukuran layar dan resolusi.

2.3 *Automation Testing*

Automation testing adalah proses pengujian perangkat lunak yang menggunakan alat dan skrip otomatis untuk menjalankan pengujian, menggantikan pengujian manual yang dilakukan oleh penguji [22], [23], [24]. Tujuan utama dari *automation testing* adalah untuk meningkatkan efisiensi, akurasi, dan kecepatan dalam proses pengujian [3], [25], [26]. Salah satu keuntungan utama dari pengujian otomatisasi adalah efisiensi. Dengan kemampuan untuk menjalankan tes lebih cepat dibandingkan pengujian manual, pengujian otomatisasi sangat bermanfaat untuk tugas-tugas yang berulang, seperti pengujian regresi.



Gambar 2. 4 *Automation Testing* [27]

Selain efisiensi, pengujian otomatisasi juga memiliki akurasi yang lebih tinggi. Metode ini mengurangi risiko kesalahan manusia yang dapat terjadi dalam pengujian manual, memastikan bahwa tes dijalankan secara konsisten dan akurat. Aspek ini penting untuk menjaga kualitas perangkat lunak, terutama ketika kesalahan kecil dapat menyebabkan masalah besar pada sistem yang lebih besar.

Pengujian otomatisasi juga lebih mudah dipelihara karena skrip uji yang dapat digunakan kembali. Dalam lingkungan pengembangan perangkat lunak yang dinamis, seperti pada metodologi tangkas, perangkat lunak sering mengalami pembaruan. Skrip uji otomatisasi yang dapat digunakan kembali ini membantu pengujian untuk terus relevan meski aplikasi berkembang. Kemudian skalabilitasnya juga menjadi aspek penting. Pengujian otomatis dapat menangani volume besar uji coba dan data, yang memungkinkan tim pengembang untuk menguji lebih banyak fitur dalam waktu singkat. Hal ini sangat berguna ketika menghadapi proyek dengan persyaratan pengujian yang ekstensif dan tenggat waktu yang ketat.

Selain itu, pengujian otomatisasi memungkinkan umpan balik yang lebih cepat kepada pengembang. Dengan proses pengujian yang lebih cepat, pengembang dapat segera mengidentifikasi dan memperbaiki masalah, terutama dalam pengembangan berbasis metodologi tangkas yang mengandalkan iterasi cepat. Meski pengujian otomatisasi memerlukan investasi awal dalam alat dan pelatihan,

keuntungan jangka panjangnya, seperti penghematan waktu dan biaya, sangat besar.

Pengujian otomatisasi juga memperluas cakupan pengujian yang memungkinkan pengujian skenario yang mungkin sulit diuji secara manual, memberikan gambaran yang lebih lengkap tentang fungsionalitas aplikasi [3], [25].

2.4 *Automation Testing Tools*

Automation testing tools adalah perangkat lunak yang dirancang untuk membantu tim pengujian dalam mengotomatiskan proses pengujian perangkat lunak. Alat-alat ini memungkinkan pengujian dilakukan dengan lebih cepat, efisien, dan akurat dibandingkan dengan pengujian manual.

Gambar 2. 5 Top 10 Automation Tools 2025 [28]

Dengan menggunakan *automation testing tools*, penguji dapat menjalankan serangkaian tes berulang kali tanpa memerlukan intervensi manusia, yang sangat berguna dalam pengujian regresi, pengujian fungsional, dan pengujian beban [1], [14]. Salah satu kategori utama dari *automation testing tools* adalah alat pengujian fungsional. Dalam hal ini, alat seperti Katalon dan Selenium memainkan peran penting. Berdasarkan informasi yang dirilis oleh *website* resmi Katalon, kedua alat ini termasuk dalam daftar Top 15 *Automation Testing Tools* 2025, yang

menggarisbawahi keberhasilannya dalam mendukung pengujian otomatisasi di berbagai aplikasi.

2.4.1 Katalon Studio

Katalon Studio yang dikembangkan oleh Katalon LLC adalah platform pengujian perangkat lunak berbasis open source yang dirancang untuk otomatisasi pengujian aplikasi web, mobile, dan API [29]. Aplikasi ini kompatibel dengan berbagai sistem operasi, termasuk Windows, macOS, dan Linux. Katalon Studio dilengkapi dengan antarmuka khusus berbasis IDE yang dirancang untuk mendukung proses pengujian.

Gambar 2. 6 Katalon Studio [30]

Alat ini dirancang untuk memberikan solusi pengujian yang efisien dan efektif, baik bagi penguji berpengalaman maupun pemula, karena antarmuka penggunanya yang ramah dan fitur-fitur yang canggih. Salah satu keunggulan utama Katalon Studio adalah kemampuannya untuk mendukung pengujian pada berbagai platform, termasuk aplikasi web yang diuji melalui berbagai browser utama seperti Chrome, Firefox, Safari, dan Edge, aplikasi mobile di perangkat Android dan iOS, serta API untuk layanan RESTful dan SOAP. Katalon Studio menawarkan dua mode penggunaan, yakni Manual Mode yang memungkinkan pengujian dilakukan tanpa menulis kode, dan *Script Mode* yang memberikan fleksibilitas bagi pengguna untuk menulis skrip pengujian menggunakan bahasa pemrograman Groovy.

Selain itu, Katalon Studio dilengkapi dengan fitur rekam dan putar kembali (*Record and Playback),* yang memungkinkan penguji untuk merekam interaksi dengan aplikasi dan memutar kembali langkah-langkah pengujian tersebut tanpa menulis kode secara manual. Fitur ini sangat berguna untuk pengujian otomatis, terutama bagi penguji yang tidak terbiasa dengan penulisan kode.

Alat ini juga memungkinkan integrasi dengan berbagai alat *Continuous Integration/Continuous Deployment* (CI/CD), seperti Jenkins, Git, GitHub, dan Bitbucket, yang mendukung otomatisasi pengujian setiap kali ada perubahan dalam kode sumber. Dengan demikian, pengujian dapat dilakukan secara berkelanjutan dalam siklus pengembangan perangkat lunak yang cepat.

