**TRƯỜNG ĐẠI HỌC KINH TẾ**

**KHOA THỐNG KÊ – TIN HỌC**

–––––––––––––––––––––––––––––––



**BÁO CÁO THỰC TẬP NGHỀ NGHIỆP**

**THỰC HIỆN KIỂM THỬ DỰ ÁN VỀ WEB CHẤM CÔNG**

**NHÂN VIÊN NỘI BỘ**

**Sinh viên thực hiện**: Trần Tiên Ngọc Yến

**Lớp:** 44K21.2

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

**Đà Nẵng, 7/2021**

**LỜI CẢM ƠN**

**LỜI CAM ĐOAN**

Em xin cam đoan dự án **Thực hiện kiểm thử dự án về web chấm công nhân viên nội bộ** là kết quả nghiên cứu độc lập dưới sự hướng dẫn của giáo viên hướng dẫn TS. Cao Thị Nhâm. Ngoài ra không có bất cứ sự sao chép nào của người khác.

Dự án, nội dung báo cáo là sản phẩm mà em đã nỗ lực nghiên cứu trong quá trình học tập cũng như tìm hiểu ở nhiều phương tiện. Các dữ liệu, kết quả trình bày trong báo cáo là hoàn toàn trung thực, em xin chịu hoàn toàn trách nhiệm, kỷ luật của bộ môn và nhà trường đề ra nếu như có vấn đề xảy ra.

Mục lục

Danh mục hình ảnh

Danh mục bảng biểu

Danh mục các từ viết tắt

**LỜI MỞ ĐẦU**

1. **Mục tiêu của đề tài**
2. **Nhiệm vụ của đề tài**
3. **Kết cấu của đề tài**

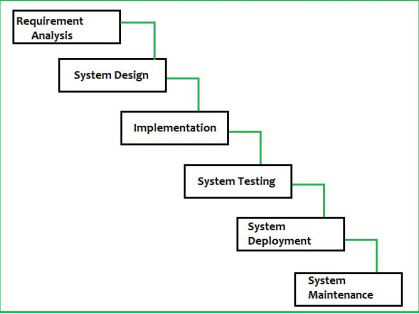
Đề tài được tổ chức gồm 3 chương nội dung và phần kết luận.

* **Chương 1**: Tổng quan về lý thuyết
* **Chương 2**: Tổng quan về hệ thống cần test
* **Chương 3**: Thiết kế testcase và thưc hiện test

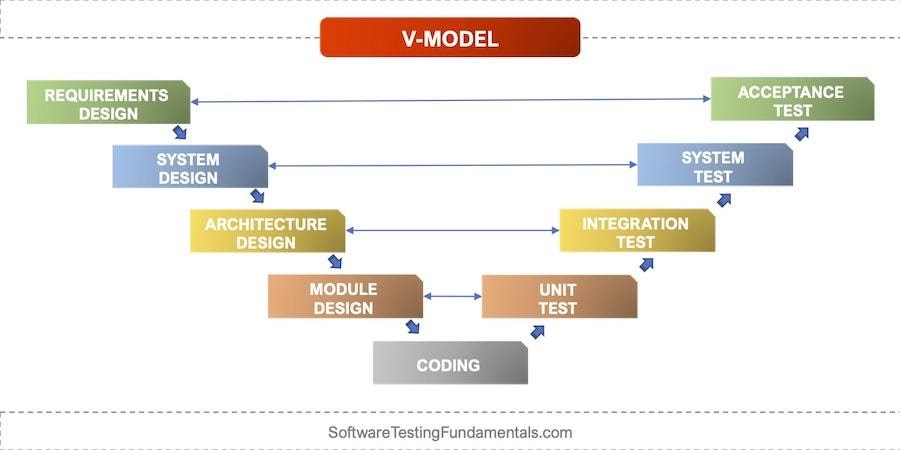
Kết luận và hướng phát triển

**CHƯƠNG 1: CƠ SỞ LÝ THUYẾT**

* 1. **Tổng quan về kiểm thử phần mềm**
     1. **Vòng đời phát triển phần mềm**
        1. **Mô hình thác nước (Water Fall)**



