

Pengujian Aplikasi *Mobile* untuk Lelang Mobil dengan Metode *Black Box* menggunakan *Automation Testing Tool*

Geralda Ken Sadhani Tresnavitane¹, Paulus Mudjihartono², Yulius Harjoseputro³
Program Studi Informatika, Fakultas Teknologi Industri, Universitas Atma Jaya Yogyakarta
Jl. Babarsari No. 43, Caturtunggal, Kab. Sleman, 55281, Daerah Istimewa Yogyakarta, Indonesia
Email: ¹geraldakensadhani@gmail.com, ²paulus.mudjihartono@uajy.ac.id,
³yulius.harjoseputro@uajy.ac.id

Abstrak. Aplikasi ACCBid merupakan aplikasi lelang mobil secara online yang dirilis oleh Astra Credit Companies pada tahun 2018. ACCBid terus melakukan pengembangan pada fitur-fitur yang dimilikinya dan untuk memastikan fitur tersebut berjalan dengan baik, diperlukan adanya pengujian untuk menjamin kualitas aplikasi tetap terjaga. Dalam penelitian ini, dilakukan pengujian secara manual dan otomatis menggunakan Katalon Studio sebagai automation testing tool terhadap 14 dari 47 fungsi yang ada pada aplikasi ACCBid menggunakan metode black box. Penelitian ini secara lebih lanjut membandingkan keefektifan pengujian otomatis dengan pengujian manual. Berdasarkan pengujian yang dilakukan, ditemukan bahwa 1 dari 14 fungsi masih belum berjalan baik dengan tingkat kepastian kebenaran fungsionalitas sebesar 92.86%, serta pengujian otomatis tidak cukup efektif dibandingkan pengujian manual dari segi waktu dengan rata-rata persentase penurunan waktu sebesar 48.15177%. Meskipun demikian, pengujian otomatis menggunakan Katalon Studio memiliki keunggulan yaitu mudah digunakan, meminimalisir kesalahan manusia serta pelaporan hasil pengujian yang di-generate secara otomatis.

Kata Kunci: Black Box Testing, ACCBid, Katalon Studio, pengujian manual, pengujian otomatis

1. Pendahuluan

1.1. Latar Belakang

Perkembangan *smartphone* yang semakin pesat di era ini salah satunya disebabkan oleh berbagai macam aplikasi-aplikasi *mobile* yang dapat semakin mempermudah manusia. Pada kuartal ke-4 tahun 2019, Statista mencatat jumlah aplikasi yang tersedia pada Google Play Store mencapai 2,57 juta sedangkan Apple's App Store hadir dengan hampir 1,84 juta aplikasi yang tersedia [1]. Hal ini menunjukkan bahwa pengguna *smartphone* sebagai *end user* sudah dimanjakan dengan begitu banyaknya aplikasi yang dapat mereka unduh baik secara gratis maupun berbayar pada ponsel cerdas mereka. Akan tetapi, tentunya dalam proses pengembangan aplikasi *mobile* tidak terlepas dari kesalahan. Maka dari itu, sebelum aplikasi tersebut dapat dirilis atau dilakukan pembaharuan, diperlukan adanya pengujian perangkat lunak untuk meminimalisir kesalahan pada aplikasi-aplikasi tersebut.

Berbagai metode pengujian perangkat lunak telah diciptakan, salah satunya adalah metode pengujian *black box*. Metode ini hanya berfokus pada keluaran yang dihasilkan tanpa mengetahui *source code* dari aplikasi tersebut. Metode *black box* sebagai salah satu metode yang paling banyak diminati untuk lingkungan pengujian saat ini telah banyak menghasilkan analisa dan kesimpulan mengenai penggunaannya. Hal tersebut telah dilakukan oleh Ningrum dkk dalam penelitiannya yang digunakan untuk menguji aplikasi sistem seleksi sales terbaik. Dari penelitian ini, telah berhasil mendapatkan kesimpulan bahwa metode *black box* dapat membantu dalam proses pembuatan *case* pengujian, pengujian kualitas serta menemukan kesalahan yang tidak terdeteksi yang disebabkan oleh kesalahan pengetikan [2]. Melihat keberhasilan yang telah dilakukan [2], maka metode *black box* akan digunakan juga untuk pengujian aplikasi mobile ACCBid.

