Trong quá trình thực hiện công việc **kiểm thử phần mềm**, chúng ta gặp phải vô số các khái niệm khác nhau. Các khái niệm này có thể đã quen thuộc với nhiều người, nhưng cũng có rất nhiều khái niệm mà chúng ta lạ lẫm và chưa từng nghe thấy.

Kiểm thử phần mềm (Software Testing)

Kiểm thử phần mềm là quá trình thực thi một chương trình với mục đích tìm ra lỗi. **Kiểm thử phần mềm** đảm bảo sản phẩm phần mềm đáp ứng chính xác, đầy đủ và đúng theo yêu cầu của khách hàng, yêu cầu của sản phẩm đề đã đặt ra. **Kiểm thử phần mềm** cũng cung cấp mục tiêu, cái nhìn độc lập về phần mềm, điều này cho phép việc đánh giá và hiểu rõ các rủi ro khi thực thi phần mềm. Kiểm thử phần mềm tạo điều kiện cho bạn tận dụng tối đa tư duy đánh giá và sáng tạo để bạn có thể phát hiện ra những điểm mà người khác chưa nhìn thấy.

Kiểm thử hộp đen (Black Box Testing)

Kiểm thử hộp đen là 1 phương pháp kiểm thử mà tester sẽ chỉ xem xét đến đầu vào và đầu ra của chương trình mà không quan tâm code bên trong được viết ra sao. Tester thực hiện kiểm thử dựa hoàn toàn vào đặc tả yêu cầu. Mục đích của kiểm thử hộp đen là tìm ra các lỗi ở giao diện, chức năng của phần mềm. Các trường hợp kiểm thử sẽ được xây dựng xung quanh đó.

Xem thêm: Kiểm thử hộp đen – Black Box Testing

Kiểm thử hộp trắng (White Box Testing)

Kiểm thử hộp trắng là phương pháp kiểm thử mà cấu trúc thuật toán của chương trình được đưa vào xem xét. Các trường hợp kiểm thử được thiết kế dựa vào cấu trúc mã hoặc cách làm việc của chương trình. Người kiểm thử truy cập vào mã nguồn của chương trình để kiểm tra nó.

Xem thêm: Kiểm thử hộp trắng – White Box Testing

Kiểm thử đơn vị (Unit Test)

Kiểm thử đơn vị là hoạt động kiểm thử nhỏ nhất. Kiểm thử thực hiện trên các hàm hay thành phần riêng lẻ. Đây là một công việc mà để thực hiện được nó thì người kiểm thử sẽ phải hiểu biết về code, về chương trình, các hàm... Nếu bạn đang lo lắng vì bạn không có nhiều kiến thức về code thì không sao cả, vì bạn sẽ không phải thực hiện bước kiểm thử này, lập trình viên sẽ làm nó trước khi giao cho bạn.

Mục đích của việc thực hiện kiểm thử đơn vị là cô lập từng thành phần của chương trình và chứng minh các bộ phận riêng lẻ chính xác về các yêu cầu chức năng.

Xem thêm: Kiểm thử đơn vị – Unit Testing

Kiểm thử tích hợp (IntegrationTesting)

Như chúng ta đã biết, một phần mềm được tạo ra sẽ bao gồm rất nhiều module trong đó, để chắc chắn rằng phần mềm hoạt động tốt thì chúng ta cần phải gom các module lại với nhau

để kiểm tra sự giao tiếp giữa các module cũng như bản thân từng thành phần từng module... Kiểm thử tích hợp bao gồm hai mục tiêu chính là : Phát hiện lỗi giao tiếp xảy ra giữa các Unit Tích hợp các Unit đơn lẻ thành các hệ thống nhỏ và cuối cùng là 1 hệ thống hoàn chỉnh để chuẩn bị cho bước kiểm thử hệ thống.

Xem thêm: Kiểm thử tích hợp – Integration Testing

Kiểm thử hệ thống (System test)

Kiểm thử một hệ thống đã được tích hợp hoàn chỉnh để xác minh rằng nó đáp ứng được yêu cầu Kiểm thử hệ thống thuộc loại kiểm thử hộp đen. Kiểm thử hệ thống tập trung nhiều hơn vào các chức năng của hệ thống. Kiểm tra cả chức năng và giao diện, các hành vi của hệ thống 1 cách hoàn chỉnh, đáp ứng với yêu cầu.

Xem thêm: Kiểm Thử Hệ Thống – System Testing

Kiểm thử chấp nhận (Acceptance test)

Trong kiểu kiểm thử này, phần mềm sẽ được thực hiện kiểm tra từ người dùng để tìm ra nếu phần mềm phù hợp với sự mong đợi của người dùng và thực hiện đúng như mong đợi. Trong giai đoạn test này, tester có thể cũng thực hiện hoặc khách hàng có các tester của riêng họ để thực hiện. Có 2 loại kiểm thử chấp nhận đó là kiểm thử Alpha và kiểm thử Beta:

Kiểm thử Alpha: là loại kiểm thử nội bộ . Tức là phần mềm sẽ được 1 đội kiểm thử độc lập hoặc do khách hàng thực hiện tại nơi sản xuất phần mềm.

