

BAB IV TEST CASES

A. What is Test Case?

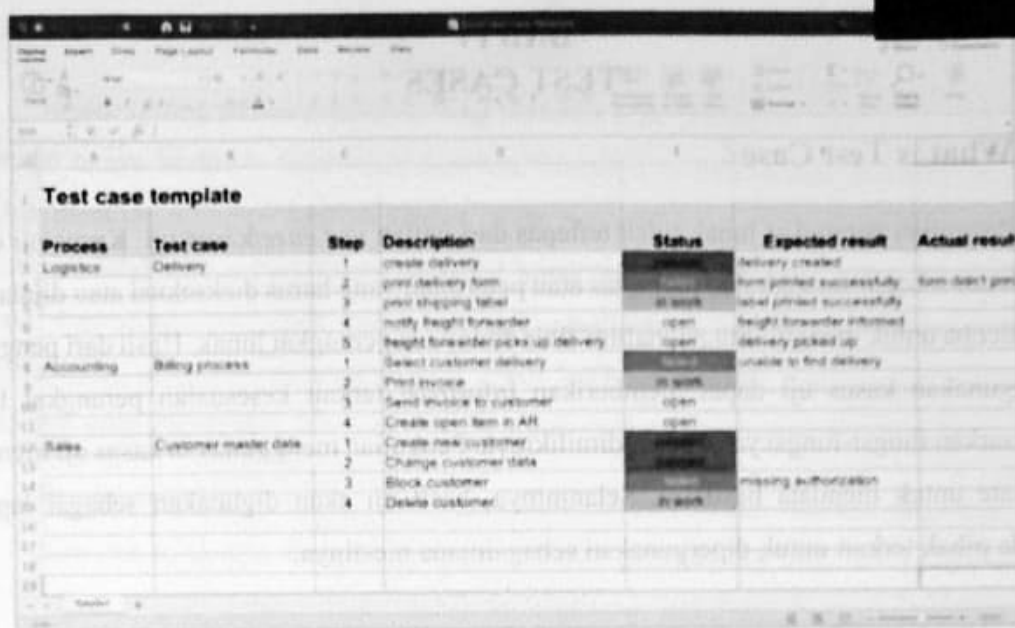
Pengujian perangkat lunak tidak terlepas dari istilah *test case*/kasus uji. Kasus uji dapat dijelaskan sebagai serangkaian daftar tes atau pengujian yang harus dieksekusi atau dijalankan atau dicoba untuk menguji fungsionalitas fitur-fitur pada perangkat lunak. Hasil dari pengujian menggunakan kasus uji dapat memberikan informasi terkait kesesuaian perangkat lunak berdasarkan fungsi-fungsi yang harus dimilikinya. Pengujian menggunakan kasus uji memiliki template untuk mendata hasil uji. Selanjutnya, hasil uji akan digunakan sebagai laporan kepada pihak terkait untuk dipergunakan sebagaimana mestinya.

B. Why Test Cases is Important?

Pengujian menggunakan kasus uji penting untuk dilakukan, banyak dampak positif yang dapat dihasilkan dari pengujian ini. Pengujian menggunakan kasus uji dilakukan pada semua fungsi perangkat lunak, sehingga dapat membantu memperbaiki kelemahan/kekurangan pada perangkat lunak agar lebih baik walaupun tidak akan sempurna. Pengujian menggunakan kasus uji juga dapat membantu menemukan *bug* baru yang mungkin terlewatkan pada pengujian sebelumnya. Pengujian menggunakan kasus uji dapat memastikan kualitas dari perangkat lunak yang diuji, semakin sedikit *bug* yang ditemukan maka semakin baik perangkat lunak tersebut.

