Day 4

Bug là gì

Đầu tiên ta cần hiểu Bug là gì? Theo wikipedia định nghĩa: "Bug là những error, flaw, failure, hay fault tạo ra một kết quả sai, hoặc không lường đến được."

Đơn giản có thể coi nó như một thứ gì đó không hoạt động đúng theo thiết kế. Tuy nhiên còn nhiều thứ phát sinh không được mô tả trong thiết kế và requiment của khách hàng, vậy nó có phải là Bug?

Bug có đáng sợ?

Trong thực tế, nếu khi bạn đang trong quá trình kiểm thử trên môi trường test hoặc đang chạy thử trên các môi trường dev mà phát hiện ra bug. Đấy là 1 điều cực vui và may mắn. Vì ở đây các bạn có thể hiểu được tại sao lại có bug và các bước để tái hiện lại bug đó cho lập trình viên để có thể fix chúng. Trường hợp này thường xảy ra khi các bạn thực hiện testcase hoặc test free trên các màn hình hoặc các chức năng nhất định không theo 1 rule nhất định nào.

Trên môi trường của khách hàng hoặc trong bản release cho khách hàng. Nó như một quả bom sẽ phát nổ và có sức công phá khủng khiếp. Thường các công ty sẽ phạt rất nặng khi còn bug để khách hàng phát hiện ra trên phần mềm hoặc ứng dụng của họ. Nó làm ảnh hưởng trực tiếp đến quá trình sử dụng phần mềm, ảnh hưởng đến dữ liệu hoặc nghiêm trọng hơn có thể gây lỗi toàn hệ thống.

Các loại bug thường gặp.

**Block (Lỗi nghiêm trọng)**

Lỗi này ngăn không cho người dùng làm điều anh ta muốn mà bị chặn lại, không thể đi xa hơn. 99% lỗi này thuộc về lỗi chức năng (sẽ được nhắc tới sau).

Ví dụ:

* link hỏng
* trang không tải được
* không đổi được mật khẩu
* không nộp được biểu mẫu
* thông tin hợp lệ nhưng không lưu được khi điền biểu mẫu
* đơn không được xác nhận khi người dùng trở về trang chủ
* ứng dụng bị crash khi làm gì đó
* trang/ứng dụng đơ khi làm gì đó
* mật khẩu mới không được nhận
* thanh toán đơn hàng bị từ chối
* không thêm mục vào giỏ hàng được

**Major (Lỗi lớn)**

Lỗi này khiến người dùng rất khó chịu nhưng không đến mức ngăn anh ta thực hiện hành động nào đó.

Ví dụ:

* hiển thị mã HTML trên trang
* pop-up thông tin không chứa văn bản
* trang phải mất 10 giây mới tải được
* quảng cáo che nội dung trang nhưng tắt được
* ảnh sản phẩm bị kéo méo mó
* trang hiển thị dưới dạng HTML
* mất âm thanh (khi xem video)
* không chia sẻ trên mạng xã hội được
* trường không được đánh dấu là bắt buộc nhưng thực tế là có
* trường văn bản hiển thị 2 lần
* một phần trang web hiển thị ngôn ngữ khác

**Minor (Lỗi nhỏ)**

Lỗi này không có ảnh hưởng nhiều tới người dùng khi duyệt web hay mở ứng dụng, nhưng vẫn cần phải xem xét. Người quản lý sản phẩm sẽ quyết định có ưu tiên chúng hay không. Việc sửa lỗi thường tạo ra ROI thấp (Returen on Investment - tức là lợi ích có được trên 1 khoản bỏ ra) nhưng trong một số trường hợp, rất quan trọng tới hình ảnh thương hiệu.

Ví dụ:

* lỗi chính tả
* ảnh bị mờ
* các mục không thẳng hàng
* font chữ không đồng nhất
* mỏ neo dẫn tới sai chỗ
* trỏ chuột trên link nhưng không sáng lên
* vấn đề về thương hiệu (màu sắc, nền…)
* mô tả sai sản phẩm
* nội dung không được dịch

**Suggestion (Gợi ý)**

Đây thực chất không phải là lỗi. Tức là sản phẩm hoạt động bình thường theo đúng những gì hy vọng nhưng có thể bạn cho rằng nó gây bối rối hoặc không tuân theo chuẩn web.