Katalon Studio juga mendukung *data-driven* testing, yang memungkinkan pengujian berbagai skenario menggunakan data dari f*ile* CSV atau Excel. Hal ini memungkinkan pengujian otomatis yang lebih menyeluruh, dengan cakupan skenario yang lebih luas. Selain itu, Katalon Studio menyediakan fitur pelaporan dan dasbor *built-in* yang memungkinkan tim pengujian untuk memantau dan menganalisis hasil tes dengan lebih efisien. Laporan pengujian yang dihasilkan dapat diekspor dalam berbagai format seperti HTML atau PDF, memberikan informasi yang jelas mengenai keberhasilan atau kegagalan tes serta area yang membutuhkan perbaikan.

2.4.2 Selenium IDE

Berdasarkan halaman resmi Selenium, Selenium IDE (*Integrated Development Environment*) adalah alat otomatisasi pengujian perangkat lunak yang digunakan untuk merekam, memutar ulang, dan menyunting pengujian aplikasi web. Berdasarkan penelitian yang dilakukan oleh Hawari, Kusumo, & Firdaus, Selenium IDE merupakan sebuah *plugin* untuk browser yang berfungsi sebagai alat bantu dalam pengujian otomatis.

Gambar 2. 7 Selenium IDE [31]

Alat ini merupakan bagian dari rangkaian proyek Selenium yang lebih besar, yang dikenal dengan kemampuannya dalam mendukung pengujian otomatis untuk aplikasi web di berbagai browser. Selenium IDE awalnya dikembangkan sebagai ekstensi untuk browser Firefox. Kemudian, dalam versi-versi berikutnya, pengembangannya diperluas sehingga mendukung browser seperti Google Chrome dan Firefox. Namun, saat ini dukungan untuk Google Chrome telah dihentikan, sehingga Selenium IDE hanya tersedia dan dapat digunakan melalui Firefox. Alat ini dirancang untuk memudahkan pengujian bagi pengguna yang tidak memiliki keterampilan pengkodean, tetapi juga fleksibel bagi penguji berpengalaman yang ingin menyesuaikan dan memperluas fungsionalitas pengujian.

Salah satu fitur utama dari Selenium IDE adalah rekam dan putar kembali (*Record and Playback),* yang memungkinkan penguji untuk merekam interaksi dengan aplikasi web tanpa menulis satu baris kode pun. Penguji hanya perlu menjalankan aplikasi web di browser dan melakukan serangkaian tindakan, seperti menekan tombol atau mengisi formulir, yang kemudian direkam oleh Selenium IDE. Setelah itu, penguji dapat memutar ulang langkah-langkah tersebut untuk memastikan bahwa aplikasi berfungsi sesuai dengan yang diharapkan. Fitur ini sangat berguna dalam pengujian fungsional, terutama untuk pengujian regresi dan pengujian yang melibatkan tindakan berulang.

BAB III

METODOLOGI PENELITIAN

3.1 Waktu dan Tempat Penelitian

Waktu dan tempat penelitian dilakukan pada:

1. Waktu penelitian: November 2024 - Juni 2025
2. Tempat penelitian: Gedung Jurusan Teknik Elektro Universitas Lampung

Tabel 3. 1 Jadwal Penelitian

No	Aktivitas	Bulan							
		Nov 2024	Des 2024	Jan 2025	Feb 2025	Mar 2025	Apr 2025	Mei 2025	Jun 2025
1	Pengumpulan jurnal dan referensi yang mendukung topik penelitian								
2	Penelitian literatur terkait perbandingan *automation testing tools*								
3	Analisis aplikasi dan kebutuhan								

No	Aktivitas	Bulan							
		Nov 2024	Des 2024	Jan 2025	Feb 2025	Mar 2025	Apr 2025	Mei 2025	Jun 2025
4	Penyusunan proposal penelitian								
5	Rancangan rencana penelitian pembuatan t*est case*								
6	Pembuatan daftar uji (*test case)*								
7	Eksekusi pengujian menggunakan Katalon dan Selenium								
8	Analisis hasil pengujian								
9	Penyusunan laporan *bug* yang ditemukan dan rekomendasi perbaikan								
10	Perbandingan kedua alat yang digunakan selama pengujian								

No	Aktivitas	Bulan							
		Nov 2024	Des 2024	Jan 2025	Feb 2025	Mar 2025	Apr 2025	Mei 2025	Jun 2025
11	Penyusunan laporan hasil perbandingan								

3.2 Perangkat Penelitian

Alat yang digunakan dalam penelitian ini meliputi:

Tabel 3. 2 Perangkat Penelitian

No	Komponen	Deskripsi
1.	Katalon Studio	Perangkat lunak yang digunakan untuk melakukan pengujian otomatis pada Siakadu Unila dan VClass Unila
2.	Selenium IDE	Perangkat lunak yang digunakan untuk melakukan pengujian otomatis pada Siakadu Unila dan VClass Unila
3.	XCode	Xcode digunakan untuk pengujian aplikasi *mobile* berbasis iOS, khususnya pada perangkat Apple.
4.	Microsoft Excel dan Word	Digunakan untuk mencatat dan mendokumentasikan hasil pengujian dari kedua *tools*

