**ĐẠI HỌC ĐÀ NẴNG**

**TRƯỜNG ĐẠI HỌC KINH TẾ**

🙣 🕮 🙡

**KHOA THỐNG KÊ – TIN HỌC**

–––––––––––––––––––––––––––––––



**BÁO CÁO THỰC TẬP TỐT NGHIỆP**

**TÊN ĐỀ TÀI**

**Manual Testing websit Customer Portal**

**Đơn vị thực tập: Enclave**

**Giảng viên hướng dẫn:** Ths. Cao Thị Nhâm

**Sinh viên thực hiện:**

**Họ và tên:** Phan Quốc Hưng

**Lớp:** 44K21.1

**Công ty thực tập:** Enclave

**Vị trí thực tập:** Tester

**Đà Nẵng, 9/2022**

**LỜI CẢM ƠN**

Em xin gửi lời cảm ơn chân thành và sự tri ân sâu sắc đối với các thầy cô của trường Đại học Kinh Tế Đà Nẵng, đặc biệt là các thầy cô ở khoa Thống kê-Tin học của trường đã tạo điều kiện cho em thực tập ở Công Ty Phần mêm Enclave. Và em cũng xin chân thành cám ơn cô Cao Thị Nhâm đã nhiệt tình hướng dẫn hướng dẫn em hoàn thành tốt khóa thực tập.

Trong quá trình thực tập và làm bài báo cáo thực tập, do kiến thức cũng như kinh nghiệm thực tế còn nhiều hạn chế nên bài báo cáo không thể tránh khỏi những thiếu sót, em rất mong nhận được ý kiến đóng góp thầy, cô để em học hỏi được nhiều kĩ năng, kinh nghiệm.

Em xin chân thành cảm ơn!

**LỜI CAM ĐOAN**

Em xin cam đoan đây là bài báo cáo thực tập Tester của em trong thời gian qua. Những số liệu và kết quả nghiên cứu là trung thực,hoàn toàn được thực hiện tại Công ty Phần mềm Enclave, không sao chép bất kỳ nguồn nào khác. Ngoài ra, trong bài báo cáo có sử dụng một số nguồn tài liệu tham khảo đã được trích dẫn nguồn và chú thích rõ ràng. Em xin hoàn toàn chịu trách nhiệm trước bộ môn, khoa và nhà trường về sự cam đoan này.

# MỤC LỤC

[MỤC LỤC ii](#_Toc101017017)

[LỜI MỞ ĐẦU 1](#_Toc101017018)

[CHƯƠNG 1. TỔNG QUAN VỀ kiểm thử phầm mềm 3](#_Toc101017019)

[1.1. Tester là gì? 3](#_Toc101017020)

[1.2. Các loại Testing: 3](#_Toc101017021)

[1.3. Quy trình kiểm thử: 4](#_Toc101017022)

[1.3.1. Requirement analysis – Phân tích yêu cầu: 6](#_Toc101017023)

[1.3.2. Test planning – Lập kế hoạch kiểm thử: 6](#_Toc101017024)

[1.3.3. Test case development – Thiết kế kịch bản cho quy trình kiểm thử: 7](#_Toc101017025)

[1.3.4. Test environment set up – Thiết lập môi trường kiểm thử: 7](#_Toc101017026)

[1.3.5. Test execution – Thực hiện kiểm thử: 7](#_Toc101017027)

[1.3.6. Test cycle closure – Đóng chu trình kiểm thử: 8](#_Toc101017028)

[CHƯƠNG 2. Manual Testing 9](#_Toc101017029)

[2.1. Các kỹ thuật về testing: 9](#_Toc101017030)

[2.1.1. Blackbox testing: 9](#_Toc101017031)

[2.1.2. Whitebox testing: 9](#_Toc101017032)

[2.2. Tài liệu kiểm thử: 11](#_Toc101017033)

[2.3. Test case: 12](#_Toc101017034)

[2.4. Bug, Bug life cycle: 13](#_Toc101017035)

[2.5. Test Report: 15](#_Toc101017037)

[CHƯƠNG 3. kiểm thử một số chức năng của website Customer Portal 20](#_Toc101017041)

[3.1. Research requiredment của website: 20](#_Toc101017042)

[3.2. Thiết kế test cases cho quá trình kiểm thử: 25](#_Toc101017043)

[3.3. Triển khai kiểm thử các chức năng của website Customer Portal: 28](#_Toc101017044)

[CHƯƠNG 4. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN 33](#_Toc101017047)

[TÀI LIỆU THAM KHẢO 35](#_Toc101017048)

# Danh sách sinh viên thực hiện

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **STT** | **Họ và tên** | **Lớp** | **Nhiệm vụ** | **DVTT** |
| 1 | Phan Quốc Hưng | 44K21.1 | Tester | Enclave |

# LỜI MỞ ĐẦU

1. **Giới thiệu chung về Enclave:**

**Enclave** được thành lập năm 2007, một công ty của các chuyên gia kỹ thuật phần mềm. Chúng tôi đã cung cấp chất lượng nổi bật cho các dịch vụ kiểm tra phần mềm và kỹ thuật phần mềm kể từ năm 2007. Dựa trên các tính năng yêu cầu thu thập từ nhiều tên tuổi lớn trong CNTT và ITO Industries, chúng tôi đã tạo ra môi trường làm việc sáng tạo và các giải pháp hiệu quả hiện có sẵn cho tất cả các công ty được kích hoạt.

Với một nhóm các kỹ sư CNTT có kinh nghiệm trong việc phát triển, thiết kế và kỹ thuật SQA, chúng tôi đáng tin cậy cung cấp cho khách hàng các giải pháp giá trị gia tăng, kỹ năng chuyên nghiệp, trách nhiệm và kiến ​​thức lĩnh vực công nghiệp để giảm chi phí vận hành, loại bỏ rủi ro, cung cấp giải pháp đúng hạn và ngân sách cũng như đảm bảo quyết định đúng đắn đã được đưa ra như là giải pháp và dịch vụ tốt nhất của chúng tôi.

1. **Mục tiêu nghiên cứu của đề tài**

* Đề tài này nghiên cứu về công việc Testing.
* Hiện nay cùng sự phát triển mạnh mẽ cũng như bước chuyển mình nhanh chóng của các xu thế công nghệ thông tin trên thế giới đã mang lại cho Việt Nam đồng thời thuận lợi và khó khăn. Do đó, những dự án, những chương trình quốc gia nhằm thúc đẩy hiệu quả ứng dụng CNTT trong mọi mặt đời sống kinh tế chính trị xa hội đang ngày càng được chú trọng và triển khai. Kéo theo đó là nhu cầu về sử dụng về lĩnh vực kiểm thử, đặc biệt là kiểm thử phần mềm tự động. Kiểm thử phần mềm là một thành phần quan trọng trong qui trình phát triển phần mềm. Nó đóng vào trò quan trọng trong việc kiểm định chất lượng của phần mềm, đảm bảo phần mềm tạo ra có chạy đúng với yêu cầu của khách hàng hay không, có xảy ra những sao sót nó khác với bảng phân tích thiết kế ban đầu không.

