MAKALAH

TEKNIK INFORMATIKA

“PENGGUNAAN METODE ACCEPTENCE TEST DRIVEN DEVELOPMENT PADA PROSES ACCEPTENCE TEST MENGGUNAKAN CODECEPTION”

Nama Anggota :

5200411146 Gabriel Sumampow

5200411527 Muhammad Daffa K R

5200411528 Noviyan Syamsuwardi

5200411547 Yoga Maulana Qhadafi

UNIVERSITAS TEKNOLOGI YOGYAKARTA

PROGRAM STUDI INFORMATIKA

2021

DAFTAR ISI

DAFTAR ISI **2**

BAB I **3**

PENDAHULUAN **3**

Latar Belakang 3

Rumusan Masalah 4

Tujuan Penelitian 4

BAB II **5**

PEMBAHASAN **5**

Pengertian ATDD 5

Langkah-langkah proses ATDD 5

Kelebihan dan Kekurangan ATDD 6

BAB III **7**

REVIEW JURNAL **7**

Hasil dan Pembahasan 7

kesimpulan : 10

BAB IV **12**

PENUTUP **12**

I.Perbandingan metode Acceptance Test Development Driven Dengan metode Waterfall, Prototype, RAD 12

BAB I

PENDAHULUAN

A. Latar Belakang

Pada zaman sekarang perkembangan teknologi perangkat lunak telah berkembang pesat dan menjadi pendukung utama bagi sebuah perusahaan. Suatu perusahaan atau lembaga yang menempatkan teknologi perangkat lunak menjadi salah satu pendukung dalam kemajuan perusahaan dapat mencapai rencana strategis organisasi. Perangkat lunak (software) sendiri merupakan program komputer yang terasosiasi dengan dokumentasi perangkat lunak seperti dokumentasi kebutuhan, model desain, dan cara penggunaannya (user manual). Perangkat lunak pada saat ini sudah menjadi kebutuhan khalayak umum di setiap usaha karena dengan memanfaatkan teknologi perangkat lunak akan membantu suatu perusahaan untuk memecahkan sebuah permasalahan yang terjadi.

Dalam pembangunan perangkat lunak sistem informasi terdapat proses yang kompleks, tidak hanya melibatkan proses coding tetapi juga proses testing menjadi salah satu kunci keberhasilan dalam mencapai tujuan system informasi tersebut.Proses testing sebagai salah satu cara untuk menjaga agar sistem informasi dikembangkan sesuai dengan tujuan awal dan tidak melebar ke permasalahan lain dan juga untuk memastikan alur bisnis berjalan sesuai dengan skenario yang sudah disepakati dengan klien sehingga testing menjadi sarana penting untuk menjaga kepercayaan klien.

Acceptance test adalah salah satu jenis pengujian pada perangkat lunak. Acceptance test dilakukan untuk menyesuaikan perangkat lunak yang dibangun sesuai dengan requirement yang ada atau kontrak yang telah disepakati dimana proses tersebut akan menentukan diterima atau tidaknya perangkat lunak yang sedang atau telah dibangun. Dalam acceptance test, tester memposisikan diri sebagai pengguna aplikasi. Perilaku pengguna diterjemahkan dalam bentuk skenario. Acceptance test. berkaitan dengan antarmuka (interface) aplikasi yang terdiri dari form-form inputan maupun tombol-tombol. Pada proses pengembangan perangkat lunak, acceptance test tidak hanya dilakukan satu kali melainkan berkali-kali jika program banyak mengalami perubahan. Salah satu metode pengembangan perangkat lunak yang terbilang baru dan menjadi sorotan selama beberapa tahun terakhir ini adalah Acceptance Test Driven Development (ATDD) (Kamalrudin, Sidek, Moketar, & Robinson, 2013). ATDD merupakan bagian dari metode agile software development, secara garis besar ATDD melibatkan proses acceptance testing sebelum implementasi kode program yang dapat meningkatkan produktivitas waktu pengembangan perangkat lunak serta kesesuaian requirement dari perangkat lunak yang dibangun (Pugh,2010). Biasanya, pengembang aplikasi melakukan proses acceptance testing pada metode ATDD secara manual yaitu menggunakan interaksi manusia dan ini menguras waktu dalam pengembangan

aplikasi tersebut. Proses pengujian yang manual ini dapat diotomatisasi dengan menggunakan tools Codeception.