ACCBid adalah sebuah aplikasi mobile untuk lelang mobil secara online yang dapat dilakukan dimanapun dan kapanpun sesuai dengan event lelang yang tersedia. Aplikasi ini menyediakan fitur-fitur menarik seperti penawaran secara otomatis, pengadaan lelang secara tertutup, fasilitas pembiayaan yang dapat dilakukan secara kredit bagi pemenang lelang, dan beragam fitur lainnya yang mempermudah penggunaannya dalam melakukan lelang [3]. Meskipun demikian, aplikasi yang dikembangkan oleh Astra Credit Companies di bawah naungan PT. Astra Sedaya Finance ini masih tidak luput dari kesalahan-kesalahan baik kesalahan *interface*, input-output maupun ketidaksesuaian fungsionalitas yang terdapat pada aplikasi tersebut. Sehingga perlu dilakukan pengujian untuk mengidentifikasi kesalahan-kesalahan yang ada.

Pengujian pada perangkat lunak dapat dilakukan baik secara manual maupun otomatis. Pengujian manual adalah sebuah metode pengujian dimana seorang *tester* akan berperan dalam setiap proses pengujianya yaitu melakukan inisialisasi pada test, berinteraksi, menganalisa hasil pengujian dan juga membuat laporan terhadap hasil pengujian yang telah dilakukan [4]. Akan tetapi, pengujian secara manual tidak cukup efektif [5] dan diperlukan juga pengujian secara otomatis. Maka dari itu, pada penelitian ini penulis membahas mengenai “Pengujian Aplikasi *Mobile* untuk Lelang Mobil dengan Metode *Black Box* menggunakan *Automation Testing Tool*”. Penelitian ini menerapkan pengujian secara otomatis menggunakan Katalon Studio sebagai *Automation Testing Tool* pada fungsionalitas yang terdapat pada aplikasi *ACCBid*. Penelitian ini bertujuan untuk memastikan bahwa fungsionalitas yang diuji pada aplikasi telah berjalan dengan baik serta melihat perbandingan tingkat keefektifan dari pengujian secara otomatis dibandingkan pengujian manual serta kelebihan dan kekurangan dari masing-masing tipe pengujian.

2. Tinjauan Pustaka

Dalam penelitian ini, penulis mengumpulkan informasi dari penelitian-penelitian sebelumnya untuk dijadikan bahan perbandingan dan acuan dalam proses pengujian. Penulis juga mengumpulkan informasi dari jurnal, buku-buku dan skripsi yang berkaitan dengan judul penelitian ini untuk mendapatkan landasan teori ilmiah.

Beberapa penelitian menyangkut pengujian perangkat lunak dengan *black box* testing bukan merupakan suatu hal yang baru. Penelitian menggunakan metode ini pernah dilakukan oleh Yasa pada tahun 2013 berjudul “Analisis dan Pengujian Perangkat Lunak dengan Metode *Black Box*, Studi Kasus BRS Online Universitas Sanata Dharma”. Penelitian tersebut bertujuan untuk melakukan analisa terhadap setiap fungsi pada modul yang terdapat di BRS Online Universitas Sanata Dharma. Dalam penelitiannya, didapatkan hasil bahwa pengujian yang dilakukan dengan metode *black box* terbukti efektif untuk menguji perangkat lunak BRS Online ini dikarenakan mampu memecahkan persoalan yang ada serta mampu memberikan solusi dari permasalahan tersebut [6].

Penelitian lainnya dilakukan oleh Purnomo pada tahun 2017 dengan judul “Software Testing Aplikasi Website PT Gramedia menggunakan Metode *Black Box* pada PT WGS Bandung” dengan tujuan untuk mengukur kualitas perangkat lunak berdasarkan metode *black box* testing pada aplikasi berbasis web PT. Gramedia Pustaka Utama. Hasil dari penelitian ini didapatkan bahwa pengujian pada situs website Gramedia telah terpenuhi/sukses secara keseluruhan sesuai dengan skenario/*user story* [7].

Penelitian yang hampir serupa dilakukan oleh Andry dan Reinaldo pada tahun 2017 dengan judul “Testing dan Implementasi Aplikasi Parkir di PT ABC Menggunakan Metode *Black Box*”. Penelitian ini bertujuan untuk mengukur kualitas perangkat lunak berdasarkan metode *black box testing* pada aplikasi parkir di PT. ABC. Penelitian tersebut menghasilkan kesimpulan bahwa pengujian *black box* yang telah dilakukan pada aplikasi parkir telah terpenuhi atau sukses secara keseluruhan sesuai dengan skenario atau *user acceptance test* (UAT) yang telah dilakukan [8].