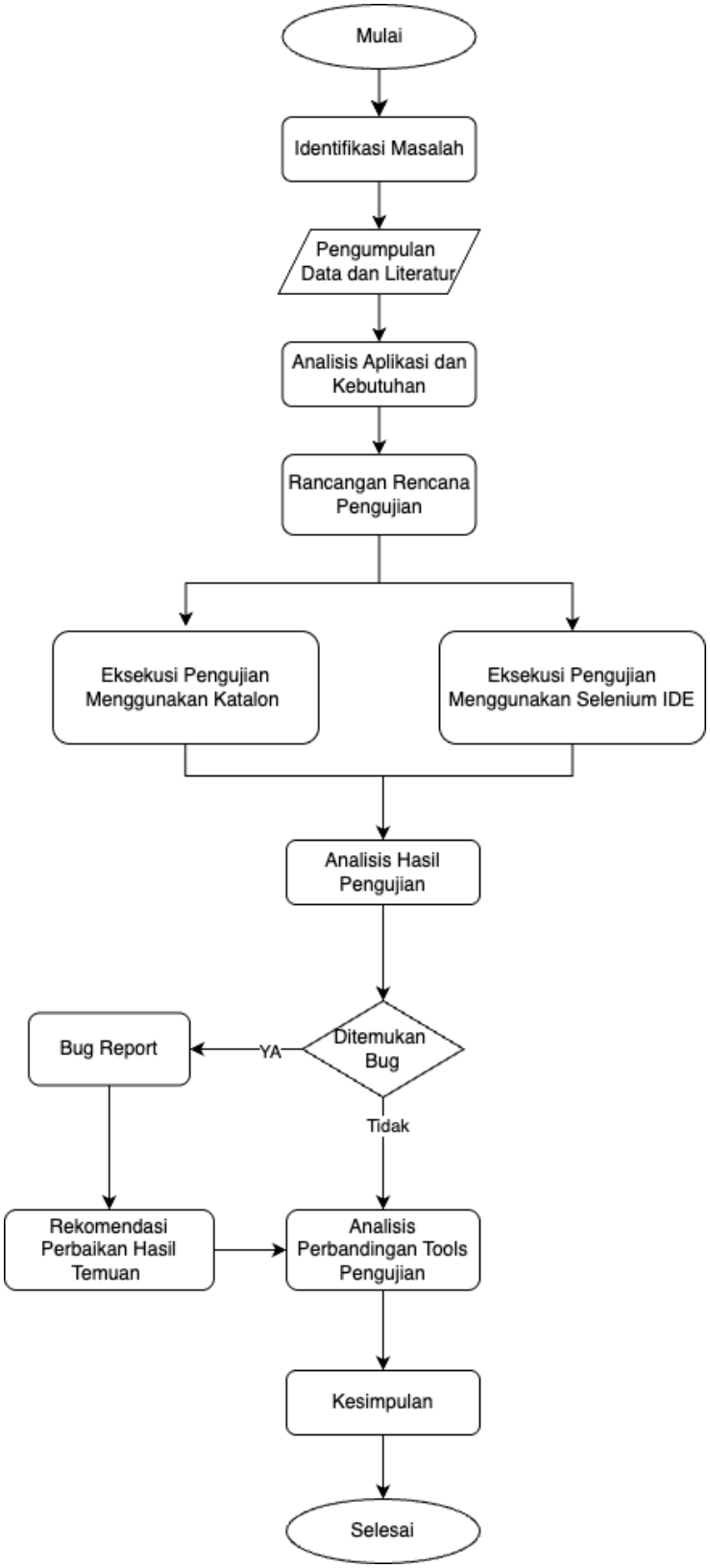
No	Komponen	Deskripsi
5.	MacBook Pro M1 2020	a. Processor : M1 b. Memory : 8 GB c. OS : MacOS Sonoma 14.5 d. Web Browser: Chrome, Safari dan Firefox Digunakan sebagai media penelitian.

3.3 Tahapan Penelitian

Gambar 3.1 menggambarkan proses sistematis untuk menganalisis dan membandingkan pengujian perangkat lunak menggunakan *tools automation* testing, yaitu Katalon Studio dan Selenium. Proses dimulai dengan identifikasi masalah, yang berfokus pada kendala atau kebutuhan yang dihadapi dalam pengujian perangkat lunak. Setelah itu, dilakukan pengumpulan data dan literatur untuk mendukung pemahaman mengenai metode dan *tools* yang digunakan.

Langkah selanjutnya adalah analisis aplikasi dan kebutuhan, yang bertujuan untuk menentukan cakupan dan spesifikasi pengujian. Berdasarkan analisis tersebut, dibuatlah rancangan rencana pengujian sebagai acuan untuk proses pengujian otomatis menggunakan Katalon Studio dan Selenium. Setelah melakukan pengujian, penulis akan melakukan analisis terkait hasil pengujian. Jika pengujian menemukan bug atau kesalahan, maka akan dihasilkan *bug report*, yang berisi detail temuan untuk ditindaklanjuti.

Jika diperlukan, rekomendasi perbaikan dari hasil temuan diberikan. Langkah terakhir adalah analisis perbandingan *tools* pengujian untuk mengukur efektivitas dan efisiensi antara Katalon Studio dan Selenium IDE. Hasil analisis ini kemudian dirangkum dalam kesimpulan, sehingga proses penelitian selesai.



Gambar 3. 1 Diagram Alir Tahapan Penelitian

3.4 Identifikasi Masalah

Dalam pengembangan perangkat lunak, pengujian adalah tahap yang sangat krusial untuk memastikan kualitas serta fungsionalitas aplikasi sesuai dengan kebutuhan pengguna. Namun, metode pengujian manual yang sering digunakan memiliki berbagai kendala, seperti waktu pelaksanaan yang lama, kebutuhan sumber daya manusia yang besar, dan keterbatasan dalam mendeteksi bug pada aplikasi kompleks. Maka dari itu, pengujian otomatis dapat menjadi pendekatan yang lebih efisien.

Penggunaan alat pengujian otomatis memungkinkan pengujian dilakukan secara berulang tanpa campur tangan manusia, sehingga menghemat waktu dan tenaga. Namun, penulis juga mengidentifikasi bahwa pemilihan alat pengujian otomatis memerlukan pertimbangan mendalam karena banyaknya alat yang tersedia dengan fitur, jenis pengujian, biaya, serta kompatibilitas yang berbeda. Berdasarkan penelitian terdahulu yang meneliti terkait *tools automation* testing yang paling banyak digunakan adalah Katalon dan Selenium, khususnya pada aplikasi berbasis *website*. Hal ini mendasari penelitian ini untuk membandingkan kedua *tools automation* testing tersebut dengan parameter-parameter yang telah ditentukan.

Parameter penting yang akan diuji terkait dengan pengujian antarmuka pengguna (GUI), pengujian *end-to-end*, pengujian lintas browser, dan kompatibilitas, terutama pada aplikasi dengan tingkat penggunaan yang tinggi.

