e-ISBN 978-623-93343-0-7
Seminar Nasional Mahasiswa Ilmu Komputer dan Aplikasinya (SENAMIKA)
Jakarta-Indonesia, 28 Januari 2020

PEMBANGUNAN FRAMEWORK WEB AUTOMATION TESTING MENGGUNAKAN SERENITY BDD PADA STUDI KASUS APLIKASI SUPPLY CHAIN

Maria Magdalena Panjaitan[1], IGN Mantra[2]
Information System, FTI, Perbanas Institute
Jl. Perbanas Setiabudi, Kuningan, Jakarta 12940
maria.m.panjaitan@gmail.com,
Ign.mantra@perbanas.id

Abstrak: Pengujian aplikasi merupakan elemen penting dalam menentukan kualitas suatu aplikasi. Pengujian aplikasi bermaksud untuk menguji setiap fungsionalitas maupun fitur yang ada pada aplikasi dan menemukan kemungkinan kesalahan yang terjadi pada aplikasi sedini mungkin. Pengujian umumnya dilakukan secara manual dengan dilakukan berulang-ulang terhadap area fungsionalitas untuk memastikan bahwa aplikasi tersebut minim dari kesalahan/bug. Begitu juga halnya yang terjadi pada pengujian Aplikasi Supply Chain berbasis web yang dijadikan pada studi kasus ini. Tujuan dari penelitian ini adalah untuk membangun sebuah Framework Web Automation Testing menggunakan Serenity BDD yang dapat digunakan untuk membantu pengujian aplikasi pada studi kasus aplikasi Supply Chain. Framework ini diharapkan dapat membantu perusahaan dalam menjamin kualitas produk aplikasi yang dibangun. Hal ini karena dengan adanya framework ini dapat menjadi solusi untuk mengurangi atau menghilangkan pengujian berulang yang dilakukan secara manual sehingga mengurangi human error yang mungkin terjadi saat melakukan pengujian aplikasi.

Kata Kunci: framework, web automation testing, serenity bdd, aplikasi supply chain

1 Pendahuluan

Seiring dengan perkembangan jaman, teknologi cenderung menjadi alat yang dapat menyelesaikan banyak hal. Salah satu teknologi yang semakin dikembangkan adalah perangkat lunak ataupun aplikasi.

Tujuan dari pengembangan perangkat lunak ataupun aplikasi adalah untuk menghasilkan perangkat lunak yang berkualitas unggul yang dapat diandalkan dan memberikan kepuasan pada penggunanya dengan memberikan kemudahan sehingga dapat meningkatkan kinerja bagi suatu perusahaan, industri, instansi, organisasi maupun usaha mandiri [1]. Untuk mencapai tujuan tersebut maka sangat diperlukan pengukuran kualitas perangkat lunak dengan dilakukannya pengendalian, pengujian dan pengelolaan yang mengacu pada kualitas perangkat lunak [2]. Jaminan kualitas perangkat lunak merupakan aktivitas mendasar untuk menghasilkan produk yang dapat digunakan oleh pengguna.

Ada dua cara pengujian yang biasa dilakukan yaitu pengujian dengan cara manual dan pengujian dengan cara otomatisasi. Semua jenis perangkat lunak dapat diuji dengan cara manual dengan melakukan mencoba satu persatu menu, fungsionalitas ataupun fitur pada perangkat lunak ketika perangkat lunak yang dikembangkan sudah jadi dan pengujian ini dilakukan oleh *tester* (orang yang melakukan *testing*) atau disebut penguji [3]. Cara pengujian ini akan membutuhkan sumber daya dan waktu yang lama sebab dilakukan pengujian dengan berulang-ulang terhadap area fungsionalitas untuk memastikan bahwa perangkat lunak tersebut minim dari kesalahan/bug. Hal ini terjadi karena banyaknya fitur baru dan pengembangan yang berkelanjutan pada perangkat lunak. Begitu juga halnya yang terjadi pada pengujian Aplikasi Supply Chain berbasis web yang dijadikan pada studi kasus ini. Pada studi kasus ini menggunakan Aplikasi Supply Chain yang merupakan salah satu aplikasi yang dikembangkan oleh perusahaan Advotics yaitu tempat peneliti bekerja. Namun dengan kurangnya/terbatasnya sumber daya penguji, maka dari itu, diperlukan perangkat lunak (*tool*) yang dapat mengotomasi pengujian aplikasi Supply Chain tersebut.

