ANALISA METODE FIT DALAM *USER ACEPTANCE TESTING*

Venda Triandito, Dana Sulistiyo Kusumo, S.T., M.T., Ph.D.

[1,2,3]Fakultas Informatika, Universitas Telkom, Bandung
[4]Divisi Digital Service PT Telekomunikasi Indonesia
[1]venda.trian1993@icloud.com, [2]danakusumo@telkomuniversity.ac.id.

Abstrak

Pada tugas akhir ini akan membahas tentang salah satu *teknik* dalam *user acceptance testing* yang disebut dengan *FIT table.* Dalam karya tulis ini penulis akan mengeksplorasi lebih jauh tentang teknik *FIT table* dalam *user acceptance test* untuk dapat lebih menjelaskan apa itu *FIT table.* Tugas akhir ini juga mengeksplorasi bagaimana cara melakukan testing dengan menggunakan *software requirement specification* untuk mengatahui bagaimana hubungan *FIT table* dan software requirement. Dalam membantu proses penyusunan FIT *table* dengan *software requiremen spesificationt,* dalam tugas akhir ini disusun juga *requirement traceability matrix* dan juga *test case*. Untuk mengamati bagaimana *software requirement specification* berpengaruh dalam proses pengetestan dalam tugas akhir ini penulis akan membandingkan 4 skenario berbeda dalam pengetestan. Hasilnya adalah pemahaman user yang baik terhadap software requirement yang dibantu dengan *requirement traceability matrix* dan *test case* akan mempermudah dalam penyusunan FIT t*able.*

.

Kata kunci: *software requirement specification, user acceptance testing, FIT table.*

Abstract

This paper would explain about a technic on the *user* acceptance testing called *FIT table*. In this paper author will explore about *FIT table* on the *user* acceptance testing to explain what *FIT table* is. This paper also explore how to run testing using software requirement specification to explain connection between *FIT table* and software requirement. To ease in form of FIT table with software requirement, in this paper also create requirement traceability matrix and test case. In order to observe how software requiremet take effect on testing, in this journal author will compare 4 testing scenario. The result is user understanding of software requirement that helped by requirement traceability matrix and test case will ease in the making of FIT table.

Keywords: software requirement specification, user acceptance testing, FIT table.

1. Pendahuluan

Latar Belakang

Dalam siklus pengembangan aplikasi atau yang biasa disebut dengan SDLC (*software development life cycle*), *testing* adalah salah satu proses yang harusnya dilakukan pada setiap pengembangan suatu sistem[12]. Hal ini untuk mengecek, melacak dan juga membantu pihak developer dalam proses perbaikan dari sistem tersebut sehingga sesuai dengan apa yang diharapkan oleh konsumen [13]. Untuk itu maka sangatlah penting bagi *tester* atau *quality assurance* untuk melakukan testing terhadap suatu aplikasi sebelum aplikasi tersebut di serah terimakan kepada konsumen. Test tersebut disebut dengan *user acceptance testing* [1]. Tujuan utama dari proses *testing* ini adalah untuk memastikan bahwa setiap skenario yang ada dalam aplikasi telah berjalan sesuai dengan yang diharapkan dan memenuhi standar *requirement* yang ditulis dalam *software requirement spesification*. *Software requirement specification* merupakan hal yang menjadi dasar dalam pengetesan karena apa yang telah dikerjakan oleh developer dijelaskan dalam software requirement specification dan *user* acceptance test akan mengetest hal tersebut.

Secara umum *user acceptance test* terbagi menjadi dua yaitu otomatis atau dengan bantuan *tools* dan juga manual tanpa bantuan *tools* apapun. Salah satu teknik yang digunakan dalam *user acceptance testing* adalah FIT. FIT adalah kepanjangan dari *framework integration test,* FIT merupakan *acceptance test* yang berbasis tabel [4]. Dalam FIT *customer* dan *analyst* menuliskan semua yang telah dilakukan dalam *acceptance testing* dalam suatu tabel. Hal ini tentu sangat membantu karena kesalahan komunikasi antar pihak yang telibat dalam pembuatan *software*

dan juga spesifikasi kasus uji merupakan masalah yang sering dihadapi dalam UAT. Dalam FIT terdapat 3 hal dasar yaitu *FIT table, fixture* dan juga *test-runner* [10].

FIT merupakan *user acceptance testing* yang bersifat otomatis dan dapat berjalan setiap kali suatu proses dilakukan [4]. FIT juga dapat berjalan sepanjang aplikasi tersebut dijalankan. Selain itu FIT juga merupakan media komunikasi antara *tester, programmer dan user* [10]. Dengan hal tersebut FIT diharapkan dapat menyelesaikan testing dengan cepat dan mudah dibandingkan dengan metode manual yang melakukan segala sesuatunya dengan cara manual tanpa bantuan *tools* maupun *script*.