1. **Nhiệm vụ của đề tài**

* Nghiên cứu về Tester là gì, các loại testing. Tìm hiểu và cách sử dụng các phần mềm và công cụ dùng trong lĩnh vực testing.
* Nâng cao kiến thức và kinh nghiệm trong lĩnh vực testing và làm việc như một tester với các công việc công ty Enclave đã giao và trainning.

1. **Phương pháp nghiên cứu**

* Nghiên cứu với kiến thức và kinh nghiệm làm việc đã được trainning trong quá trình thực tập tại Enclave.

# TỔNG QUAN VỀ kiểm thử phầm mềm

## Tester là gì?

Người kiểm thử là người thực hiện kiểm tra trên phần mềm / phần cứng được phát triển để đảm bảo rằng sản phẩm chất lượng cao, đáp ứng các chức năng tiềm ẩn & rõ ràng đã nêu trước khi được giao cho khách hàng cuối cùng. Người kiểm thử đạt được điều này bằng cách áp dụng các phương pháp kiểm tra nghiêm ngặt để xác định các lỗi / lỗi sau đó được nhóm phát triển loại bỏ.

## Các loại Testing:

1. **Manual testing? (Kiểm thử thủ công)**

Kiểm thử thủ công là quá trình kiểm thử phần mềm thủ công để tìm sai sót. Nó yêu cầu một kiểm thử viên đóng vai trò người dùng cuối, theo đó họ sử dụng hầu hết tính năng của ứng dụng để bảo đảm hành vi đúng.

1. **Automation testing? (Kiểm thử tự động)**

Trong kiểm thử phần mềm, kiểm thử tự động là việc sử dụng phần mềm đặc biệt để kiểm soát việc thực hiện các bài kiểm tra và so kết quả thực tế với kết quả dự đoán.

1. **Performance testing? (Kiểm thử hiệu năng)**

Trong bảo đảm chất lượng phần mềm, kiểm thử hiệu năng nói chung là một thực hành kiểm thử được thực hiện để xác định cách hệ thống thực hiện theo mức độ đáp ứng và độ ổn định trong một khối lượng công việc cụ thể.

1. **Security testing? (Kiểm thử bảo mật)**

Kiểm thử bảo mật là một quá trình nhằm tiết lộ các lỗ hổng trong cơ chế bảo mật của hệ thống thông tin nhằm bảo vệ dữ liệu và duy trì chức năng như dự kiến.

1. **Integration testing? (Kiểm thử tích hợp)**

Kiểm thử tích hợp thỉnh thoảng còn gọi là tích hợp và kiểm thử là một giai đoạn trong kiểm thử phần mềm mà mỗi môđun phần mềm riêng biệt được kết hợp lại và thử nghiệm theo nhóm. Nó xảy ra sau kiểm thử đơn vị và trước kiểm thử xác nhận.

1. **Stress test? (Kiểm thử tải)**

Là một loại của Performance Test

Stress Test là một loại kiểm thử xác định sự ổn định và tính mạnh mẽ của hệ thống. Đây là một kỹ thuật kiểm thử không chức năng.

1. **Load test? (Kiểm thử tải)**

Là một loại của Performance Test

Load Test là quá trình mô phỏng độ chịu tải thực tế của bất kỳ ứng dụng hoặc trang web nào. Nó kiểm thử cách ứng dụng hoạt động trong điều kiện hoạt động bình thường và hoạt động hiệu suất cao.

## Quy trình kiểm thử:

**6 giai đoạn cơ bản trong quy trình kiểm thử phần mềm**

1. Requirement analysis – Phân tích yêu cầu.
2. Test planning – Lập kế hoạch **kiểm thử**
3. Test case development – Thiết kế kịch bản cho **quy trình kiểm thử**
4. Test environment set up – Thiết lập môi trường **kiểm thử**
5. Test execution – Thực hiện **kiểm thử**
6. Test cycle closure – Đóng chu trình kiểm thử

### Requirement analysis – Phân tích yêu cầu:

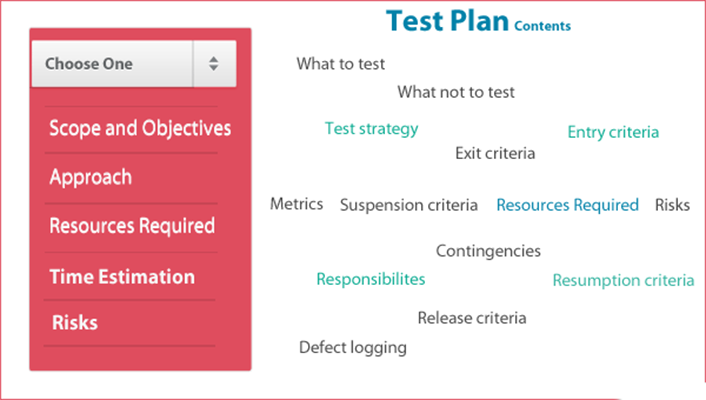
Giai đoạn đầu tiên của quy trình kiểm thử là phân tích các yêu cầu thông qua những tài liệu bao gồm: tài liệu yêu cầu của khách hàng, prototype của khách hàng, tài liệu đặc tả yêu cầu của phần mềm, tài liệu thiết kế hệ thống…

QA team có nhiệm vụ phân tích và xác định những yêu cầu của khách hàng, trong đó có yêu cầu về kiểm thử chức năng/phi chức năng của phần mềm. Trong quá trình phân tích, QA team có thể đặt ra câu hỏi để hiểu chính xác hơn về yêu cầu của sản phẩm, đồng thời hỗ trợ đưa ra giải pháp thích hợp cho khách hàng.