C. How to Write Test Cases?

Isi template laporan hasil uji berupa kolom modul proses yang akan diuji, kasus uji, langkah-langkah proses terkait kasus uji, deskripsi atau penjelasan yang lebih merinci terkait kasus uji (hal-hal yang akan diuji), status pengujian, hasil uji yang diharapkan dan kolom hasil uji yang didapatkan. Ketika status pengujian deskripsi kasus uji gagal (*failed*), hasil yang gagal tersebut tetap harus dimasukkan ke laporan hasil uji dengan tambahan penjelasan terkait hasil uji yang didapatkan. Hal ini nantinya akan digunakan oleh tim pengembang untuk memperbaiki *bug* tersebut sehingga mendapatkan hasil uji sesuai dengan yang diharapkan.



Test case template						
Process	Test case	Step	Description	Status	Expected result	Actual result
Logistics	Delivery	1	create delivery	passed	delivery created	
		2	print delivery form	open	form printed successfully	form didn't print
		3	print shipping label	in work	label printed successfully	
		4	notify freight forwarder	open	freight forwarder informed	
		5	freight forwarder picks up delivery	open	delivery picked up	
Accounting	Billing process	1	Select customer delivery	passed	unable to find delivery	
		2	Print invoice	in work		
		3	Send invoice to customer	open		
		4	Create open item in AR	open		
Sales	Customer master data	1	Create new customer	passed		
		2	Change customer data	passed		
		3	Block customer	passed	missing authorization	
		4	Delete customer	in work		

Gambar 4.1 Test Case Template

Cara penentuan/penulisan kasus uji yang baik didasarkan pada beberapa hal, yaitu persyaratan awal (*preconditions*), langkah-langkah proses (*steps*), dan hasil yang diharapkan (*expected results*). Persyaratan awal digunakan untuk mendeskripsikan kesepakatan antara pihak pengembang dengan pelanggan terkait spesifikasi perangkat lunak, contohnya interaksi pengguna pada halaman *login*. Langkah-langkah proses digunakan untuk mendeskripsikan lebih rinci terkait interaksi yang harus dilakukan oleh pengguna dengan perangkat lunak, contohnya pengguna dapat melakukan *login* dengan data-data yang benar/valid. Hasil yang diharapkan digunakan untuk mengetahui apa yang akan terjadi kemudian setelah langkah sebelumnya selesai dilakukan, contohnya pengguna akan mendapatkan *pop-up*/pesan penyambutan setelah berhasil *login*.

Salah satu perangkat lunak yang populer digunakan untuk pengujian perangkat lunak adalah Zephyr. Zephyr menawarkan berbagai fitur yang mendukung pengujian yang lebih baik. Namun, masih banyak perangkat lunak lain yang juga dapat digunakan untuk pengujian perangkat lunak.

D. Smoke Testing

Smoke testing adalah pengujian yang berfokus pada perangkat lunak secara keseluruhan, namun hanya berfokus pada *main function*, biasanya mengutamakan *positive case*, namun tidak menutup kemungkinan dilakukan pada *negative case* yang sifatnya *critical*. *Smoke Testing* adalah proses pengujian perangkat lunak yang menentukan apakah perangkat lunak yang digunakan stabil atau tidak. *Smoke Testing* adalah konfirmasi bagi tim QA untuk melanjutkan pengujian perangkat lunak lebih lanjut. Ini terdiri dari serangkaian pengujian minimal yang dijalankan pada setiap build untuk menguji fungsionalitas perangkat lunak.