Akan tetapi dalam penyusunan FIT *table* pemahaman terhadap *software requirement* akan sangat membantu. Maka *software requirement* yang baik akan mempermudah dalam penyusunan FIT *table*. Namun ketika proses penyusunan Fit table pemahaman user terhadap software requirement terkadang berbeda dikarenakan user yang tidak memiliki latar belakang teknis. Maka dalam proses pemahaman software requirement, user akan dibantu dengan adanya *requirement traceability matrix, test case* dan pendampingan dari tester. *requirement traceability matrix* berfungsi untuk memahami fungsionalitas yang terdapat pada program, sedangkan *test case* berfungsi untuk memberikan gambaran output yang dihasilkan ketika suatu input dengan kriteria tertentu dilakukan terhadap suatu fungsionalitas tersebut.

Topik dan Batasannya

Berdasarkan latar belakang diatas maka dapat disimpulkan rumusan masalah sebagai berikut:

a. Bagaimana FIT diterapkan dalam proses *user acceptance testing* menggunakan *software requirement spesification*
b. Bagaimana evalusi hasil penerapan FIT pada pengujian berdasar SRS?

1.1. BatasanMasalah

Untuk menghindari terjadinya perluasan ruang lingkup, batasan masalah sebagai berikut:

a. Metode FIT diimplementasikan di aplikasi yang berbasis *desktop.*
b. *fixture* disusun berdasarkan *test case* yang didapat dari *requirement traceability matrix* yang telah disusun sebelumnya.
c. *Software requirement* yang disediakan berupa poin poin spesifikasi yang ada didalam aplikasi.
d. Output yang dihasilkan berupa hasil testing atau laporan hasil testing.
e. *User* yang berperan merupakan admin program yang memiliki akses ke semua fungsi dalam program.

Tujuan

Berdasarkan rumusan masalah diatas, adapun tujuan dari penelitian ini adalalah menggunakan metode FIT dalam *user acceptance testing* adalah untuk melakukan testing dengan cara otomatis untuk menguji suatu aplikasi. Test dilaksanaan untuk memastikan bahwa prosedur dalam aplikasi sudah berjalan sesuai dengan fungsi seperti yang sudah dituliskan dalam *software requirement specification*. Dan dari hasil testing tersebut akan diketahui berapa skenario testing yang berhasil dilakukan dan sesuai dengan hasil yang diharapkan oleh skenario testing yang dituliskan dalam *FIT table*.

2. Studi Terkait

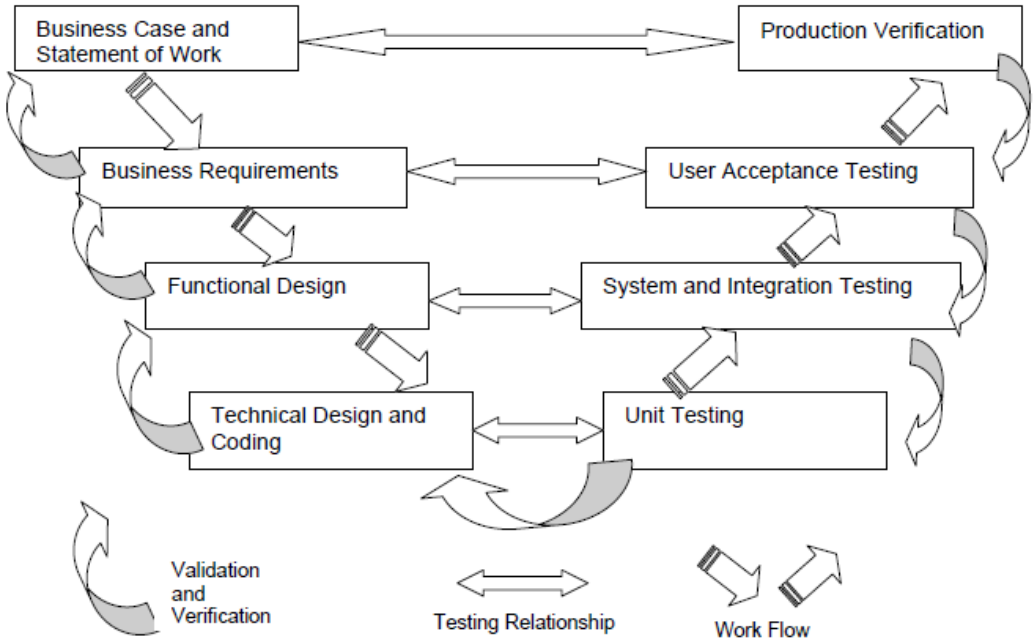
2.1 Software testing

Definisi dari Testing adalah suatu metode atau langkah yang dilakukan untuk mengevaluasi apa yang sudah dikerjakan sesuai dengan apa yang diinginkan atau belum[1]. Hal tersebut sangat penting dilakukan dalam setiap hal yang telah dikerjakan. Termasuk juga dalam siklus pengembangan *software* atau yang lebih dikenal dengan *software testing. Software testing* adalah suatu metode yang untuk melakukan verifikasi dan validasi apakah *software* yang telah selesai dibuat sudah memenuhi kriteria bisnis dan teknis yang telah ditentukan sebelumnya dan berjalan sesuai yang diharapkan[2]. Secara garis besar proses yang ada dalam software testing aada 3 yaitu validasi, verifikasi dan penemuan kesalahan. *Software testing* bukanlah pekerjaan yang dilakukan oleh satu individu melainkan harus dikerjakan oleh suatu tim. Kerjasama dan komunikasi yang baik antara *tester, developer, dan user* sangatlah menentukan kesuksesan jalannya test.

