Jurnal Ilmiah Wahana Pendidikan, Mei 2023, 9(10), 114-126
DOI: https://doi.org/10.5281/zenodo.7983405.
p-ISSN: 2622-8327 e-ISSN: 2089-5364
Accredited by Directorate General of Strengthening for Research and Development
Available online at https://jurnal.peneliti.net/index.php/JIWP

Automation Testing Api, Android, dan Website Menggunakan *Serenity Bdd* Pada Software Sistem Manajemen Rumah Sakit

Bagas Dwi Saputra[1], Arnisa Stefanie[2]

[1,2]Jurusan Teknik Elektro Universitas Singaperbangsa Karawang

Received: *11 Februari 2023*
Revised: *27 Februari 2023*
Accepted: *3 Maret 2023*

Abstract

System testing has an important part to produce a product with high quality and time saving. Manual testing has many drawbacks compared to automated testing. This research discusses automation testing. Automation testing will greatly help quality application developers and time efficiency. Script automation testing based on cucumber serenity running behind the scenes using jenkis. Test scenario system, running with the Java programming language. Test conclusions drawn from testing automatically generate testing reports from the scenarios that have been made. Testing can be seen more easily in the form of an idex website compared to manual testing.

Keywords: *Automation Testing, Proses Tesing, Serenity BDD, Quality Assurance*

(*) Corresponding Author: *bagas@gmail.com*

How to Cite: Saputra, B., & Stefanie, A. (2023). Automation Testing Api, Android, dan Website Menggunakan Serenity Bdd Pada Software Sistem Manajemen Rumah Sakit. *Jurnal Ilmiah Wahana Pendidikan, 9*(10), 114-126. https://doi.org/10.5281/zenodo.7983405.

PENDAHULUAN

Pada pengembangan sistem, pengujian mempunyai bagian yang sangat penting untuk menghasilkan keluaran sistem yang berkualitas tinggi dan mengurai adanya kesalahan pada sistem. Testing sistem merupakan bagian yang tersusun dalam pembuatan *system development*. *Software* yang bernilai merupakan *software* yang dapat memenuhi keinginan pengguna dan sesuai perancangan, serta sedikit adanya *bugs*. Testing juga merupakan suatu alat ukur dalam menentukan atau meninjau kualitas, kinerja, ataupun keandalan dari suatu aplikasi perangkat lunak sebelum dilakukan implementasi di ranah publik (*real production*) [1].

Maka dalam menjaga kualitas dari *software*, sebelum dirilis akan melalui proses pengujian sistem. Untuk pengujian secara manual, keluaran bisa jadi tidak akurat dikarenakan human *error*. Pengujian manual masih bisa dilakukan yang sederhana saja, contoh seperti *review* aplikasi pada Play Store. Pengujian secara manual kurang dapat dipertanggungkan jika disandingkan oleh pengujian otomatis yang memanfaatkan *tools* dan *script*.

Software quality assurance adalah tindakan yang dilakukan sesuai dengan pola terencana dan sistematis untuk memastikan barang atau produk tersebut telah sesuai dengan persyaratan teknis yang telah ditetapkan. Serangkaian kegiatan yang dirancang untuk melakukan evaluasi produk yang sedang dikembangkan atau diproduksi [2]

Dengan menggunakan *automation* testing akan menghemat waktu dalam pengujian dibandingkan dengan manual testing. Melakukan penelitian dengan membandingkan teknik *scripting* yang menggunakan *automation testing*. Teknik *scripting* dapat mengurangi biaya dalam melakukan *automation* testing *software*

secara keseluruhan, mengoptimalkan kecepatan dalam pengujian, mempercepat *development process* [3]

Mengimplementasikan testing secara otomatis akan memudahkan perancangan sistem dan efisien dalam mengetahui kesalahan sistem. Development akan dipermudah dalam pengujian yang mendeteksi adanya perubahan pada source code.

Automation testing diimplementasikan pada aplikasi *Hospital Management System. Hospital management system* merupakan aplikasi berbasis *website* dan android serta API. Hospital management system berguna untuk mengatur penerimaan pasien di admisi, data pasien rawat inap dan pasien rawat jalan dapat dilakukan lebih mudah. Selain itu, Hospital *management system* adalah sistem yang berbasis perangkat lunak *mobile* dan *website* dikembangkan oleh *mentee* program studi independent bersertifikat dari kampus merdeka dengan mitra alterra academy.

METODE

Metode yang digunakan dalam penelitian dengan inplementasikan *cucumber scenario* memakai *serenity framework*.

A. *Software* IntelliJ IDEA

IntelliJ IDEA merupakan kepanjangan dari integrated *development environment*, yang dikembangkan dari JetBrains. Dapat digunakan oleh *development* untuk mengembangkan program atau aplikasi. Antar muka IntelliJ IDEA dapat terhubung keberbagai *platfom*, seperti VCS, Git, SVN, Mercurial, CVS, Perforce, dan TFS.

IntelliJ IDEA pertama kali diluncurkan pada januari 2001 dan diusung sebagai aplikasi pengembang program java pertama dengan penavigasian dan perekstrusi kode program tingkat lanjut [4].