Kiểm thử Beta: là loại kiểm thử mà khách hàng thực hiện kiểm thử ở chính môi trường của họ. Loại kiểm thử này được thực hiện sau kiểm thử Alpha.

Xem thêm: Kiểm Thử Chấp Nhận – Acceptance Testing

Kiểm thử chức năng (Functional testing)

Kiểm thử chức năng là một loại kiểm thử hộp đen (black box) và các trường hợp kiểm thử của nó được dựa trên đặc tả của ứng dụng phần mềm/thành phần đang test. Các chức năng được test bằng cách nhập vào các giá trị nhập và kiểm tra kết quả đầu ra, và ít quan tâm đến cấu trúc bên trong của ứng dụng (không giống như kiểm thử hộp trắng - white-box testing). Có thể hiểu 1 cách đơn giản, kiểm thử chức năng là xác nhận tất cả các chức năng của hệ thống. Nó đánh giá ứng dụng và xác nhận liệu ứng dụng có đang hoạt động theo yêu cầu hay không.

Xem thêm: Tìm hiểu về Functional Testing

Test cấu hình (Shakeout Testing)

Kiểu kiểm thử này cơ bản là kiểu kiểm thử về khả năng của hệ thống mạng, kết nối dữ liệu và sự tương tác của các module. Thông thường thì kiểu test này là do nhóm quản lý cấu hình chuẩn bị thiết lập các môi trường test thực sự. Họ cũng kiểm tra xem liệu các thành phần

chính của phần mềm có hoạt động bất thường không. Kiểu kiểm thử này thực hiện trước khi tiến hành thực hiện trong môi trường test. Sau khi test shakeout, bước kế tiếp là test smoke (kiểu test được thực hiện bởi tester sau khi biên dịch, được tiến hành trong môi trường test).

Smoke Testing

Smoke Testing là quá trình để kiểm tra liệu bản build có ổn định hay không? Để xem bản build có đủ ổn định để thực hiện test chi tiết hay không (trong trường hợp bản build không ổn định, lỗi luôn chức năng chính hoặc build bị lỗi thì trả lại Dev, yêu cầu sửa luôn). Hay kiểm tra các tính năng quan trọng có đang hoạt động hay không. Nó là 1 bài test hồi quy nhỏ đơn giản và nhanh của các chức năng chính, cho thấy sản phẩm đã sẵn sàng cho việc test hay chưa.

Xem thêm: Tìm hiểu về Smoke Testing và Sanity Testing

Adhoc Testing

Thuật ngữ Adhoc testing là phương pháp kiểm thử dạng Black box test mà không theo cách thông thường. Với quy trình test thông thường là phải có tài liệu yêu cầu, kế hoạch test (test plan), kịch bản kiểm thử. Kiểu test này không theo bất cứ loại kỹ thuật test nào để tạo test case.

Xem thêm: Tìm hiểu về Adhoc Testing

Monkey Testing

Monkey testing được định nghĩa rất ngắn gọn: là một phương pháp kiểm thử với đầu vào ngẫu nhiên, không theo test case hay một chiến lược test nào. Trong Monkey testing thì các tester (đôi khi cả developer nữa) được coi như là 1 con khỉ vậy. Bạn thử nghĩ mà xem, nếu 1 con khỉ mà sử dụng máy tính thì nó sẽ làm những gì nhỉ? Tuy loài khỉ rất thông minh nhưng khi cho nó sử dụng máy tính, nó sẽ thực hiện những hành động bất kỳ trên hệ thống , điều mà chính nó cũng không thể hiểu được. Nó cũng giống như khi tester thực hiện monkey testing, họ sẽ áp dụng các kịch bản kiểm thử ngẫu nhiên trên hệ thống để tìm ra lỗi mà không cần xác định trước. Trong 1 số trường hợp, Monkey testing chỉ dành cho Unit Testing hoặc GUI Testing (Kiểm thử giao diện người dùng)

Kiểm thử hiệu suất (Performance Testing)

Trong loại test này, ứng dụng được test dựa vào sức nặng như sự phức tạp của giá trị, độ dài của đầu vào, độ dài của các câu truy vấn… Loại test này kiểm tra bớt phần tải (stress/load) của ứng dụng có thể được chắc chắn hơn.