e-ISBN 978-623-93343-0-7
Seminar Nasional Mahasiswa Ilmu Komputer dan Aplikasinya (SENAMIKA)
Jakarta-Indonesia, 28 Januari 2020

Banyak *tools* yang menawarkan dan menyediakan kemudahan dalam pengujian perangkat lunak secara terotomasi seperti Katalon Studio, Selenium Webdriver, SoapUI, Apache JMeter, Cypress dan lain sebagainya. Dari beragam *tools* tersebut, poin penting yang menjadi bahan pertimbangan dalam pemilihan *tool* yang akan digunakan yaitu memiliki fitur yang dapat membantu proses pengujian dengan kemudahan dalam pemeliharaannya, mempercepat pengujian, mudah digunakan dan tidak menitikberatkan tim penguji harus memiliki kemampuan *programming*. Salah satu metode pengujian perangkat lunak yang dapat menghemat waktu pengujian dan fungsionalitasnya terjaga adalah menggunakan *tool* yang menerapkan Behaviour Driven Development (BDD) yang dapat menjadi solusi untuk pengujian perangkat lunak *automated acceptance testing* dengan menghasilkan laporan pengujian yang digambarkan dengan baik [4].

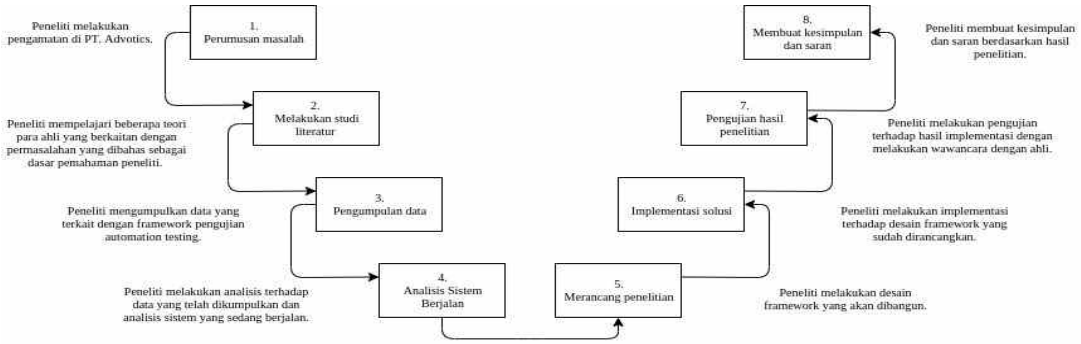
Tujuan dari penelitian ini adalah untuk mengembangkan Framework Web Automation Testing dengan menggunakan Tool Serenity BDD pada aplikasi Supply Chain dengan memberikan informasi mengenai contoh-contoh *tools* yang dapat digunakan untuk melakukan pengujian web secara terautomasi dan memberikan informasi mengenai manfaat penggunaan *framework web automation testing* pada pengujian aplikasi.

Penelitian ini dibatasi pada beberapa masalah berikut:

1. Pada penelitian ini tidak akan membahas semua *tools* yang dapat digunakan untuk *web automation testing*, hanya akan membahas lebih spesifik *tool* Serenity BDD.
2. Pengujian yang akan dilakukan menggunakan *framework* yang akan dikembangkan ini hanyalah pengujian web dengan aspek pengujian *functional test* dan aspek u*ser interface test,* tidak termasuk pengujian API.
3. Pada penelitian ini hanya akan mengembangkan *framework web automation testing*, yang hanya dapat digunakan untuk pengujian web dengan aspek pengujian *functional test* dan aspek u*ser interface test*, tidak termasuk pengujian API.

2 Metode

Tahapan atau alur penelitian yang dilakukan dalam penelitian ini digambarkan sebagai berikut:



Gambar 1 Tahapan atau alur penelitian

2.1 Jenis dan Sumber Data

Sumber data yang dipakai dalam penelitian ini berasal dari data primer dan data sekunder. Data primer yang digunakan dalam penelitian ini adalah data dari hasil wawancara pihak manajemen, developer dan juga tester perusahaan supply chain yang dimaksud. Sedangkan untuk data sekunder yang digunakan dalam penelitian ini adalah data yang diperoleh melalui studi kepustakaan, literatur, artikel, jurnal serta situs di internet yang berkenaan dengan penelitian yang dilakukan.