B. *Testing Automation*

Automation testing melakukan pemeriksaan pada *software* atau sistem dengan otomatis memakai perangkat yang membantu mengerjakan *scenario*. Perangkat yang digunakan bisa mengelola data yang sedang dalam pengujian, dan menghasilkan detail laporan dari pengujian yang sudah dilakukan. *Automation Testing* menggabungkan berbagai bahasa dalam pembuatan *test script*, bahasa yang digunakan seperti *python, java script*, dan *tool command language* (TCL) [5]

Dalam pengembangan sistem atau aplikasi oleh *development* biasanya memerlukan *test* atau pengujian sebelum sistem diliris ke pasaran. Semenjak diperkenalkan mulai banyak yang beralih ke *automation testing* dari manual testing. Tujuan dari *automation testing* ialah mengurangi eksekusi *sekenario* secara manual dan menghemat biaya dan waktu, mempercepat proses pengujian hingga sistem dapat diliris. Terdapat beberapa *prosess* dari *automation testing* seperti ditunjukan pada Gambar 1, berikut; (1) Pelilihan *tool* untuk Pemeriksaan, (2) Pemaparan cakupan dan otomatis, (3) Perencanaan, desain ruang lingkup, dan pengembangan, (4) Eksekusi, dan (5) Pemeliharaan.

Gambar 1. Bagian proses pada *automation testing*

C. *Framework Serenity*

BDD merupakan perpustakaan *opensource* untuk mempermudah pengkerjaan. Serenity memakai keluaran dari pengujian untuk membentuk gambaran, deskripsi lamporan, menerangkan tentang aplikasi dan cara kerja sistemnya. *Framework* adalah dasar dari pembuatan *software*, dengan adanya kerangka dasar *developer* tidak perlu membuat dari awal [6]

D. Jenkins

Jenkins merupakan suatu server yang bekerja secara otomatis berbahasa java. Terintergrasi berkelanjutan dan memiliki aspek teknis yang dapat melakukan pengiriman secara kontinu. Jenkins dapat membantu *developer* untuk memproses pengembangan sistem.

E. Appium

Appium merupakan *open-source* yang menghubungkan antara aplikasi di platform ios atau android, dan website. Aplikasi website dapat diakses melalui browser ios atau pun android. Appium ialah *cross-platform* memungkinkan penggunaan testing dalam banyak platform ios, android, windows [7]

F. *Automation Testing* pada *Project*

Automation Testing pada *Project* yang mempunyai fungsi dalam menuliskan sebuah *soure code* skenario yang akan di lakukan. Automation Testing yang akan dilakukan pengetesan adalah aplikasi buatan sendiri oleh mentee pada program studi *independent* yaitu *Hospital Management* System. Pada sistem *Hospital Management System* terdapat 2 dua role dan berbagai fungsi, seperti berikut:

1. Role Pasien
   a. Data Pasien, dapat melakukan penambahan data pasien, pengeditan data pasien, pembatalan penambahan atau pun pengeditan data, dan bisa mencari berbagai data sesuai kategori (nama, id, nik, dan usia),

b. Data Dokter, dapat melakukan penambahan data pasien, pengeditan data pasien, pembatalan penambahan atau pun pengeditan data, dan bisa mencari berbagai data sesuai kategori (nama, npa idi, spesialis, username, kata sandi, dan konfirmasi kata sandi),
c. Kelola Jadwal, dapat menambahkan atau mengatur pertemuan antara pasien dengan dokter sesuai dengan tanggal dan jenis perawatan, dan dapat melakukan pencarian sesuai kategori, dan
d. Arsip Jadwal, pada fiture ini dapat melihat hasil penyimpan yang sudah di buat pada fitur Kelola Jadwal dan catatan dokter.

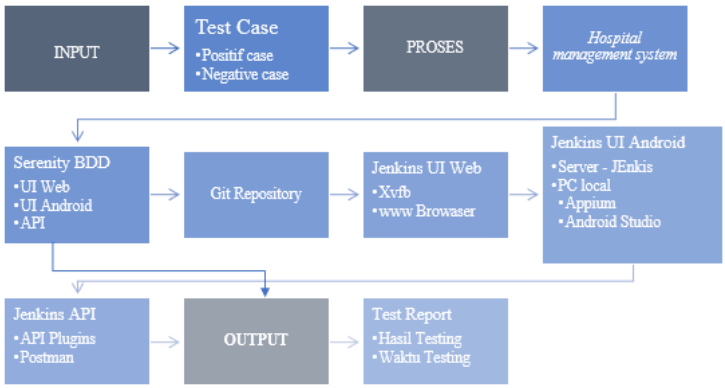
2. Role Dokter
   a. Pertemuan hari ini, pada fitur ini user dapat memperoses atau menambahkan catatan medis pada proses data pasien. Setelah proses selesai data akan dikirim ke Arsip Jadwal, dan
   b. Histori data pertemuan pasien, pada fitur ini dapat melihat melalui bagian aksi atara pasien rawat inap atau pasien rawat jalan. Terdapat fitur pencarian sesuai kategori yang disediakan.

G. Android Studio

Integrated Development Environment (IDE) berguna membuat software dalam platform android atau pun ios. Android studio dapat digunakan untuk pengujian aplikasi android seperti emulator.