Menurut dengan tipenya *software testing* dibagi menjadi dua yaitu *manual testing* dan juga *automated testing*. Manual testing adalah test yang bersifat manual tanpa menggunakan tool atau script apapun sedangkan automated testing adalah test dengan menggunakan bantuan tools atau script sehingga test yang diakukan dapat disellesaikan lebih cepat[1].automated testing memiliki beberapa keunggulan antara

lain dapet menekan biaya yanng disebabkan karena kegagalan, dapat menghemat waktu dan juga dapet meningkatkan produktivitas dari *resouces* [3]. Selain itu dalam software testing terdapat juga istilah *black box testing* dan *white box testing.* Perbedaan dari keduanya adalah jika white box testing maka tester dituntut untuk mengetahui sorce code dari suatu software termasuk struktur serta logika dari code tersebut. Sedangkan pada black box testing, tester hanya berfokus pada input dan outputnya tanpa mengetahui logika yang berjalan didalam software tersebut.

Dalam testing terdapat proses dan metodologi yang dipilih dan digunakan untuk mencapai hasil yang diinginkan dan menunjukan setiap tahapan yang terjadi selama proses yang dilakukan[5]. Model yang paling kerap digunakan adalah V-Model. Model ini disebut juga sebagai *verification and validation model* [5]. Dibawah ini merupakan gambaran dari v model.



Gambar 1 V model [3]

V model merupakan model yang simple dan mudah untuk di impementasikan. Selain itu testing dapat dilakukan dalam setiap tahapan rancangan sehingga setiap kesalahan dapat di tangani lebih awal sehingga dapat mempermudah tahapan selanjutnya dan mengurangi resiko kesalahan di akhir.

2.2 Acceptance testing

Selain untuk mengetahui apakah *code* yang telah dibuat telah bekerja dengan benar, acceptance test dilakukan untuk mengetahui apakah software sudah memenuhi spesifikasi yang sudah ditentukan oleh klien atau belum. Test ini dilakukan oleh tim dari *quality assurance* dan *user*. Untuk melakukan test ini tim dan *user* akan bekerja sama untuk menyiapkan skenario dan kasus yang tertulis untuk dilakukan dengan software yang akan di test[1].

Dalam *acceptance test* dikenal juga dengan *automated acceptance test*, test ini adalah *acceptance test* yang bersifat otomatis yang berfungsi untuk mengecek apa yang sudah dikerjakan oleh *developer* dan juga menspesifikkan kriteria yang ada dalam *software* tersebut. *Automated acceptance* test biasanya diterapkan pada setiap iterasi dari setiap hal yang telah di eksekusi atau dilakukan dari *requirement* yang telah ditentukan [11].

2.3 User Acceptance testing

User acceptance testing adalah proses testing untuk memastikan bahwa software yang dibuat sudah sesuai dengan pengguna atau belum. UAT dilakukan untuk memastikan bahwa solusi yang ditawarkan aplikasi sudah sesuai dengan yang diinginkan oleh pengguna. UAT merupakan testing yang bersifat seperti black box testing. Yaitu testing tanpa mengetahui struktur code yang ada dalam program.

2.4 Software requirement spesification

Software requirement specification atau yang kerap disingkat SRS adah suatu dokumen yang berisi tentang apa saja komponen atau kebutuhan yang harus dipenuhi yang terdapat didalam suatu program. Dokumen ini dibuat *requirement engineer* untuk developer program dalam menggali apa saja yang dibutuhkan end *user* dalam programnya. SRS merupakan sebuah komponen penting yang harus ada dalam suatu program karena hal ini dapat membantu developer dalam pengerjaaan suatu program agar lebih optimal baik dari segi waktu maupun biaya.

2.5 FIT

Fit adalah sebuah metode testing dalam *user acceptance test* yang membantu tester dalam berkomunikasi dan bekerjasama dalam menyusun skenario testing dengan user. FIT merupakan *automated acceptance* test yang bersifat *black box*. Fit membuat feedback yang menghubungkan antara customer dan juga programmer. FIT juga mengijinkan customer dan tester untuk menggunakan alat bantuan lain seperti microsoft office untuk mengetahui atau mensimulasikan bagaimana seharusnya program berjalan nantinya[4]. Fit menggunakan tabel dalam menuliskan test dan penyajian data yang telah secara otomatis dilakukan selama test. Dalam FIT terdapat beberapa hal yang penting yaitu

2.5.1 *Testcase*

Dalam tabel berisi *test case* atau proses yang ada dalam sistem dan apa hasil yang seharusnya disajikan oleh sistem setelah proses input dengan kriteria tertentu dijalankan. Dalam *test case* akan dituliskan apa saja hasil yang diharapkan ketika sebuah prosedur dengan kriteria tertentu dijalankan pada sebuah fitur yang terdapat pada program.