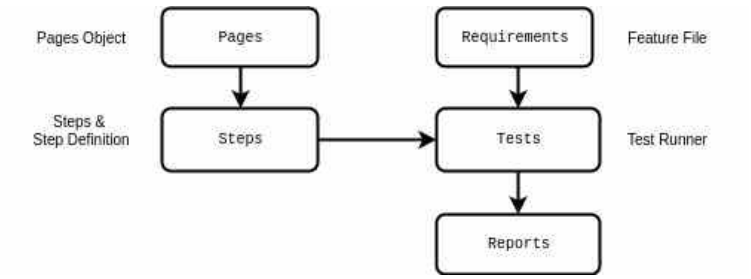
2.2 Arsitektur Dasar Serenity BDD

Serenity BDD merupakan library open source yang membantu tester dalam menulis *automation acceptance test* yang

e-ISBN 978-623-93343-0-7
Seminar Nasional Mahasiswa Ilmu Komputer dan Aplikasinya (SENAMIKA)
Jakarta-Indonesia, 28 Januari 2020

terstruktur dan lebih terawat dengan baik dan juga menghasilkan laporan pengujian yang kaya makna (yang biasa dikenal dengan istilah "living documentation" atau dokumentasi hidup) yang tidak hanya melaporkan pada hasil pengujian, tetapi juga fitur apa yang telah diuji.

Serenity BDD memiliki arsitektur sebagai berikut:

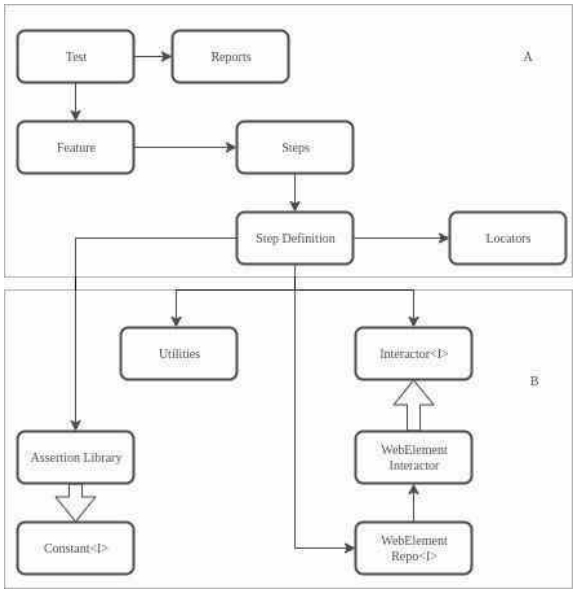


Gambar 2 Arsitektur Dasar Serenity BDD

Dengan arsitektur dasar Serenity BDD tersebut, maka dilakukan perancangan tambahan arsitektur agar framework yang dikembangkan dapat lebih mudah digunakan kembali dan dipahami oleh tim penguji aplikasi.

2.3 Arsitektur Pengembangan Serenity BDD

Perancangan tambahan arsitektur tersebut kemudian dijadikan sebagai rancangan framework yang disarankan. Berikut ini rancangan framework yang disarankan [5]:



Gambar 3 Arsitektur Pengembangan Serenity BDD

Berdasarkan gambar hasil rancangan sistem yang disarankan tersebut, terdapat dua bagian besar rancangan yaitu bagian A dan B. Bagian A merupakan rancangan ulang dari konsep dasar Serenity BDD sedangkan bagian B merupakan rancangan tambahan dari konsep dasar Serenity BDD. Konsep bagian B tersebut secara umum merupakan sebuah *class* atau *interface* yang berisikan method-method dasar yang dirancang se-general mungkin sehingga dapat digunakan kembali oleh penguji dalam menulis skrip. *Class* ataupun *Interface* tersebut dapat dikatakan sebagai *core class* dari rancangan framework yang disarankan.
Berikut ini penjelasan rancangan framework yang disarankan:

e-ISBN 978-623-93343-0-7
Seminar Nasional Mahasiswa Ilmu Komputer dan Aplikasinya (SENAMIKA)
Jakarta-Indonesia, 28 Januari 2020

a. Test

Kelas test atau test runner digunakan untuk menjalankan test pada pengujian automation test. Tes yang dijalankan berdasarkan file fitur yang dideskripsikan pada anotasi @CucumberOptions dengan memberikan path direktori dimana file fitur tersebut berada.