Xem thêm: Tìm hiểu về Performance Testing

Kiểm thử hồi quy (Regression Testing)

Test hồi quy là test lại 1 chức năng đã được làm xong, đã được test xong rồi, đã hết lỗi nhưng do có sự sửa đổi 1 chức năng khác mà lại có ảnh hưởng đến nó, thì phải test 1 chức năng đã xong rồi thì gọi là test hồi quy. Ví dụ có 3 chức năng A B C đã hoàn thành, 3 chức năng này đều có mối quan hệ với nhau và chức năng A cần phải sửa đổi thêm về nghiệp vụ, và việc sửa chức năng A sẽ làm ảnh hưởng đến chức năng B, C và việc phải test lại chức năng B và C thì gọi là test hồi quy. Hoặc khi Developer sửa 1 chức năng mà chức năng này có làm ảnh hưởng đến chức năng đã xong rồi thì cũng phải thực hiện test lại chức năng đã xong rồi kia thì gọi là test hồi quy

Hoặc ngay cả khi re-test để đóng bug, mà thấy chức năng Developer sửa có thể làm ảnh hưởng đến 1 chức năng khác đã xong rồi thì tester cũng nên test hồi quy lại chức năng đó để tránh có lỗi tiềm ẩn.

Accessibility Testing

Là một loại kiểm thử dành cho các hệ thống được thiết kế cho người khuyết tật (khiếm thính, khiếm thị, thiểu năng tâm thần...), xác định khả năng tiếp cận và sử dụng của họ với ứng dụng. Ví dụ như các thiết lập Accessibility của smart phone: Text-to-Speech, "Magnification gestures"... Người khuyết tật thường sử dụng những tính năng hỗ trợ họ sử dụng các phần mềm. Loại test này có thể được thực hiện theo 2 cách là Manual và Automated.

Active Testing

Loại kiểm thử bao gồm việc đưa ra dữ liệu test và phân tích kết quả thực hiện. Nó thường được tiến hành bởi đội tester. Trong quá trình active testing, tester sẽ hình dung để dựng một mô hình của phần mềm để test, cái này sẽ được phát triển và làm hoàn thiện hơn các tương tác của nó với những phần mềm đang chạy.

Agile Testing

Kiểm thử Agile là việc kiểm thử phần mềm được thực hiện theo một số quy định của bản tuyên ngôn (Manifesto) Agile, xem việc phát triển phần mềm như là khách hàng của việc kiểm thử. Kiểm thử Agile thực hiện kiểm thử theo quan điểm của khách hàng càng sớm càng tốt, thử nghiệm sớm và thường xuyên ngay khi code vừa xong và đủ ổn định để test từ level unit test.

API Testing

- API (Application Programming Interface): cho phép kết nối và trao đổi dữ liệu giữa hai hệ thống phần mềm riêng biệt. Một hệ thống phần mềm có thể nhúng các API bao gồm các hàm/thủ tục con (functions/sub-routines) mà có thể chạy bởi một hệ thống phần mềm khác.
- Kiểm thử API khác hoàn toàn với kiểm thử GUI và các thành phần chủ yếu khác trong tầng business logic của kiến trúc phần mềm. Loại kiểm thử này không tập trung vào phần giao diện và thao tác giao diện của một ứng dụng. Thay vì sử dụng các đầu vào (bàn phím) và đầu ra tiêu chuẩn, trong kiểm thử API, bạn có thể sử dụng một phần mềm để gửi các yêu cầu đến API, nhận đầu ra và ghi lại phản hồi của hệ thống.

All-pairs Testing

All pair testing (Kiểm thử tất cả các cặp) hay còn gọi là pairwise testing, là một phương pháp test được thực hiện để kiểm thử các phần mềm sử dụng phương pháp tổ hợp. Đó là một phương pháp để kiểm tra tất cả các kết hợp rời rạc có thể có của các thông số liên quan, phương pháp test ít nhất sao cho chất lượng tốt nhất.

Basis Path Testing

Thuộc phương pháp kiểm thử hộp trắng (White box test) là kỹ thuật kiểm thử mà phần mềm được chia thành các lộ trình, đảm bảo các lộ trình độc lập qua một module mã sẽ được kiểm thử đầy đủ. (Được thực hiện bởi developer).

Backward Compatibility Testing

Kiểm tra sự tương thích ngược: là việc kiểm tra xem các sản phẩm của ứng dụng cũ có tiếp tục làm việc tốt trên phiên bản mới của ứng dụng đó hay không.

Benchmark Testing

Là việc xác định performance hiện tại của hệ thống và thực hiện các thay đổi để cải thiện performance của sản phẩm cho tốt hơn. Như việc refactor code, tinh chỉnh cơ sở dữ liệu để người dùng có thể trải nghiệm những cải tiến hiệu suất. Được thực hiện bởi đội dev hoặc Database admin (DBAs).