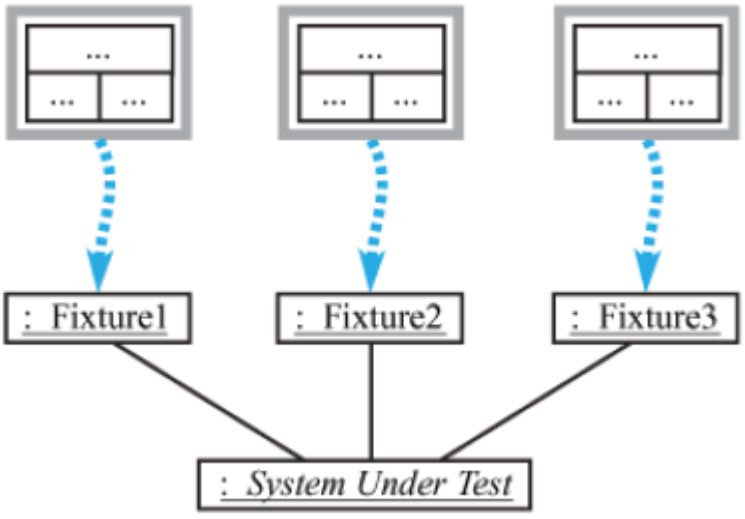
2.5.2 Fixture

Dalam melakukan pengecekan data yang telah sebelumnya dituliskan dalam tabel maka diperlukanlah skenario pengetesan. Dalam FIT skenario tersebut disebut dengan *fixture. Fixture* bekerja dengan cara mengecek proses yang ada dalam tabel dan menghasilkan output yang berisi apakah proses yang berjalan sudah benar atau belum.

2.5.3 System under test

System under test adalah sebuah system atau proses yang akan di lakukan test .

Secara umum gambaran FIT adalah seperti gambar dibawah ini



Gambar 2 Alur proses fit secara umum [10]

2.6 Requirement traceability matrix

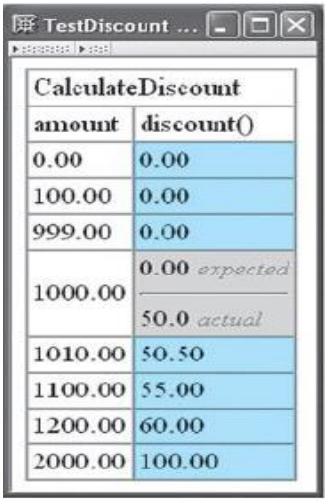
RTM adalah sebuah tabel yang digunakan untuk menelusuri apa saja fungsionalitas yang terdapat dalam program. Dalam RTM dijelaskan asalah apa saja yang ditangani program serta hal apa saja yang diinginkan oleh stakeholder dan bentukan penyelesaian yang ditawarkan dalam menu tertentu. RTM dibuat untuk lebih

memudahkan proses testing karena fungsi dari RTM yang digunakan untuk menelusuri fungsionalitas yang ada pada use case yang kemudian akan digunakan untuk menyusun *test case* yang diaplikasikan dalam pembuatan fixture.dalam RTM akan dijelaskan proses yang terjadi pada setiap activity atau menu dalam aplikasi.

2.7 *FIT table*

FIT table merupakan suatu table yang berisi gambaran skenario proses pengujian yang akan dilakukan terhadap system. *FIT table* berfungsi untuk menjadi media komunikasi antara tester dan juga *end user*. Dalam *FIT table* ini akan dituliskan skenario pengetestan yang akan dilakukan yang telah disusun oleh *user* sebelumnya. *End user* cukup berperan dalam pembuatan *FIT table* karena komunikasi yang baik antara *tester* dan *end user* akan menghasilkan *FIT table* yang sesuai dengan kriteria input atau output yang ditentukan.