Berikut ini contoh dari kelas test:

```
@RunWith(CucumberWithSerenity.class
)
@CucumberOptions(features="src/test/resources/features/consult_dictionary/
LoginAdvowork.feature")public class LoginAdvoworkTest {}
```

b. Feature

File Fitur menggunakan bahasa tingkat tinggi ataupun dapat dikatakan bahasa sehari-hari yang biasa digunakan yang secara umum dapat dibaca dan dimengerti oleh tim tester, product ataupun tim sales yang berisi skenario pengujian.

Berikut ini contoh dari file fitur:

```
@LoginAdvoworkTes
t
Feature:            Login
Advowork
Scenario: Looking up activity in dashboard menu
Staff
            Given I login into 'advowork-integration' as
  'testingstl' When I select menu 'Staff'
            Then they should see the titlebar 'Staff'
```

c. Step

Class step adalah kelas yang memetakan setiap stories yang ada pada file fitur ke dalam sebuah abstract yang menggambarkan “apa” yang dilakukan oleh test. Pemetaan ini digambarkan cukup sederhana dengan menggunakan keyword yang sama pada file fitur yaitu Given When Then, sehingga teknik ini dianggap cukup mudah dimengerti dan diimplementasikan.

Berikut ini contoh class step:

```
public class LoginAdvoworkSteps
{
    @Steps
    LoginStepsDefinition loginStepDef;
    @Given("I login into '(.*)' as '(.*)'")
        //implementation
```

Class step pada arsitektur pengembangan ini tidak berbeda dengan konsep class step pada arsitektur dasar Serenity

BDD. d. Step Definition

Dengan menggunakan class Step Definition atau Step Library pada arsitektur dasar Serenity BDD dapat menambahkan lapisan abstract antara “apa” dan “bagaimana” dari *acceptance test*. Pendekatan berlapis ini membuat tes lebih mudah dimengerti dan dipertahankan, dan membantu membangun *library* yang hebat dari *business-level steps* yang dapat digunakan kembali dalam pengujian lain. Tanpa pendekatan berlapis seperti ini, step definitions cenderung menjadi sangat teknis, yang dapat membatasi penggunaan kembali dan membuatnya lebih sulit untuk dipahami dan dipelihara (*maintain*).

Berikut ini contoh class step definitions:

```
public class LoginStepsDefinition
{
        public LoginStepsDefinition() {
        @Step
        public void login_into_web(String env, String userLogin) {
            //Implementation login(username, password);
        }
```

e. Locator

Class locator adalah class yang pada konsep dasar framework Serenity BDD disebut sebagai Page Object. Pada

e-ISBN 978-623-93343-0-7
Seminar Nasional Mahasiswa Ilmu Komputer dan Aplikasinya (SENAMIKA)
Jakarta-Indonesia, 28 Januari 2020

class Page Object sebelumnya terdapat variabel penyimpan web element sebuah halaman web dan method-method yang digunakan untuk berinteraksi dengan web element tersebut. Namun pada class Locator ini, hanya berfokus menyimpan web element sebuah halaman web. Untuk method-method yang digunakan untuk berinteraksi dengan halaman web tersebut terdapat di class lain (Interactor). Hal ini bertujuan agar method interaksi tersebut lebih mudah dimaintain dan digunakan kembali pada step definition yang ada.
Berikut ini contoh class locator:

```
public class LoginPage {
        public HashMap<String, String> locators;
        public LoginPage() {
            locators = new HashMap<>();
            locators.put("username", "//input[contains(@id,'username')]"); }
        public String getLocator(String key) {
            return locators.get(key);
        }}
```

f. Interactor <I>
Interactor adalah sebuah interface yang berisi method interaksi dengan web element yang hanya dideklarasikan dengan tidak ditulis secara utuh. Method dalam interface ini adalah method yang digunakan untuk berinteraksi dengan web element seperti mencari elemen, memasukan teks, mengklik button maupun mengambil nilai dari sebuah element.
Berikut ini contoh interface Interactor:

```
public interface Interactor {
        void inputText(String locator, String text);}
```

g. Web Element Interactor
Web Element Interactor merupakan sebuah class implementasi dari interface Interactor. Pada class Web Element
Interactor akan dijabarkan secara utuh apa maupun bagaimana method tersebut berinteraksi dengan web element. h. Web Element Repo <I>
Web Element Repo merupakan sebuah interface yang berisi method yang hanya dideklarasikan dengan tidak ditulis secara utuh. Method-method dalam interface ini adalah method yang berguna untuk menghubungkan secara langsung dengan web element dengan memanfaatkan web driver dalam cara komunikasinya.
Berikut ini contoh Web Element Repo:

```
public interface WebElementRepository {
        WebElementFacade findByLocator(String locator);
        void openBrowser();
    }
```

i. Web Element Repo Serenity
Web Element Repo Serenity merupakan sebuah class implementasi dari interface Web Element Repo. Pada class
Web Element Repo Serenity akan dijabarkan secara utuh apa maupun bagaimana method tersebut bekerja.