### Test planning – Lập kế hoạch kiểm thử:

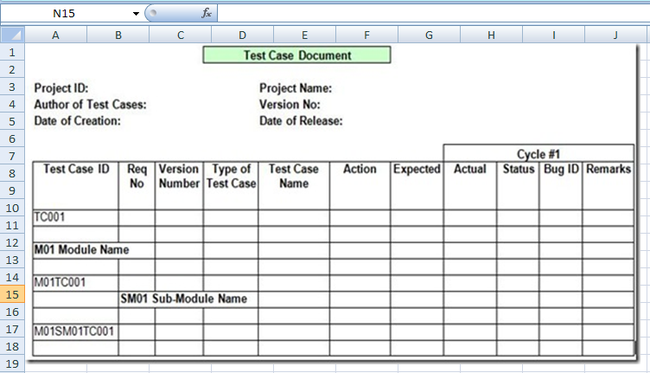
Dựa vào tài liệu nhận được trong giai đoạn đầu, Test Lead hoặc Test Manager sẽ lên kế hoạch kiểm thử phần mềm cho QA team để xác định một số yếu tố:

* Phạm vi dự án: Thời gian thực hiện dự án bao lâu? Trong từng khoảng thời gian sẽ có những công việc gì?
* Phương pháp tiếp cận: Dựa vào yêu cầu chất lượng của khách hàng, thời gian test, kỹ thuật phát triển ứng dụng, lĩnh vực của sản phẩm… Test Manager sẽ đưa ra phương pháp tiếp cận sao cho đảm bảo tiến độ và chất lượng sản phẩm. Sau khi kết thúc giai đoạn này, QA team cần nhận được test plan, test schedule, test estimation.

Lên kế hoạch kiểm thử

### Test case development – Thiết kế kịch bản cho quy trình kiểm thử:

Trong giai đoạn này, các Tester sẽ đọc hiểu tất cả các tài liệu, từ đó xác định những việc cần làm, chức năng nào cần test hoặc không. Sau đó, dựa vào kế hoạch và kỹ thuật thiết kế kịch bản kiểm thử, Tester sẽ bắt đầu viết test case. Yêu cầu của test case: Thể hiện tất cả các trường hợp kiểm thử có thể phát sinh để đáp ứng yêu cầu sản phẩm. Ngoài test case, Tester cũng cần chuẩn bị các dữ liệu cần thiết khác như test data, test script, test design, test automation script.

Thiết kế kịch bản cho quy trình kiểm thử

### Test environment set up – Thiết lập môi trường kiểm thử:

Đây là một trong những giai đoạn đóng vai trò rất quan trọng trong Software Testing Life Cycle (vòng đời phát triển phần mềm). Dựa trên yêu cầu khách hàng và đặc thù của sản phẩm, môi trường kiểm thử sẽ được xác định. Tester cần chuẩn bị smoke test case để kiểm tra môi trường cài đặt đã đáp ứng yêu cầu và sẵn sàng cho giai đoạn kiểm thử tiếp theo hay chưa.

### Test execution – Thực hiện kiểm thử:

Theo test case đã thiết kế và môi trường kiểm thử đã hoàn tất cài đặt, Tester sẽ báo cáo bug lên tool quản lý lỗi và theo dõi đến khi fix bug thành công. Tiếp đó, Tester thực hiện retest để verify các fix bug và regression test trong trường hợp có sự thay đổi. Sau khi hoàn tất giai đoạn này, các chuyên viên kiểm thử cần có được test results (kết quả kiểm thử) và defect reports (danh sách các lỗi tìm được).

### Test cycle closure – Đóng chu trình kiểm thử:

Để đóng chu trình kiểm thử, QA team cần có được những tài liệu đã được tổng hợp và hoàn thiện từ những giai đoạn trước: tài liệu phân tích đặc tả yêu cầu, test plan, defect reports, test results… Tiếp đó, QA team sẽ tổng kết, báo cáo về quá trình kiểm thử, có bao nhiêu bug đã được fix, bug có nghiêm trọng hay không, chức năng nào còn lỗi, chức năng nào đã hoàn thành…

# Manual TEsting

## Các kỹ thuật về testing:

### Blackbox Testing:

Kiểm thử hộp đen là một kỹ thuật kiểm thử phần mềm kiểm tra chức năng của phần mềm mà không cần nhìn vào cấu trúc bên trong hoặc mã hóa của nó. Nguồn chính của thử nghiệm hộp đen là một đặc điểm kỹ thuật của các yêu cầu được nêu bởi khách hàng. Trong phương pháp này, người thử nghiệm chọn một chức năng và đưa ra giá trị đầu vào để kiểm tra chức năng của nó và kiểm tra xem chức năng có đưa ra đầu ra dự kiến hay không. Nếu chức năng tạo ra đầu ra chính xác, sau đó nó được vượt qua trong thử nghiệm, nếu không sẽ thất bại. Nhóm kiểm tra báo cáo kết quả cho nhóm phát triển và sau đó kiểm tra chức năng tiếp theo. Sau khi hoàn thành kiểm tra tất cả các chức năng nếu có vấn đề nghiêm trọng, sau đó nó được trao lại cho nhóm phát triển để sửa chữa.

##### a. Phân vùng tương đương:

Phân vùng tương đương là một kỹ thuật kiểm tra phần mềm trong đó dữ liệu đầu vào được chia thành các phân vùng của các giá trị hợp lệ và không hợp lệ và bắt buộc tất cả các phân vùng phải thể hiện cùng một hành vi. Nếu một điều kiện của một phân vùng là đúng, thì điều kiện của một phân vùng bằng nhau cũng phải là đúng, và nếu một điều kiện của một phân vùng là sai, thì điều kiện của một phân vùng bằng nhau cũng phải là sai. Nguyên tắc phân vùng tương đương là, các trường hợp thử nghiệm nên được thiết kế để bao phủ mỗi phân vùng ít nhất một lần. Mỗi giá trị của mỗi phân vùng bằng nhau phải thể hiện hành vi tương tự như những giá trị khác. Các phân vùng tương đương được lấy từ các yêu cầu và thông số kỹ thuật của phần mềm. Ưu điểm của cách tiếp cận này là, nó giúp giảm thời gian kiểm tra do một số lượng nhỏ hơn các trường hợp kiểm tra từ vô hạn đến hữu hạn. Nó được áp dụng ở tất cả các cấp độ của quá trình thử nghiệm. Ví dụ, sau khi nhập tên người dùng và mật khẩu, chúng ta cũng cần nhập OTP để đăng nhập, OTP phải có sáu chữ số vì vậy bên dưới là các lớp tương đương.