Pada tahun 2018, Cholifah dkk melakukan penelitian dengan judul “Pengujian *Black Box Testing* pada Aplikasi *Action & Strategy* Berbasis Android dengan Teknologi *Phonegap*”. Penelitian ini membahas mengenai pengujian yang dilakukan secara menyeluruh terhadap

interface aplikasi *Action & Strategy* berbasis Android untuk mengetahui kepastian kebenaran fungsional sistem dan mengevaluasi kesesuaian aplikasi dengan kebutuhan pengguna. Pengujian dilakukan secara manual oleh 16 orang anggota BRIMOB dan instruktur sebagai pengguna aplikasi dan sistem administrator. Dari penelitian ini tidak ditemukan adanya kesalahan pada *interface* sehingga sistem aplikasi dapat dikatakan telah berjalan sesuai dengan tujuan yang diharapkan [9].

Penelitian lainnya berjudul “*Automation Testing Tool* dalam Pengujian Aplikasi Belajar Tajwid pada Platform Android” yang dilakukan oleh Herlinda dkk pada tahun 2019 dilakukan dengan cara melakukan record dan playback. Tujuan dari penelitian ini untuk menganalisis efektivitas aplikasi yang sudah dibangun dengan menggunakan pengujian otomatis menggunakan Katalon. Dari penelitian ini dapat diketahui bahwa secara garis besar Katalon sudah memenuhi fungsinya sebagai *Automation Testing tool* yang baik untuk melakukan pengujian otomatis meskipun masih terdapat beberapa kekurangan [10].

Dalam penelitian yang dilakukan oleh Muhtadi, dkk pada tahun 2019 berjudul Analisis GUI Testing pada Aplikasi *E-Commerce* menggunakan Katalon” dilakukan pengujian secara otomatis menggunakan Katalon untuk menganalisis pengaruh kompleksitas rancangan GUI terhadap *response time* dari beberapa halaman web. Dari penelitian ini dapat diketahui bahwa kompleksitas rancangan GUI tidak sepenuhnya mempengaruhi *response time* dari sebuah halaman GUI website.

3. Metodologi Penelitian

Berikut adalah lima tahapan yang digunakan dalam penelitian ini: (1) Tahap pertama merupakan studi pustaka dimana pada tahap ini dilakukan pencarian referensi-referensi dan sumber pustaka yang memiliki keterkaitan dengan penelitian yang dilakukan. Sumber pustaka tersebut berkaitan dengan pengujian perangkat lunak, *black box testing*, *automated testing tool* serta referensi lainnya yang mendukung penulisan. Langkah ini diharapkan dapat memberikan pengetahuan dan teori yang berguna bagi penulis agar mendapatkan hasil penelitian yang sesuai. (2) Tahap kedua adalah analisis kebutuhan pengujian yang dilakukan guna menemukan alur berjalannya masing-masing fungsi sehingga nantinya dapat digunakan untuk merancang *test case*. (3) Tahap ketiga yaitu perancangan skenario *test case* dimana dalam fase ini dilakukan penulisan *test case* pengujian berdasarkan analisis kebutuhan yang telah dirancang pada tahap sebelumnya. Pada tahap ini akan dibentuk skenario positif dan skenario negatif. Skenario positif merancang pengujian dengan data-data yang valid, sementara skenario negatif dilakukan dengan data-data yang invalid. Pembentukan skenario positif dan skenario negatif ini sangat penting agar pengujian tidak hanya memastikan *flow* positif dapat berjalan secara normal, akan tetapi juga memastikan bahwa sistem dapat mengendalikan *flow* negatif dengan baik [11]. (4) Tahap keempat dilakukan pengujian secara manual. Pengujian pada tahapan ini tidak menggunakan bantuan *automation testing tool* dan dilakukan dengan memperhatikan atribut yang telah ditentukan. Pada tahap ini dilakukan pemberian status terhadap *test case* yang ada yaitu ‘Sesuai Harapan’ ketika fungsi telah berjalan sesuai dengan kebutuhan, dan ketika sebaliknya status akan menjadi ‘Tidak Sesuai Harapan’. Pada tahap ini juga dibuat laporan mengenai hasil pengujian yang telah dilakukan. (5) Tahap selanjutnya ialah pengujian secara otomatis yang dilakukan dengan bantuan Katalon Studio sebagai *automation testing tool*. Pada tahap ini, dilakukan juga pembuatan *test data* dan pembuatan *script automation test* merujuk pada *test case* yang ada. Pengujian pada tahapan ini dilakukan dengan memperhatikan atribut yang telah ditentukan. Pada tahap ini dilakukan pemberian status terhadap *test case* yang ada yaitu ‘Pass’ ketika fungsi telah berjalan sesuai dengan kebutuhan, dan ketika sebaliknya status akan menjadi ‘Fail’. Pada tahap ini laporan mengenai hasil pengujian diambil dari hasil eksekusi yang telah di-*generate* secara otomatis pada Katalon Studio pada akhir pengujian, dan juga pada Katalon TestOps untuk melakukan analisis lebih lanjut terhadap hasil pengujian.