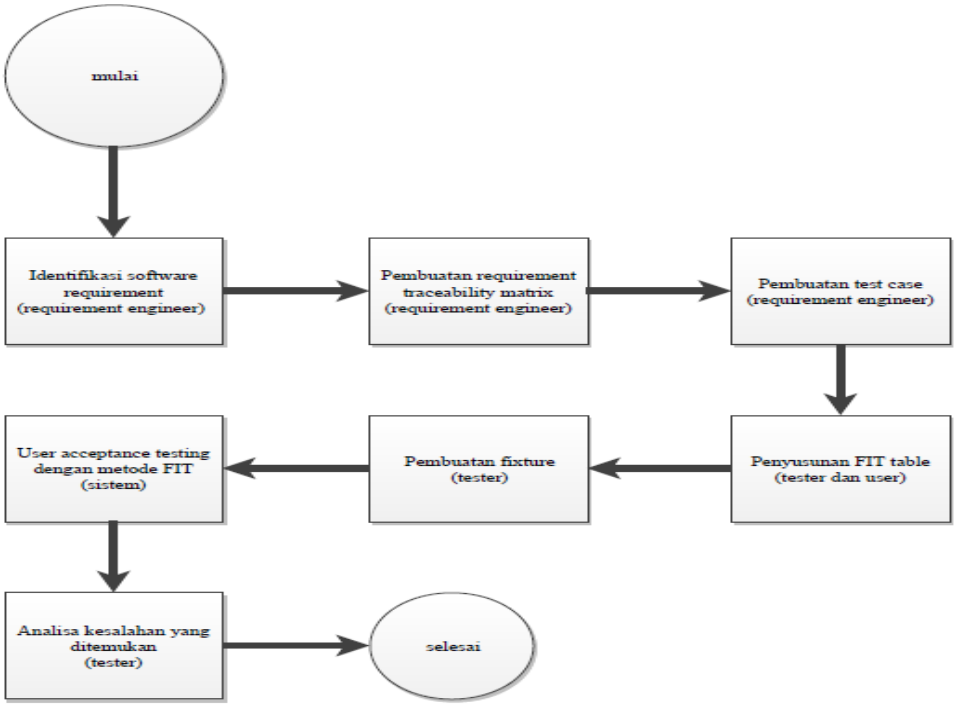
Tabel 1. *FIT table*



3. Sistem yang Dibangun

3.1.Model dan deskripsi sistem

FIT terdiri dari beberapa tahapan. Tahapan pertama yang dilakukan adalah menemukan fungsional atau fungsi apa saja yang terdapat dalam aplikasi tersebut serta kriteria atau *system under test*. Setelah itu menjadikan fungsi-fungsi tersebut kedalam bentuk tabel. Dalam membuat tabel dapat menggunakan tools seperti misalnya *microsoft excel*. Yang kemudian akan digunakan untuk membuat *fixture*. Tabel disini merupakan media komunikasi antara *tester, programmer dan customer* [10]. Komunikasi tersebut berfungsi untuk menentukan skenario pengujian terhadap fungsi yang telah ada dalam tabel yang telah dibuat. *Fixture* akan mengecheck apakah fungsi telah berjalan sesuai dengan yang diharapkan oleh *system under test* atau belum.



Gambar 3 Alur penelitian

Flowchart diatas adalah gambaran alur proses pengujian yang dilakukan oleh penulis dalam proses pengujian menggunakan metode FIT dengan menggunakan *software requirement spesification* dalam tugas akhir ini. Dari hasil proses pengujian tersebut maka akan diketahui prosedur mana saja yang telah berjalan sesuai dengan skenario pengujian yang telah disusun.

3.1.2. *Identifikasi software requirement*

Pada tahap ini akan dilakukan identifikasi dan pemahaman pada spesifikasi dari aplikasi yang akan dilakukan proses testing. Hal ini diperlukan untuk memahami lebih lanjut tentang apa saja poin poin yang terdapat dalam program dan harus dipenuhi oleh pembuat program. Pemahaman pada software requirement akan sangat membantu dalam penyusunan *step* selanjutnya. Dalam tugas akhir ini software requirement telah disedikan oleh pengembang aplikasi berupa poin-poin berdasarkan spesifikasi yang ada pada aplikasi. Dalam SRS terdapat 7 fungsional yang dibuat oleh pengembang dan telah dituliskan dalam aplikasi dalam bentuk tampilan dari fitur tersebut pada bagian product function. Dan penjelasan fungsi dari fitur tersebut terdapat pada bagian *functional requirement.* Pada bagian tersebut terdapat 11 poin penjelasan tentang fitur yang terdapat pada aplikasi. Pada identifikasi ini akan ditentukan keyword yang akan menjadikan dasar dalam pembuatan *requirement traceability matrix.* Keyword yang disusun dalam tugas akhir ini adalah setiap kata benda yang ditemukan dalam penjelasan fitur dalam SRS dan memiliki kesamaan dengan penjelasan fitur lain, kesamaan diperlukan untuk pengelompokan suatu fitur dengan fungsi yang sama. Seperti dalam fitur login dan logout kata benda yang ditemukan antara lain keamanan, aplikasi dan fitur. Dan hanya kata keamanan yang memiliki kesamaan sehingga kata keamanan yang akan menjadi keyword. Kata benda digunakan karena biasanya kata benda menjelaskan tentang *input* atau *output* dari proses komputasi yang dilakukan.