* Invalid: alphabet/ special character
* Invalid: digits >=7
* Valid: Digits =6
* Invalid: Digits<=5
* Invalid: blank space
* Invalid: Expired OTP
* Invalid: valid format but invalid value

##### b. Phân tích giá trị ranh giới:

Phân tích giá trị ranh giới là một trong những kỹ thuật thiết kế trường hợp được sử dụng rộng rãi để kiểm tra hộp đen. Nó được sử dụng để kiểm tra các giá trị ranh giới vì các giá trị đầu vào gần ranh giới có cơ hội sai số cao hơn. Nó bao gồm tối đa, tối thiểu, ranh giới bên trong hoặc bên ngoài, giá trị điển hình và giá trị lỗi. Bất cứ khi nào chúng tôi thực hiện thử nghiệm bằng cách phân tích giá trị biên, người thử nghiệm tập trung vào, trong khi nhập giá trị biên, cho dù phần mềm có tạo ra đầu ra chính xác hay không. Ví dụ: đối với OTP yêu cầu phải có sáu chữ số, giá trị ranh giới là:

* Digits >=7: 6789045, 7890904643
* Digits =6: 651904
* Digits<=5: 67054

##### c. Kiểm tra bảng quyết định:

Kỹ thuật bảng quyết định là một trong những kỹ thuật thiết kế trường hợp được sử dụng rộng rãi để thử nghiệm hộp đen. Đây là một cách tiếp cận có hệ thống trong đó các kết hợp đầu vào khác nhau và hành vi hệ thống tương ứng của chúng được nắm bắt ở dạng bảng. Đó là lý do tại sao nó cũng được biết đến như là một bảng hiệu quả nguyên nhân. Kỹ thuật này được sử dụng để chọn các ca kiểm thử một cách có hệ thống; giúp tiết kiệm thời gian kiểm thử và cung cấp độ bao phủ tốt cho khu vực kiểm thử của ứng dụng phần mềm. Kỹ thuật bảng quyết định thích hợp cho các chức năng có mối quan hệ logic giữa hai và nhiều hơn hai đầu vào. Kỹ thuật này liên quan đến sự kết hợp chính xác của đầu vào và xác định kết quả của các kết hợp đầu vào khác nhau. Để thiết kế các trường hợp thử nghiệm theo kỹ thuật bảng quyết định, chúng ta cần xem xét các điều kiện như đầu vào và hành động như đầu ra. Dưới đây là ví dụ để tạo Bảng Cơ sở Quyết định cho Màn hình Đăng nhập. Điều kiện rất đơn giản nếu người dùng cung cấp đúng tên người dùng và mật khẩu, người dùng sẽ được chuyển hướng đến trang chủ. Nếu bất kỳ thông tin nhập sai nào, thông báo sẽ hiển thị.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Conditions | Rule 1 | Rule 2 | Rule 3 | Rule 4 |
| Username (T/F) | F | T | F | T |
| Password (T/F) | F | F | T | T |
| Output (E/H) | E | E | E | H |

* T – Correct username/password
* F – Wrong username/password
* E – Error message is displayed
* H – Home screen is displayed

##### d. Use-case dựa vào kiểm thử:

Trường hợp sử dụng là kiểm tra chức năng của kiểm tra hộp đen được sử dụng để xác định các trường hợp kiểm tra từ đầu đến cuối của hệ thống theo cách sử dụng của hệ thống. Bằng cách sử dụng kỹ thuật này, nhóm thử nghiệm tạo ra một kịch bản thử nghiệm có thể thực hiện toàn bộ phần mềm dựa trên chức năng của từng chức năng từ đầu đến cuối. Nó là một minh họa đồ họa về nhu cầu kinh doanh, mô tả cách người dùng cuối sẽ hợp tác với phần mềm hoặc ứng dụng. Các trường hợp sử dụng cung cấp cho chúng tôi tất cả các kỹ thuật có thể về cách người dùng cuối sử dụng ứng dụng. ​​Ví dụ, dưới đây là sơ đồ sử dụng cho ATM ngân hàng



Dựa trên trường hợp sử dụng này, chúng tôi có các trường hợp dưới đây trong ATM ngân hàng:

* + Khách hàng kiểm tra số dư
  + Tiền gửi của khách hàng
  + Khách hàng rút tiền mặt
  + Khách hàng chuyển tiền
  + Kỹ thuật viên bảo trì ATM ngân hàng
  + Kỹ thuật viên ATM sửa chữa ATM ngân hàng.
  + Kiểm tra số dư ngân hàng
  + Tiền gửi ngân hàng
  + Ngân hàng rút tiền mặt
  + Chuyển khoản ngân hàng
  + Bảo trì ngân hàng ATM ngân hàng
  + Ngân hàng sửa chữa ATM ngân hàng

##### e. Kiểm tra đoán lỗi:

Đoán lỗi là một kỹ thuật trong đó không có phương pháp cụ thể để xác định lỗi. Nó dựa trên kinh nghiệm của chuyên gia phân tích kiểm thử, trong đó người kiểm thử sử dụng kinh nghiệm để đoán các khu vực có vấn đề của phần mềm. Đây là một loại kỹ thuật kiểm tra hộp đen không có bất kỳ cấu trúc xác định nào để tìm lỗi. Trong phương pháp này, mỗi kỹ sư thử nghiệm sẽ thu được các giá trị hoặc đầu vào dựa trên sự hiểu biết hoặc giả định của họ về các yêu cầu, và chúng tôi không tuân theo bất kỳ loại quy tắc nào để thực hiện các kỹ thuật đoán lỗi. Việc hoàn thành kỹ thuật đoán lỗi phụ thuộc vào khả năng và kiến thức sản phẩm của người thử nghiệm vì một kỹ sư thử nghiệm giỏi biết được lỗi ở đâu, giúp tiết kiệm rất nhiều thời gian.

### Whitebox Testing:

White box testing là kỹ thuật phần mềm trong đó cấu trúc bên trong, thiết kế và mã hóa của phần mềm được kiểm tra để xác minh luồng đầu vào, đầu ra và để cải thiện thiết kế, khả năng sử dụng và bảo mật. Trong thử nghiệm hộp trắng, người thử nghiệm có thể nhìn thấy mã nên nó còn được gọi là Thử nghiệm hộp trong, Thử nghiệm hộp mở, Thử nghiệm hộp trong suốt, Thử nghiệm dựa trên mã và Thử nghiệm hộp thủy tinh.

##### a. Kiểm tra luồng dữ liệu:

Kiểm thử luồng dữ liệu là một chiến lược cụ thể của kiểm thử phần mềm tập trung vào các biến dữ liệu và các giá trị của chúng. Nó được sử dụng để phân tích luồng dữ liệu trong chương trình. Đó là quá trình thu thập thông tin về cách các biến lưu chuyển dữ liệu trong chương trình. Nó cố gắng để có được thông tin cụ thể của từng điểm cụ thể trong quá trình. Nó được thực hiện để bao gồm các thử nghiệm PATH và khoảng cách thử nghiệm nhánh.