Ví dụ:

* hiển thị số bình luận
* cung cấp thông tin lỗi chính xác trong mọi tình huống
* dùng hệ thống Captcha chuẩn
* tăng kích thước chữ trên di động để dễ dàng click vào link
* vuốt trái/phải để đi tới/lui trang.

## Phân loại lỗi theo hình thức

Đây là bản chất của lỗi, tức là luôn giống nhau dù ở trên sản phẩm nào. Dưới đây là 5 lỗi và ví dụ để giúp bạn nhận diện chúng.

**Lỗi chức năng (Functional Bug)**

Đây là lỗi quan trọng liên quan tới thao tác bạn thực hiện. Lỗi này chỉ có thể tìm thấy khi thao tác gì đó và sản phẩm không phản hồi như mong muốn.

Ví dụ:

* cuộn trang không được dù có nội dung bên dưới
* có thể cuộn trang ngang nhưng lại không được
* nút ấn không hoạt động
* email gửi không tới
* bàn phím chữ hiện lên mặc định thay vì bàn phím số
* không thể đăng nhập
* núi chuyển hướng tới trang 404
* thanh tải mãi không kết thúc
* người dùng quay về trang chủ trong khi muốn truy cập trang khác
* ứng dụng không hoạt động ở chế độ máy bay
* chạm nhưng không có tác động gì
* [mất âm thanh video](https://quantrimang.com/hoi-tai-sao-may-tinh-cua-toi-khong-co-am-thanh-116776)
* không thể tắt popup
* trang bị đơ
* tính năng tự động hoàn thiện không điền đủ thông tin
* chỉ hiển thị có lỗi xảy ra nhưng không nói rõ, ví dụ “an error has occurred, please try again later”, thay vì “please enter a valid email address”
* không thể phóng to, thu nhỏ trang
* không thể đăng ký
* thông tin địa lý không hoạt động hoặc sai
* bộ lọc không hoạt động
* mỏ neo sai
* không thể viết trong trường điền văn bản
* thông tin cũ (với sản phẩm)

**Lỗi đồ họa (Graphical Bug)**

Đây là lỗi tĩnh, liên quan tới giao diện, dàn trang.

Ví dụ:

* hình ảnh mờ
* không cân xứng
* giữa các yếu tố trên trang không đồng nhất, thiếu…
* trường nằm ngoài vùng hiển thị của màn hình
* hình ảnh, video méo mó
* hình ảnh, văn bản, link… bị chèn lên nhau

**Lỗi từ ngữ (Wording Bug)**

Lỗi ngày thuộc về nội dung văn bản.

Ví dụ:

* dịch dở
* văn bản hiển thị khác so với bản mockup
* kí tự đặc biệt không được mã hóa
* lỗi chính tả, ngữ pháp
* văn bản gây hiểu nhầm nghĩa
* thiếu từ
* từ viết thường và viết hoa không đồng nhất

**Yếu tố con người (Ergonomics)**

Các vấn đề này liên quan tới trải nghiệm người dùng, có thể là lỗi nhưng phần lớn chỉ là gợi ý.

Ví dụ:

* click quá nhiều lần mới xem được sản phẩm
* logo đặt ở vị trí không thích hợp
* thanh điều hướng quá lớn
* chữ quá nhỏ
* chữ và nền không đủ đối lập, khó nhìn
* trang không phản hồi khi dùng trên di động hay máy tính bảng
* nút đặt ở vị trí không thích hợp
* trường thông tin bắt buộc nên được nói rõ
* thời gian tải trang không nên quá lâu
* popin và popup nên có biểu tượng để tắt
* hành động không thể thao tác lại nên có thông báo “bạn có chắc chắn muốn thực hiện?”
* thao tác thành công nên có tin xác nhận “đã thành công”
* xanh lá cây, cam, đỏ thường dùng với ý nghĩa thành công, cảnh báo, thất bại, các thông điệp đang truyền tải thì màu sắc sẽ được dùng kết hợp

Quy tắc xác định bug

### Một số quy tắc xác định bug

Một lỗi phần mềm xuất hiện khi 1 hoặc nhiều hơn trong 4 quy tắc dưới đây là đúng:

#### 1. Nó không được thực hiện giống như mô tả trong bản đặc tả phần mềm

Áp dụng quy tắc này, tester chỉ cần luôn nắm được yêu cầu phần mềm thì việc xác định bug cũng sẽ đơn giản hơn.