Dengan menggunakan objek penelitian *website* akademik dari Universitas Lampung, yaitu SIAKADU dan VClass, diharapkan penelitian dapat membantu tim pengembang untuk menentukan alat pengujian otomatis yang sesuai dengan kebutuhan aplikasi untuk meningkatkan kepuasan pengguna khususnya dengan peran mahasiswa.

3.5 Pengumpulan Data dan Studi Literatur

Dalam melaksanakan penelitian ini, penulis membutuhkan berbagai materi, referensi, informasi, dan data yang relevan untuk mendukung analisis dan

pembahasan penelitian yang sedang dilakukan. Data dan informasi yang dikumpulkan harus memiliki keterkaitan dengan topik penelitian agar dapat memperkuat validitas serta keabsahan hasil penelitian yang dihasilkan.

Untuk memperoleh data yang dibutuhkan, penulis menggunakan metode studi literatur. Studi literatur dilakukan dengan cara membaca, meninjau, dan menganalisis berbagai penelitian sebelumnya yang memiliki kesamaan tema atau topik. Hal ini bertujuan untuk memperoleh sumber informasi yang dapat dijadikan sebagai acuan, landasan teori, serta dasar bagi pengembangan penelitian ini. Penulis memanfaatkan beragam jenis literatur, seperti skripsi, jurnal ilmiah baik nasional maupun internasional, buku teks, dan artikel ilmiah yang tersedia.

Selain itu, penulis juga memperhatikan kualitas dan kredibilitas literatur yang digunakan agar data yang diperoleh memiliki tingkat kepercayaan yang tinggi. Literatur yang dipilih tidak hanya relevan dengan materi penelitian, tetapi juga berasal dari sumber yang diakui, seperti *database* jurnal terindeks, dan repositori akademik. Dengan pendekatan ini, penulis berharap dapat menghasilkan uraian dan pembahasan yang mendalam, akurat, dan mendukung tujuan dari penelitian yang dilakukan.

3.6 Analisis Aplikasi & Kebutuhan

Pada tahap analisis kebutuhan, peneliti melakukan identifikasi kebutuhan spesifik untuk pengujian perangkat lunak seperti *automation tools* yang akan digunakan, serta objek penelitian, yaitu aplikasi SIAKADU dan VCLASS Unila. Analisis Katalon dan Selenium IDE dilakukan dengan menginstall kedua aplikasi pada perangkat yang akan digunakan untuk pengujian. Sedangkan analisis objek penelitian melibatkan penentuan modul atau fitur utama yang sering digunakan oleh pengguna dengan peran mahasiswa. Berdasarkan analisis yang telah dilakukan, berikut ini merupakan fitur utama yang akan diuji pada kedua website.

Tabel 3. 3 Fitur yang akan diuji

No.	Fitur	Keterangan
1	Login Siakadu	Fitur ini memastikan bahwa hanya pengguna yang telah terdaftar dapat mengakses sistem, sebagai bentuk validasi keamanan dan otentikasi pengguna.
2	Pengisian Kartu Rencana Studi pada Siakadu	Fitur ini memungkinkan mahasiswa untuk memilih dan mendaftarkan mata kuliah yang akan diambil pada semester berjalan.
3	Cetak Kartu Rencana Studi pada Siakadu	Fitur ini menyediakan opsi bagi mahasiswa untuk mencetak Kartu Rencana Studi (KRS) yang telah diisi sebagai dokumen pendukung akademik.
4	Akses Kartu Hasil Studi pada Siakadu	Mahasiswa dapat melihat hasil studi, termasuk nilai akhir dari mata kuliah yang telah diambil, melalui fitur ini.
5	Cetak Transkrip Nilai pada Siakadu	Fitur ini memberikan kemampuan kepada mahasiswa untuk mencetak transkrip nilai mereka sebagai dokumen resmi untuk keperluan akademik maupun non-akademik.
6	Logout pada Siakadu	Fitur ini memastikan bahwa pengguna dapat keluar dari sistem dengan aman setelah selesai menggunakan layanan Siakadu.
7	Login VClass	Fitur ini menyediakan akses bagi pengguna yang terdaftar untuk masuk ke platform pembelajaran daring VClass.
8	*Enrollment Class* di VClass	Fitur ini memastikan bahwa hanya mahasiswa yang memiliki *enrollment-key* yang dapat bergabung ke kelas daring tertentu yang sesuai dengan mata kuliah yang mereka ikuti.

No.	Fitur	Keterangan
9	Absensi kelas di *VClass*	Melalui fitur ini, mahasiswa dapat mengisi absensi kehadiran mereka pada pertemuan kelas daring.
10	Pengunggahan tugas di *VClass*	Fitur ini memfasilitasi mahasiswa untuk mengunggah tugas yang diberikan oleh dosen dalam bentuk *file* digital.
11	*Logout VClass*	Fitur ini memastikan bahwa pengguna dapat keluar dari sistem *VClass* dengan aman setelah selesai menggunakan *VClass*.

Kebutuhan teknis juga diidentifikasi selama analisis kebutuhan ini. Hal ini mencakup spesifikasi perangkat keras dan perangkat lunak yang diperlukan agar proses pengujian dapat berjalan dengan lancar, kemudian proses pengujian akan lebih terstruktur dan hasilnya akan lebih akurat dalam menggambarkan kualitas perangkat lunak secara keseluruhan.

3.7 Rancangan Rencana Pengujian

Rencana pengujian merupakan hal-hal yang harus dipersiapkan untuk panduan dalam proses pengujian perangkat lunak.

3.7.1 Pendekatan Pengujian

Pengujian dilakukan dengan metode *black-box testing*, yang berfokus pada pengujian fungsionalitas fitur berdasarkan masukan (*input*) dan keluaran (*output*) tanpa memeriksa kode sumbernya.

3.7.2 Lingkup Pengujian

Fitur-fitur yang diuji berdasarkan tabel sebelumnya:

Tabel 3. 4 Rancangan Fitur Pengujian

No.	Siakadu	VClass
1	Login	Login
2	Pengisian Kartu Rencana Studi	*Enrollment Class*
3	Cetak Kartu Rencana Studi	Absensi Kelas
4	Akses Kartu Hasil Studi	Pengunggahan Tugas
5	Cetak Transkrip Nilai	Logout
6	Logout	

3.7.3 Jenis Pengujian

Jenis pengujian yang akan dilakukan dalam rencana ini mencakup berbagai aspek untuk memastikan kualitas dan fungsionalitas sistem.