Codeception merupakan automated framework testing yang menyederhanakanpenulisan test script yang dapat mempercepat dan meningkatkan keefektifan pada proses acceptance testing. Sebagai contoh, terdapat beberapa penelitian yang membahas mengenai implementasi metode TDD dalam pembuatan aplikasi. Salah satu penelitian tersebut adalah penelitian yang dilakukan oleh [7] yang melakukan implementasi TDD untuk pengembangan sisten informasi rawat jalan. Kemudian implementasi TDD juga dilakukan dalam pembuatan perangkat lunak pengajuan tugas akhir [8]. Hasil yang diperoleh menunjukan bahwa dengan menerapakan TDD pengecekan error lebih mudah dilakukan. Selain itu pengembang dapat bekerja lebih cepat karena simulasi manual untuk pengujian fungsi perangkat lunak menjadi berkurang

B. Rumusan Masalah

1. Apa itu Model Acceptence Test Driven Development?
2. Bagaimana langkah-langkah pengerjaan ATDD?
3. Apa kelebihan dan kekurangan ATDD?

C. Tujuan Penelitian

1. Mengetahui apa itu metode ATDD.
2. Mengetahui langkah-langkah pengerjaan ATDD.
3. Mengetahui kelebihan dan kekurangan ATDD.
4. Mengetahui tujuan model ATDD.
5. Mengetahui penyelesaian pada suatu masalah dengan model ATDD.

BAB II

PEMBAHASAN

A. Pengertian ATDD

Acceptence test merupakan salah satu jenis pengujian pada pengembangan perangkat lunak, *acceptance test* tidak hanya dilakukan satu kali melainkan berkalli-kali jika program banyak mengalami perubahan, dalam *accpetence test*, tester memposisikan diri sebagai pengguna aplikasi. Perilaku pengguna dibentukan dalam sebuah scenario ujicoba, dengan mengadpatasi metode *Acceptence Test Driven Development* (ATDD), scenario pengujian dan pembuatan script pengujian dilakukan di awal pengembangan perangkat lunak.

Pengujian dilakukan untuk menguji aktivitas tambah, ubah, dan hapus pada beberapa form sistem atau aplikasi yang sudah ada. Penelitian ini membuat sebuah proses *Automated Acceptence Testing* yang biasanya dilakukan secara manual. Pembuatan proses *Automated Acceptence Test* menggunakan *tools codeception*. Hasil dari pengujian semua scenario tersebut adalah Passed (Lolos). Namun hasil pengujian sangat bergantung dengan perangkat keras yang digunakan untuk pengujian.

B. Langkah-langkah proses ATDD

1. Tuliskan Acceptance Test.
   Pastikan tes dibuat berdasarkan use case yang diterima dari klien, dan tuliskan semua kemungkinan yang dapat diterima untuk input dan output-nya agar semua
   harapan klien terpenuhi sesuai standar.
2. Pastikan Acceptance Test Gagal.
   Karena belum ada kode apapun yang membuat tes tersebut lolos (passed).
3. Tuliskan Unit Test.
   Penting untuk menulis unit test sebelum kode fungsional. Ini membantu memastikan bahwa setiap baris kode dapat diuji.
4. Pastikan Unit Test Gagal.
   Sama seperti acceptance test, karena kode fungsional belum siap, unit test harus gagal.
5. Tuliskan Kode Fungsional.
   Terapkan tujuan dari kode fungsional dan pastikan bahwa unit test dan acceptance test lolos (passed).
6. Refactor Code.
   Lakukan perubahan kode jika merasa kode yang sudah ditulis tidak rapi. Selama unit test tidak ada yang gagal berarti tidakmasalah terhadap kode yang di-refactor tersebut.

C. Kelebihan dan Kekurangan ATDD

Kelebihan dari model ATDD :

1. Pengembanganperangkat lunak serta kesesuaian requirement dari perangkat lunak yang dibangun.

Kekurangan dari model ATDD :

1. Testing pada metode ATDD secara manual yaitu menggunakan interaksi manusia dan ini menguras waktu dalam pengembangan aplikasi tersebut.