* VD 1: Thử áp dụng quy tắc trên để phân tích lỗi trong phần mềm Calculator, với yêu cầu nói rõ rằng: 'nó phải thực thi phép cộng, phép trừ, phép nhân, phép chia đúng'.

Nếu bạn nhấn phím “+” và không có chuyện gì xảy ra, đó là một bug theo đúng quy tắc này do nó không thực hiện giống như mô tả. Tương tự, nếu bạn nhận được câu trả lời sai, cũng có nghĩa rằng đó là một bug.

* VD2: Kiểm tra chức năng của nút Back ở trang kết quả tìm kiếm của một website:



Ở VD này nút 'Back' không được làm việc theo thiết kế (Not Working as Designed). Vì nó không làm đúng chức năng yêu cầu nên nó rõ ràng là một bug.

#### 2. Bản đặc tả yêu cầu nó không được thực hiện nhưng nó vẫn xảy ra

Spec yêu cầu: 'Calculator sẽ không bao giờ bị đột ngột ngưng hoạt động, bị khóa lại hoặc bị đóng băng'. Nếu bạn nhấn liên tiếp lên các phím và nhận được thông báo từ calculator “not responding”, đây là một lỗi theo theo quy tắc 2.

#### 3. Nó là chức năng mà bản đặc tả không đề cập tới

Khi bạn phát hiện ra Calculator không chỉ thực hiện các phép tính công, trừ, nhân, chia mà còn thực hiện các phép căn bậc 2. Điều này không được nêu trong spec, một lập trình viên có nhiều tham vọng vừa thêm nó vào bởi vì anh ta cảm thấy nó sẽ là một tính năng hữu ích. Đây không phải là một một feature, nó thật sự là một bug theo quy tắc 3. Phần mềm đang thực hiện một số công việc không được yêu cầu.

Mặc dù feature này có thể là tốt, nhưng nó sẽ yêu cầu thêm những nỗ lực lập trình và kiểm thử (vì có thể sẽ xuất hiện thêm nhiều lỗi). Có thể làm cho chi phí sản xuất phần mềm lớn hơn, làm giảm hiệu quả kinh tế của quá trình sản xuất phần mềm.

#### 4. Nó là chức năng không được thực hiện mà bản đặc tả cũng không đề cập tới nhưng là những việc nên làm

Đọc quy tắc 4 có thể thấy hơi lạ với sự phủ định kép, nhưng mục đích của nó là tìm thấy những đặc điểm bị lãng quên, không được nhắc tới trong requirement. VD ở phần mềm Calculator khi Pin yếu, bạn không nhận được những câu trả lời đúng cho quá trình tính toán của bạn nữa. Chưa ai từng xem xét xem calculator phản ứng lại như thế nào trong chế độ này. Giả dụ pin luôn được nạp đầy thì những phép tính đúng đã không xảy ra. Tuy nhiên spec lại không nêu rõ khi pin yếu thì cần làm những gì, do đó việc thông báo pin yếu và không trả về kết quả tính toán sai là một việc cần thiết nên làm.

**Khi nào một bug không phải là bug?**

Một bug có thể đúng với 1 hoặc nhiều hơn trong 4 quy tắc trên. Vậy ngược lại khi nó không đúng với bất kỳ nguyên tắc nào trên đó nhưng vẫn chưa xác định được chính xác và rõ ràng là bug hay không? Hãy cùng thử trả lời mỗi câu hỏi dưới đây cho mỗi vấn đề đang gặp, có thể bạn sẽ biết được có nên đưa nó vào danh sách bugs không hay là feedback nó:

1. Nó có khó hiểu, khó sử dụng hay cản trở khả năng của người dùng sử dụng ứng dụng không?
2. Bạn có thể làm nó xảy ra từ hai lần trở lên không?
3. Nếu chỉ xảy ra 1 lần, nó có tạo ra kết quả tiêu cực đáng kể không?
4. Nó có làm mất hứng thú của người dùng sử dụng không?
5. Nó có gì trái ngược hay mâu thuẫn không?
6. Nó có phải là cách tối ưu nhất không?
7. Bạn có mong đợi nó xảy ra theo một cách khác?