1. Pengujian Antarmuka Pengguna (GUI Testing) bertujuan untuk memastikan elemen visual, seperti tombol, menu, dan formulir, sesuai dengan desain dan berfungsi dengan baik.
2. Pengujian *End-to-End* akan menguji keseluruhan alur kerja sistem, mulai dari *login* hingga *logout*, untuk memastikan setiap fitur terintegrasi dengan benar dan berjalan sesuai harapan.
3. Pengujian Lintas Browser (*Cross-Browser Testing*) dilakukan untuk memeriksa kompatibilitas sistem pada berbagai browser populer seperti Chrome, Firefox, dan Safari. Pengujian ini penting untuk memastikan konsistensi tampilan dan fungsi di berbagai platform.
4. Pengujian Kompatibilitas dilakukan untuk memastikan bahwa sistem dapat berjalan lancar pada perangkat yang berbeda, seperti desktop dan ponsel. Pengujian ini akan mengidentifikasi potensi masalah yang mungkin terjadi di perangkat tertentu.

3.8 Eksekusi Pengujian Otomatis

Setelah rancangan *test case* selesai disusun, langkah selanjutnya dalam penelitian ini adalah melakukan eksekusi pengujian otomatis dengan menguji *website* dengan pengujian yang telah ditentukan. Pengujian otomatis dilakukan untuk memastikan bahwa aplikasi yang diuji, yaitu *website* VCLASS dan SIAKADU, berfungsi sesuai dengan spesifikasi yang telah ditetapkan berdasarkan parameter pengujian yang telah ditentukan. Dalam penelitian ini, penulis menggunakan dua alat pengujian otomatis, yaitu Katalon Studio dan Selenium.

Pada tahap ini, *test case* yang telah disusun sebelumnya akan dijalankan secara otomatis menggunakan kedua alat tersebut. Proses pengujian ini dilakukan dengan mengikuti urutan dan langkah-langkah yang telah ditentukan dalam *test case*, mulai dari memverifikasi *input* yang dimasukkan oleh pengguna, mengecek navigasi halaman, hingga memastikan bahwa sistem dapat memberikan hasil yang sesuai dengan ekspektasi pengguna. Setiap hasil pengujian, baik itu berhasil atau gagal, akan dicatat dan dianalisis untuk mendeteksi adanya *bug* atau masalah lainnya yang perlu diperbaiki.

3.9 Analisis Hasil Pengujian

Analisis hasil pengujian merupakan tahap krusial dalam proses evaluasi perangkat lunak untuk mengukur efektivitas dan efisiensi metode pengujian yang digunakan. Pada tahap ini, disusun ringkasan dari periode pengujian, tim yang terlibat, dan metodologi yang digunakan untuk memberikan pemahaman hasil pengujian. Lingkungan pengujian seperti perangkat keras, perangkat lunak, dan konfigurasi sistem juga dicatat untuk memahami faktor-faktor yang mempengaruhi hasil.

Dalam analisis ini, perhitungan *test case* menjadi elemen kunci. Jumlah *test cas*e yang berhasil dieksekusi tanpa kesalahan dicatat, serta jumlah yang gagal menunjukkan adanya *bug* atau kesalahan perangkat lunak. Hal ini memberikan gambaran tentang kualitas perangkat lunak yang diuji dan efektivitas metode

pengujian. Analisis waktu eksekusi juga dilakukan untuk membandingkan efisiensi kedua *automation tools* yang digunakan.

Kemudian, disusun daftar fitur-fitur perangkat lunak yang telah diuji beserta status pengujian masing-masing, yang memberikan wawasan tentang performa fitur-fitur tersebut. Terakhir, status pengujian diintegrasikan dalam bentuk persentase sebagai ringkasan dari semua *test case*, hasil pengujian, dan rekomendasi perbaikan, termasuk analisis risiko terkait fitur yang tidak berhasil diuji, untuk memberikan gambaran komprehensif tentang kualitas perangkat lunak yang dievaluasi.

Tabel 3. 5 Klasifikasi Status Pengujian

Status	**Deskripsi**
Passed	*Test case* berhasil dijalankan dan hasil aktual sesuai dengan hasil yang diharapkan.
Failed	*Test case* gagal dijalankan hasil aktual tidak sesuai dengan hasil yang diharapkan.
Skipped	*Test case* dilewati dan tidak dijalankan, karena langkah sebelumnya gagal atau kondisi tidak terpenuhi.
Error	Terjadi kesalahan teknis saat menjalankan *test cas*e, seperti masalah pada *environment*, atau koneksi.
Blocked	*Test case* tidak dapat dijalankan karena ada kendala eksternal.
Not Executed	*Test case* belum dijalankan karena belum dijadwalkan atau karena pertimbangan tertentu seperti fitur belum tersedia.

3.10 *Bug Report*

Jika terdapat kesalahan terdeteksi selama proses uji coba, maka *bug* atau kesalahan tersebut akan dicatat dalam bentuk laporan dokumentasi rinci. Laporan *bug* mencakup beberapa elemen kunci seperti *test case* ID untuk referensi spesifik ke skenario uji tertentu, deskripsi kesalahan agar tim developer memahami isu dengan jelas, serta tanggal dilakukan pengujian sebagai catatan waktu kejadian.

Selain itu, rincian tambahan seperti platform apa saja yang digunakan saat melakukan uji coba, termasuk sistem operasi (OS) dan browser juga dicantumkan agar developer bisa mereproduksi kondisi saat *bug* ditemukan dengan tepat. Tingkat *severity* (keparahan) dan prioritas juga dinyatakan pada laporan tersebut. Kemudian, diperlukan dokumentasi tahapan untuk mereproduksi *bug* agar tim developer dapat mengetahui dengan jelas cara mengatasi *bug* tersebut.