4. Hasil dan Diskusi

Dari hasil pengujian yang telah dilakukan, dapat diketahui bahwa kepastian kebenaran fungsionalitas dari 14 fungsi yang diuji pada sistem ACCBid mendapatkan persentase senilai 92.86% dimana hal ini menunjukkan bahwa 1 dari 14 fungsi masih belum berjalan dengan baik. Maka dari itu, aplikasi ini masih membutuhkan perbaikan pada *bug* yang ditemukan sehingga menghasilkan kualitas aplikasi yang lebih baik ketika digunakan oleh *customer* ACCBid. Berdasarkan hasil pengujian yang telah dilakukan, berikut adalah perbandingan durasi waktu yang dibutuhkan untuk melakukan pengujian manual dan pengujian otomatis yang telah tercantum pada Tabel 1.

Tabel 1. Perbandingan Waktu Pengujian Manual dan Pengujian Otomatis

No	Fungsi	Waktu Pengujian		Selisih Waktu
		Manual	Otomatis	
1.	Login	4 menit 19 detik	6 menit 47 detik	2 menit 28 detik
2.	Ubah No. Handphone	3 menit 49 detik	5 menit 27 detik	1 menit 38 detik
3.	Ubah Kata Sandi	13 menit 14 detik	15 menit 56 detik	2 menit 42 detik
4.	Unggah Dokumen	2 menit 33 detik	4 menit 13 detik	1 menit 40 detik
5.	Tambah Akun Bank	12 menit 15 detik	15 menit 38 detik	3 menit 23 detik
6.	Hapus Akun Bank	50 detik	1 menit 15 detik	25 detik
7.	Top Up Saldo (Astria)	1 menit 46 detik	2 menit 28 detik	42 detik
8.	Top Up Saldo (IBID)	13 menit 55 detik	19 menit 54 detik	5 menit 59 detik
9.	Jadwal Lelang	5 menit 22 detik	8 menit 46 detik	3 menit 24 detik
10.	Cari Mobil	5 menit 12 detik	7 menit 33 detik	2 menit 21 detik
11.	Ikut Lelang Terbuka – Bid Otomatis	4 menit 15 detik	6 menit 11 detik	1 menit 56 detik
12.	Ikut Lelang Terbuka – Bid Manual	2 menit 36 detik	4 menit 6 detik	1 menit 30 detik
13.	Ikut Lelang Tertutup	3 menit 7 detik	5 menit 24 detik	2 menit 17 detik
14.	Kalkulator	6 menit 59 detik	10 menit	3 menit 1 detik

Pada Tabel 1, dapat dilihat bahwa waktu pengujian manual secara keseluruhan lebih cepat dibandingkan waktu untuk melakukan pengujian otomatis. Apabila ditinjau lebih lanjut, dalam menjalankan 14 fungsi ini total selisih waktu yang dimiliki untuk melakukan pengujian otomatis dan pengujian manual cukup jauh yaitu sebesar 2006 detik atau sekitar 33 menit 26 detik. Hal ini menandakan bahwa dari segi waktu, pengujian manual jauh lebih efektif dibandingkan pengujian otomatis. Berdasarkan tabel 1 pula, dapat dilakukan perhitungan untuk persentase penurunan waktu eksekusi dengan rumus seperti yang terlihat pada rumus 1.

$$\text{Persentase Penurunan Waktu} = \frac{\text{Waktu Pengujian Otomatis} - \text{Waktu Pengujian Manual}}{\text{Waktu Pengujian Manual}} \quad (1)$$

Perhitungan yang telah dilakukan untuk penurunan waktu eksekusi dan peningkatan kecepatan pengujian otomatis dibandingkan pengujian manual dapat dilihat pada Tabel 2.