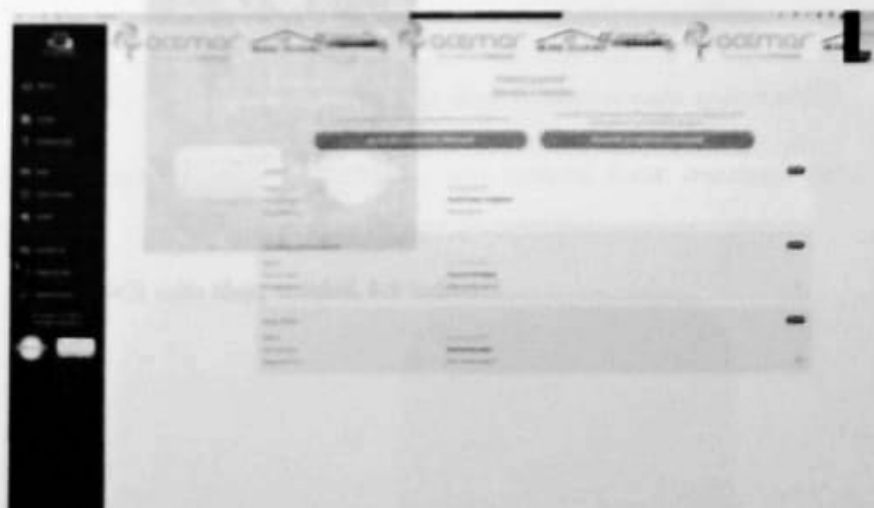
Secara sederhana, *Smoke Testing* berarti memverifikasi bahwa fitur-fitur penting berfungsi dan tidak ada *showstoppers* dalam *build* yang sedang diuji. Ini adalah tes regresi mini dan cepat dari fungsionalitas utama. Ini adalah tes sederhana yang menunjukkan produk siap untuk pengujian. Ini membantu menentukan apakah *build* tersebut cacat sehingga pengujian lebih lanjut akan membuang-buang waktu dan sumber daya.

E. How to Write Smoke Suite?

Kwidos ~~ini~~ adalah platform yang menghubungkan kontraktor dan konsumen. Berikut penjelasan tentang pengujian *smoke suite* yang dilakukan pada aplikasi Kwidos.

1. Halaman *Homepage / Layout*

1.1 *Homepage* merupakan halaman beranda yang pasti dijumpai oleh pengunjung pada situs Kwidos. Tampilan halaman beranda pada situs Kwidos sangatlah umum dan tidak ada yang istimewa.



Gambar 4.2 Halaman beranda situs Kwidos

1.2 Hal terpenting pada halaman beranda pada situs Kwidos adalah ^{publikasi iklan} banner/spanduk yang harus ditampilkan.



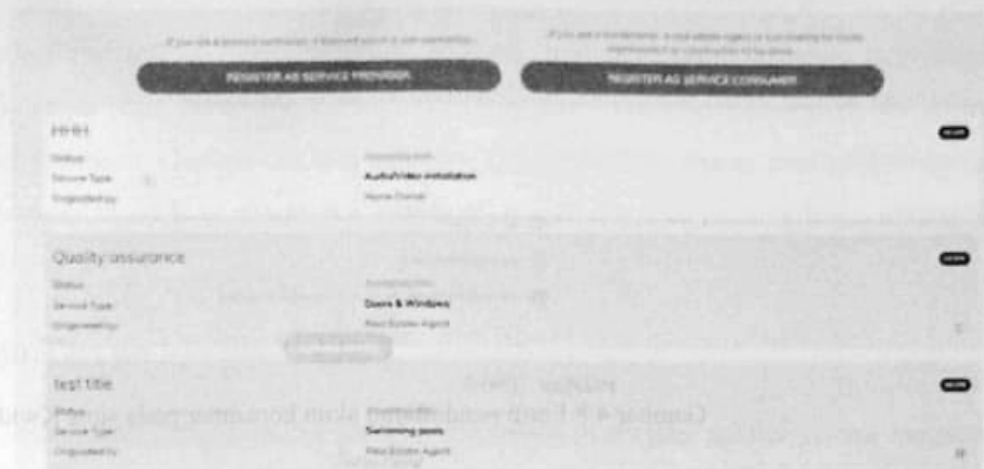
^{publikasi iklan}
Gambar 4.3 Banner pada situs Kwidos

1.3 Hal berikutnya yaitu *sidebar* di sebelah kiri layar. Panel ini harus ditampilkan karena berisikan berbagai macam navigasi seperti, *sign up*, *offers* dll



Gambar 4.4 *Sidebar* pada situs Kwidos

1.4 Kemudian blok pendaftaran akun yang terletak ditengah, daftar pekerjaan atau daftar proyek



Gambar 4.5 Tampilan pendaftaran akun pada situs Kwidos

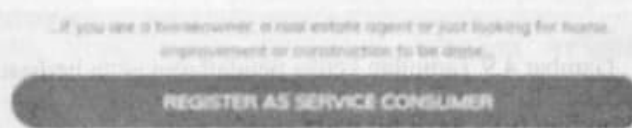
1.5 Selanjutnya pengisian hasil uji pada ^{halaman smoke} smoke suite.