Dokumentasi *bug report* tidak hanya membantu tim developer dalam memperbaiki kesalahan tetapi juga berfungsi sebagai catatan historis bagi proyek tersebut di masa mendatang. Dengan adanya laporan terperinci tentang *bug-bug* sebelumnya beserta status perbaikannya memungkinkan tim untuk belajar dari kesalahan masa lalu dan mencegah terulangnya isu serupa di versi berikutnya dari perangkat lunak.

Klasifikasi dari tingkat *serverity* atau *keparahan* tercantum dalam tabel berikut [34].

Tabel 3. 6 Klasifikasi *Bug Severity*

S*everity*	**Deskripsi**	**Dampak pada Sistem**
Low	Cacat kecil yang tidak berdampak signifikan pada sistem.	Tidak memengaruhi fungsi atau kinerja sistem secara keseluruhan.
Minor	Kesalahan kecil yang menganggu, tapi tidak memengaruhi fungsionalitas utama	Sistem tetap berfungsi normal

Severity	**Deskripsi**	**Dampak pada Sistem**
Major	*Bug* yang memengaruhi fungsi penting dalam sistem	Beberapa fungsi inti terganggu, tapi sistem masih berjalan
Critical	*Bug* yang menyebabkan sistem gagal total	Sitem tidak dapat digunakan, fungsionalitas utama rusak

Klasifikasi *bug priority* atau tingkat prioritas penanganan tecantum dalam tabel berikut [34].

Tabel 3. 7 Klasifikasi *Bug Priority*

Priority	**Deskripsi**	**Tindakan yang Diperlukan**
Blocker	Harus ditangani segera. Bug memblokir seluruh sistem	Perbaikan segera, sebelum rilis dilanjutkan
Critical	Sangat penting, berdampak pada fungsi utama aplikasi	Harus diperbaiki dalam beberapa hari
Major	Penting tapi tidak darurat	Dapat diperbaiki di sprint selanjutnya
Minor	Tidak terlalu penting	Ditangani setelah bug prioritas tinggi
Trivial	Sangat kecil, tidak berdampak pada fungsi sistem	Diperbaiki jika ada waktu, Bisa diabaikan sementara

3.11 Rekomendasi Perbaikan Hasil Temuan

Berdasarkan hasil temuan dalam pengujian, rekomendasi perbaikan disusun untuk membantu pengembang perangkat lunak dalam meningkatkan kualitas aplikasi VCLASS dan SIAKADU Unila. Rekomendasi ini difokuskan pada prioritas

perbaikan *bug* yang ditemukan selama proses pengujian, dengan mempertimbangkan tingkat *severity* dan *priority* dari masing-masing *bug*. *Bug* yang memiliki tingkat *severity* tinggi, seperti yang dapat menyebabkan kerusakan sistem atau mengganggu fungsi utama aplikasi, akan ditempatkan pada urutan perbaikan pertama. Hal ini bertujuan untuk meminimalkan dampaknya terhadap pengalaman pengguna dan memastikan stabilitas aplikasi.

Selain itu, *bug* dengan *priority* tinggi, yang meskipun mungkin tidak memiliki *severity* yang sangat tinggi tetapi memiliki dampak besar pada penggunaan aplikasi sehari-hari, juga akan segera diperbaiki. Penyelesaian masalah yang terkait dengan *bug* untuk perbaikan yang lebih rendah, baik dari segi *severity* maupun *priority*, dapat dijadwalkan pada fase perbaikan lanjutan setelah masalah kritis diselesaikan.

Dengan melakukan perbaikan berdasarkan urutan prioritas dan *severity* yang tepat, diharapkan dapat meningkatkan kinerja dan keamanan aplikasi secara keseluruhan, serta memberikan pengalaman pengguna yang lebih optimal. Proses ini juga dapat mempercepat waktu pemulihan dari masalah yang terjadi, mengurangi potensi gangguan terhadap operasional pengguna, dan meningkatkan kepuasan pengguna aplikasi.

3.12 Analisis Perbandingan Alat Pengujian

Setelah menyelesaikan tahap pengujian dengan kedua *tools*, analisis perbandingan alat pengujian menjadi langkah penting untuk menentukan kelebihan dan kekurangan dari masing-masing alat pengujian. Dalam analisis ini, peneliti akan menilai berbagai aspek dari aplikasi berdasarkan parameter yang telah di tentukan. Parameter ini ditentukan sebagai tolak ukur perbandingan antar kedua alat yang digunakan. Penentuan parameter ini bertujuan untuk memberikan tolak ukur perbandingan yang objektif antara kedua alat yang digunakan, sehingga dapat memberikan gambaran yang jelas tentang alat mana yang lebih efektif dalam konteks pengujian perangkat lunak pada sistem yang diuji.

3.12.1 Parameter

Parameter yang digunakan pada penelitian ini mengacu pada penelitian terdahulu dimana aspek yang menjadi perbandingan adalah sebagai berikut [5], [17], [33]:

1. Kecepatan Eksekusi: Parameter ini mengukur seberapa cepat alat dapat menjalankan skrip pengujian.
2. Efektivitas Perekaman: Efektivitas perekaman mengacu pada kemampuan alat untuk merekam tindakan pengguna dan menghasilkan tahapan pengujian otomatis serta dapat memutar kembali rekaman tersebut.
3. Kemudahan Instalasi dan Penggunaan: Parameter ini mengevaluasi seberapa mudah alat tersebut digunakan oleh pengguna baru dan tingkat kompleksitas proses instalasinya. Aspek yang dinilai mencakup kejelasan dokumentasi, kemudahan konfigurasi awal, antarmuka pengguna yang intuitif, serta seberapa cepat pengguna baru dapat memahami dan mulai menggunakan fitur-fitur utama dari alat tersebut tanpa memerlukan pelatihan yang mendalam.
4. Kemampuan Membuat Skrip Pengujian: Parameter ini mengacu pada kemampuan alat dalam menghasilkan skrip pengujian secara otomatis berdasarkan tindakan pengguna atau input manual.
5. Laporan Hasil Pengujian: Mengacu pada kemampuan alat untuk menghasilkan laporan yang terstruktur dan informatif mengenai hasil pengujian. Laporan yang berkualitas mencakup informasi seperti jumlah tes yang berhasil, tes yang gagal, waktu eksekusi, dan detail kesalahan jika ada.
6. Biaya: Parameter ini mengukur biaya yang dibutuhkan untuk menggunakan alat pengujian, termasuk biaya lisensi, pelatihan, dan pemeliharaan. Biaya menjadi faktor penting, terutama untuk organisasi yang memiliki anggaran terbatas.
7. Kompatibilitas pengujian : Parameter ini menilai kemampuan alat uji otomatis dalam mendukung berbagai jenis aplikasi dan lingkungan pengujian. Parameter ini mencakup dukungan terhadap aplikasi berbasis web, desktop, *mobile*, serta sistem operasi dan peramban yang berbeda.