Tabel 2. Persentase Penurunan Waktu Eksekusi Pengujian Otomatis Dibandingkan Pengujian Manual

No	Fungsi	Waktu Pengujian (detik)		Persentase Penurunan Waktu Eksekusi (%)	Keterangan
		Manual	Otomatis		
1.	Login	259	407	57.14286	1.571x lebih lambat
2.	Ubah No. Handphone	229	327	42.79476	1.428x lebih lambat
3.	Ubah Kata Sandi	794	956	20.40302	1.204x lebih lambat
4.	Unggah Dokumen	153	253	65.35948	1.654x lebih lambat
5.	Tambah Akun Bank	735	938	27.61905	1.276x lebih lambat
6.	Hapus Akun Bank	50	75	50	1.5x lebih lambat
7.	Top Up Saldo (Astria)	106	148	39.62264	1.396x lebih lambat

No	Fungsi	Waktu Pengujian (detik)		Persentase Penurunan Waktu Eksekusi (%)	Keterangan
		Manual	Otomatis		
8.	Top Up Saldo (IBID)	835	1194	42.99401	1.43x lebih lambat
9.	Jadwal Lelang	322	526	63.35404	1.634x lebih lambat
10.	Cari Mobil	312	453	45.19231	1.452x lebih lambat
11.	Ikut Lelang Terbuka – Bid Otomatis	255	371	45.4902	1.455x lebih lambat
12.	Ikut Lelang Terbuka – Bid Manual	156	246	57.69231	1.577x lebih lambat
13.	Ikut Lelang Tertutup	187	324	73.26203	1.733x lebih lambat
14.	Kalkulator	419	600	43.19809	1.432x lebih lambat
Rata-Rata				48.15177	1.482x lebih lambat

Dari hasil yang didapatkan pada Tabel 2, didapatkan rata-rata persentase penurunan waktu eksekusi untuk 14 fungsi yang diuji adalah 48.15177%. Sehingga, secara keseluruhan pengujian otomatis pada Katalon Studio dalam penelitian ini menunjukkan penurunan waktu dimana rata-rata durasi waktu untuk pengujian otomatis 1.482x lebih lambat dibandingkan pengujian secara manual.

Analisis lebih lanjut dilakukan terhadap data yang tercantum pada Tabel 1 dan 2. Hal ini meninjau faktor apa saja yang membuat pengujian otomatis pada Katalon Studio memakan waktu lebih lama dibandingkan pengujian manual. Maka ditemukan salah satu faktor yang mempengaruhi lama pengujian otomatis yaitu pada saat melakukan *StartApplication* atau memulai aplikasi pada *device smartphone*. *StartApplication* pada Katalon sendiri dilakukan untuk melakukan *setting* seperti yang ditunjukkan pada Gambar 1.

```

1 - callTestCase(findTestCase("StartApplication"), [:], STOP_ON_FAILURE) (18.785s)
  1 - Test Cases/StartApplication (18.668s)
    1 - appPath = relativeToAbsolutePath(ApkName, getProjectDir()) (0.059s)
    2 - startApplication(appPath, false) (18.554s)
      User set preference: ['unicodeKeyboard', 'true']
      User set preference: ['deviceld', 'R9CMB06TMDJ']
      Appium server started on port 56,318
      appiumVersion = 1.12.1
      appiumServerStatus = {"status":0,"value":{"build":{"version":"1.12.1"},"sessionId":null}}
      deviceld = R9CMB06TMDJ
      deviceName = samsung SM-A207F (Android 9)
      deviceModel = SM-A207F
      deviceManufacturer = samsung
      deviceOS = Android
      deviceOSVersion = 9

```

Gambar 1. Contoh StartApplication pada Katalon Studio Saat Menjalankan Pengujian

Gambar 1 menunjukkan bahwa pada *StartApplication* dilakukan pengaturan dan pengenalan pada perangkat *smartphone* yang digunakan untuk melakukan pengujian, di antaranya pengaturan untuk *keyboard* menjadi *unicodeKeyboard*, *peng-install-an* atau menjalankan Appium, dan lain sebagainya. Proses ini memakan waktu yang cukup lama dan dijalankan secara berulang setiap kali ingin menuju ke *data binding* berikutnya pada saat menjalankan *test suite* pada sebuah fungsi yang dilakukan pengujian. Hal ini berbeda dengan pengujian manual yang tidak membutuhkan *setting* setelah melakukan *peng-install-an* aplikasi. Maka dari itu, dilakukan analisis terhadap waktu untuk menjalankan *StartApplication* pada salah satu fungsi yang diuji yaitu Ubah Kata Sandi seperti yang terlihat pada Tabel 3.

Tabel 3. Perbandingan Waktu untuk Membuka Aplikasi / Menjalankan StartApplication pada Test Case ‘Ubah Kata Sandi’.

