Để mang đến một sản phẩm phần mềm chất lượng đáng tin cậy thì việc phân tích yêu cầu là khâu vô cùng quan trọng trong quá trình xây dựng phần mềm. Hoạt động này đòi hỏi sự kết hợp rất chặt chẽ giữa khách hàng và người phân tích để vạch ra được xem chúng ta phải phát triển cái gì.

Yêu cầu của phần mềm là tất cả các yêu cầu về phần mềm do người dùng nêu ra bao gồm các chức năng của phần mềm, hiệu năng của phần mềm, giao diện của phần mềm và một số các yêu cầu khác.

Thông thường các yêu cầu phần mềm được phân loại dựa trên 4 thành phần của phần mềm như sau:

* Các yêu cầu về phần mềm
* Các yêu cầu về phần cứng
* Các yêu cầu về dữ liệu
* Các yêu cầu về con người

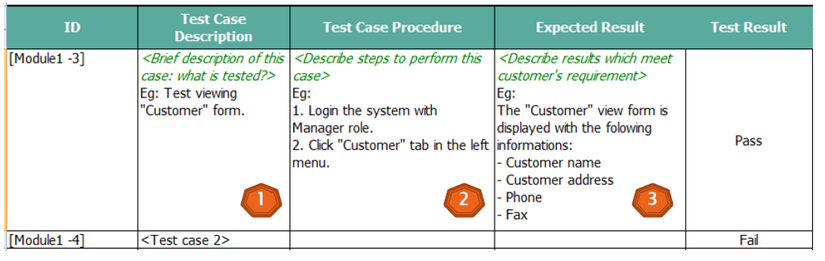
Mục tiêu quan trọng nhất đối với chất lượng phần mềm là phần mềm phải thỏa mãn được các yêu cầu và mong muốn của người dùng.

Người dùng thường chỉ đưa ra những ý tưởng, nhiều khi rất mơ hồ về phần mềm mà họ mong muốn xây dựng. Và việc của các kỹ sư phát triển phần mềm đó là phải giúp họ đưa những ý tưởng mơ hồ đó thành hiện thực và xây dựng được một phần mềm có đầy đủ các tính năng cần thiết thỏa mãn yêu cầu của người dùng. Hơn thế nữa, ý tưởng của người dùng thường xuyên thay đổi và việc của nhà phát triển là phải nắm bắt và đáp ứng được các yêu cầu thay đổi đó một cách hợp lý.

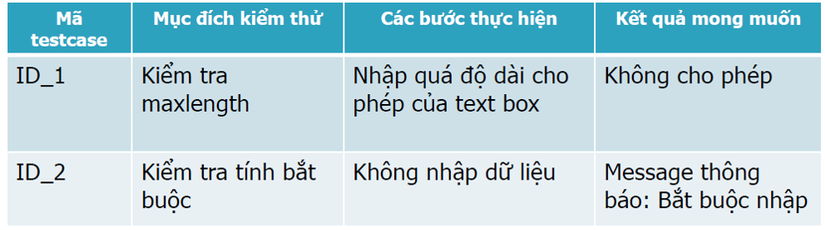
* Đọc và cố gắng hiểu mục đích của ứng dụng đang mong muốn là gì?
* Vừa đọc và hình dung, tưởng tượng xem phần mềm /màn hình đó sẽ chạy như thế nào.
* Thẩm định từng yêu cầu phần mềm để xác định xem chúng có khả năng thực hiện được hay không.
* Xác định các rủi ro có thể xảy ra với từng yêu cầu cụ thể.
* Thảo luận với BA về những băn khoăn, vướng mắc, bất hợp lý, chưa rõ ràng trong tài liệu đặc tả yêu cầu
* Viết Q & A gửi cho Khách Hàng