j. Utilities
Utility class adalah kelas yang mendefinisikan sekumpulan metode yang melakukan fungsi-fungsi umum yang sering digunakan kembali. Sebagian besar kelas utilitas mendefinisikan metode umum dalam lingkup statis.

k. Assertion Library
Dalam automation testing untuk memutuskan sebuah tes lulus uji atau dapat diterima, dibutuhkan sebuah validasi yang biasa dikenal dengan istilah assertion.

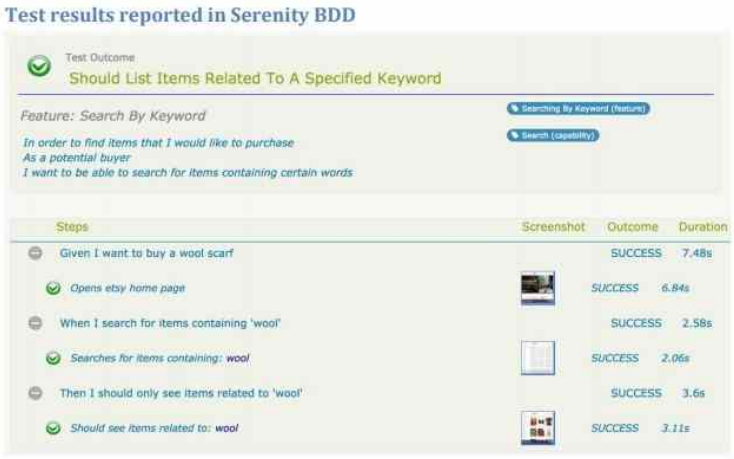
l. Constant
Interface Constant digunakan sebagai tempat menyimpan suatu nilai variabel yang bersifat tetap (konstan) dan tidak bisa diubah sepanjang program berjalan. Interface ini dapat diimplemen langsung oleh Class Step Definitions maupun class Assertion Library. Untuk membuat konstanta, berikut ini format penulisan yang dapat digunakan [6]:

```
<access_modifier> static final <tipe_data> <NAMA_KONSTANTA>= nilai_konstanta;
```

m. Report
Report pada arsitektur pengembangan framework yang baru, tidak berbeda dengan report pada arsitektur dasar

e-ISBN 978-623-93343-0-7
Seminar Nasional Mahasiswa Ilmu Komputer dan Aplikasinya (SENAMIKA)
Jakarta-Indonesia, 28 Januari 2020

Serenity BDD. Report yang dihasilkan dari Serenity BDD berisi detail sebagai berikut :



Gambar 4 Report Serenity BDD

Dalam Serenity Report diberitahukan Scenario mana yang lulus uji dan tidak. Selain informasi tersebut, report yang dihasilkan akan menampilkan hasil tangkapan layar saat melakukan pengujian aplikasi dan memberitahu durasi pengujian yang dilakukan.

2.4 Perbandingan Arsitektur Dasar dan Arsitektur Pengembangan Serenity BDD

Berikut ini tabel hasil perbandingan arsitektur dasar dan arsitektur pengembangan framework Serenity BDD:

Tabel 1 Perbandingan Arsitektur Dasar dan Arsitektur Pengembangan Serenity BDD

Kategori	Framework Serenity BDD			
	Arsitektur Dasar		Arsitektur Pengembangan	
	Keunggulan	Kelemahan	Keunggulan	Kelemahan
Instantiate Web Element pada Page Object	Instansiasi web element pada Page Object tidak memungkinkan terjadinya redundansi karena penamaan elemen harus menggunakan nama yang unik.	Penulisan/instansiasi web element pada page object membutuhkan setidaknya 2 baris kode untuk masing-masing elemen.	Penulisan/instansiasi web element pada page object menggunakan konsep HashMap<Key, String> sehingga cukup satu baris kode untuk masing-masing elemen.	Penggunaan konsep HashMap pada page object memberikan peluang terjadinya redudansi key.
Komponen yang dimiliki Class Page Object	N/A	Page Object menyimpan web element dan fungsi-fungsi interaksi web element pada	Page Object hanya berfokus menyimpan web element dari masing-masing page	N/A