<i>Data binding ke-</i>	Waktu untuk membuka aplikasi / <i>StartApplication</i>	
	Pengujian Manual	Pengujian Otomatis
1.	3 detik	17 detik
2.	2 detik	15 detik
3.	2 detik	15 detik
4.	3 detik	15 detik
5.	2 detik	15 detik
6.	3 detik	16 detik
7.	3 detik	15 detik
8.	3 detik	15 detik
9.	2 detik	19 detik
10.	4 detik	18 detik
11.	3 detik	18 detik
12.	2 detik	18 detik
13.	2 detik	18 detik
14.	2 detik	19 detik
15.	3 detik	18 detik
16.	2 detik	18 detik
17.	3 detik	17 detik
18.	3 detik	19 detik
Total	47 detik	305 detik
Rata-rata	2.611 detik	16.944 detik

Use case Ubah Kata Sandi memiliki jumlah *data binding* yang paling banyak dengan inputan data yang cukup rumit dibandingkan dari keseluruhan fungsi yang ada dalam penelitian ini. Pada tabel 3 terlihat bahwa total waktu yang diperlukan untuk menjalankan *StartApplication* pada pengujian otomatis cukup lama yaitu 305 detik atau sekitar 5 menit 5 detik dengan rata-rata waktu 16.944 detik. Berbeda dengan waktu untuk membuka aplikasi pada pengujian manual yang hanya membutuhkan 47 detik. Dalam hal ini, *StartApplication* / pembukaan aplikasi dihitung sejak logo ACCBid muncul pada layar. Apabila diasumsikan bahwa waktu untuk membuka aplikasi / menjalankan *StartApplication* tidak dihitung, maka didapatkan total waktu pengujian manual untuk fungsi Ubah Kata Sandi ialah sebesar 747 detik (12 menit 27 detik), sedangkan waktu untuk pengujian otomatis setelah dikurang waktu untuk *StartApplication* ialah 651 detik (10 menit 51 detik).

Analisis juga dilakukan terhadap *use case* Hapus Akun Bank dengan jumlah data binding yang lebih sedikit. Pada Tabel 4, terlihat perbandingan waktu untuk membuka aplikasi / menjalankan *StartApplication*.

Tabel 4. Perbandingan Waktu untuk Membuka Aplikasi / Menjalankan StartApplication pada Test Case ‘Hapus Akun Bank’

<i>Data binding ke-</i>	Waktu untuk membuka aplikasi / <i>StartApplication</i>	
	Pengujian Manual	Pengujian Otomatis
1.	3 detik	18 detik
2.	2 detik	15 detik
Total	5 detik	33 detik
Rata-rata	2.5 detik	16.5 detik

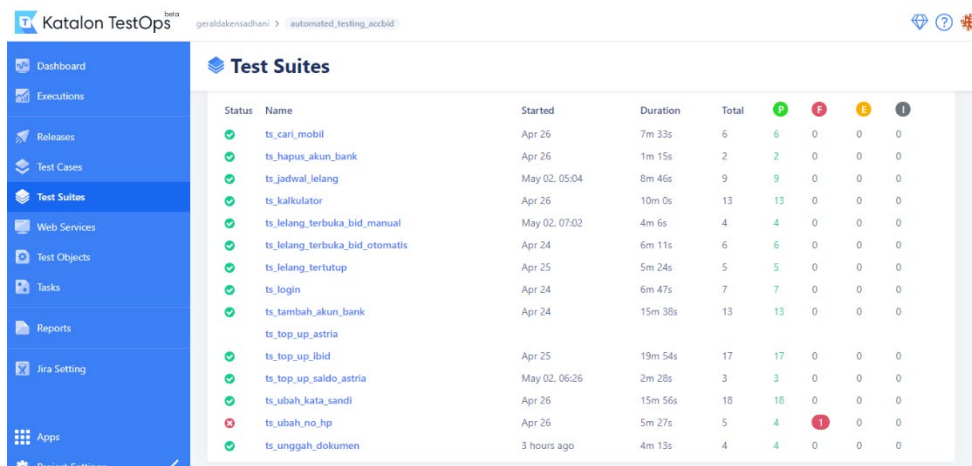
Apabila diasumsikan bahwa waktu untuk membuka aplikasi / menjalankan *StartApplication* tidak dihitung, maka didapatkan total waktu pengujian manual untuk fungsi

Hapus Akun Bank ialah sebesar 45 detik, sedangkan waktu untuk pengujian otomatis setelah dikurang waktu untuk StartApplication ialah 42 detik.

Maka dari itu dapat dikatakan bahwa *StartApplication* sebagai salah satu faktor yang mempengaruhi lamanya waktu pengujian otomatis. Selain itu, waktu pengujian otomatis menggunakan Katalon Studio juga dipengaruhi oleh waktu pendeteksian dari masing-masing objek yang digunakan dalam *test case*. Dari hasil pengujian yang telah dilakukan, diketahui bahwa pengujian manual terbukti cukup efektif dibandingkan pengujian otomatis menggunakan Katalon Studio. Salah satu kelebihan yang dimiliki pengujian manual ialah tidak diperlukan waktu yang lama untuk membuka aplikasi karena tidak diperlukan proses *setting* pada *device* yang digunakan.