##### b. Kiểm tra mức độ bao phủ:

Phạm vi tuyên bố: Phạm vi tuyên bố là một kỹ thuật kiểm tra hộp trắng, liên quan đến việc thực hiện tất cả các tuyên bố ít nhất một lần trong mã nguồn. Đây là một số liệu, được sử dụng để tính toán và đo lường số lượng tuyên bố trong mã nguồn đã được thực thi. Bảo hiểmchi nhánh: Bảo hiểm chi nhánh là yêu cầu, đối với mỗi chi nhánh trong chương trình (ví dụ: nếu có tuyên bố, vòng lặp (trong khi, cho)), mỗi chi nhánh đã được thực hiện ít nhất một lần trong quá trình thử nghiệm.

## Tài liệu kiểm thử:

### Các kịch bản kiểm thử:

Đây là một tài liệu xác định nhiều cách hoặc kết hợp thử nghiệm ứng dụng. Nói chung, nó được chuẩn bị để hiểu dòng chảy của một ứng dụng. Nó không bao gồm bất kỳ đầu vào và các bước điều hướng nào.

### Kế hoạch kiểm thử

Đây là một tài liệu được chuẩn bị bởi các nhà quản lý hoặc trưởng nhóm kiểm tra. Nó bao gồm tất cả thông tin về các hoạt động thử nghiệm. Kế hoạch kiểm tra bao gồm nhiều thành phần như Mục tiêu, Phạm vi, Phương pháp tiếp cận, Môi trường kiểm tra, Phương pháp kiểm tra, Mẫu, Vai trò & Trách nhiệm, Ước tính nỗ lực, Tiêu chí Nhập và Xuất cảnh, Lịch trình, Công cụ, Theo dõi lỗi, Kiểm tra có thể phân phối, Giả định, Rủi ro và Kế hoạch giảm thiểu hoặc Kế hoạch dự phòng.

### Ma trận truy xuất yêu cầu (RTM):

Ma trận xác định nguồn gốc yêu cầu [RTM] là một tài liệu đảm bảo rằng tất cả các trường hợp thử nghiệm đã được đề cập. Tài liệu này được tạo trước quá trình thực thi thử nghiệm để xác minh rằng chúng tôi không bỏ sót việc viết bất kỳ trường hợp thử nghiệm nào cho yêu cầu cụ thể.

### Dữ liệu kiểm thử:

Nó là dữ liệu xảy ra trước khi kiểm tra được thực hiện. Nó chủ yếu được sử dụng khi chúng tôi đang triển khai trường hợp thử nghiệm. Hầu hết, chúng tôi sẽ có dữ liệu thử nghiệm ở định dạng trang tính Excel và được nhập theo cách thủ công trong khi thực hiện trường hợp thử nghiệm. Dữ liệu thử nghiệm có thể được sử dụng để kiểm tra kết quả mong đợi, có nghĩa là khi dữ liệu thử nghiệm được nhập, kết quả mong đợi sẽ đáp ứng kết quả thực tế và cũng kiểm tra hiệu suất ứng dụng bằng cách nhập dữ liệu đầu vào đúng.

### Báo cáo Bugs/ Danh sách theo dõi bugs:

Báo cáo lỗi là một tài liệu mà chúng tôi lưu giữ một bản tóm tắt về tất cả các lỗi đã xảy ra trong quá trình thử nghiệm. Đây là một tài liệu quan trọng cho cả nhà phát triển và kỹ sư kiểm tra vì, với sự trợ giúp của báo cáo lỗi, họ có thể dễ dàng theo dõi các lỗi, báo cáo lỗi, thay đổi trạng thái của các lỗi đã được sửa thành công và cũng tránh lặp lại chúng trong quá trình tiếp theo .

### Báo cáo thực hiện kiểm thử:

Đây là tài liệu được chuẩn bị bởi các khách hàng tiềm năng sau khi toàn bộ quá trình thực thi thử nghiệm được hoàn thành. Báo cáo tóm tắt thử nghiệm xác định tính ổn định của sản phẩm và nó chứa thông tin như các mô-đun, số lượng các trường hợp thử nghiệm đã viết, đã thực hiện, đạt, không đạt và tỷ lệ phần trăm của chúng. Và mỗi mô-đun có một bảng tính riêng của mô-đun tương ứng của chúng

## Test case:

### Thế nào là test case?

Test case là một tập hợp các hành động được thực hiện trên một hệ thống để xác định xem nó có đáp ứng các yêu cầu và chức năng phần mềm một cách chính xác hay không. Mục đích của trường hợp thử nghiệm là để xác định xem các tính năng khác nhau trong hệ thống có đang hoạt động như mong đợi hay không và để xác nhận rằng hệ thống đáp ứng tất cả các tiêu chuẩn, hướng dẫn liên quan và các yêu cầu của khách hàng. Quá trình viết một trường hợp kiểm thử cũng có thể giúp tiết lộ các lỗi hoặc khiếm khuyết trong hệ thống.

### Các thành phần của một trường hợp kiểm thử bao gồm:

**Test name**: Tiêu đề mô tả chức năng hoặc tính năng mà thử nghiệm đang xác minh.

**Test ID:** Thông thường là mã nhận dạng số hoặc chữ và số mà các kỹ sư QA và người thử nghiệm sử dụng để nhóm các trường hợp thử nghiệm thành các bộ thử nghiệm.

**Objective:** Còn được gọi là mô tả, thành phần quan trọng này mô tả những gì mà bài kiểm tra dự định xác minh trong một đến hai câu.

**References:** Liên kết đến User Story, thông số kỹ thuật thiết kế hoặc các yêu cầu mà thử nghiệm dự kiến sẽ xác minh.

**Prerequisites:** Bất kỳ điều kiện nào cần thiết để người thử nghiệm hoặc kỹ sư QA thực hiện thử nghiệm.

**Test setup:** Thành phần này xác định những gì trường hợp thử nghiệm cần để chạy chính xác, chẳng hạn như phiên bản ứng dụng, hệ điều hành, yêu cầu ngày giờ và thông số kỹ thuật bảo mật.