BAB V

KESIMPULAN DAN SARAN

5.1 Kesimpulan

Berdasarkan hasil penelitian dan pengujian, diketahui bahwa Selenium IDE dan Katalon Studio memiliki karakteristik yang berbeda dalam aspek efisiensi dan efektivitas pengujian. Selenium IDE menonjol dalam kecepatan eksekusi, efektivitas perekaman, dan kemudahan penggunaan sebagai ekstensi browser *open-source*. Meskipun mampu menyesuaikan perubahan elemen secara otomatis, Selenium IDE memiliki keterbatasan dalam menangani skenario pengujian kompleks dan pembuatan laporan yang mendalam.

Katalon Studio memiliki performa yang lebih efektif untuk pengujian yang kompleks. Dengan dukungan *scripting* lanjutan dan kemampuan integrasi lintas platform serta browser, Katalon Studio cocok untuk pengujian yang membutuhkan dokumentasi lengkap dan laporan uji yang komprehensif.

Oleh karena itu, pemilihan alat *automation testing* sebaiknya disesuaikan dengan kebutuhan teknis, skala pengujian, dan tingkat kompleksitas sistem yang diuji. Selenium IDE direkomendasikan untuk pengujian sederhana dan cepat, sementara Katalon Studio direkomendasikan untuk proyek yang menuntut pengujian menyeluruh, terdokumentasi, dan stabil di berbagai lingkungan.

5.2 Saran

Penelitian ini masih memiliki potensi untuk diperluas untuk membandingkan *automation testing tools* dengan berbagai aspek lain. Untuk pengembangan penelitian di masa mendatang, disarankan hal-hal berikut:

1. Menambahkan skenario pengujian regresi dalam evaluasi *automation testing tools* untuk melihat sejauh mana kemampuan *tools* dalam mengelola pengujian berulang ketika terjadi perubahan pada kode sistem.
2. Melibatkan uji beban *load testing* untuk mengukur performa *tools* dalam menangani simulasi banyak pengguna atau data secara bersamaan, serta untuk mengetahui dampaknya terhadap kestabilan dan kecepatan eksekusi.

DAFTAR PUSTAKA

[1] D. Sapto Prasetyo dan W. Silfianti, "Analisis Perbandingan Pengujian Manual dan Automation Testing pada Website E-Commerce," *Jurnal Informatika dan Teknologi,* vol. 2, no. 2, pp. 45-52, 2023.

[2] J. Lian Min, A. Istiqomah, A. Rahmani, "Evaluasi Penggunaan Manual dan Automated Software Testing pada Pelaksanaan End-To-End Testing," *Jurnal Teknologi Terapan*, vol. 6, no. 1, pp. 12-20, 2020.

[3] Y. Kosasih dan A. B. Cahyono, "Automation Testing Tool dalam Pengujian Aplikasi The Point of Sale (Studi Kasus TPOS PT. Javasigna Intermedia)," *Jurnal Teknologi dan Sistem Komputer, vol. 9, no. 1, pp. 1-8, 2021.*

[4] M. Khan dan A. Salam, "Comparative Analysis of Automated Software Testing Tools," *International Journal of Computer Applications*, vol. 139,no. 9, pp. 1-5, 2016. [Online]. Available: https://www.researchgate.net/publication/373389405

[5] H. V. Gamido dan M. V. Gamido, "Comparative Review of the Features of Automated Software Testing Tools," *International Journal of Electrical and Computer Engineering,* vol. 9, no. 5, pp. 4473–4478, Oct. 2019, doi: 10.11591/ijece.v9i5.pp4473-4478.

[6] Refano Trinanda Saputra, Hariz Farizi, dan Mochamad Chandra Saputra, "Analisis Perbandingan Performansi Tool Testing Antara Appium dan Katalon dalam Pengujian Otomatisasi Perangkat Lunak pada Aplikasi Berbasis Mobile," J*urnal Teknologi dan Sistem Komputer*, vol. 5, no. 2, pp. 100-110, 2017.

[7] Y. Kumar dan Y. Kumar, "Comparative Study of Automated Testing Tools: Selenium, SoapUI, HP Unified Functional Testing and TestComplete," *International Journal of Computer Applications*, vol. 115, no. 12, pp. 1-5, 2015. [Online]. Available: https://www.researchgate.net/publication/354191481

[8] A. Indrayanti, B. A. Wardijono, D. Nur, dan R. Aulia, "Analisis Pengujian Graphical User Interface E-Commerce dengan Menggunakan Katalon Studio," *Seminar Nasional Teknologi Informasi dan Komunikasi STI&K (Sentik),* vol. 5, no. 1, pp. 15-20, 2021.

[9] S. E. Anggrainy dan A. H. Muhammad, "Efficiency Development Analysis Use of Automation Testing Using Katalon on Mobile-Based Applications (Case Insurance Company)," *International Journal of Information System & Technology Akreditasi*, vol. 8, no. 158, pp. 54–59, 2024.

[10] Gesang Ibnu Safaat dan Viany Utami Tjhin, "Analysis of Quality Assurance Performance in the Application of Manual Testing and Automation Testing for Software Product Testing," *Indonesian Interdisciplinary Journal of Sharia Economics (IIJSE)*, vol. 7, no. 2, pp. 1987–1996, 2024.

[11] Ahmad Jamalludin, Aliv Rivaldi, Faizal Maulana, dan Riyan Kristian, "Otomatisasi Pengujian Aplikasi Blibli Menggunakan Selenium IDE," *Jurnal Batirsi,* vol. 8, no. 1, pp. 2502–3691, Jul. 2024.