Hãy thử áp dụng với một số tình huống lỗi sau:

* VD1: Trong ứng dụng Calculator có những nút có kích thước quá nhỏ. Hoặc có thể sự sắp xếp của các nút đã làm cho nó khó sử dụng. Hoặc sự bố trí màu sắc làm cho nó rất khó nhìn... Tất cả những điều này đều có câu trả lời là có cho câu hỏi số 1. Nên nó được xác định là bug.
* VD2: Đối với những lỗi mà nó không thể được tái hiện ở lần thứ hai (và không chỉ ra được kết quả ảnh hưởng) thì nó sẽ bị ưu tiên thấp và có khả năng sẽ bị từ chối. Lỗi này được gọi tên là 'Once Upon a Time Bug'.





Có thể ban đầu chúng ta bắt gặp nó là lỗi nhưng thật ra lại chỉ bị với trình duyệt của bạn, đó có thể là những lỗi như: hình ảnh bị hỏng, các nút không click được, lỗi đồng bộ video, ... Cách thông minh nhất để bug đó không bị từ chối là xóa bộ nhớ cache, khởi động lại trình duyệt và re-test lại để xác nhận lỗi.

* VD3: Không nhập gì vào ô tìm kiếm, khi nhấn Search thì load lại một trang trắng.

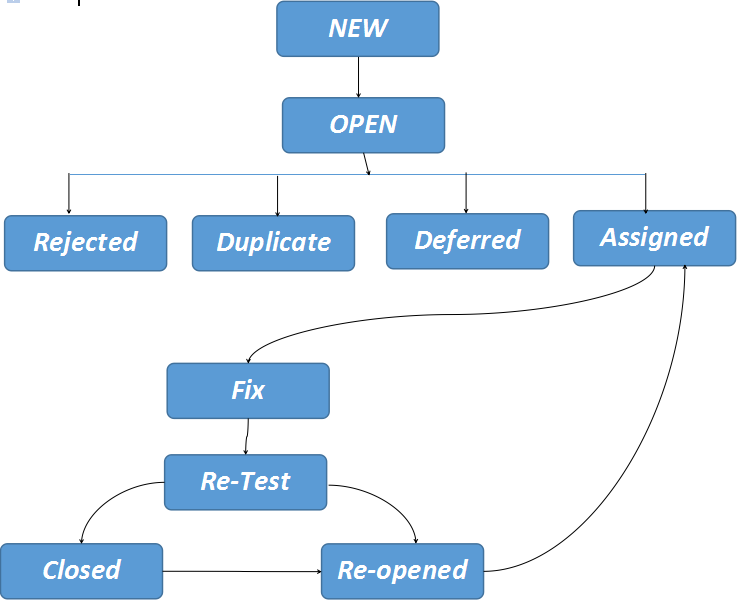


Bản đặc tả đã không yêu cầu về tính năng reload lại trang trong trường hợp này, nó có thể không ảnh hưởng đến việc sử dụng phần mềm của người dùng. Tuy nhiên người dùng không mong đợi như thế, nó được coi là một lỗi UX.

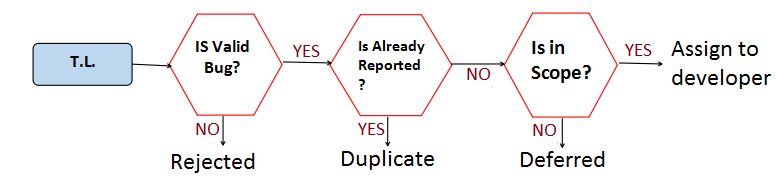
Vòng đời của BUG

Mục tiêu chính của tester không chỉ là tìm ra bug/khiếm khuyết của phần mềm mà còn phải theo dõi lỗi đó cho đến khi close nó. Vì vậy, vòng đời của bug là từ lúc tester tìm thấy bug đó đến khi close nó.