3.1.3. Penyusunan *requirement traceability matrix*

Requirement traceability matrix diperlukan untuk mempermudah pemahaman fungsionalitas yang terdapat pada program sehingga akan membantu dalam proses testing dan juga membantu *end user* untuk lebih memahami *software requirement* yang terdapat dalam program. *Requirement traceability matrix* di tugas akhir ini dibuat untuk mengetahui aspek apa sajakah yang sudah dipenuhi oleh program berdasarkan dengan fitur yang telah dijelaskan dalam *software requirement specification*.

3.1.4. Penyusunan *test case*

Test case merupakan suatu tes yang dilakukan berdasarkan pada prosedur input ataupun masukan yang memiliki suatu kriteria tertentu dan hasil yang akan diharapkan dari suatu input tersebut. Dalam pembuatan *test case* dalam tugas akhir ini dikelompokkan pada keyword yang ditemukan dalam problem description pada requirement traceability matrix. Keyword tersebut merupakan keyword yang didapatkan dari peng identifikasian software requirement yang telah dikembangkan menjadi sebuah pertanyaan di problem description pada requirement traceability matrix.

Tabel 2. *Test case*

Keyword dalam SRS	*Test case* ID	*Test case* name	*Test case* description	Pre condition	Step	Expect	Post condition
Keamanan	TC-01	Login	User melakukan login untuk mengakses menu dalam aplikasi	1. Akun sudah terdaftar pada system 2. Akun berada pada halaman awal atau halaman login	1. User memasukkan username 2. User memasukkan password(password bersifat case sensitive) 3. User mengklik login	User berhasil login	User berada pada halaman utama aplikasi
	TC-02	Logout	User melakukan logout untuk mengakhiri akses menu utama pada aplikasi	a. User sudah berl melakukan pros login sebelumn	i.User mengklik pilihan oke dalam menu logout	User berhasil logout	User berhasil keluar dari semua akses aplikasi

3.1.5. Pembuatan *FIT table*

FIT table berisi tentang skenario proses pengetestan serta expected result ketika proses dijalankan. Dalam memudahkan penyusunan *FIT table* maka *user* akan dibantu oleh adanya *test case* serta requirement traceability matrix. Karena dari *test case* tersebut *user* mendapat gambaran akan hasil suatu prosedur ketika dilakukan input tertentu dalam fitur atau prosedur tersebut. *FIT table* ini dibuat berdasarkan *test case* yang telah disusun sebelumnya. *Test case* akan memberikan penjelasan tentang hasil ketika dilakukan suatu prosedur dengan kriteria tententu terhadap fungsionalitas tersebut beserta dengan hasil yang akan didapatkan ketika prosedur tersebut dilakukan. Hal ini yang akan membantu dalam penyusunan skenario pengetesan setelah didapatkan gambaran dari proses tersebut ketika dilakukan suatu prosedur dengan kriteria tertentu. Dibawah ini merupakan contoh dari *FIT table* yang telah dibuat

Table 3. *FIT table* menu login

TC-01 LOGIN

Input		**Expected Output**
Username	**Password**	
Admin	Siposfit	TRUE
Admin	Siposfit	TRUE
ADMIN	SIPOSFIT	FALSE
Admin	Sipos	FALSE
Admi	Siposfit	FALSE

3.1.6. Penyusunan *fixture*

Fixture merupakan sekumpulan script yang berguna untuk melakukan atau mengekseskusi test yang telah disusun dalam *FIT table*. Pembuatan script fixture dalam tugas akhir ini didasarkan dengan dengan skenario yang telah dituliskan dalam *FIT table*. Fixture akan menerjemahkan skenario tersebut kedalam Bahasa pemrograman sehingga dapat dilakukan pengetestan. Dibawah ini merupakan salah satu contoh fixture yang telah dibuat.

```
public void testTC01_Section1() {
    System.out.println("TC-01 Login with User:admin and Password:siposfit; expected:TRUE");
    frmMain frmmain = new frmMain();

    dlgLogin instance = new dlgLogin(frmmain, true, frmmain);

    boolean expected = true;
    boolean result = instance.login("admin", "siposfit");
    assertEquals(expected, result);

}
```

gambar 4 fixture

3.1.7. Pengujian dengan skema pengujian yang berbeda

Pengujian ini dimaksudkan untuk membandingkan pengujian dengan skema yang telah disusun sebelumnya dengan skema berbeda. Ada 4 skema berbeda yang akan dijadikan pembanding. Yaitu pembuatan *FIT table* oleh *user* tanpa tester menjelaskan proses-proses yang akan diuji, pembuatan *FIT table* oleh *user* hanya dengan memberikan SRS tanpa penjelasan lebih lanjut tentang SRS akan tetapi dengan penjelasan secara singkat tentang proses yang akan diuji dan yang terakhir adalah penyusunan *FIT table* oleh *user* dengan diberikan SRS dan *test case*. Perbandingan ini dilakukan untuk membuktikan sejauh mana SRS dapat membantu *user* dalam menyusun *FIT table* serta proses apa saja yang dibutuhkan untuk menyusun *FIT table*.