[12] M. A. Jamil, M. Arif, N. S. A. Abubakar, dan A. Ahmad, "Software Testing Techniques: A Literature Review," *Institute of Electrical and Electronics Engineers (IEEE)*, Jan. 2017, pp. 177–182, doi: 10.1109/ICT4M.2016.045.

[13] R. Swanda Narastu, S. Zai, V. Handrian us Pranatawijaya, "Analisis Kualitas dan Penerapan Software Quality Assurance pada Web K24klik Menggunakan Model ISO/IEC 9126," J*urnal Teknik Informatika*, vol. 10, no. 1, pp. 30-40, 2024.

[14] Rahmat Fauzan, Ferina Putri Soedjono, Annisa Ayu Permadani, dan Muhammad Ainul Yakin, "Perbandingan Pengujian Manual dan Terotomasi pada Software Enterprise Resource Planning," *Journal of Advances in Information and Industrial Technology,* vol. 5, no. 1, pp. 23– 30, May 2023, doi: 10.52435/jaiit.v5i1.318.

[15] Maulidatul Muauwanah, I. Romadloni, M. A. Muqid, M. H. Al Arif, dan R. Purbaningtyas, "Pengujian Kualitas Perangkat Lunak Website Jurusan Teknologi Informasi Politeknik Negeri Jember Menggunakan ISO 9126," *Jurnal Teknik Informatika dan Teknologi Informasi*, vol. 3, no. 3, pp. 01–14, Oct. 2023, doi: 10.55606/jutiti.v3i3.2798.

[16] Utor, "GUI Testing: What, Why, How?" [Online]. Available: https://utor.com/topic/gui-testing

[17] S. Melia dan F. P. Putra, "Comparative Analysis of Automated Testing Tools on GUI Web-Based Application," International *Journal of Computer Applications, Aug.* 2023, [Online]. Available: https://journal.irpi.or.id/index.php/sentimas

[18] Muhtadi, Matin & Friyadi, Moch & Rahmani, Ani, "Analisis GUI Testing pada Aplikasi E-Commerce Menggunakan Katalon," *Jurnal Teknologi dan Sistem Komputer,* 2019, [Online]. Tersedia: 10.35313/irwns.v10i1.1443.

[19] M. Ghozy Alkhairi, S. Putri, A. Alkadri, Y. Utami, dan M. G. Alkhairi, "Implementasi Unit Testing dan End-To-End Testing pada Sistem Informasi Akademik Teknik Informatika," *JIPI (Jurnal Ilmiah Penelitian dan Pembelajaran Informatika),* vol. 9, no. 4, pp. 2208–2219, 2024, doi: 10.29100/jipi.v9i4.5626.

[20] Zhimin Zhan, "Cross Browser Testing Clarified." [Online] Available: https://zhiminzhan.medium.com/

[21] B. Kaalra dan K. Gowthaman, "Cross Browser Testing Using Automated Test Tools," *International Journal of Advanced Computer Science and Applications,* 2014, [Online]. Available: www.ijascse.org

[22] Merina, C, "Analisis Perbandingan Kinerja Test Automation Framework untuk Functional Testing pada Aplikasi Berbasis Android dengan Metode The Distance to the Ideal Alternative," 2017, [Online]. Available: https://repository.uinjkt.ac.id/dspace/handle/123456789/53151

[23] Umar, Mubarak Albarka & Chen, Zhanfang, “A Study of Automated Software Testing: Automation Tools and Frameworks,” 2019, [Online]. Available: 10.5281/zenodo.3924795.

[24] S. Sri Rahayu, D. Aris Firmansyah, S. Susanti, dan U. Adhirajasa Reswara Sanjaya Bandung, “Analisis Penggunaan Tools Automation Testing pada Aplikasi: Systematic Literature Review,” Remik: Riset dan *E-Jurnal Manajemen Informatika Komputer*, vol. 8, no. 1, 2024, doi: 10.33395/remik.v8i1.13241.

[25] Y. Kosasih dan A. B. Cahyono, “Automation Testing Tool dalam Pengujian Aplikasi The Point of Sale (Studi Kasus TPOS PT. Javasigna Intermedia),” *Jurnal Teknologi dan Sistem Komputer*, vol. 9, no. 1, pp. 1-8, 2021.

[26] I. Nurkholis dan A. H. Yunial, “Automation Testing Tool dalam Pengujian Website Sistem Informasi Pembayaran Iuran dan LKS di MTs Al-Ittihad Menggunakan Puppeteer,” *Jurnal Media Publikasi,* [Online]. Available : https://journal.mediapublikasi.id/index.php/logic

[27] Karinka Priskila, “Automation Testing and How to Start,” [Online]. Available: https://icehousecorp.com/automation-testing-and-how-tostart/

[28] Katalon Studio, “Top Automation Testing Tools for 2025,” [Online]. Available: www.katalon.com

[29] Y. Kosasih dan A. B. Cahyono, “Automation Testing Tool dalam Pengujian Aplikasi The Point of Sale (Studi Kasus TPOS PT. Javasigna Intermedia),” *Jurnal Teknologi dan Sistem Komputer*, vol. 9, no. 1, pp. 18, 2021.

[30] Katalon, “Katalon Studio Official Web Page.” [Online] Available: www.katalon.com

[31] Selenium, “Selenium Ide Official Web Page.” [Online] Available: www.selenium.dev

[32] A. Pratama Ginting, Z. Abidin, A. Asari, dan A. Saifudin, "Otomatisasi Pengujian Aplikasi Web Toko Sembako Menggunakan Selenium IDE," Jurnal Media Publikasi, [Online]. Available: https://journal.mediapublikasi.id/index.php/logic

[33] M. Kaur dan R. Kumari, "Comparative Study of Automated Testing Tools: TestComplete and QuickTest Pro," *International Journal of Computer Applications,* 2011.

[34] S. Desikan and G. Ramesh, Software Testing: Principles and Practices. New Delhi: Pearson Education, 2006, pp. 435.

[35] F. E. Gunawan, "Editorial Page and Table of Content", *CommIT (Comm unication and Information Technology) Journal*, vol. 11, no. 1, Aug. 20 17.