e-ISBN 978-623-93343-0-7
Seminar Nasional Mahasiswa Ilmu Komputer dan Aplikasinya (SENAMIKA)
Jakarta-Indonesia, 28 Januari 2020

		masing-masing page web.	web.	
Fungsi-fungsi interaksi Web Element	N/A	N/A	Fungsi-fungsi interaksi web element disimpan pada class interactor sehingga lebih mudah untuk digunakan kembali.	N/A
Stories pada file fitur	Stories pada file fitur dapat digunakan untuk hal/contoh yang berbeda dari skenario yang sama.	N/A	Stories pada file fitur dirancang lebih umum agar dapat digunakan untuk hal/contoh yang berbeda dari skenario yang sama.	Dengan desain stories yang umum menjadikan setiap stories memiliki setidaknya lebih dari satu parameter.
Assertion yang digunakan	N/A	Class step definition mendefinisikan sendiri assertion yang digunakan sebagai validasi tes.	Memiliki class assertion library yang menyimpan semua assertion yang dapat digunakan kembali pada class step definition.	N/A
Penggunaan variabel yang sama	N/A	Menyimpan nilai variabel pada step definition yang memungkinkan terjadi pengulangan penulisan pada class step definition yang berbeda untuk nilai variabel yang sama.	Memiliki class constant sebagai tempat menyimpan suatu nilai variabel yang bersifat tetap (konstan) dan tidak bisa diubah sepanjang program berjalan. Jika ingin mengubah nilai dari suatu variabel, maka cukup mengubahnya di class constant tanpa harus mengubah di semua class step definition yang menggunakannya.	N/A

Berdasarkan tabel hasil perbandingan tersebut dapat dilihat bahwa framework Serenity BDD dengan arsitektur pengembangan memiliki keunggulan yang lebih dibandingkan framework Serenity BDD dengan arsitektur dasar. Hal ini dikarenakan pada framework Serenity BDD dengan arsitektur pengembangan memiliki core class yang dapat

e-ISBN 978-623-93343-0-7
Seminar Nasional Mahasiswa Ilmu Komputer dan Aplikasinya (SENAMIKA)
Jakarta-Indonesia, 28 Januari 2020

langsung digunakan kembali saat menulis skrip karena class-class tersebut berisi metode interaksi yang umum digunakan saat pengujian dan menjembatani cara test berhubungan dengan elemen web dari aplikasi yang akan diuji.

3 Hasil dan Analisa

Berdasarkan hasil implementasi dilakukan tahap pengujian agar framework yang dibangun memiliki nilai uji dan validitas yang baik sebagai solusi untuk permasalahan yang dihadapi.
Hasil pengujian berdasarkan hasil wawancara dengan ahli dapat dilihat pada tabel berikut:

Tabel 2 Hasil Wawancara dengan Ahli

Pertanyaan	Hasil				
	NA	AT	TA	FH	CP
a. Apakah framework dapat dan siap digunakan untuk pengujian automation test?	5	4	5	4	4
b. Apakah framework dapat menangani masalah pengujian aplikasi yang dilakukan berulang-ulang?	4	4	4	4	4
c. Apakah framework dapat meminimalisir human error dalam melakukan pengujian aplikasi?	3	3	4	4	4
d. Apakah konsep penggunaan framework mudah digunakan dan dipahami?	4	4	3	4	4
e. Apakah framework menggunakan/menyediakan bahasa level tingkat tinggi?	4	4	4	4	4
f. Apakah framework memungkinkan digunakan oleh penguji yang tidak memiliki latar belakang coding?	2	3	2	3	2
g. Apakah framework menyediakan fitur yang configurable?	3	4	4	3	4
h. Apakah framework menyediakan abstract class yang digunakan sebagai penghubung dengan library yang dimiliki Serenity BDD?	5	5	4	5	5
i. Apakah framework memiliki fungsi interaksi umum yang dapat digunakan untuk memudahkan penulisan script automation testing?	4	3	4	4	3
j. Apakah waktu eksekusi test efektif (lebih cepat) dengan menggunakan framework?	3	3	4	4	4
k. Apakah framework dapat menghasilkan report dari hasil pengujian aplikasi?	5	4	4	4	4