4. Evaluasi.

4.1 Hasil Pengujian

Tabel 4. Tabel hasil pengujian

No	Activity	Data	Pass	not pass
1	login	5	5	0
2	logout	3	3	0
3	add user	8	8	0
4	add item	8	8	0
5	add member	8	8	0
6	sales transaction	5	5	0
7	top up transaction	5	5	0

Tabel 4 merupakan hasil dari pengujian dengan skenario pertama (ketiga skema lainnya tidak menghasilkan tabel hasil pengujian). Dalam pengujian pertama dilakukan penyusunan RTM dan *test case* serta bantuan penjelasan tester dalam proses penyusunan *FIT table*. Dalam pengujian ini disusun RTM agar *user* dapat memahami fungsionalitas yang terdapat dalam program yang disusun setelah mengidentifikasi SRS. Kemudian dari RTM akan disusun *test case* untuk mempermudah *user* dalam menyusun *FIT table* dikarenakan dalam testcase berfungsi untuk menggambarkan hasil tertentu ketika suatu prosedur input dilakukan, dan kemudian skenario dalam *FIT table* tersebut yang kemudian diterjemahkan menjadi fixture sehingga dapat diperoleh hasil pengujian. Pengujian yang kedua dilakukan adalah pembuatan *FIT table* oleh *user* tanpa tester menjelaskan proses-proses yang akan diuji, dalam pengujian ini *user* tidak dapat menyusun *FIT table* dikarenakan *user* tidak mengerti apa yang harus diuji dan bagaimana suatu prosedur dalam program berjalan serta apa fungsinya. Yang ketiga adalah pembuatan *FIT table* oleh *user* hanya dengan memberikan SRS tanpa penjelasan lebih lanjut tentang SRS akan tetapi dengan penjelasan secara singkat tentang proses yang akan diuji dalam pengujian ini *user* mulai mengerti gambaran umum

proses dalam program, akan tetapi *user* belum dapat menyusun *FIT table* dikarenakan *user* belum memiliki gambaran hasil suatu prosedur. Dan yang terakhir adalah penyusunan *FIT table* oleh *user* dengan diberikan SRS dan *test case.* Dalam pengujian ini *user* sudah memiliki gambaran tentang proses yang terdapat dalam program serta gambaran hasil ketika suatu prosedur dilakukan. Disini *user* masih kebingungan dalam menyusun *FIT table* akan tetapi sudah mulai mendapat gambaran tentang bagaimana cara menyusun *FIT table*. Pendampingan *tester* dalam penyusunan *FIT table* diperlukan karena *user* tidak memiliki latar belakang teknis serta pemahaman yang berbeda terhadap proses yang terjadi. Seperti pada skema pengujian keempat *user* masih belum mampu menyusun *FIT table* meskipun sudah diberikan SRS, RTM dan juga *test case*.

4.2 Analisis Hasil Pengujian

Dari hasil 4 pengujian yang telah dilakukan dapat dianalisis bahwa SRS memiliki peranan yang penting dalam pengujian ini. SRS akan memberikan gambaran tentang apa saja hal yang kan dilakukan pengujian. Hal ini terbukti pada pengujian kedua ketika *user* diminta menyusun *FIT table* tanpa diberikan SRS, *user* tidak mengerti apa yang harus diuji. Akan tetapi dalam pemahaman SRS untuk menyusun suatu *FIT table, user* memerlukan RTM dan juga *test case*. Karena dalam RTM akan menggambarkan fungsionalitas yang terdapat pada program dan *test case* akan memberikan gambaran tentang hasil suatu fitur ketika prosedur dengan kriteria tertentu dilakukan. Pada pengujian ketiga *user* hanya diberikan SRS tanpa diberikan RTM dan *test case*, pada pengujian ini *user* hanya mengerti gambaran umum program dan belum memiliki gambaran hasil suatu prosedur. Pendampingan *tester* dalam penyusunan FIT *table* diperlukan agar *user* memahami RTM dan *test case.* Hal ini terbukti pada pengujian keempat ketika *user* diberikan SRS, RTM dan *test case* tanpa penjelasan dari *tester*, pada skema pengujian ini *user* masih kebingungan dalam menyusun *FIT table.*

Tabel 5. Tabel perbandingan skema pengujian

Skema	Diberikan penjelasan			Bantuan tester	Pembuatan FIT table
	SRS	RTM	*Test case*		
1	Ya	Ya	Ya	Ya	Berhasil
2	Tidak	Tidak	Tidak	Tidak	Tidak
3	Ya	Tidak	Tidak	Tidak	Tidak
4	Ya	Ya	Ya	Tidak	Tidak

Untuk skema pengujian yang pertama, dari *test case* yang telah dibuat oleh *tester, user* dengan dibantu oleh *tester* menyusun *FIT table*. *FIT table* ini yang kemudian diterjemahkan menjadi fixture oleh *tester*. Dari proses pengujian ini ditemukan bahwa pendampingan *user* dalam penyusunan *testcase* masih diperlukan dikarenakan *user* tidak memiliki latar belakang teknis. Hal ini ditunjukkan pada skema pengujian kedua, ketiga dan keempat ketika *user* diminta untuk menyusun FIT *table* tanpa pendampingan *tester* maka *user* tidak mampu menyusun FIT *table.* Pendampingan ini berfungsi untuk membantu *user* dalam proses pemahaman terhadap RTM dan *test case*. Sehingga dapat disimpulkan bahwa pemahaman *user* terhadap SRS didapatkan dari pemahaman terhadap RTM dan *test case* serta pendampingan tester yang membantu *user* dalam penyusunan FIT table, seperti yang terlihat dalam pengujian yang pertama.