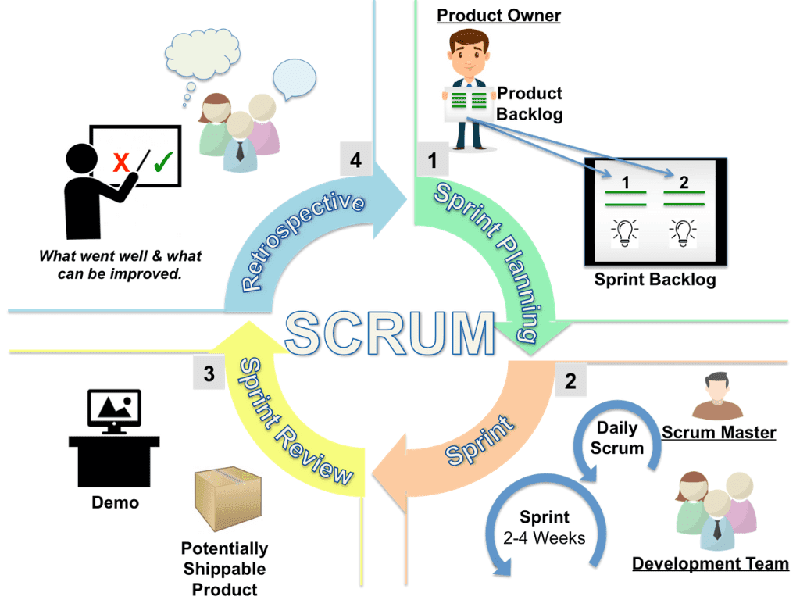
* Các giai đoạn phát triển nối tiếp nhau, giống như thác nước, nối tiếp nhau theo thời gian.
* Việc kiểm thử chỉ được thực hiện sau khi code xong toàn bộ phần mềm.
* Các giai đoạn trong mô hình Water Fall
* **Requirements Definition**: Thu thập và phân tích yêu cầu được ghi lại vào tài liệu đặc tả yêu cầu trong giai đoạn này.
* **System and software design**: Phân tích thiết kế hệ thống phần mềm đáp ứng yêu cầu của khách hàng như trong tài liệu SRS.
* **Implementation and Unit Testing**: Hệ thống được phát triển theo từng unit và được tích hợp trong giai đoạn tiếp theo. Mỗi Unit được phát triển và kiểm thử bởi dev được gọi là Unit Test.
* **Integration and system testing**: Kiểm thử tích hợp và kiểm thử hệ thống Công việc chính của giai đoạn này là kiểm tra và sửa tất cả những lỗi tìm được sao cho phần mềm hoạt động chính xác và đúng theo tài liệu đặc tả yêu cầu.
* **Operation and Maintenance**: Bảo trì hệ thống khi có bất kỳ thay đổi nào từ phía khách hàng, người sử dụng.
* Ưu điểm của mô hình Water Fall
* Thích nghi tốt với những nhóm linh hoạt: có phác thảo và tự động hóa tài liệu từ ngay giai đoạn đầu tiên
* Cho phép những thay đổi thiết kế sớm: Việc thay đổi này được diễn ra thuận lợi hơn là bởi chưa có mã hoặc triển khai nào ở giai đoạn này.
* Áp đặt một tổ chức có kết cấu chặt chẽ: giúp xây dựng dự án nghiêm ngặt, chính xác, tuân thủ theo thiết kế và cấu tạo của sản phẩm
* Thích hợp cho những dự án theo hướng đến mốc: Với các khung thời gian rõ ràng và cụ thể, các thành viên trong nhóm có thể dễ dàng hiểu, làm đúng theo tiến trình thời gian.
* Nhược điểm của mô hình Water Fall
* Thực tế cho thấy đến những giai đoạn cuối của dự án mới có khả năng nhận ra sai sót.
* Các lỗi được tìm thấy sau giai đoạn phát triển rất tốn thời gian và chi phí để sửa các lỗi.
  + - 1. **Mô hình V\_model**



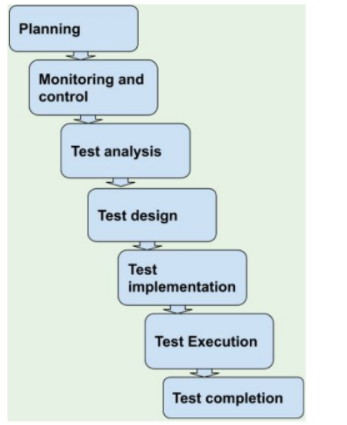
* Là quy trình phát triển phần mềm mở rộng của quy trình phát triển phần mềm thác nước.
* Toàn bộ quy trình được chia làm hai nhánh: Phát triển và kiểm thử.
* Ưu điểm của mô hình V\_model
* Mỗi giai đoạn phát triển sẽ tiến hành song song với một giai đoạn kiểm thử tương ứng => Các lỗi sẽ được phát hiện sớm ngay từ đầu
* Nhược điểm của mô hình V\_moldel
* Vì ở mô hình chữ V này người ta vẫn phát triển cùng lúc cả hệ thống nên mô hình vẫn gặp rắc rối khi khách hàng thường xuyên thay đổi yêu cầu

=> Mô hình thích hợp cho các dự án ngắn và có yêu cầu rõ ràng ít có sự thay đổi.