Big Bang Integration Testing

Khi mà tất cả mọi thứ sẵn sàng thì kĩ thuật kiểm tra này thực hiện để tích hợp các module độc lập của chương trình.

Binary Portability Testing

Kĩ thuật test tính di động của phần mềm bằng cách chạy phần mềm trên các nền tảng và môi trường khác nhau. Nó được sử dụng cho cấu tạo của Application Binary Interface (ABI). Binary Portability testing nên được tiến hành trên các loại khác nhau của platform, như Windows (x86, X86-64), Linux, Mac OS, Java, Solaris, and Android. Nếu ứng dụng có tính di động cao thì người dùng có thể chạy ứng dụng trên bất kì nền tảng (platform) nào. Do đó để test Binary portability thì tức là khi xây dựng 1 phần mềm thì cần test việc nó thực thi được ứng dụng đó trên nhiều hệ điều hành khác nhau, nếu nó là website thì cần chạy được trên nhiều trình duyệt khác nhau để kiểm tra tính di động của nó. Được thực hiện bởi nhóm test.

Boundary Value Testing

Kĩ thuật kiểm thử phần mềm trong đó test case được thiết kế để bao gồm những giá trị giới hạn tiêu biểu. (Được thực hiện bởi đội tester).

Bottom Up Integration Testing

Trong kiểm thử tích hợp từ dưới lên, module ở mức thấp nhất được phát triển đầu tiên và các module khác đi theo hướng chương trình chính được tích hợp và kiểm thử cùng một lúc. (Được thực hiện bởi đội tester).

Branch Testing

Kỹ thuật kiểm thử, trong đó tất cả các nhánh trong mã nguồn của chương trình sẽ được kiểm tra ít nhất một lần. (Được thực hiện bởi các developer).

Breadth Testing

Là việc kiểm tra bằng các test suit thực hiện đầy đủ các chức năng của một sản phẩm nhưng không kiểm tra từng tính năng chi tiết. (Được thực hiện bởi đội tester)

Code-driven Testing

Kỹ thuật kiểm thử sử dụng framework để kiểm thử (như xUnit) cho phép thực hiện các kiểm thử đơn vị (unit test) để xác định xem các phần khác nhau của mã đang hoạt động như mong đợi trong những tình huống khác nhau. (Được thực hiện bởi các developer).

Compatibility Testing

Kỹ thuật kiểm thử xác nhận làm thế nào một phần mềm thực hiện tốt trong một phần cứng / phần mềm / hệ điều hành/ môi trường hệ thống / môi trường mạng. (Được thực hiện bởi đội tester).

Comparison Testing

Kỹ thuật kiểm thử so sánh điểm mạnh và điểm yếu của sản phẩm với các phiên bản trước đó hoặc các sản phẩm tương tự khác. Thực hiện bởi nhân viên kiểm thử, các nhà phát triển, quản lý sản phẩm hoặc chủ sở hữu sản phẩm.

Component Testing

Kỹ thuật kiểm thử tương tự với kiểm thử đơn vị, nhưng với một mức độ kiểm thử tích hợp cao hơn được thực hiện kiểm thử các ứng dụng thay vì chỉ trực tiếp kiểm thử một phương thức cụ thể. (Được thực hiện bởi đội tester hoặc đội developer).

Configuration Testing

Kỹ thuật kiểm thử quyết định cấu hình tối thiểu và tối ưu của phần cứng và phần mềm, hiệu quả của việc thêm hoặc sửa đổi các nguồn tài nguyên như bộ nhớ, ổ đĩa và CPU. (Thực hiện bởi các kỹ sư kiểm thử hiệu năng.)

Condition Coverage Testing

Loại kĩ thuật kiểm thử phần mềm mà mỗi điều kiện được gán 2 giá trị đúng và sai ít nhất 1 lần. (Thực hiện bởi các nhóm kiểm thử tự động.)

Compliance Testing

Loại kiểm thử mà kiểm tra xem hệ thống đã được xây dựng phù hợp với các tiêu chuẩn, thủ tục và hướng dẫn hay không. Người thực hiện: Các công ty bên ngoài, các công ty cung cấp thương hiệu "chứng nhận tuân thủ OGC ".

Concurrency Testing

Kiểm thử đa người dùng hướng tới việc xác định ảnh hưởng của việc tiếp cận cùng một mã ứng dụng, module hoặc cơ sở dữ liệu. (Thực hiện bởi các kỹ sư hiệu năng).

Conformance Testing

Quá trình kiểm thử một sản phẩm hoặc hệ thống phù hợp với đặc điểm kỹ thuật của spec. (Thực hiện bởi các đội kiểm thử).