**Test steps:** Mô tả chi tiết các hành động tuần tự phải thực hiện để hoàn thành kiểm tra.

**Expected results:** Bản phác thảo về cách hệ thống sẽ phản ứng với mỗi bước kiểm tra.

### Một test case hiệu quả là gì?

* Chính xác, hoặc cụ thể về mục đích.
* Tiết kiệm, nghĩa là không sử dụng các bước hoặc từ ngữ không cần thiết.
* Có thể truy nguyên, nghĩa là các yêu cầu có thể được truy nguyên.
* Có thể lặp lại, nghĩa là tài liệu có thể được sử dụng để thực hiện thử nghiệm nhiều lần.
* Có thể tái sử dụng, nghĩa là tài liệu có thể được sử dụng lại để thực hiện lại thành công kiểm tra trong tương lai.
* Để đạt được những mục tiêu này, các kỹ sư kiểm tra và chất lượng có thể sử dụng các phương pháp hay nhất sau:
  + Ưu tiên những trường hợp kiểm thử nào cần viết dựa trên các mốc thời gian của dự án và các yếu tố rủi ro của hệ thống hoặc ứng dụng.
  + Tạo các trường hợp thử nghiệm duy nhất và tránh các trường hợp thử nghiệm không liên quan hoặc trùng lặp.
  + Xác nhận rằng bộ kiểm tra kiểm tra tất cả các yêu cầu cụ thể được đề cập trong tài liệu đặc điểm kỹ thuật.
  + Viết các trường hợp kiểm thử một cách minh bạch và dễ hiểu. Tiêu đề của mỗi trường hợp kiểm thử phải ngắn gọn.
  + Các bước test case nên được chia thành các đoạn nhỏ nhất có thể để tránh nhầm lẫn khi thực thi.
  + Các trường hợp kiểm thử nên được viết theo cách cho phép người khác dễ dàng hiểu chúng và sửa đổi tài liệu khi cần thiết.
  + Lưu ý người dùng cuối bất cứ khi nào một trường hợp thử nghiệm được tạo.
  + Không giả định các tính năng và chức năng của hệ thống.
  + Mỗi trường hợp kiểm thử phải được xác định dễ dàng.
  + Mô tả phải rõ ràng và ngắn gọn.

### Test suit:

Test suit là một vùng chứa có một tập hợp các thử nghiệm giúp người thử nghiệm thực thi và báo cáo trạng thái thực thi thử nghiệm

Một trường hợp thử nghiệm có thể được thêm vào nhiều bộ thử nghiệm và kế hoạch thử nghiệm. Sau khi tạo kế hoạch thử nghiệm, các bộ thử nghiệm được tạo ra, từ đó có thể có bất kỳ số lượng thử nghiệm nào. Các bộ thử nghiệm được tạo dựa trên chu kỳ hoặc dựa trên phạm vi.

Ví dụ: chúng tôi có thể có bộ kiểm tra hồi quy hoặc chúng tôi có thể có test suit Thêm khách hàng mới.

### Các loại test case:

##### Các test case kiểm tra chức năng:

#### Đây là một loại kiểm tra hộp đen có thể tiết lộ xem giao diện của ứng dụng có hoạt động với phần còn lại của hệ thống và người dùng hay không bằng cách xác định xem các chức năng mà phần mềm dự kiến thực hiện là thành công hay thất bại. Các trường hợp kiểm thử chức năng dựa trên các đặc tả hệ thống hoặc câu chuyện người dùng, cho phép thực hiện các bài kiểm tra mà không cần truy cập vào cấu trúc bên trong của phần mềm. Trường hợp thử nghiệm này thường được viết bởi nhóm QA.

##### Các test case kiểm ra hiệu năng:

Các trường hợp thử nghiệm này có thể giúp xác nhận thời gian phản hồi và xác nhận hiệu quả tổng thể của hệ thống. Các trường hợp kiểm tra hiệu suất bao gồm một bộ tiêu chí thành công rất nghiêm ngặt và có thể được sử dụng để hiểu cách hệ thống sẽ hoạt động trong thế giới thực. Các trường hợp kiểm thử hiệu suất thường được viết bởi nhóm kiểm thử, nhưng chúng thường được tự động hóa vì một hệ thống có thể yêu cầu hàng trăm nghìn bài kiểm tra hiệu suất.

##### Các test case kiểm thử đơn vị:

Kiểm thử đơn vị liên quan đến việc phân tích các đơn vị hoặc thành phần riêng lẻ của phần mềm để xác nhận từng đơn vị hoạt động như mong đợi. Đơn vị là phần tử nhỏ nhất có thể kiểm tra được của phần mềm. Thường cần một vài đầu vào để tạo ra một đầu ra duy nhất.

##### Các test case kiểm tra giao diện người dùng:

Loại trường hợp thử nghiệm này có thể xác minh rằng phần tử cụ thể của giao diện người dùng đồ họa (GUI) trông và hoạt động như mong đợi. Các trường hợp kiểm tra giao diện người dùng có thể tiết lộ lỗi trong các yếu tố mà người dùng tương tác, chẳng hạn như lỗi ngữ pháp và chính tả, liên kết bị hỏng và sự không nhất quán về thẩm mỹ. Kiểm tra giao diện người dùng thường yêu cầu chức năng trên nhiều trình duyệt để đảm bảo ứng dụng hoạt động nhất quán trên các trình duyệt khác nhau. Các trường hợp kiểm thử này thường được viết bởi nhóm kiểm thử với một số trợ giúp từ nhóm thiết kế.

##### Các test case kiểm tra bảo mật:

Các trường hợp kiểm tra này được sử dụng để xác nhận rằng hệ thống hạn chế các hành động và quyền khi cần thiết để bảo vệ dữ liệu. Các trường hợp thử nghiệm bảo mật thường tập trung vào xác thực và mã hóa và thường xuyên sử dụng các thử nghiệm dựa trên bảo mật, chẳng hạn như thử nghiệm thâm nhập. Nhóm bảo mật chịu trách nhiệm viết các trường hợp thử nghiệm này - nếu một trường hợp tồn tại trong tổ chức.

##### Các test case kiểm tra tích hợp:

Một trường hợp kiểm thử tích hợp được viết để xác định cách các mô-đun phần mềm khác nhau tương tác với nhau. Mục đích chính của trường hợp thử nghiệm này là để xác nhận rằng các giao diện giữa các mô-đun khác nhau hoạt động chính xác. Các trường hợp kiểm thử tích hợp thường được viết bởi nhóm kiểm thử, với đầu vào do nhóm phát triển cung cấp.

##### Các test case kiểm tra cơ sở dữ liệu:

Loại trường hợp kiểm thử này nhằm mục đích kiểm tra những gì đang xảy ra bên trong, giúp người kiểm tra hiểu được dữ liệu đang đi đến đâu trong hệ thống. Các nhóm kiểm thử thường sử dụng các truy vấn SQL để viết các trường hợp kiểm thử cơ sở dữ liệu.

##### Các test case kiểm tra khả năng sử dụng:

Một trường hợp kiểm tra khả năng sử dụng có thể được sử dụng để tiết lộ cách người dùng tiếp cận và sử dụng ứng dụng một cách tự nhiên. Thay vì cung cấp chi tiết từng bước, một trường hợp kiểm tra khả năng sử dụng sẽ cung cấp cho người kiểm tra một kịch bản hoặc nhiệm vụ cấp cao cần hoàn thành. Các trường hợp thử nghiệm này thường được viết bởi nhóm thiết kế và thử nghiệm và phải được thực hiện trước khi thử nghiệm chấp nhận của người dùng.