Gambar 4.6 Pengisian hasil pengujian

2. Create a consumer account

2.1 Pengujian yang akan dilakukan adalah pembuatan akun sebagai ^{halaman smoke} Konsumer



Gambar 4.7 Tampilan pendaftaran akun konsumen pada situs Kwidos

2.2 Setelah tombol registrasinya di-klik maka muncul *form* registrasi yang harus diisikan.

Register as consumer

Name *

Last name *

Phone *

Email address *

Password *

☐ Link the Facebook

☐ I am a new estate agent

☒ I have read and agree to the Terms of Use and Privacy Policy

Create Account

mubtikan itahiz
Gambar 4.8 Form pendaftaran akun konsumen pada situs Kwidos

2.3 Ketika proses pendaftaran akun *berhasil* dilakukan, akan muncul pesan bahwa pembuatan akun telah sukses seperti gambar 4.9



Gambar 4.9 Tampilan ketika pendaftaran akun berhasil dilakukan pada Kwidos

2.4 Kemudian tulis hasil uji pada *smoke suite*

Summary	Preconditions	Steps	Test data	Expected result
Home Page Login	At least one user is logged in to the system	Click on the user name	Click on the user name	Redirect to the user profile page
Click on the user name	At least one user is logged in to the system	Click on the user name	Click on the user name	Redirect to the user profile page

Gambar 4.10 Pengisian hasil pengujian

BAB V

CROSS-PLATFORM AND CROSS-BROWSER TESTING

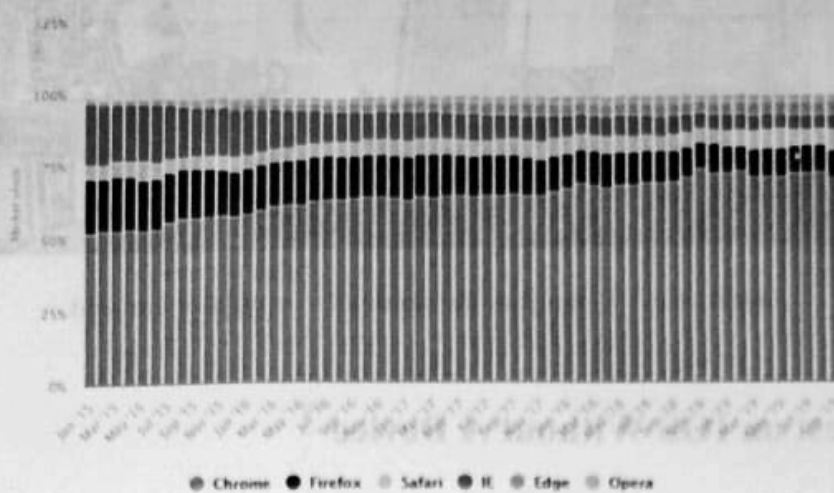
Cross-platform testing mengharuskan aplikasi berjalan sama baiknya (sebaiknya sempurna) di semua jenis komputer, perangkat seluler dan sistem operasi seperti Windows, OS, Android, Macintosh, Chrome OS, dan lain-lain. *Cross-platform testing* menentukan semua kemungkinan masalah *caching* dan perilaku aplikasi seluler atau web pada berbagai perangkat yang berbeda.