Android studio ini berbasis pada IntelliJ IDEA, sebuah IDE untuk Bahasa pemrograman Java. Bahasa pemrograman utama yang digunakan adalah Java, sedangkan untuk membuat tampilan atau *layout*, digunakan bahasa *XML Invalid source specified* [8].

H. Desain Sistem



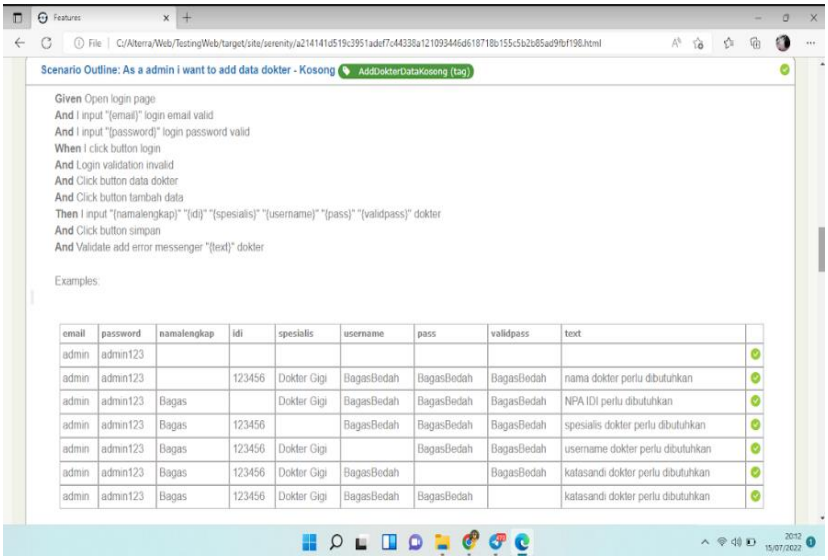
Gambar 2. Design Sistem

Pada gambar 2, Design sistem mengambaran testing yang dilakukan. Bagian-bagian yang ada di dalam gambar diatas, terdapat banyak fokus yang berbeda-beda.

I. Automation Testing User Interface Website

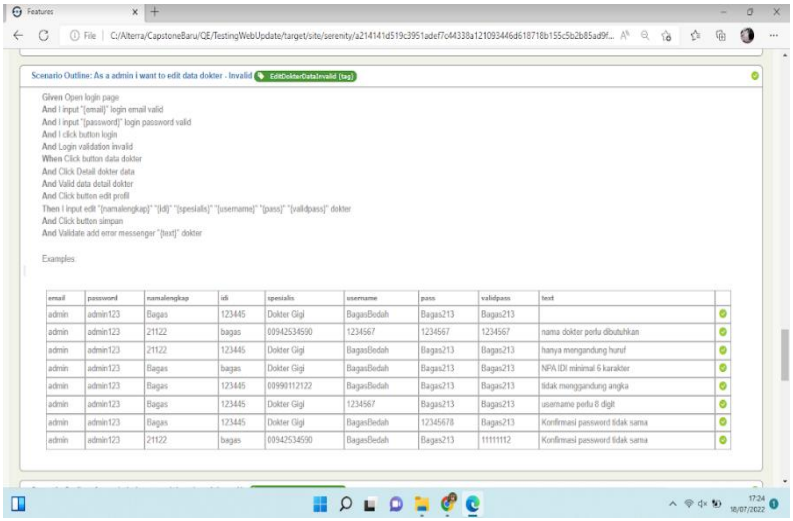
Website adalah sebutan atau panggilan dari sekelompok halaman web (page web), umumnya termasuk dalam bagian domain atau subdomain di world wide web (www) [9]

Automation Testing *UI Web* adalah pengujian atau testing untuk mencari bila adanya *bugs* atau *error* pada sistem atau *software* di Website. Dibutuhkan *tools* tambahan untuk melakukan testing Website browser seperti chrome, firefox, atau safari. Untuk dapat mengakses *button* atau *push button* pada website biasanya melalui *inspect code*.



email	password	namalengkap	idi	spesialis	username	pass	validpass	text	
admin	admin123								
admin	admin123		123456	Dokter Gigi	BagasBedah	BagasBedah	BagasBedah	nama dokter perlu dibutuhkan	
admin	admin123	Bagas		Dokter Gigi	BagasBedah	BagasBedah	BagasBedah	NPA IDI perlu dibutuhkan	
admin	admin123	Bagas	123456		BagasBedah	BagasBedah	BagasBedah	spesialis dokter perlu dibutuhkan	
admin	admin123	Bagas	123456	Dokter Gigi		BagasBedah	BagasBedah	username dokter perlu dibutuhkan	
admin	admin123	Bagas	123456	Dokter Gigi	BagasBedah		BagasBedah	katasandi dokter perlu dibutuhkan	
admin	admin123	Bagas	123456	Dokter Gigi	BagasBedah	BagasBedah		katasandi dokter perlu dibutuhkan	