##### Các test case kiểm tra hồi quy:

Kiểm tra này xác nhận các thay đổi mã hoặc chương trình gần đây không ảnh hưởng đến các tính năng hiện có của hệ thống. Kiểm thử hồi quy bao gồm việc chọn tất cả hoặc một số trường hợp kiểm thử đã thực thi và chạy lại chúng để xác nhận các chức năng hiện có của phần mềm vẫn hoạt động một cách thích hợp.

## Bug, bug life cycle:

### Thế nào là Bug?

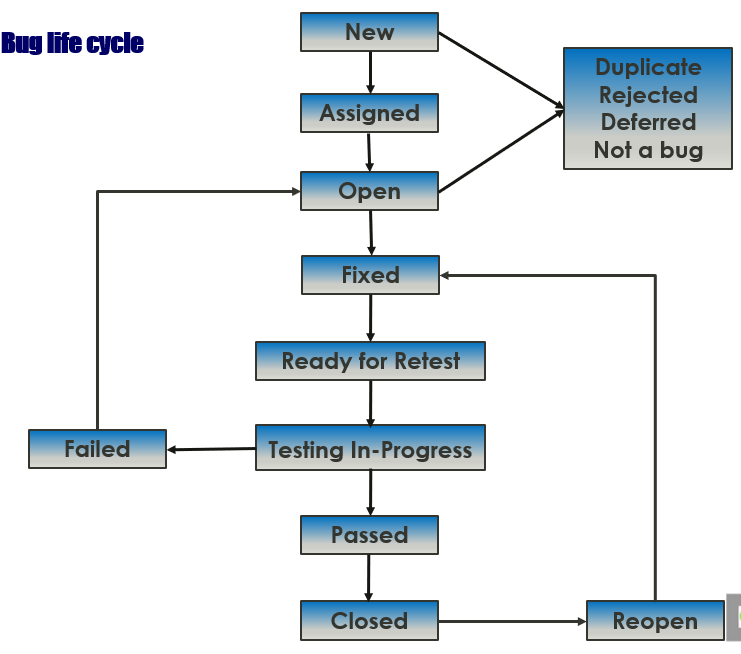
Lỗi là tên không chính thức của các lỗi, có nghĩa là phần mềm hoặc ứng dụng không hoạt động theo yêu cầu. Trong kiểm thử phần mềm, một lỗi phần mềm cũng có thể là sự cố, lỗi, lỗi hoặc hỏng. Lỗi xảy ra khi các nhà phát triển mắc bất kỳ sai sót hoặc lỗi nào trong khi phát triển sản phẩm.

Dưới đây là một số thuật ngữ khác nhau về lỗi:

|  |  |  |
| --- | --- | --- |
| **Terms** | **Description** | **Raised by** |
| Defect | When the application is not working as per the requirement. | Test Engineer |
| Bug | Informal name of defect | Test Engineer |
| Error | Problems in code lead to the errors. | Developer, Automation Test Engineer |
| Issue | When the application is not meeting the business requirement. | Customer |
| Mistake | Problem in the document is known as a mistake | —--- |
| Failure | Lots of defects lead to failure of the software. | —--- |

### Vòng đời của Bug:

Số lượng trạng thái mà một khiếm khuyết trải qua khác nhau giữa các dự án. Dưới đây là sơ đồ vòng đời, bao gồm tất cả các trạng thái có thể có



### Độ nghiêm trọng của Bug:

Dựa trên nhu cầu của khách hàng, mức độ nghiêm trọng của lỗi được xác định cho từng dự án và sẽ là điều cơ bản để nhóm Kiểm tra tập trung vào.

* Critical - 0: Thiết bị / Hệ thống / Phần mềm KHÔNG hoạt động chút nào: Cần được khắc phục càng sớm càng tốt
* Major - 1: Chức năng chính bị vô hiệu hóa hoặc không chính xác: Cần được sửa càng sớm càng tốt
* Minor - 2: Một số bất tiện cho người dùng: có thể khắc phục sau
* Cosmetic - 3: Lỗi chính tả, v.v., không ảnh hưởng đến hoạt động: có thể sửa sau

### Mức độ ưu tiên của Bug:

Dựa trên đặc điểm kỹ thuật của dự án, mức độ ưu tiên lỗi được xác định cho từng dự án và sẽ là cơ sở cho chiến lược Kiểm tra.

* Khẩn cấp - 0: Công việc phải dừng lại cho đến khi đóng lỗi này.
* Cao - 2: Công việc bị cản trở khi lỗi này đang mở
* Trung bình - 3: Một số tác động tiêu cực đến công việc tồn tại cho đến khi lỗi này được đóng lại
* Thấp - 4: Lỗi này phải được đóng lại, nhưng nó không ảnh hưởng đến công việc hiện tại.