Context Driven Testing

Một kỹ thuật kiểm tra Agile, khuyến khích các tester tự lựa chọn test techniques, test deliverable, test documentation và test objectives, theo chủ trương đánh giá liên tục và sáng tạo các cơ hội kiểm thử của những thông tin tiềm năng, các giá trị của thông tin để tổ chức kiểm thử tại một thời điểm cụ thể. (Thực hiện bởi các đội kiểm thử Agile).

Conversion Testing

Được thực hiện khi chuyển đổi dữ liệu từ hệ thống hiện có sang hệ thống mới thay thế. Kiểm thử các chương trình hoặc các thủ tục được sử dụng để chuyển đổi dữ liệu từ hệ thống hiện có để sử dụng trong các hệ thống thay thế. (Thực hiện bởi đội ngũ QA).

Decision Coverage Testing

Loại hình kiểm thử phần mềm mà mỗi quyết định được thực hiện ít nhất một lần đảm bảo rằng tất cả các code được thực hiện. Mỗi quyết định được thực hiện bằng cách đặt vào 2 giá trị đúng và sai. (Thực hiện bởi developer hoặc các nhóm kiểm thử tự động hóa).

Destructive Testing

Loại kiểm thử trong đó các kiểm thử được tiến hành với các dữ liệu kiểm thử thất bại , để hiểu về hiệu suất ,hành vi quan trọng của một mẫu kiểm thử theo tải trọng khác nhau. (Thực hiện bởi đội ngũ QA).

Dependency Testing

Loại kiểm thử trong đó xem xét các yêu cầu của một ứng dụng cho các phần mềm tiền đề, trạng thái ban đầu và cấu hình để duy trì chức năng thích hợp. (Thực hiện bởi các đội kiểm thử).

Dynamic Testing

Kiểm thử liên quan đến việc thực thi các thành phần hoặc toàn bộ hệ thống phần mềm. (Thực hiện bởi đội tester).

Domain Testing

Là loại kiểm thử kiểm tra những thông tin mà người dùng sử dụng để nhập vào trên các vùng dữ liệu, kiểm tra các kết quả nhận được và xem xét chúng có đúng không. (Thực hiện bởi đội phát triển phần mềm và team test automation).

Error Handling Testing

Loại kiểm thử phần mềm xác định trách nhiệm của hệ thống xử lý các giao dịch có lỗi một cách thích hợp. (Thực hiện bởi đội tester).

End-to-end Testing

Tương tự với system testing, liên quan đến việc kiểm thử trong môi trường tương tự với môi trường sử dụng thật, ví dụ tương tác với DB, sử dụng giao tiếp mạng, hoặc tương tác với các phần cứng, ứng dụng, hoặc hệ thống nếu phù hợp. (Thực hiện bởi đội QA)

Endurance Testing

Loại test để check việc thiếu bộ nhớ hoặc các vấn đề có thể xảy ra khi ứng dụng được thực hiện trong thời gian dài. (Thực hiện bởi đội hiệu năng).

Fault Injection Testing

Là kỹ thuật mà tester tập trung vào phương pháp mà hệ thống xử lý ngoại lệ thông qua việc truyền mã độc vào hệ thống. (Được thực hiện bởi đội QA).

Formal Verification Testing

Kiểm định hình thức theo kiểu chứng minh định lý (Theorem Proving), kiểm tra logic hệ thống xem có phù hợp với spec không. (Thường được thực hiện bởi đội QA).

Fuzz Testing

Kỹ thuật kiểm thử phần mềm dùng các dữ liệu input invalid, không mong muốn hoặc random - nó là trường hợp đặc biệt của mutation testing. (Fuzz Testing được thực hiện bởi đội test).

Gorilla Testing

Là kỹ thuật mà tester thực hiện test lại phần đã test một vài lần trước đó để đảm bảo tính chắc chắn của hệ thống (Được thực hiện bởi tester hoặc developer).

Glass Box Testing

Nó khá giống với kiểm thử hộp trắng. Nó cũng sử dụng kiến thức về xử lý code bên trong để test. (Được đội developer thực hiện là chủ yếu).

GUI Software Testing

Test giao diện người dùng của sản phẩm để đảm bảo đáp ứng được các yêu cầu. (Thường được thực hiện bởi đội test).

Globalization Testing

Phương pháp kiểm thử kiểm tra các chức năng thích hợp của sản phẩm trong điều kiện thiết lập culture / locale sử dụng tất cả các loại đầu vào quốc tế có thể. (Nó được thực hiện bởi đội test).

Hybrid Integration Testing

Kết hợp 2 kỹ thuật kiểm thử tích hợp từ trên xuống và tích hợp từ dưới lên để tận dụng những ưu điểm của cả 2 kỹ thuật đó. (Thường được thực hiện bởi đội tester).