Gambar 3. *Automated Add* dokter testing website



email	password	namalengkap	idi	spesialis	username	pass	validpass	text	
admin	admin123	Bagas	123445	Dokter Gigi	BagasBedah	Bagas213	Bagas213		
admin	admin123	21122	bagas	08942534590	1234567	1234567	1234567	nama dokter perlu dibutuhkan	
admin	admin123	21122	123445	Dokter Gigi	BagasBedah	Bagas213	Bagas213	hanya mengandung huruf	
admin	admin123	Bagas	bagas	Dokter Gigi	BagasBedah	Bagas213	Bagas213	NPA IDI minimal 6 karakter	
admin	admin123	Bagas	123445	08990112122	BagasBedah	Bagas213	Bagas213	tidak menggandung angka	
admin	admin123	Bagas	123445	Dokter Gigi	1234567	Bagas213	Bagas213	username perlu 8 digit	
admin	admin123	Bagas	123445	Dokter Gigi	BagasBedah	12345678	Bagas213	Konfirmasi password tidak sama	
admin	admin123	21122	bagas	08942534590	BagasBedah	Bagas213	11111112	Konfirmasi password tidak sama	

Gambar 4. *Automated* Edit dokter testing website

J. *Automation Testing* API

Application Programming Interface adalah pemeriksaan atau testing mencari bila terdapat kesalahan pada sistem pada bagian *backeand*. Dibutuhkan perangkat tambahan dalam melakukan pemeriksaan seperti *rest assured*.

Rest Assured merupakan bahasa java DSL (*Domain Specific Language*). DSL berguna dalam menyederhanakan pemeriksaan REST *service*. Rest

Assured memiliki *request* seperti Head, Patch, Put, Delete, Post, Options, Get, dan Post berguna untuk mengkonfirmasi dari *request* yang diberikan.

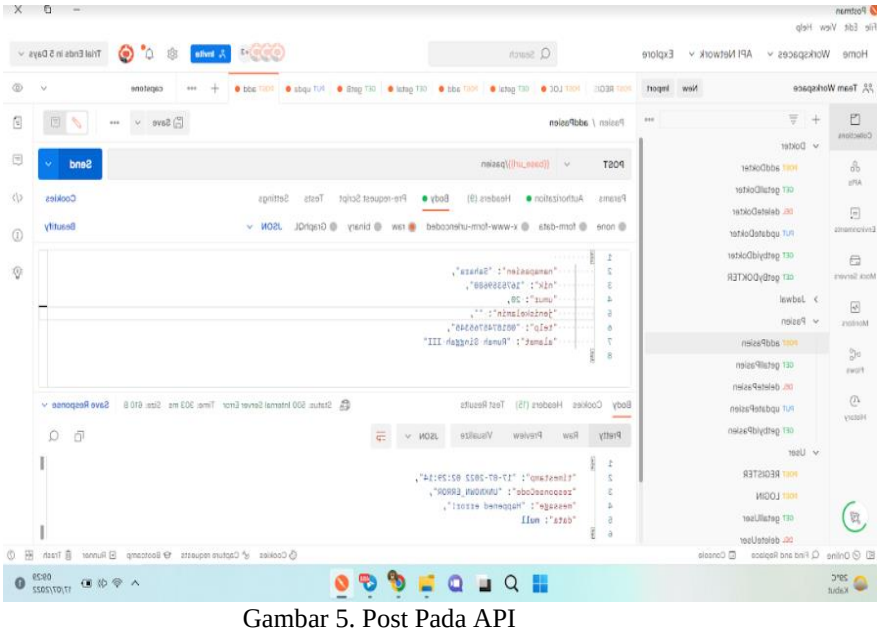
1. Susunan POM

   Dalam *Automation Testing* pada API membutuhkan sebuah file *dependency* yang berguna sebagai library API dalam Rest Assured. Rest Assured merupakan suatu DSL akan menyederhanakan pemeriksaan *rest serevices* yang ditingkatkan pada HTTP *Builder*. Rest Assured dapat membantu macam-macam *request* seperti Head, Patch, Put, Delete, Post, Options, Get, dan Post yang dapat berguna dalam menguji *respon* dari *request*.
2. Susunan *class serenity.properties*

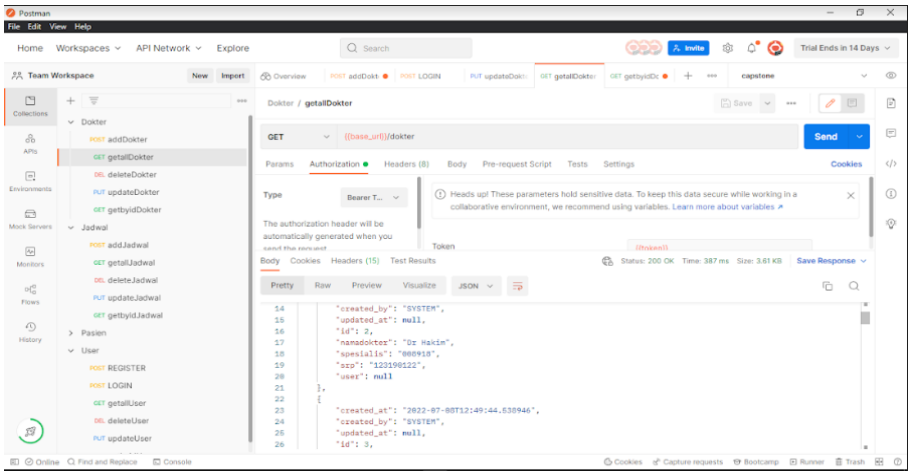
   Dalam *class.properties* sebuah *class* yang dapat konfigurasi *webdriver* secara manual.
3. Susunan *report test*