BAB III

REVIEW JURNAL

A. Hasil dan Pembahasan

Pada bagian ini akan dipaparkan hasil dari penelitian jurnal yang kami pilih. Hasil yang akan ditampilkan adalah bagaimana langkah-langkah pembuatan script test dan hasil pengujian dari script test tersebut.

1. Pembuatan skenario pengujian Dalam penelitian ini akan ditampilkan contoh skenario login dengan data valid dan tidak valid

Tabel 1. Skenario Login dengan Data Valid

skanario	langkah tes	jenis inputan	hasil yang di harapkan
Login dengan data valid	1. ke login page		masuk ke home
	2. masukan username = tu@sma.com	textfield	
	3. masukan password =1234	textfield	
	4. pilih tahun ajaran = 2014/2015 Genap	combobox	
	5. klik sign in/enter	button	

Tabel 2. Skenario Login dengan Data Tidak Valid

skanario	langkah tes	jenis inputan	hasil yang dsi harapkan

Login TUdengan salah satu data tidak valid	1. Ke login page		munculkan pesanan error 'Kombinasi username/password salah. Silahkan coba lagi.
	2. Masukkan username = tu@sma.com	textfield	
	3. Masukkn password = assdfghj	textfield	
	4. Pilih tahun ajaran = 2015/2016 Genap	combobox	
	5. Klik Sign In / Tekan Enter	button	

2. implementasi Codeception

a. instalasi Codeception
   Install dependency Codeception yang paling stabil via Composer, caranya yaitu buka command prompt, lalu masuk ke direktori dimana file acceptance test berada, ketikkan perintah berikut :
   ```
   composer require
   "codeception/codeception" --dev
   ```

b. Membuat Bootstrap File Eksekusi perintah untuk membuat bootstrap-nya, ketika perintah ini dijalankan maka codeception akan meng-generate file-file yang dibutuhkan untuk proses testing. Hal ini akan membuat file codeception.yml dan direktori tests dan default tes.
   ```
   “codecept bootstrap”
   ```

c. Konfigurasi Web Driver

Pastikan bahwa aplikasi yang akan diuji berjalan dengan baik. Buka file tests/acceptance.suite.yml, lalu lakukan beberapa konfigurasi seperti URL Aplikasi yang akan diuji dan WebDriver. Webdriver yang digunakan untuk penelitian ini adalah chromedriver yang dapat mengeksekusi test script pada real browser google chrome layaknya pengujian secara manual, namun google chrome akan dikontrol oleh sebuah servis.

```
"actor: AcceptanceTester
modules:
enabled:
- WebDriver:
url:
http://localhost/eschoolface-
lift/publicwindow_size: false
port: 9515
browser: chrome"
```

d. Membuat File Accpetence Test

```
"codecept generate:cest acceptance
LoginValid"
"codecept generate:cest acceptance
LoginTidakValid"
```

e. Membuat Script Pengujian Pengujian Login Valid

```
"class LoginValidCest
{
public function
_before(AcceptanceTester $I){
}
// tests
public function
tryToTest(AcceptanceTester $I)
{
$I->wantTo('perform actions and
see result');
$I->amOnPage('/');
$I-
>fillField('email','tu@sma.com');
$I->wait(2);
$I->fillField('password','1234');
$I>selectOption('semester_id','201
4/2015 Ganjil');
$I>click('//select[@name="semester
_id"]');
$I>click('//select[@name="semester
_id"]/option[@value="20152"]');
$I->wait(2);
$I->click('SIGN IN');
$I->see('dashboard');
$I->wait(3);"
```

f. Membuat Script Pengujian Login Tidak Valid

```
"Public function
```

```
tryToTest(AcceptanceTester $I)
{
$I->wantTo('perform actions and
see result');
$I->amOnPage('/');
$I->fillField('email','tu');
$I->wait(2);
$I>fillField('password','assdfghjk
uheghb');
$I->click('SIGN IN');
$I->see('Kombinasi
username/password salah. Silahkan
coba lagi’);
$I->wait(2);
}”
```

g. Menjalakan Pengujian

Dalam penelitian ini hanya akan ditampilkan skenario tes login valid dan tidak valid dan hasil pengujian skenario tes login valid dan tidak valid karena keterbatasan aturan pembuatan dokumen jurnal penelitian, hal ini dikarenakan pengujian ini menggunakan chromedriver, maka sebelum menjalankan pengujian pastikan bahwa servis chromedriver telah jalan. Untuk menjalankan servis chromedriver dapat dilakukan dengan cara seperti berikut :

```
Chromedriver –url-base=”\wd\hub”
```