Namun, pengujian manual juga memiliki kekurangan. Berikut adalah penjabaran kekurangan dari pengujian manual yang didapatkan selama penelitian ini. (1) Masih terdapat kesalahan yang disebabkan oleh kesalahan manusia seperti kesalahan pengetikan dan kehilangan fokus pada saat melakukan pengujian. (2) Kurang efektif apabila dilakukan untuk menjalankan pengujian pada data dalam jumlah yang banyak dikarenakan proses yang dilakukan berulang secara terus-menerus dapat menjadi aktivitas yang membosankan bagi *tester* dan menyebabkan *tester* kelelahan. (3) Hasil pengujian dan waktu pengujian masih harus direkapitulasi secara manual.

Selanjutnya kita melihat ke sisi pengujian secara otomatis menggunakan Katalon Studio. Meskipun terbukti tidak cukup efektif dari segi waktu dibandingkan pengujian manual, berikut adalah kelebihan penggunaan Katalon Studio sebagai *automation testing tool* yang ditemukan selama penelitian ini. (1) *Interface* pada Katalon Studio yang mudah digunakan (*user-friendly*). (2) Hasil pengujian yang dapat secara langsung digenerate secara otomatis ketika eksekusi selesai dilakukan. Selain itu, adanya integrasi dengan Katalon TestOps membantu mempermudah dalam menganalisis riwayat hasil pengujian yang telah dilakukan seperti yang dapat dilihat pada gambar 2. (3) Dapat menjalankan *data binding* dalam jumlah yang banyak tanpa dibutuhkan banyak *tester*. (4) Hasil inputan yang akurat sesuai dengan yang telah dirancang sehingga meniadakan resiko kesalahan manusia pada saat eksekusi. (5) *Test case* dapat dilakukan pembaruan apabila sewaktu-waktu terdapat pembaruan objek pada aplikasi.



Status	Name	Started	Duration	Total	Pass	Fail	Warning	Info
✓	ts_cari_mobil	Apr 26	7m 33s	6	6	0	0	0
✓	ts_hapus_akun_bank	Apr 26	1m 15s	2	2	0	0	0
✓	ts_jadwal_lelang	May 02, 05:04	8m 46s	9	9	0	0	0
✓	ts_kalkulator	Apr 26	10m 0s	13	13	0	0	0
✓	ts_lelang_terbuka_bid_manual	May 02, 07:02	4m 6s	4	4	0	0	0
✓	ts_lelang_terbuka_bid_otomatis	Apr 24	6m 11s	6	6	0	0	0
✓	ts_lelang_tertutup	Apr 25	5m 24s	5	5	0	0	0
✓	ts_login	Apr 24	6m 47s	7	7	0	0	0
✓	ts_tambah_akun_bank	Apr 24	15m 38s	13	13	0	0	0
✓	ts_top_up_astria							
✓	ts_top_up_ibid	Apr 25	19m 54s	17	17	0	0	0
✓	ts_top_up_saldo_astria	May 02, 06:26	2m 28s	3	3	0	0	0
✓	ts_ubah_kata_sandi	Apr 26	15m 56s	18	18	0	0	0
✗	ts_ubah_no_hp	Apr 26	5m 27s	5	4	1	0	0
✓	ts_unggah_dokumen	3 hours ago	4m 13s	4	4	0	0	0

Gambar 2. Report Hasil Pengujian Otomatis yang Digenerate Secara Langsung ke Katalon TestOps

Adapun kekurangan dari pengujian otomatis menggunakan Katalon Studio: (1) Diperlukan waktu yang cukup lama untuk menjalankan *StartApplication* dan juga melakukan pendeteksian objek. (2) Kurang efektif apabila digunakan untuk menjalankan pengujian pada data dengan jumlah yang tidak banyak. (3) Kesalahan penginputan data yang sering terjadi pada saat pengujian manual terlebih pada inputan yang rumit seperti yang ada pada saat menjalankan

test case ubah kata sandi. Hal ini membuat *tester* menjadi kehilangan fokus dan melakukan kesalahan yang tidak disengaja.

