

## **BUG (LANJUTAN)**

Pengelompokan bug pada proses pengujian kualitas perangkat lunak merupakan proses pencarian sejumlah bug pada perangkat lunak kemudian memilahnya berdasarkan tingkatan dampak atau resiko kegagalan perangkat lunak. Pada bug yang memiliki tingkat resiko kegagalan yang besar akan lebih diprioritaskan perbaikannya daripada bug yang tingkat resiko kegagalannya kecil. Hal ini bertujuan untuk mempermudah penyelesaian/perbaikan bug pada perangkat lunak. Namun, hal ini tidak menjamin perbaikan secara total pada perangkat lunak karena pada dasarnya bug pada perangkat lunak tidak mungkin dapat diselesaikan secara total.

Bug dengan resiko kegagalan kecil yang tidak mengganggu atau merusak alur proses perangkat lunak dapat dikesampingkan, karena kecacatan kecil pada perangkat lunak pasti akan selalu ada. Pernyataan tersebut dapat dianalogikan seperti goresan pada mobil. Goresan pada mobil tidak mempengaruhi fungsi dari mobil itu sendiri, mobil tetap dapat digunakan sebagaimana fungsinya walaupun pada mobil terdapat banyak goresan yang mungkin merupakan hasil dari ketidaksengajaan pengemudi saat mengemudikan mobil tersebut.

Terdapat dua istilah umum pada proses pengujian perangkat lunak yaitu severity dan priority. Severity dapat didefinisikan sebagai tingkatan resiko dalam proses pengembangan perangkat lunak. Sedangkan, priority dapat didefinisikan rentetan bug pada perangkat lunak yang harus diselesaikan. Kedua istilah tersebut saling melengkapi satu sama lainnya, maksudnya adalah dengan dilakukan pengelompokan bug berdasarkan tingkat resiko maka dapat ditentukan bug mana yang mendapatkan prioritas lebih besar sehingga proses perbaikan pada bug dapat lebih terorganisir dan tepat.

Priority pada pengujian perangkat lunak terbagi menjadi tiga kelompok, yaitu low, medium dan high priority. Bug yang berdampak besar terhadap alur proses perangkat lunak bahkan bisa merusak alur proses perangkat lunak adalah bug yang sangat diprioritaskan perbaikannya atau bisa disebut sebagai bug dengan prioritas tinggi (high priority). Setiap bug dengan prioritas tinggi harus dilakukan perbaikan/penyelesaian secepat mungkin agar proses pengembangan dapat dilanjutkan dan tidak terhambat. Sedangkan bug dengan prioritas sedang (medium priority) yang bisa dibilang sebagai bug yang tidak terlalu mengganggu alur proses tidak harus langsung dilakukan perbaikan, bug dengan prioritas sedang dapat menunggu giliran perbaikannya setelah bug dengan prioritas tinggi terselesaikan terlebih dahulu. Begitupun halnya dengan bug prioritas rendah (low priority), bug dengan prioritas rendah merupakan kesalahan-kesalahan kecil yang tidak berdampak pada alur proses sama sekali. Bahkan bug prioritas rendah sering kali dibiarkan begitu saja tanpa adanya perbaikan.

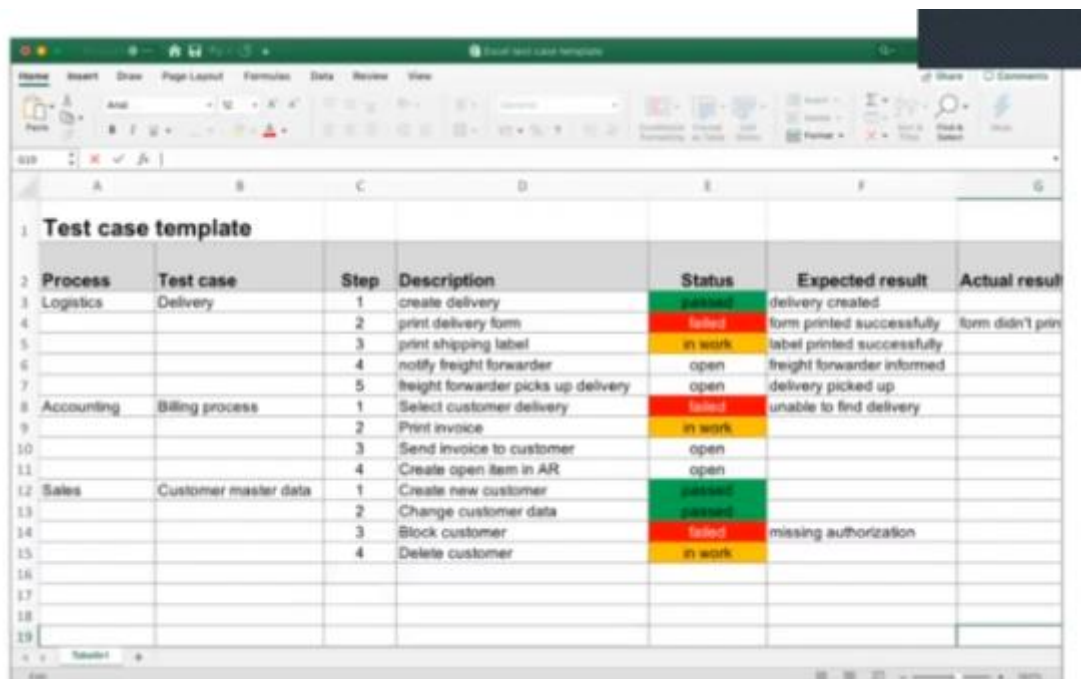
Proses perbaikan bug dimulai dari bug prioritas tinggi kemudian dilanjutkan dengan bug prioritas sedang yang diproses perbaikannya setelah semua bug prioritas tinggi diselesaikan dan diakhiri dengan perbaikan pada bug prioritas rendah yang diproses ketika semua bug dengan prioritas tinggi dan sedang selesai diperbaiki. Perbaikan pada bug prioritas rendah tidak begitu disarankan karena hampir tidak berpengaruh pada perangkat lunak. Kesalahan kecil dalam pengkodean proses perangkat lunak akan berdampak besar pada jalannya proses tersebut bahkan bisa merusak alur proses perangkat lunak, sehingga perbaikan pada kesalahan tersebut harus lebih diprioritaskan walaupun hanya kesalahan kecil. Berikut contoh bug berdasarkan prioritas perbaikannya:

- a. Bug prioritas tinggi : kegagalan salah satu fitur penting pada perangkat lunak
- b. Bug prioritas sedang : kegagalan pada fitur yang jarang digunakan pada perangkat lunak
- c. Bug prioritas rendah : kesalahan jenis huruf pada keterangan gambar pada perangkat lunak

## 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.

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.



Process	Test case	Step	Description	Status	Expected result	Actual result
Logistics	Delivery	1	create delivery	passed	delivery created	
		2	print delivery form	failed	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	failed	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	failed	missing authorization	
		4	Delete customer	in work		

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.

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.

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.