   Pada pengelolan test *resport*, sistem perlu ditambahkan beberapa <plugin><groupId>net.serenity-bdd.maven.plugins</groupId> <plugins> dengan versi 3.2.1
4. Penambahan penyimpanan API *Controll* dipackage main

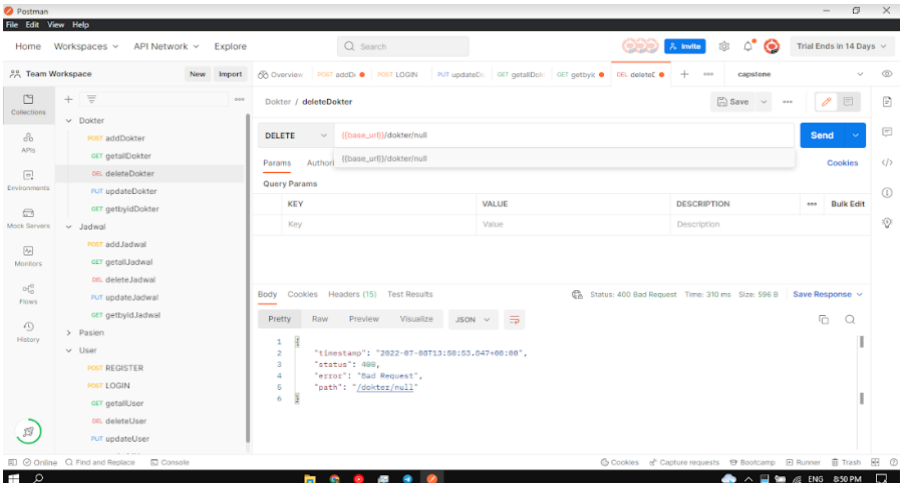
   Dalam pembuatan pengkontrolan kelas di pemeriksaan API, menggunakan direktori tambahan direktori controller.



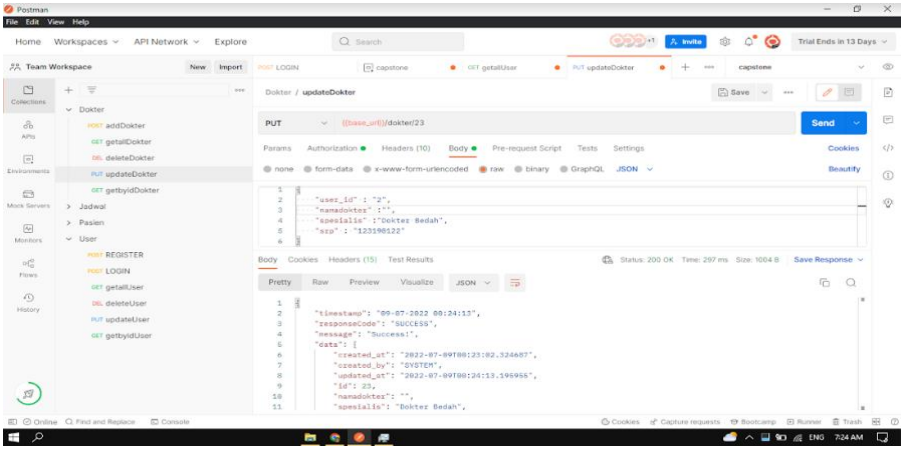
Gambar 5. Post Pada API



Gambar 6. Get pada API



Gambar 7. Delete pada API



Gambar 8. Put pada API

K. Automation Testing User Interface Android

Pemeriksaan Otomatis *User Interface Testing* Android adalah pemeriksaan atau testing dalam mencari bila adanya kesalahan pada sistem

pada bagian Android. Dibutuhkan *tools* tambahan untuk melakukan pemeriksaan seperti *Appium*. Selain itu diperlukan juga *Andoird Studio* yang berguna untuk menjalankan aplikasi android.

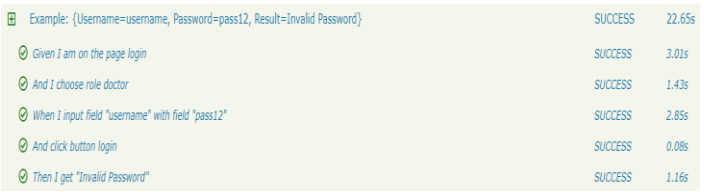
Jenkins adalah sebuah *opensource automation server* untuk mengotomatiskan tugas-tugas di dalam *proses continuous integration* and *delivery* sebuah perangkat lunak. Jenkins merupakan aplikasi berbasis Java yang dapat dipasang dari *repositori* Ubuntu atau dengan mengunduh dan menjalankan file *Web applicatino ARchive* (WAR), sebuah koleksi file yang sudah lengkap dan tinggal dijalankan disebuah *server* [10]

1. Susunan *class serenity.properties*

   Merupakan sebuah kelas untuk dapat mesusun secara manual pada Android *Driver*, terdapat berbagai bentu. *Class* properties mempunyai fungsi untuk mengabungkan *soure automation testing* dengan Appium.
2. Susunan *class driver Android*

   Dalam memproses *user interface* android memburuhkan driver yang dapat terhubung antar kode *automation testing* dengan *device* android.