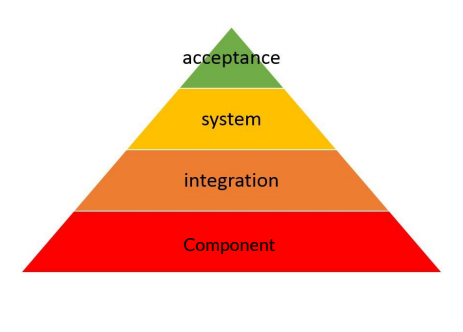
* + - 1. **Mô hình scrum**



* Mỗi một sprint kéo dài khoảng từ 1 tuần đến 4 tuần ( ko dài hơn 1 tháng).
* Đầu sprint sẽ lên kế hoạch làm những yêu cầu nào. Sau đó, sẽ thực hiện code và test. Cuối sprint là 1 sản phẩm hoàn thiện cả code lẫn test có thể demo và chạy được.
* Hoàn thành sprint 1, tiếp tục làm sprint 2, sprint... cho đến khi hoàn thành hết các yêu cầu.
* Trong mỗi 1 sprint thì sẽ có họp hàng ngày – daily meeting từ 15 – 20 phút. Mỗi thành viên sẽ báo cáo: Hôm qua tôi đã làm gì? Hôm nay tôi sẽ làm gì? Có gặp khó khăn gì không?
* Scrum là mô hình hướng khách hàng (Customer oriented).
* Ưu điểm của mô hình Scrum
* Phù hợp với những dự án có sự thay đổi về yêu cầu của khách hàng,làm theo giai đoạn ngắn, có thể nhìn thấy những rủi ro, hay những điểm chưa phù hợp để thay đổi.
* Nhược điểm của mô hình Scrum
* Nhóm sản xuất phải có kỹ năng và hiểu biết về mô hình Agile, quy trình Scrum.
* Khó khăn trong việc xác định ngân sách và thời gian.
  + 1. **Quy trình kiểm thử phần mềm**