**Hướng dẫn viết TESTCASES**

**Testcase là gì?**

* Quá trình phát triển test case có thể giúp tìm ra lỗi trong các yêu cầu hoặc thiết kế của ứng dụng, vì nó đòi hỏi phải tư duy hoàn toàn thông qua các hoạt động của ứng dụng. Vì lý do này, việc chuẩn bị test case sớm nhất có thể trong qui trình phát triển phần mềm là rất hữu ích
* Các trường hợp kiểm thử phải bao phủ được toàn bộ luồng xử lý chức năng mô tả trong tài liệu phân tích và thiết kế; các yêu cầu về bảo mật an toàn thông tin, yêu cầu hiệu năng của hệ thống. 
* Testcase gồm 3 bước cơ bản: **Mục đích kiểm thử** ( Miêu tả testcase-Testcase Description ): + Miêu tả của test case là phần bạn sẽ đề cập một cách chi tiết những gì mà bạn sẽ test và cách xử lý riêng biệt được kiểm tra bằng test.
* “Miêu tả” của một test case nên đưa ra được “Mình sẽ test những gì”?
* Ví dụ: Test nhập quá max length cho username

**Các bước thực hiện / Testcase Procedure:**

* Dữ liệu đầu vào của test: định nhập cái gì để ra được kết quả mong muốn
* Việc xác định dữ liệu đầu vào của test thực sự là hoạt động tốn nhiều thời gian.
* Dữ liệu test chính là phần Input dữ liệu đầu vào, để hệ thống xử lý và trả ra Kết quả mong đợi
* Ví dụ: tester nhập dữ liệu test: 1.Nhập username = anhdt 2.Nhập password = abc@123 \*\*Kết quả mong đợi / Expected result: \*\*
* Một test case được viết tốt cần phải đề cập một cách rõ ràng kết quả mong đợi của ứng dụng hoặc hệ thống.
* Mỗi bước thiết kế test nên chỉ ra rõ ràng những gì bạn mong đợi
* Phần mềm sẽ phải chạy đúng như Kết quả mong đợi, nếu ko giống thì sẽ là Lỗi ( bug/ defect) và test case đó là Fail **Cột Kết quả test/Test result:** Thông thường sẽ là pass, fail, và pending. Đây là kết quả thực tế khi thực hiện test theo test case trên môi trường của hệ thống **Xác định trường hợp kiểm tra.** Với 1 giá trị cần kiểm tra luôn luôn có 3 trường hợp lớn cần kiểm tra có thể xảy ra. Normal case: Các trường hợp kiểm thử thông thường Abnormal case: Các trường hợp kiểm thử bất bình thường Boundary case: Các trường hợp kiểm tra boundary ( phân tích giá trị biên).  **Đối với testcase chức năng**:
* Các bước thực hiện chỉ mô tả các bước thực hiện đứng từ phía người dùng cuối bao gồm nhập dữ liệu, nhấn button.
* Việc kiểm tra dữ liệu trong DB so với hiện thị trên màn hình nằm ở kết quả mong muốn. Thường được dùng cho các trường hợp kiểm thử kiểm tra lưu, cập nhật, xóa DB SELECT \* FROM … WHERE… Ví dụ: Tạo 1 email đăng ký thành công :
* Test tạo 1 email đăng ký thành công. Đã đăng nhập thành công bằng email mới trên giao diện
* Vào DB check xem email đó có được lưu vào DB hay không? ( tuy nhiên nhiều nơi ko yêu cầu tester vào CSDL để check) **Ví dụ** : Thực hiện viết TCs cho chức năng đăng nhập facebook  **Xác định Yêu cầu**: Form login bao gồm: 2 text box email/điện thoại và mật khẩu, 1 button đăng nhập, 1 link quên mật khẩu. **Xây dựng TCs**:

1. Xác định các case UI: Bao gồm UI chung của cả form: màu sắc, font, size, color của label, chiều dài, rộng, cao, loại của các textbox, button, vị trí của form, textbox, button, link trên trang
2. Xác định case test chức năng: Ở đây chức năng là đăng nhập gồm 2 text box email/điện thoại và mật khẩu, 1 button đăng nhập, 1 link quên mật khẩu. Cho nên sẽ có những case như sau. **Đối với email/ điện thoại textbox**:

* Normal case sẽ gồm: đăng nhập với đúng sdt, địa chỉ email đã đăng ký với hệ thống facebook trước đó và đăng nhập với blank, sai sdt, địa chỉ email đã đăng ký với hệ thống facebook trước đó.
* Abnormal case sẽ gồm: Đăng nhập với số điện thoại mà thêm mã vùng, mã nước vào trước đó (ví dụ: +849....) hoặc email mà không nhập @ cho nó. Ngoài ra, còn có các cases: offline mode ( ko có internet), đang đăng nhập mà có điện thoại gọi đến nếu là mobile…
* Boundary case sẽ gồm: Text số lương ký tự tối thiểu và tối đa mà ô text cho nhập. Có thể tạo ra 1 email với nhiều ký tự để test, hoặc 1 email ngắn nhất có thể để test.( Với trường hợp này tôi ko kiểm tra tính đúng đắn của follow đăng nhập mà chỉ kiểm tra khả năng cho nhập tối thiểu và tối đa của ô text.) **Tương tự với ô mật khẩu**:
* Tương tự với ô mật khẩu: nhưng thêm vào nữa ở ô mật khẩu cần kiểm tra thêm tính mã hóa password nữa.
* Đối với button đăng nhập:
* Normal case sẽ gồm: Nhập giá trị vào text, click button đăng nhập, bấm phím enter từ bàn phím.
* Abnormal case sẽ gồm: Click liên tục or enter liên tục vào button
* Boundary case sẽ gồm: không cần check trường hợp này **Lưu ý khi viết testcases**:
* File testcase cần có những step test đơn giản, minh bạch, dễ hiểu:
* Step test phải viết chi tiết rõ ràng để ngay cả khi tester khác đọc có thể thực hiện được
* Mục đích và phạm vi của testcase cũng được mô tả chi tiết.
* Điều kiện tiền đề, data test cũng phải được ghi ở từng testcase nếu cần.
* Testcase nên được review chéo bởi member trong team.
* Không nên gộp quá nhiều kết quả confirm vào 1 case mà nên tách mỗi kết quả confirm ra từng case.
* Khi tạo testcase nên đứng ở vị trí End user.
* Testcase nên cover các trường hợp kiểm thử như: Phân lớp tương đương, giá trị biên, điều kiện normal và abnormal. Cả các trường hợp free test không có trong đặc tả yêu cầu. **Mục đích tạo testcase**:
* Mục tiêu cơ bản của việc tạo tescase là để xác nhận độ bao phủ kiểm thử của ứng dụng, là bằng chứng quan trọng để xác nhận phần mềm có đủ tiêu chuẩn để triển khai hay không?
* Testcase được tạo là một tài liệu chính thức bắt buộc của dự án để kiểm soát chất lượng phần mềm.
* Tái sử dụng khi thay đổi, nâng cấp hệ thống
* Testcase hiệu quả
* Testcase tốt/hiệu quả là testcase có khả năng tìm ra lỗi cao. \*\*Một testcase hiệu quả cần có các đặc điểm sau: \*\*
* Chính xác, đầy đủ nghiệp vụ hệ thống
* Độc lập (có thể thực hiện mà không phụ thuộc vào các testcase khác, dễ dàng chia cho nhiều người cùng kiểm thử)
* Nội dung đơn giản, có mục đích rõ ràng và ai đọc cũng hiểu theo một cách duy nhất. (đầu vào, đầu ra, các bước thực hiện rõ ràng)
* Trình bày mạch lạc thống nhất cho toàn bộ tài liệu.
* Có khả năng tái sử dụng (có thể dễ dàng cập nhật và sửa đổi)