Untuk menjalankan pengujian login valid, masukkan perintah berikut

```
“codecept run --steps
LoginValidCest.php”
```

Untuk menjalankan pengujian login tidak valid, masukkan perintah berikut.

```
“codecept run --steps
LoginTidakValidCest.php”
```

Dari hasil pengujian login dengan data valid sesuai dengan Tabel 2 Skenario Login dengan Data Tidak Valid, terlihat bahwa antara hasil yang diharapkan dengan hasil pengujian sesuai ekspektasi yaitu berhasil (passed).

kesimpulan :

Dalam penelitian Penggunaan Metode Acceptance Test Driven Development Pada Proses Acceptance Test Menggunakan Codeception, dapat ditarik beberapa kesimpulan sebagai berikut :

1. Penggunaan codedeption pada proses acceptance testing terbilang lebih cepat dari segi waktu dan lebih tinggi tingkat keefektifan setiap skenario tes. Bagaimanapun juga penggunaan automated framework test memerlukan usaha yang lebih dibanding pengujian secara manual. Namun jika developer dan tester sudah terlatih dan terbiasa menggunakan codeception maka dampak positif penggunaan automated framework test akan lebih terasa.

2. Hasil pengujian script test dengan codeception sangat bergantung dengan perangkat keras yang digunakan ketika pengujian. Dikarenakan proses tunggu (wait) tidak berefek sama pada semua perangkat.
3. Dibalik hal positif dari penggunaan automated framework testing yang sudah disebutkan, automated framework testing juga memicu munculnya defect tersendiri, yaitu defect yang disebabkan penulisan script test. Penyebab munculnya kesalahan pada script test disebabkan oleh human error, kesalahan dalam menerjemahkan skenario tes kedalam format script test, maupun ketidaksesuaian script test dengan sumber kode yang telah ditulis.
4. Menggabungkan manual acceptance test dan automated acceptance test merupakan salah satu upaya mengoptimalkan proses acceptance testing pada ATDD.

BAB IV

PENUTUP

I.Perbandingan metode Acceptance Test Development Driven Dengan metode Waterfall, Prototype, RAD

1. Acceptence Test Development Driven
   Acceptance test dilakukan untuk menyesuaikan perangkat lunak yang dibangun sesuai dengan requirement yang ada atau kontrak yang telah disepakati dimana proses tersebut akan menentukan diterima atau tidaknya perangkat lunak yang sedang atau telah dibangun. Dalam acceptance test, tester memposisikan diri sebagai pengguna aplikasi.
2. Model Waterfall
   Model pengembangan Waterfall cocok digunakan untuk sistem atau perangkat lunak yang bersifat generik, artinya sistem dapat diidentifikasi semua kebutuhannya dari awal dengan spesifikasi yang umum serta sesuai untuk perangkat lunak yang memiliki tujuan untuk membangun sebuah sistem dari awal yang mengumpulkan kebutuhan sistem yang akan dibangun sesuai dengan topik penelitian yang dipilih sampai dengan produk tersebut diuji.
3. Model Prototype
   Model pengembangan Prototype lebih cocok untuk sistem atau perangkat lunakyang bersifat customize, artinya software yang diciptakan berdasarkan permintaan dan kebutuhan (bahkan situasi atau kondisi) tertentu dan sesuai untuk perangkat lunak memiliki tujuan untuk mengimplementasikan sebuah metode atau algoritma tertentu pada suatu kasus.
4. Model RAD
   Model pengembangan RAD lebih cocok untuk sistem atau perangkat lunak yang bersifat customize, berskala besar dan memerlukan waktu yang lebih singkat artinya software yang diciptakan berdasarkan permintaan dan kebutuhan (bahkan situasi atau kondisi) tertentu dan sesuai untuk perangkat lunak memiliki tujuan untuk mengimplementasikan sebuah metode atau algoritma tertentu pada suatu kasus, serta memiliki kemungkinan untuk kebutuhan pengembangan kembali dalam jangka waktu yang cukup panjang.