Gambar 9. Pengujian *Automation testing* aplikasi android



Gambar 10. Pengujian Automation testing aplikasi android

HASIL DAN PEMBAHASAN

Pada saat dilakukan pengujian, memakai tiga metode pengambilan data sebagai berikut:

1. Membandingkan langkah yang digunakan untuk memproses pengkerjaan manual dan *automation testing*,
2. Membandingkan waktu *respon* antara manual dan *automation* testing, dan
3. Membandingkan kecepatan dalam menemukan *error* dan *bugs* pada *software* antara manual dan *automation testing*.

Melakukan pengujian API, UI Web, dan Android di *software Hospital Management* System. Skenario testing yang ditampilkan pada Tabel 1.

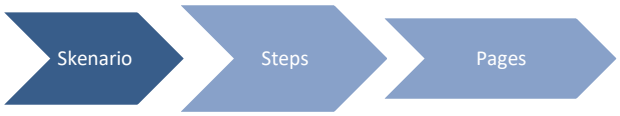
Tabel 1. Pengujian kuantitas fitur dan skenario.

No	Pengujian	Kuantitas Fitur	Kuantitas Skenario
1	API	25	148

2	UI Web	19	72
3	UI Android	18	74

A. Penerapan *Automation Testing* pada *User Interface*

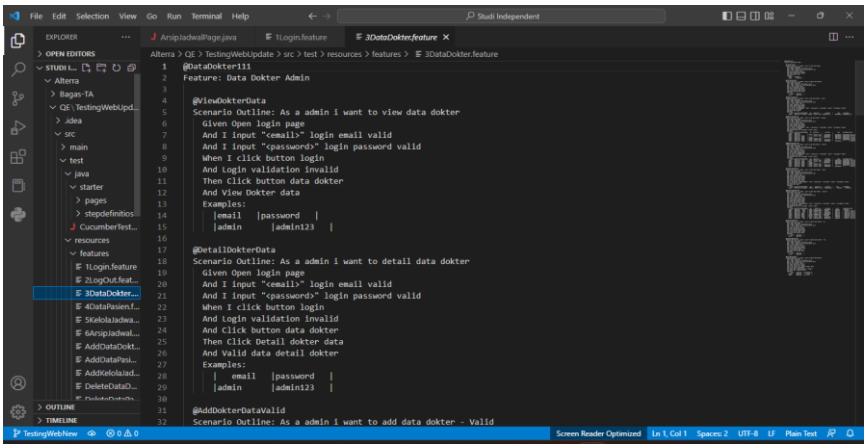
Dalam pengujian otomatis antarmuka pengguna ada *class java automation* testing seperti pada Gambar 11.



Gambar 11. Diagram *Automation Testing*

1. *Feature*

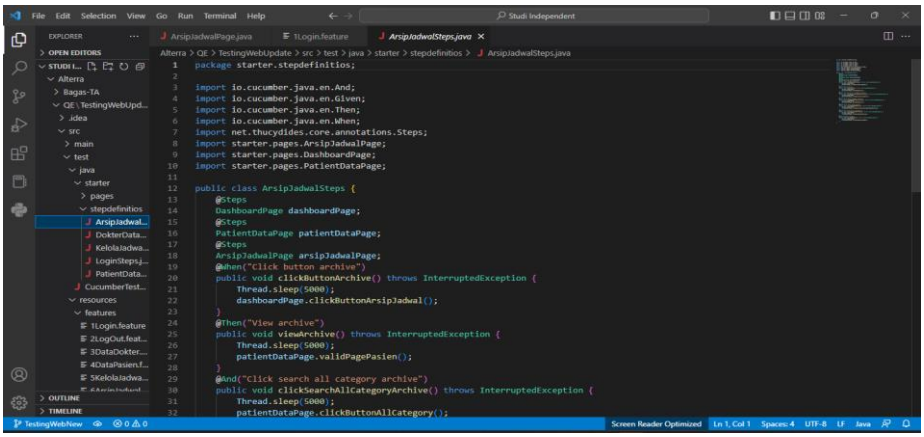
Pada *serenity cucumber*, *skenario* tersimpan di folder *feature*. Berisikan penjelasan dari detail *feature* beserta sejumlah skenario. Ada dua jenis skenario pada *testing*, pertama *positive testing* dan kedua *negative testing*. Pada skenario *positive* testing dibuat dengan menginputkan data *username* dan *password* yang benar. N*egative* testing dibuat dengan menginpukan data *username* dan *password* yang salah atau kosong.