### Các thành phần của Bug:

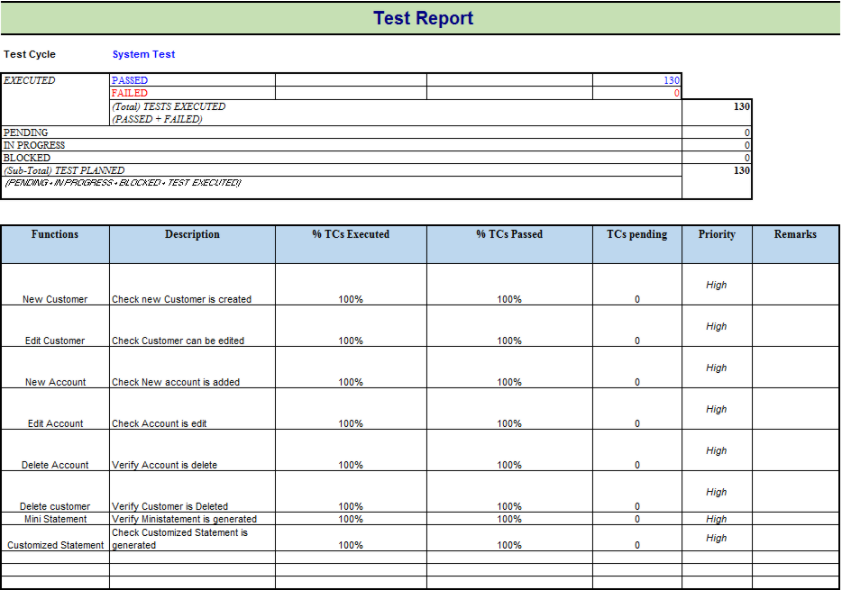
* Bug Number/id: Bug number hoặc số nhận dạng (như swb001) giúp báo cáo lỗi và quá trình đề cập đến lỗi dễ dàng hơn nhiều. Nhà phát triển có thể dễ dàng kiểm tra xem một lỗi cụ thể đã được sửa hay chưa. Nó làm cho toàn bộ quá trình kiểm tra và kiểm tra lại trơn tru và dễ dàng hơn. Nếu chúng tôi có hệ thống quản lý lỗi / lỗi, số lỗi / Id sẽ được tạo tự động
* Bug Title: Là được đọc thường xuyên hơn bất kỳ phần nào khác của báo cáo lỗi. Điều này sẽ giải thích tất cả về những gì xuất hiện trong lỗi. Tiêu đề Bug phải đủ gợi ý để người đọc có thể hiểu được nó. Tiêu đề lỗi rõ ràng giúp dễ hiểu và người đọc có thể biết lỗi đã được báo trước đó hay đã được sửa.
* Mức độ nghiêm trọng: Dựa trên nhu cầu của khách hàng, mức độ nghiêm trọng của lỗi sẽ được xác định cho từng lỗi, đó là nghiêm trọng, chính, nhỏ, thẩm mỹ.
* Quyền ưu tiên: Dựa trên mức độ nghiêm trọng của lỗi, bạn có thể đặt mức độ ưu tiên cho nó. Một lỗi có thể là một Chặn, Quan trọng, Chính, Nhỏ, Nhỏ hoặc một đề xuất. Các ưu tiên lỗi có thể được đưa ra từ P1 đến P5 để những lỗi quan trọng được xem trước.
* Platform/Environment: Cấu hình hệ điều hành và trình duyệt là cần thiết để có báo cáo lỗi rõ ràng. Đó là cách tốt nhất để thông báo về cách có thể tái tạo lỗi. Nếu không có nền tảng hoặc môi trường chính xác, ứng dụng có thể hoạt động khác và lỗi ở phía người kiểm tra có thể không tái tạo ở phía nhà phát triển. Vì vậy, tốt nhất là đề cập rõ ràng môi trường mà lỗi được phát hiện.
* Test data: Đôi khi vì một số lý do, lỗi xảy ra với dữ liệu thử nghiệm này nhưng không xảy ra với dữ liệu thử nghiệm khác nên chúng ta cần đề cập đến dữ liệu thử nghiệm trong lỗi một cách rõ ràng.
* Phần mô tả: Mô tả bug giúp nhà phát triển hiểu lỗi. Nó mô tả vấn đề gặp phải. Một mô tả kém sẽ tạo ra sự nhầm lẫn và lãng phí thời gian của các nhà phát triển cũng như người kiểm tra. Cần phải trao đổi rõ ràng về tác dụng của việc miêu tả. Sử dụng các câu hoàn chỉnh luôn hữu ích. Một thực tiễn tốt là mô tả từng vấn đề một cách riêng biệt thay vì nghiền nát chúng hoàn toàn. Không sử dụng các cụm từ như “Tôi nghĩ” hoặc “Tôi tin tưởng”.
* Các bước để tạo lại: Một báo cáo lỗi tốt nên đề cập rõ ràng các bước để tái tạo. Các bước này phải bao gồm các hành động có thể gây ra lỗi. Đừng đưa ra những tuyên bố chung chung, các bước phải rõ ràng.
* Kết quả mong đợi và thực tế: Mô tả bug không đầy đủ nếu không có kết quả mong đợi và thực tế. Cần phải phác thảo kết quả của thử nghiệm là gì và người dùng nên mong đợi điều gì. Người đọc nên biết kết quả chính xác của bài kiểm tra là gì. Rõ ràng, hãy đề cập đến những gì đã xảy ra trong quá trình kiểm tra và kết quả là gì.
* Screenshot: Một bưc tranh đang gia ngan lơi noi. Chụp ảnh màn hình của trường hợp lỗi với phụ đề thích hợp để làm nổi bật lỗi. Đánh dấu các thông báo lỗi không mong muốn bằng màu đỏ nhạt. Điều này thu hút sự chú ý đến khu vực cần thiết.
* Nhật ký bảng điều khiển: Bằng cách thu thập nhật ký bảng điều khiển, các nhà phát triển của bạn sẽ thấy việc tái tạo và giải quyết bất kỳ bug nào dễ dàng hơn rất nhiều.
* Mỗi lỗi phải có số / ID lỗi, tiêu đề, mức độ nghiêm trọng, mức độ ưu tiên, mô tả lỗi, các bước để tái tạo, kết quả mong đợi và thực tế. các thành phần khác có thể có hoặc không dựa trên từng bug.

## Test Report:

Báo cáo kiểm thử là một tài liệu chứa tóm tắt tất cả các hoạt động thử nghiệm và kết quả thử nghiệm cuối cùng của một dự án thử nghiệm. Báo cáo thử nghiệm là một đánh giá về mức độ thực hiện của Thử nghiệm. Dựa trên báo cáo thử nghiệm, các bên liên quan có thể đánh giá chất lượng của sản phẩm được thử nghiệm và đưa ra quyết định về việc phát hành phần mềm.

* Thông tin dự án: Tất cả thông tin của dự án như tên dự án, tên sản phẩm và phiên bản phải được mô tả trong báo cáo thử nghiệm.
* Mục tiêu kiểm tra: Báo cáo kiểm tra phải bao gồm mục tiêu của mỗi vòng kiểm tra, chẳng hạn như Kiểm tra đơn vị, Kiểm tra hiệu suất, Kiểm tra hệ thống…
* Tóm tắt kiểm tra: Phần này bao gồm tóm tắt về hoạt động thử nghiệm nói chung. Thông tin chi tiết tại đây bao gồm:
  + - Số lượng trường hợp thử nghiệm được thực thi
    - Số lượng các trường hợp thử nghiệm vượt qua
    - Số lượng các trường hợp thử nghiệm không thành công
    - Tỷ lệ phần trăm đạt
    - Phần trăm thất bại
    - Bình luận
* Khuyết điểm: Một trong những thông tin quan trọng nhất trong Test Report là lỗi. Báo cáo phải chứa thông tin sau.
  + - * Tổng số bug
      * Tình trạng bug (mở, đóng, phản hồi)
      * Số bug mở, giải quyết, đóng
      * Phân tích theo mức độ nghiêm trọng và mức độ ưu tiên

Ví dụ cho Báo cáo kiểm thử:



# CHƯƠNG 3. PHÂN TÍCH HỆ THỐNG

# Kiểm thử một số chức Năng của website Customer Portal.

3.1. Research Requiredment của website:

3.2. Thiết kế Test case cho quá trình kiểm thử:

3.3. Triển khai kiểm thử các chức năng cho website Customer Portal:

# CHƯƠNG 4. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

# TÀI LIỆU THAM KHẢO