Berikut adalah kendala yang dihadapi selama dilakukannya pengujian manual dan pengujian otomatis pada penelitian ini. (1) Saat menjalankan *test case* yang memiliki *data binding* dengan jumlah yang banyak pada Katalon Studio akan menyebabkan beberapa data uji tidak berjalan dengan baik sehingga dilakukan pengulangan pengujian. (2) Ada beberapa objek yang telah dideteksi pada saat melakukan *record* untuk pembuatan *test case* pada Katalon Studio yang tidak dapat berjalan dengan baik ketika dijalankan. Sehingga diperlukan lagi pengecekan terhadap *resource-id* atau *xpath* dari objek tersebut atau dilakukan pembuatan objek baru. (3) Adanya pengaruh faktor eksternal seperti koneksi internet yang menyebabkan pada saat menjalankan pengujian otomatis, terdapat objek yang masih tidak dapat terdeteksi hingga melebihi batas *timeout* sehingga menyebabkan kegagalan dan pengulangan pengujian.

5. Kesimpulan dan Saran

Berdasarkan hasil penelitian dan pembahasan yang telah dilakukan, didapatkan kesimpulan sebagai berikut: (1) Kepastian kebenaran fungsionalitas dari 14 fungsi yang diuji pada sistem ACCBid mendapatkan persentase senilai 92.86% dimana hal ini menunjukkan bahwa 1 dari 14 fungsi pada aplikasi ACCBid masih belum berjalan dengan baik. (2) Pengujian otomatis menggunakan Katalon Studio tidak cukup efektif dari segi waktu dibandingkan pengujian manual dengan rata-rata persentase penurunan waktu 48.15177% atau 1.482x lebih lambat dibandingkan pengujian secara manual. Salah satu penyumbang terbesar dalam lamanya waktu pengujian otomatis ialah ketika menjalankan *StartApplication*. (3) Kelebihan dari penggunaan Katalon Studio untuk melakukan pengujian otomatis adalah *interface* yang mudah digunakan, hasil pengujian yang dapat di-*generate* secara otomatis, serta mempermudah pengujian ketika dilakukan dalam *data binding* yang banyak dan inputan yang cukup rumit.

Referensi

- [1] Statista, "Number of apps available in leading App Stores as of 4th quarter 2019," 2020. [Online]. Available: <https://www.statista.com/statistics/276623/number-of-apps-available-in-leading-app-stores/>. [Accessed: 19-Jan-2020].
- [2] F. C. Ningrum, D. Suherman, S. Aryanti, and H. A. Prasetya, "Pengujian black box pada aplikasi sistem seleksi sales terbaik menggunakan teknik Equivalence Partitions," *J. Inform. Univ. Pamulang*, vol. 4, no. 4, pp. 125–130, Des. 2019.
- [3] M. Nabila, "ACC Rilis Aplikasi integrator layanan pembiayaan 'Acc One,'" 2019. [Online]. Available: <https://dailysocial.id/post/astra-credit-companies-aplikasi-acc-one>. [Accessed: 14-Apr-2020].
- [4] G. Hökelekli, "A comparative study on automated android application testing tools," The Middle East Technical University, 2016.
- [5] M. Komarudin, "Pengujian perangkat lunak metode black box berbasis equivalence partitions pada aplikasi sistem informasi sekolah," *J. Mikrotik*, vol. 06, no. 3, Feb. 2016.
- [6] I. K. Wi. P. Yasa, "Analisis dan pengujian perangkat lunak dengan metode black box, studi kasus BRS online Universitas Sanata Dharma," Universitas Sanata Dharma, 2013.
- [7] A. Purnomo, "Software testing aplikasi website PT Gramedia menggunakan metode blackbox pada PT WGS Bandung," vol. 91, pp. 399–404, 2017.
- [8] J. F. Andry and Reinaldo, "Testing dan implementasi aplikasi parkir di PT ABC Menggunakan Metode Black Box," *Pros. Semin. Nas. Multidisiplin Ilmu*, no. April 2017, 2017.
- [9] W. N. Cholifah, Y. Yulianingsih, and S. M. Sagita, "Pengujian black box testing pada aplikasi action & strategy berbasis Android dengan teknologi Phonegap," *STRING (Satuan Tulisan Ris. dan Inov. Teknol.)*, vol. 3, no. 2, pp. 206, 2018.
- [10] Herlinda, D. Katarina, and E. W. Ambarsari, "Automation testing tool dalam pengujian aplikasi belajar tajwid pada platform Android," *STRING (Satuan Tulisan Ris. dan Inov. Teknol.)*, vol. 4, no. 2, pp. 205–212, 2019.

- [11] P. Gaharu, "Testitude," *Medium*, 2019. [Online]. Available: <https://medium.com/doku-insight/testitude-35c57bc996a6>. [Accessed: 14-Apr-2020].