Install/ Uninstall Testing

- **Kiểm thử cài đặt:** Được thực hiện để xác minh liệu rằng các phần mềm đã được cài đặt với tất cả các thành phần cần thiết và các ứng dụng đang làm việc như mong đợi.
- **Kiểm thử gỡ bỏ cài đặt:** Được thực hiện để xác minh liệu rằng tất cả các thành phần của ứng dụng có bị loại bỏ trong quá trình hay không. Tất cả các tập tin liên quan đến ứng dụng cùng cấu trúc thư mục của nó phải được gỡ bỏ sau khi quá trình gỡ bỏ thành công.

(Được thực hiện bởi các kỹ sư kiểm thử phần mềm).

Internationalization Testing

Quá trình đảm bảo chắc chắn các chức năng của sản phẩm không bị phá vỡ và tất cả các thông điệp được đưa ra ngoài chính xác khi được sử dụng trong các ngôn ngữ và miền địa phương khác nhau. (Được thực hiện bởi đội kiểm thử).

Inter-system Testing

Kỹ thuật kiểm thử tập trung vào kiểm thử các ứng dụng nhằm đảm bảo các kết nối giữa chức năng ứng dụng luôn chính xác. (Thường được thực hiện bởi các nhóm kiểm thử).

Keyword Driven Testing

Keyword Driven Testing (hay còn gọi là as table-driven testing hoặc action word-based testing) là một kỹ thuật lập trình sử dụng các tập tin dữ liệu không chỉ chứa các dữ liệu kiểm thử và kết quả mong đợi, mà còn chứa các từ khóa liên quan đến ứng dụng đang được kiểm thử. (Thực hiện bởi nhóm kiểm thử thủ công và tự động).

Load Testing

Load testing là một quá trình thêm nhu cầu vào một hệ thống hoặc thiết bị và đo lường phản ứng của nó. Load testing được thực hiện để xác định ứng xử của hệ thống trong các điều kiện tải bình thường và cao hơn điều kiện tải dự kiến. (Được thực hiện bởi các kỹ sư hiệu năng).

Localization Testing

Localization testing là quá trình kiểm thử nhằm mục đích kiểm tra các phiên bản địa phương hóa của sản phẩm đặc trưng cho một nền văn hóa hoặc một địa phương cụ thể. Được thực hiện bởi tester. Bạn đọc có thể tham khảo thêm để biết sự khác biệt giữa Globalization và Localization Testing tại link

Loop Testing

- Kỹ thuật kiểm thử hộp trắng tạo ra một chu trình vòng lặp.
- Kiểm tra vòng lặp hoàn toàn tập trung vào tính hiệu lực của các cấu trúc vòng lặp. Nó là một trong những phần kiểm soát kiểm tra cơ cấu (kiểm tra đường dẫn, kiểm tra xác nhận dữ liệu, kiểm tra điều kiện).

(Được thực hiện bởi các kỹ sư phát triển).

Manual Scripted Testing

Kỹ thuật kiểm thử trong đó các test case được thiết kế và xem xét bởi nhóm kiểm thử trước khi thực hiện chúng. (Được hoàn thành bởi nhóm kiểm thử thủ công)

Manual Support Testing

Kiểm thử hỗ trợ thủ công là loại kiểm thử các chức năng hỗ trợ bằng tay. Nó đạt hiệu quả nhất trong giai đoạn cài đặt sản phẩm. (Được thực hiện bởi nhóm kiểm thử).

Model based Testing

Là thế hệ tự động của quy trình kiểm thử phần mềm sử dụng mô hình hành vi và yêu cầu hệ thống. (Được thực hiện bởi nhóm kiểm thử).

Mutation Testing

Kiểm thử hoán chuyển là phương pháp kiểm thử cấu trúc phần mềm nhằm mục đích đánh giá/cải thiện tính đầy đủ của điều kiện test và ước tính số lượng lỗi có thể phát sinh với hệ thống trong quá trình test. (Được thực hiện bởi tester và developer).

Modularity Driven Testing

Kiểm thử mô đun là một framework kiểm thử tự động trong đó các mô đun nhỏ, độc lập của kịch bản tự động hóa được phát triển cho ứng dụng dưới quá trình kiểm thử. (Được thực hiện bởi Tester).

Negative Testing

Để đảm bảo hệ thống chạy trơn tru và ổn định, chúng ta chỉ test những trường hợp bình thường và hợp lệ là chưa đủ. Vì vậy, để đảm bảo hệ thống chạy có thể xử lý được những trường hợp ngoại lệ ta cần test thêm những ngữ cảnh không hợp lệ. Việc test những ngữ cảnh không hợp lệ gọi là negative testing. (Được thực hiện bởi nhóm kiểm thử thủ công hoặc tự động).

Operational Testing

Kiểm thử chấp nhận hoạt động (OAT) là một kỹ thuật kiểm thử được tiến hành nhằm xác nhận sự sẵn sàng hoạt động (trước khi release) của sản phẩm hoặc ứng dụng dưới quá trình test. Kỹ thuật này chủ yếu dựa trên sự sẵn sàng hoạt động của hệ thống sao cho gần giống với môi trường production. (Được thực hiện bởi Tester).