Dưới đây là sơ đồ vòng đời của bug:



Mô tả từng trạng thái của bug:

1. NEW (Mới): Khi tester thực thi test case và đầu ra của test case đó không đúng như kết quả mong đợi, thì họ sẽ gọi đó là bug. Nghĩa là có sự khác biệt giữa kết quả mong đợi và kết quả thực tế thì gọi là bug. Do đó bug này cần phải được fix. Nhưng tester không phải là người fix bug mà là lập trình viên sẽ fix nó. Tester sẽ báo bug này như thế nào? Họ sẽ đi đến chỗ của lập trình viên và nói rằng: Này anh, tôi đã tìm thấy 1 bug, hãy fix nó sớm nhé? Câu trả lời là không. Tester cần log bug này lên cho Team Lead và Team Lead sẽ assign cho developer fix bug này sau khi đã phân tích. (Có nhiều công ty, tester sẽ trực tiếp gán bug cho developer chứ không thông qua Team Lead). ===> Khi tester tìm thấy bug thì nó sẽ có trạng thái là NEW
2. OPEN: Lỗi được log lên bởi tester. Team lead cần xác minh lại bug đó xem có đúng là bug hay không, thì bug có trạng thái OPEN. Sơ đồ dưới đây là những hoạt động cần được thực hiện bởi team lead:  H1
3. REJECTED (Từ chối): Một bug được đánh dấu là Rejected khi bug đó không hợp lệ. Nghĩa là thỉnh thoảng tester có thể hiểu sai chức năng và có thể đánh dấu chức năng là bug. Trong trường hợp này, bug sẽ bị reject sau khi team lead kiểm tra lại. ===> Khi tester báo cáo một bug nhưng nó lại là chức năng của ứng dụng thì team lead sẽ đánh dấu nó là REJECTED (H1).
4. DUPLICATE (Trùng lặp): Nếu bug là hợp lệ, thì sau đó team lead sẽ kiểm tra xem lỗi đó đã được log người khác hay chưa. Nếu đã có người khác log nó, thì team lead sẽ đánh dấu nó là DUPLICATE. Còn nếu nó chưa được báo cáo bởi tester khác thì team lead sẽ thực hiện tìm kiếm nó trong scope. Như chúng ta đã biết, chúng ta làm việc trong một team. Có khả năng rằng cùng một phần mềm hoặc một module sẽ được gán cho nhiều hơn một tester, trong trường hợp này lỗi tương tự có thể được tìm thấy bởi nhiều tester. Vì vậy, team lead cần đảm bảo rằng, cùng một lỗi sẽ không được báo cáo 2 lần hoặc nhiều hơn thế. ===> Nếu cùng một bug dược báo cáo bởi hai hay nhiều tester thì lỗi được báo cáo sau sẽ được đánh dấu là DUPLICATE (H1)
5. DEFERRED (Hoãn lại): Nếu bug không bị duplicate, nhưng lại không thuộc bản release hiện tại thì sẽ được đánh dấu là Deferred. Nghĩa là giả sử bạn đang làm theo mô hình agile, và họ chia yêu cầu dự án thành các sprint, ví dụ chia thành 10 sprint: sprint 1, sprint 2, ..., sprint 10. Hiện tại đang ở sprint 1, nhưng bug bạn tìm thấy lại có liên quan đến tính năng sẽ được phát triển ở sprint 2, thì bug này sẽ được đánh dấu là DEFERRED. Deferred bug là một bug, nhưng nó sẽ được sửa chữa trong bản release tương lai. ===> Khi một bug là một phần của bản release tương lai thì nó sẽ được đánh dấu là DEFERRED (H1)
6. ASSIGNED (Gán bug): Khi bug tìm thấy là hợp lệ, duy nhất và thuộc bản release hiện tại, thì team lead sẽ gán bug đó cho developer.
7. FIX: Khi nhận được bug từ team lead, developer sẽ thực hiện thay đổi để fix bug cho đúng với yêu cầu, và đẩy lại cho tester kiểm tra lại lỗi đó.
8. RE-TESTING (Test lại): Sau khi fix xong bug, và chức năng/tính năng đã sẵn sàng để kiểm thử, thì tester sẽ thực hiện lại những test case lỗi và xác minh lại xem nó đã chạy đúng hay chưa. Việc này gọi là RE-TESTING.
9. CLOSED Khi bug đã được fix, đã được kiểm thử lại và nó chạy đúng như yêu cầu thì tester sẽ đánh dấu nó là CLOSED.
10. RE-OPENED: Có 2 tình huống mà chúng ta cần phải re-open lại bug: Tình huống 1: Khi developer fix bug và tester thực hiện test lại nó, nhưng sau khi re-test, bug đó vẫn xảy ra thì tester sẽ RE-OPEN lại bug và assign cho developer Tình huống 2: Có trường hợp lỗi đã fix và được close xuất hiện lại. Trong trường hợp này, tester cần RE-OPEN lại bug đã close và gán nó cho developer.