A. How to Test on Mobile Devices?

Untuk melakukan pengujian dengan menggunakan perangkat seluler secara langsung tentunya dibutuhkan sebuah perangkat seluler (*smartphone*). Namun, jika pengujian secara langsung ingin dilakukan pada perangkat partikular akan sedikit menyusahkan karena keterbatasan perangkat yang dimiliki. Solusi dari permasalahan tersebut adalah menggunakan ~~simulator~~ ^{tulisan italic} atau emulator pengguna, salah satunya melalui *browser* Chrome. Selain emulator, ^{tulisan italic} juga terdapat layanan *cloud* khusus, pengguna bisa membuat akun di sana, dan menggunakan komputer dasar melalui *browser*, pengguna dapat menavigasi ke situs web tersebut dan hanya memilih perangkat yang dibutuhkan seperti sistem operasi atau memilih versi.

B. How to Use Chrome Mobile View?

Tampilan *website mobile* ditujukan untuk pengguna dengan perangkat *mobile* (*smartphone* & tablet), sementara *website desktop* adalah untuk perangkat PC (komputer ^{& laptop, laptop = tulisan italic} *desktop* & laptop). Namun, bisakah kita mengakses tampilan *website* versi *mobile* menggunakan perangkat PC? Tentu saja bisa. Google Chrome menyediakan sarana untuk membuka *website mobile* di komputer tanpa harus menginstal ekstensi tambahan. Berikut cara menggunakannya:



Gambar 5.7 Persentase penggunaan berbagai browser

D. How to Test Android Apps Without A Device ?

Untuk mendapatkan hasil uji yang menyerupai perangkat yang sebenarnya tanpa memiliki perangkat tersebut, dapat dilakukan dengan menginstal perangkat lunak khusus di komputer atau biasa disebut dengan *emulator*. Misalnya penggunaan Google untuk *emulator* android. Ada sekitar 15 *emulator* terbaik ditahun 2022, salah satunya adalah Android Studio yang dapat dijadikan pilihan pertama atau direkomendasikan.

Android studio sering digunakan oleh pengembang aplikasi dalam mengembangkan suatu aplikasi *mobile* untuk android. Terdapat banyak fitur dan banyak hal yang bisa dilakukan dalam Android Studio, sehingga akan sedikit rumit untuk digunakan oleh pemula. Android Studio memiliki *emulator* yang sudah terpasang dalam Android Studio. Selain Android Studio, terdapat juga aplikasi Jenny Motion yang juga dapat digunakan untuk mengembangkan aplikasi *mobile* untuk android. Dengan menggunakan *emulator* android, pengembang dapat melihat tampilan perangkat android seperti aslinya. Begitu pula halnya untuk tim penguji, tim penguji juga dapat memanfaatkan *emulator* untuk melakukan pengujian pada suatu perangkat lunak yang telah selesai dikembangkan.



Gambar 5.8 Pengujian menggunakan ^{tulisan italic}emulator android

E. How to Test on IOS Without A Device ?

Emulator yang paling populer untuk pengujian perangkat lunak berbasis IOS adalah X ^{tulisan italic}code. Aplikasi ini terbilang cukup berat dan memiliki banyak fitur yang sangat berguna untuk pengujian dan juga pengembangan perangkat lunak. Akan tetapi, aplikasi ini berbayar sehingga diharuskan untuk berlangganan agar bisa menggunakannya.

F. How to Test Using Cloud Services ?

Layanan *cloud* terkenal sangat mahal namun setimpal dengan berbagai fitur menarik yang ditawarkannya. Salah satu hal menarik dari layanan ini adalah pengguna layanan ini tidak hanya dapat menggunakan *emulator* tetapi pengguna juga dapat menggunakan perangkat nyata yang terhubung melalui internet. Jadi seperti menggunakan perangkat nyata yang sebenarnya terletak disuatu tempat yang lokasi dari perangkat itu sendiri dapat diketahui oleh penggunanya. Itu tentunya jauh lebih mahal dan bisa saja itu berkisar antara \$120 per bulan atau mungkin lebih.

G. What is Cross-Browser Testing ?

Pengujian *cross-browser* secara lebih sempit mempunyai target pada fungsi kualitatif aplikasi/situs web untuk memberikan pengalaman yang luar biasa pada pengguna. Sebagian besar cara kerjanya di sini terkait dengan pengembang *front-end* yang harus memberikan solusi logis dan tampilan yang sempurna.