Orthogonal Array Testing

Kiểm thử mảng trực giao là một loại kiểm thử hộp đen - kiểm tra các trường hợp tối ưu hóa kỹ thuật được sử dụng khi hệ thống được thử nghiệm có các dữ liệu đầu vào khổng lồ.

Parallel Testing

Kiểm thử song song là kiểm tra khả năng tương thích của hệ thống mới được phát triển với hệ thống cũ. (Thực hiện bởi Tester).

Path Testing

Kiểm tra đường dẫn là một phương pháp kiểm tra cấu trúc liên quan đến việc sử dụng mã nguồn của một chương trình để tìm ra tất cả các đường dẫn thực thi tốt. (Thực hiện bởi developer).

Penetration Testing

Kiểm thử thâm nhập là một loại kiểm tra an ninh dùng để kiểm tra các khu vực không an toàn của hệ thống hoặc ứng dụng. Phương pháp mà đánh giá sự an toàn của một hệ thống máy tính hoặc mạng bằng cách mô phỏng một cuộc tấn công từ một nguồn độc hại. Tạo ra bởi công ty kiểm thử thâm nhập chuyên ngành.

Qualification Testing

Là loại kiểm thử với mục đích kiểm tra lại các thông số kỹ thuật của phiên bản trước, thường được thực hiện bởi dev cho người tiêu dùng, để chứng minh rằng các phần mềm đáp ứng yêu cầu quy định của nó.

Ramp Testing

Loại thử nghiệm bao gồm việc nâng cao một tín hiệu đầu vào liên tục cho đến khi hệ thống bị phá vỡ. (Có thể được thực hiện bởi tester hoặc developer).

Requirements Testing

Kiểm tra các yêu cầu: xác nhận rằng yêu cầu này là chính xác, đầy đủ, rõ ràng, và hợp lý; phù hợp và cho phép thiết kế một bộ các test case cần và đủ của những yêu cầu. (Được thực hiện bởi đội ngũ QA).

Scenario Testing

Một kịch bản thử nghiệm là một kịch bản được trình bày từ quan điểm người dùng cuối và dựa trên kịch bản này, các thử nghiệm được tiến hành.Trong thử nghiệm kịch bản, tester đặt mình vào người dùng cuối và tìm ra các kịch bản thực tế.

Statement Testing

Là loại kiểm thử hộp trắng đáp ứng các tiêu chí mỗi câu lệnh trong một chương trình được thực hiện ít nhất một lần trong chương trình kiểm thử. (Thường được thực hiện bởi developer).

Scalability Testing

Kiểm tra khả năng mở rộng là kiểm tra hiệu suất hoạt động điều tra khả năng của một hệ thống để phát triển bằng cách tăng khối lượng công việc cho mỗi người dùng, hoặc số lượng người dùng đồng thời, hoặc kích thước của một cơ sở dữ liệu. Là một loại kiểm thử phi chức năng, là loại nhỏ trong kiểm thử hiệu suất.

Static Testing

Là một cách gọi của quá trình review code và các tài liệu của dự án. Đây là hình thức test mà không cần chạy thử phần mềm. Phương pháp này sử dụng giấy, bút để kiểm tra lần lượt từng logic, ngay sau khi lập trình xong. Chủ yếu kiểm tra tính đúng đắn của các dòn code, thuật toán và các tài liệu đặc tả. (Thực hiện bởi Developer, Code Reviewer, Business Analysts).

Stability Testing

Là kỹ thuật test nhằm xác minh sự ổn định của phần mềm qua việc kiểm tra khả năng chạy liên tục của ứng dụng trong hoặc hơn khoảng thời gian có thể chấp nhận được. Kiểm tra xem tại giới hạn nào thì ứng dụng bị crash. Là một phần của performance test, cũng thường được gọi là Load testing hoặc Endurance testing (test khả năng chịu đựng). (Thực hiện bởi performance tester/ engineer).

Smoke Testing

Là kiểu test mở đầu cho quá trình test, được thực hiện ngay khi code được build trên môi trường test. Kiểu này cơ bản giống như kiểu Adhoc testing: kiểm tra xem phần mềm có sẵn sàng cho việc test chưa, có đủ môi trường cho việc test chưa? Kiểm tra đại khái các thành phần cơ bản của hệ thống phần mềm để xem rằng các chức năng chính có bị bất thường hay không? Sau khi test smoke, các tester sẽ thực hiện test khả năng thực hiện của các chức năng. (Thực hiện bởi tester).