Gambar 12. *Feature* pada *serenity*

2. *Steps*

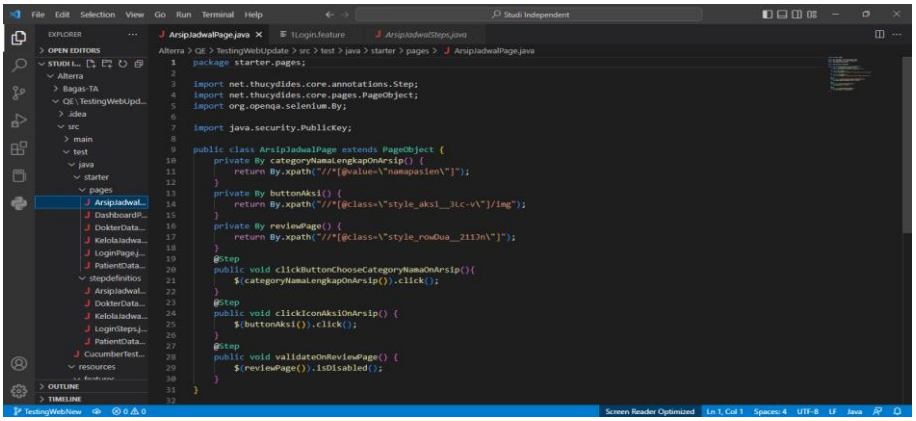
Pada *serenity cucumber*, skenario digambarkan *class steps*. *Class* ini berisikan *@Given*, *@When*, *@And* dan *@Then* sesuaikan *skenario* yang dibuat. C*lass steps* dapat didefinisikan sebagai indikasi parameter yang akan diteruskan ke metode. *Class steps* berguna untuk mengatur supaya code menjadi suatu kelompok yang dapat lebih mudah digunakan kembali. Pada bagian *@Steps* akan memberitahukan serenity suatu variable berada *class* berupa *steps library*. Pada *steps library* dapat menambahkan "*Get*" atau "*Put*" *testing*. Dalam *cucumber steps definition* menggambarkan "*Get*" yang akan dilakukan oleh testing.



Gambar 13. *Steps* pada *serenity*

3. *Pages*

Dalam *serenity* membuat kode *automation testing* memakai *Page Objects*. Merupakan cara untuk mendapatkan detail dari halaman website dalam suatu *class* dengan memanfaatkan *inspect.* Dapat dicari menggunakan *@FindBy* untuk mencari *elements web* yang akan di uji coba. *@FindBy* akan memakai *id* atau *xpath* dari *elements* di website tersebut.



Gambar 14. *Pages* pada *serenity*

B. Membandingkan Tindakan Manual dan *Automated Testing*

Dalam membandingkan tindakan dalam pengujian *manual* dan *automated* akan dibagi menjadi 2 tingkatan, pertama perancangan dan pengujian. Tabel 2 menampilkan perbandingan dalam tahapan perancangan *manual* dan *automated.* Pada tabel 3 menampilkan perbandingan pengujian *manual* dan *automated.* Untuk tabel 4 adalah hasil dari perbandingan Tindakan *Manual* dan *Automated*

Pada tabel 2 perancangan menggambarkan bahwa untuk pengujian *automated testing* membutuhkan persiapan yang lebih banyak dibandingkan manual sedikit persiapan. Untuk tabel 3 pengujian menggambarkan lebih banyak tahapan yang diperlukan manual testing sedangkan untuk *automated testing* lebih sedikit tahapan bahkan tiga kali lebih sedikit dibandingkan manual testing.

Berdasarkan dari tabel 4, tahapan-tahapan dalam *automated testing* memiliki jumlah skenario yakni 9 tahapan perancangan dan 3 tahapan pengujian. Dalam tahapan-tahapan *manual testing* memiliki jumlah skenario yakni 4 tahapan

perancangan dan 14 tahapan pengujian. Maka dari itu total tahapan yang dimiliki pengujian otomatis sedikit berbanding terbalik dengan *manual testing*. Waktu dalam pengujian yang dihasilkan pengujian otomatis memiliki waktu 20 detik lebih cepat berbanding dengan manual testing. Banyak yang dipengarui dalam pengujian, dari kekurangan perangkat, kualitas *bandwidth*, dan lain sebagainya.

Tabel 2. Membandingkan Perancangan *Manual* dan *Automated Testing*

No	***Testing Manual***	**No**	***Testing Automated***
1	Membuka website pada *software*	1	Membuka *Software* IntelliJ IDEA
2	Memahami *Flowchart* Sistem yang akan di *Test*	2	Memahami *Flowchart* Sistem yang akan di *Test*
3	Membuat skenario *testing*	3	Membuat skenario testing
4	Membuat dokumentasi hasil manual *testing*	4	Membuat project *automated testing* dengan serenity cucumber
		5	Setup project
		6	Membuat *class feature*
		7	Membuat *class steps*
		8	Membuat *class page*
		9	Membuat dokumentasi hasil *automated testing*

Tabel 3. Membandingkan Pengujian *Manual* dan *Automated Testing*

No	***Testing Manual***	**No**	***Testing Automated***
1	Membuka Website sistem pada browser	1	Menjalankan *Software* IntelliJ IDEA
2	*Login user*	2	Membuka file dan menjalkan *automated testing*
3	Pilih Role dokter atau pasien	3	Dokumentasi hasil pengujian secara otomatis
4	Masuk *kedashboard*		
5	Klik Data dokter		
6	Klik tambahkan data dokter		
7	Memasukan data dokter Nama Lengkap		
8	Memasukan data dokter IDI		
9	Memasukan data dokter *Spesialis*		
10	Memasukan data dokter *Username*		
11	Memasukan data dokter *Password*		
12	Memasukan data dokter *ValidPassword*		
13	Klik Simpan untuk menyimpan data dokter		
14	Memasukan data hasil pengujian ke dalam dokumentasi testing		