Penilaian yang digunakan pada hasil wawancara menggunakan maturity model dengan definisi [7] [8] [9]:
1: Initial/ad hoc. Tidak dilakukan pengelolaan proses yang terorganisasi. Setiap proses ditangani tanpa menggunakan standar.
2: Repeatable. Sistem/perangkat lunak sudah memenuhi spesifikasi namun masih berulang.

e-ISBN 978-623-93343-0-7
Seminar Nasional Mahasiswa Ilmu Komputer dan Aplikasinya (SENAMIKA)
Jakarta-Indonesia, 28 Januari 2020

3: Defined. Ada standar/prosedur yang harus diikuti sebagai panduan tiap elemennya dalam mengembangkan maupun mengelola aplikasi.
4: Managed. Proses dipahami dan distabilkan secara kuantitatif. Implementasi dilakukan secara baik.
5: Optimized. Proses pengembangan sistem terstandarisasi secara berkelanjutan, di monitor dan ditingkatkan berdasarkan ukuran dan analisa data pada level 4.

4 Kesimpulan

Berdasarkan penelitian yang telah dilakukan, peneliti dapat menyimpulkan beberapa hal sebagai berikut:

1. Adanya sebuah framework automation testing dapat mendukung proses pengujian aplikasi pada perusahaan dengan mempermudah penguji melakukan pengujian aplikasi dengan mengotomatiskan test case pengujian dan menjalankannya secara otomatis.
2. Framework web automation testing yang dibangun dapat memenuhi kebutuhan pengguna penguji karena pada framework ini terdapat fungsi/fitur umum seperti pada class Utilities, Interactor, Web Element Interactor, Web Element Repository, Web Element Repo Serenity, Assertion Library dan Constant yang akan mempermudah *tester* untuk tidak menuliskan *code* secara berulang saat dibutuhkan, atau dengan kata lain, *tester* dapat langsung menggunakan fungsi yang sudah disediakan.
3. Framework memiliki konsep kerja yang mudah dipahami dan digunakan oleh penguji karena memetakan step pengujian pada sebuah class step definitions yang dapat digunakan kembali.
4. Dengan menggunakan konsep file *feature*, framework memiliki fitur yang mudah dibaca dan dipahami berbagai *stakeholder* mulai dari *tester*, *product*, *sales* dan tim *management*. Atau dengan kata lain, framework menggunakan bahasa level tingkat tinggi.
5. Framework memiliki fitur yang *configurable* agar dapat digunakan pada beberapa aplikasi dan *environment* yang akan diuji.
6. Framework memiliki kemampuan melakukan eksekusi test dengan cepat untuk meningkatkan efektivitas penguji
   dalam melakukan pengujian.
7. Framework dapat melakukan generate report berdasarkan hasil pengujian, sehingga tim penguji tidak perlu mendokumentasikan hasil pengujiannya dalam sebuah Google Sheet.

Referensi

[1] D. Triwibowo, R. Kridalukmana and K. T. Martono, *Jurnal Teknologi dan Sistem Komputer,* vol. 3 no. 2, April 2015.

[2] A. Purnomo, *Software Testing Aplikasi Website PT Gramedia Menggunakan Metode Blackbox pada PT WGS Bandung,* 2013.

[3] "Wikipedia. Manual Testing," [Online]. [Accessed 05 November 2018].

[4] Baeldung, "Introduction to Serenity BDD," [Online]. [Accessed 15 November 2018]. [5] [Online]. Available: http://desy.lecturer.pens.ac.id/Workshop%20Pengembangan%20Perangkat%20Lunak/4_Class%20Diagram.pdf. [Accessed 19 December 2019].

[6] [Online]. Available: https://www.duniailkom.com/tutorial-belajar-java-pengertian-dan-cara-pembuatan-konstanta-bahasa- java/. [Accessed 21 December 2019].

[7] [Online]. Available: http://eitbokwiki.org/Enterprise_IT_Maturity_Assessments. [Accessed 18 January 2020].

[8] [Online]. Available: https://www.axelos.com/best-practice-solutions/itil/itil-maturity-model. [Accessed 18 January 2020].

[9] [Online]. Available: http://karto-iskandar.blogspot.com/2010/07/pemanfaatan-cmm-capability-maturity.html. [Accessed 18 January 2020].