Stress Testing

Stress testing là một hình thức kiểm thử được sử dụng để xác định tính ổn định của một hệ thống phần mềm. Là kiểu test kiểm tra thời gian đáp lại người dùng với số lượng người dùng bất kỳ trong nhiều ngữ cảnh khác nhau của cùng một ứng dụng tại cùng một thời điểm. Nó liên quan đến những kiểm thử vượt quá khả năng bình thường của hệ thống, thường để xác định các điểm phá vỡ của hệ thống, để quan sát các kết quả khi vượt qua ngưỡng giới hạn. (Thực hiện bởi các kỹ sư hệ thống, Testers).

Storage Testing

Là loại test nhằm xác minh chương trình qua việc kiểm tra các file dữ liệu có được lưu trữ trong các thư mục chính xác hay không? Kiểm tra xử lý của ứng dụng trong trường hợp thiếu không gian bộ nhớ thì có bị chấm dứt bất ngờ hay không? Ngoài ra còn xác định việc ứng dụng sử dụng bộ nhớ có nhiều hơn so với ước tính hay không, có khả năng gây đầy bộ nhớ và làm tăng thời gian chết hay không? (Thực hiện bởi Tester).

Structural Testing

Là một tên gọi khác của white box testing. Hay thường được gọi là clear box testing, hoặc cũng được biết đến như glass box testing. Được thực hiện bởi developers đã tạo ra code nhằm kiểm tra tính đúng đắn của các dòng code và đảm bảo các xử lý của từng hàm, từng chức năng riêng lẻ được thực hiện đúng theo yêu cầu.

Top Down Integration Testing

Là kỹ thuật test dựa trên việc xây dựng cấu trúc chương trình. Trong đó các module của phần mềm được tích hợp bằng cách di chuyển theo chiều hướng xuống theo trình tự kiểm soát cấp bậc, bắt đầu từ module chính tới các module phụ. Khi tích hợp từ trên xuống thì khung tổng thể của hệ thống được phát triển trước, các chức năng thành phần gắn vào sau.(trái ngược với Bottom up integration: Tích hợp các thành phần có cấu trúc bên dưới trước như: dịch vụ chung, mạng, truy cập cơ sở dữ liệu, sau đó gắn thêm các thành phần chức năng). (Được thực hiện bởi các tester).

Thread Testing

Thread Testing là một kỹ thuật test phần mềm được sử dụng trong giai đoạn test integration sớm để kiểm tra khả năng hoạt động của các chức năng chính. Loại kỹ thuật này rất hữu ích khi test ứng dụng có sử dụng kiến trúc client server. (Được thực hiện bởi các tester).

Upgrade Testing

Được thực hiện khi có phiên bản mới có những tính năng mới được tạo ra dựa trên một phiên bản cũ. Đây là kỹ thuật test nhằm xác minh rằng phiên bản mới được nâng cấp, hoạt động đúng và không thách thức người dùng trong việc học cách sử dụng. (Được thực hiện bởi các tester).

Usability Testing

Là kỹ thuật test xác minh ứng dụng có khả năng ứng dụng cao và dễ sử dụng: người sử dụng có thể học để thao tác, input dữ liệu, giải thích kết quả của hệ thống hoặc các thành phần một cách dễ dàng. Giao diện thân thiện với người dùng: màu sắc, font chữ, size chữ, ngữ pháp câu chữ… (Người thực hiện: end user).

Volume Testing

Volume testing đề cập tới việc kiểm thử phần mềm ứng dụng với một lượng dữ liệu nhất định. Số lượng này có thể là kích thước cơ sở dữ liệu hoặc nó cũng có thể là kích thước của 1 tập tin giao tiếp là đối tượng của volume testing (định nghĩa trong các thuật ngữ chung). (Thực hiện bởi performance engineer/tester).

Vulnerability Testing

Là một loại test bảo mật, có mục đích để ngăn chặn vấn đề có thể ảnh hưởng đến tính toàn vẹn ứng dụng và ổn định. (Được thực hiện bởi tester thông thường hoặc một đơn vị test chuyên biệt).

Workflow Testing

Workflow là một kịch bản mô phỏng tiến trình hoạt động của chức năng nào đó hay nghiệp vụ nào đó được dự kiến sử dụng bởi người dùng cuối. Nó bao gồm nhiều bước thực hiện để đạt được kết quả mong muốn của chức năng đó. Test theo workflow là quá trình đảm bảo mỗi

tiến trình làm việc phản ánh chính xác các business process. Loại test này phù hợp cho các ứng dụng workflow-based applications. (Được thực hiện bởi Tester).

iTMS Youtube Channel: https://www.youtube.com/c/TheiTMSCoaching

iTMS Fanpage: https://www.facebook.com/itms.coaching/

iTMS Radio: https://radio.itmscoaching.com

iTMS Learning: https://khoahoc.itmscoaching.com