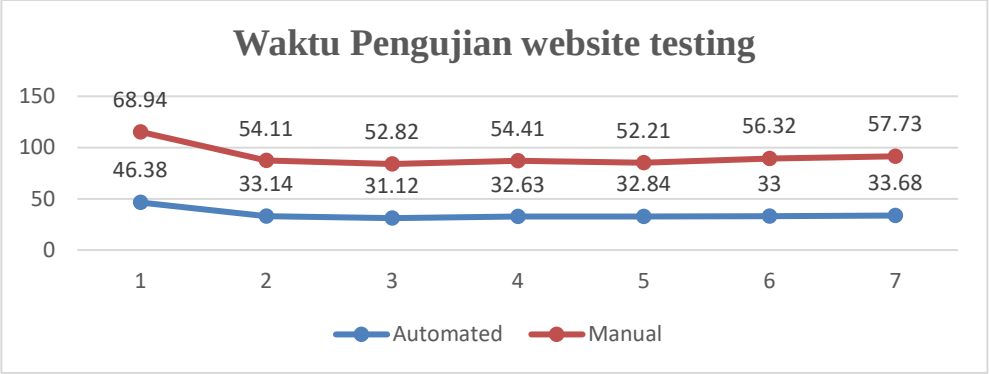
Tabel 4. Tabel Hasil Membandingkan Tindakan *Manual* dan *Automated Testing*

No	**Deskripsi**	**Langkah**	**Jumlah**

1	Perancangan	*Manual*	4
		Automated	9
2	Pengujian	*Manual*	14
		Automated	3

Tabel 5. Tabel Waktu Membandingkan Tindakan *Manual* dan *Automated Testing*

No	*Test Case*	*Type*	*Manual (s)*	*Automated (s)*
1	User menginput data dokter dengan benar	*Positive*	68,94	46,38
2	User menginput data dokter dengan salah semua	*Negative*	54,11	33,14
3	User menginput data kosong nama dokter	*Negative*	52,82	31,12
4	User menginput data kosong IDI dokter	*Negative*	54,41	32,63
5	User menginput data kosong *Username* dokter	*Negative*	52,21	32,84
6	User menginput data kosong *Password* dokter	*Negative*	56,32	33,00
7	User menginput data kosong *ValidPassword* dokter	*Negative*	57,73	32,68



Gambar 4. Grafik Waktu *Manual* dan *Automated* Testing

C. Membangkan Waktu Pengujian *Manual* dan *Automated Testing*

Dapat diambil dari tabel 3 waktu pengujian *manual* dan *automated* dan grafik pengujian *user intrface* di website dapa Gambar 4. Dapat disimpulkan bahwa *automated testing* lebih baik dari manual testing, *automated testing* dapat menghemat waktu dan mengurangi *human error*

KESIMPULAN

Pada penelitian mengambil masalah dalam pengujian otomatis pada software atau aplikasi Hospital Management System mengguakan serenity cucumber dan Jenkins. Serenity Framework yang dimanfaatkan dalam dokumen ini menghasilkan pengujian pengujian otomatis kemampuannya lebih baik dan berhasil dari pada pengujian secara manual. Dalam hasil percobaan dapat ditarik kesimpulan bahwa *automated testing* menggunakan serenity lebih baik dari pada manual testing.

REFERENSI

A. Firman, H. F. Wowor, Dan X. Najoan, *Sistem Informasi Perpustakaan Online Berbasis Web*,2016.

D. Galin Galin, *Software Quality Assurance From Theory To Implementation Software Quality Assurance From Theory To Implementation Cyan Magenta Yellow Black.* Tersedia Pada: Www.Pearsoned.Co.Uk I. Sommerville, *Software Engineering*.

G. Ayu Syafarina, M. Amin, F. Teknologi Informasi, Dan U. Islam Kalimantan Mab Banjarmasin, *Prototype Aplikasi Banjar Berbasis Android Studio Sebagai Salah Satu Petunjuk Wisata Di Banjarmasin.*

H. Mantik, *Peran Penting Testing Dan Quality Assurance Dalam Siklus Pengembangan Sistem*.

M. Destiningrum Dan Q. J. Adrian, 2017. *Sistem Informasi Penjadwalan Dokter Berbassis Web Dengan Menggunakan Framework Codeigniter* (Studi Kasus: Rumah Sakit Yukum Medical Centre)

Pearson, 2011. Oleh, *Komparasi Perangkat Lunak Pengembang Aplikasi Android Dengan Metode Qualitative Weight And Sum Studi Kasus Aplikasi Sudoku Halaman Judul*. "Quantitative Analysis Of Automation".

R. A. Putra, *Analisa Implementasi Arsitektur Microservices Berbasis Kontainer Pada Komunitas Pengembang Perangkat Lunak Sumber Terbuka (Opendaylight Devops Community)*, Teknologi Informasi Dan Komputer. [Daring]. Tersedia Pada: Https://Jurnal.Umj.Ac.Id