Akan tetapi dalam penyusunan RTM dan *testcase*, *tester* haruslah memahami SRS sehingga *tester* dapat memetakan fungsional yang terdapat dalam SRS kedalam bentuk RTM dan *test case*. Dalam pengujian ini proses yang dilakukan haruslah sekuensial dikarenakan suatu proses akan mendukung pada proses selanjutnya. Seperti misalnya mengidentifikasi SRS untuk menemukan fungsionalitas apa saja yang terdapat dalam program. Setelah fungsional tersebut ditemukan maka akan digambarkan dalam bentuk RTM dan *test case* yang digunakan untuk membantu tester dalam penyusunan FIT table bersama user.

5. Kesimpulan

5.1 Kesimpulan

Dalam penelitian *user acceptance testing* menggunakan metode FIT pada aplikasi penjualan ini dapat disimpulakn bahwa:

1. *Software requirement specification* merupakan hal yang sangat penting dalam pengujian ini. Sehingga *software requirement* yang baik akan lebih memudahkan dalam proses perancangan metode pengujian.
2. Dalam proses pemahaman terhadap SRS *user* memerlukan RTM dan *test case* serta bantuan tester dalam penyusunan *FIT table*. Serta proses yang dilakukan haruslah sekuensial atau terurut mulai dari pengidentifikasian SRS, penyusunan RTM, penyusunan *test case* dan penyusunan fit table.

3. Pemahaman *user* terhadap SRS akan berpengaruh terhadap proses penyusunan skenario dalam FIT table.

5.2 Saran

Pada penelitian kali ini hanya menggunakan empat *user* sebagai super *user* atau admin dalam pengujian ini. Alangkah lebih baik pada proses penelitian berikutnya dapat menambahkan *user* lebih banyak di berbagai macam role atau fungsi sehingga penelitian dapat mencakup menu yang lebih banyak atau mendetail. Serta pada penelitian lebih lanjut dapat menambahakan hardware tambahan yang mendukung kinerja aplikasi sehingga pengujian lebih lanjut antara aplikasi dan hardware dapat dilakukan.

Daftar Pustaka

[1] Tutorialspoint 2014. *Software testing software system evaluation* available at: *tutorialspoint.com.* [Accessed at 3 January 2018]

[2] Bentley, John E.. 2005 *Software Testing Fundamentals-concept, Roles, and Terminology.* charlotte nc: wachovia bank.

[3] Hayes, Linda G.. 2004 *automated testing handbook.* software testing institute.

[4] Inc, Cunningham & Cunningham.. 2005 *framework for integration testing.* Available at: http://fit.c2.com/wiki.cgi?introductiontofit. [Accessed at 8 January 2018]

[5] ISTBQEXAMCERTIFICATION. 2015 *what is v model advantages disadvantages and when to use it*. Available at: http://istqbexamcertification.com/what-is-v-model-advantages-disadvantages-and-when-to-use-it/.[Accessed at 10 January 2018]

[6] Richard J watt, david leigh-fellows. 2004. "acceptance testdriven planning." UK:toughtworks.

[7] Hendrickson, Elisabeth. 2008 "Driving Development with Tests: ATDD and TDD." USA:quality tree software.

[8] Offut, paul ammann & jeff. 2008. *Introducing To Software Testing.* New York: Cambridge University Press.

[9] Roy W Miller, Christoper T Collins. "Acceptance testing". USA: rolemodel software inc.

[10] Series, robert c martin. 2005. *Fit for developing software.* USA: prentice hall.

[11] Infoq. 2009. *automated acceptance test.* Available at: http://www.infoq.com/news/2009/06/automated-acceptance-tests [Accessed at 3 January 2018]

[12] Zarrina muhibah. 2014. *software testing*. Available at: https://sweetzarina.wordpress.com/2014/10/23/software-testing/ [Accessed at 10 January 2018]

[13] Tutorialspoint. SDLC tutorials. Available at: *tutorialspoint.com*

[14] Marchetto alessandro. 2015. Acceptance testing with fitnesse.ppt. lecture presentation.

[15] British Columbia. 2008. user acceptance testing process.

[16] O'hara fran. 2013. Acceptance testing- what does it mean to you?. Available at: inspireqs.ie [Accessed at 13 January 2018]