****

* + - 1. **Planning**
* Xác định chiến lược và cách tiếp cận kiểm tra
* Xác định phạm vi thử nghiệm
* Nêu tất cả các giả định và điều kiện
* Xác định các nhiệm vụ kiểm tra
* Xác định các công cụ kiểm tra
* Lập kế hoạch ngân sách thử nghiệm
* Xác định các mục tiêu và rủi ro liên quan đến thử nghiệm
* Lập kế hoạch tài nguyên, môi trường thử nghiệm và máy trạm, v.v.
* Lập kế hoạch kiểm tra cho các thử nghiệm cụ thể như (Kiểm tra hệ thống, Kiểm tra hiệu suất, v.v.)
* Lên lịch thời gian cho quá trình phân tích, thiết kế, thực hiện và kết thúc kiểm tra.
  + - 1. **Monitoring and control**
* So sánh tiến độ thực tế với tiến độ theo kế hoạch bằng cách sử dụng bất kỳ số liệu giám sát thử nghiệm nào được xác định trong kế hoạch thử nghiệm.
* Kiểm soát thử nghiệm bao gồm việc thực hiện các hành động cần thiết để đáp ứng các mục tiêu của kế hoạch thử nghiệm (có thể được cập nhật theo thời gian).
* Đánh giá các tiêu chí thoát để thực hiện kiểm
  + - 1. **Test Analysis**
* Xem lại bài kiểm tra cơ bản (về cơ bản các tài liệu như yêu cầu, kiến trúc, thiết kế cấp cao, phân tích sản phẩm)
* Phân tích cơ sở thử nghiệm phù hợp với mức thử nghiệm đang được xem xét Xác định các tính năng và tập hợp các tính năng sẽ được thử nghiệm
* Xác định và ưu tiên các điều kiện thử nghiệm cho từng tính năng dựa trên phân tích cơ sở thử nghiệm và xem xét các đặc điểm chức năng, phi chức năng và cấu trúc, các yếu tố kinh doanh và kỹ thuật khác, và mức độ rủi ro
* Nắm bắt khả năng truy nguyên hai chiều giữa từng yếu tố của cơ sở thử nghiệm và các điều kiện thử nghiệm liên quan
  + - 1. **Test Design**
* Thiết kế và ưu tiên các trường hợp kiểm thử và tập hợp các trường hợp kiểm thử
* Xác định dữ liệu thử nghiệm cần thiết để hỗ trợ các điều kiện thử nghiệm và trường hợp thử nghiệm
* Thiết kế môi trường thử nghiệm và xác định bất kỳ cơ sở hạ tầng cần thiết nào và công cụ
* Nắm bắt khả năng truy xuất nguồn gốc hai chiều giữa cơ sở thử nghiệm, thử nghiệm điều kiện và trường hợp thử nghiệm
  + - 1. **Test Implementation**
* Thực hiện các điều kiện thử nghiệm đã thiết kế
* Tạo bộ thử nghiệm (Bộ thử nghiệm - Nhóm các trường hợp thử nghiệm lại với nhau có cùng một tập hợp các hành vi)
* Phát triển và ưu tiên các thủ tục kiểm tra, và có khả năng tạo ra tập lệnh kiểm tra tự động
* Xây dựng môi trường thử nghiệm và xác minh rằng mọi thứ cần thiết đều có được thiết lập chính xác
* Chuẩn bị dữ liệu thử nghiệm và đảm bảo nó được tải đúng cách trong thử nghiệm  môi trường
* Xác minh và cập nhật khả năng truy nguyên hai chiều giữa cơ sở thử nghiệm, điều kiện thử nghiệm, trường hợp thử nghiệm, quy trình thử nghiệm và bộ thử nghiệm
  + - 1. **Test Execution**
* Ghi lại các ID và phiên bản của (các) mục thử nghiệm hoặc đối tượng thử nghiệm, thử nghiệm (các) công cụ và phần mềm kiểm tra
* Thực hiện các bài kiểm tra theo cách thủ công hoặc bằng cách sử dụng các công cụ thực thi kiểm tra
* So sánh kết quả thực tế với kết quả mong đợi
* Phân tích các điểm bất thường để xác định nguyên nhân có thể xảy ra
* Báo cáo các khiếm khuyết dựa trên các hư hỏng được quan sát thấy
* Ghi lại kết quả của việc thực hiện kiểm tra
* Lặp lại các hoạt động kiểm tra hoặc là kết quả của hành động được thực hiện cho một sự bất thường,  hoặc là một phần của thử nghiệm theo kế hoạch
* Xác minh và cập nhật khả năng truy nguyên hai chiều giữa cơ sở thử nghiệm, điều kiện thử nghiệm, trường hợp thử nghiệm, quy trình thử nghiệm và kết quả thử nghiệm.
  + - 1. **Test completion**
* Kiểm tra xem tất cả các báo cáo lỗi đã được đóng lại chưa, nhập thay đổi yêu cầu hoặc các mặt hàng tồn đọng sản phẩm cho bất kỳ lỗi nào còn sót lại chưa được giải quyết khi kết thúc quá trình thực thi thử nghiệm
* Tạo báo cáo tóm tắt thử nghiệm để thông báo cho các bên liên quan
* Hoàn thiện và lưu trữ môi trường thử nghiệm, dữ liệu thử nghiệm, thử nghiệm  cơ sở hạ tầng và phần mềm thử nghiệm khác để sử dụng lại sau này
* Bàn giao testware cho các nhóm bảo trì, dự án khác nhóm và / hoặc các bên liên quan khác có thể hưởng lợi từ việc sử dụng
* Phân tích bài học kinh nghiệm từ các hoạt động kiểm tra đã hoàn thành để  xác định các thay đổi cần thiết cho các lần lặp lại, bản phát hành và dự án trong tương lai
* Sử dụng thông tin thu thập được để cải thiện mức độ thành thục của quá trình kiểm tra
  + 1. **Bảy nguyên lý kiểm thử phần mềm**
       1. **Principle 1: Testing shows the presence of defects, not their absence ( Kiểm thử đưa ra lỗi)**
* Kiểm thử để tìm thấy rằng phần mềm đang có lỗi, dù kiểm thử nghiêm ngặt phần mềm vẫn có thể có lỗi, càng tìm nhiều lỗi càng tốt.
  + - 1. **Principle 2: Exhaustive testing is impossible (Kiểm thử cạn kiệt là không thể)**
* Nguyên tắc này nói rằng không thể kiểm tra mọi thứ một cách trọn vẹn, Thay vì kiểm thử toàn bộ, việc phân tích rủi ro và dựa trên sự mức độ ưu tiên chúng ta có thể tập trung việc kiểm thử vào một số điểm cần thiết, có nguy cơ lỗi cao hơn.
  + - 1. **Principle 3: Early testing saves time and money** (**Kiểm thử càng sớm càng tốt**)
* Yêu cầu thử nghiệm ở giai đoạn đầu giúp phát hiện bug sớm nhất có thể
* Nó giúp chuyển giao phần mềm theo yêu cầu đúng thời gian với chất lượng dự kiến.
  + - 1. **Defects cluster together** (**Sự tập trung của lỗi**)
* Để tối đa hóa hiệu quả của việc kiểm tra, chúng ta cần xác định những khu vực dễ bị lỗi và tập trung thử nghiệm vào những khu vực đó.
* Thông thường, lỗi tập trung vào những module, thành phần chức năng chính của hệ thống, Tập trung vào tìm kiếm lỗi quanh khu vực được xác định.
  + - 1. **Beware of the pesticide paradox** (**Nghịch lí thuốc trừ sâu**)
* Luôn luôn phải xem xét và sửa đổi các trường hợp kiểm thử trên một khoảng thời gian thường xuyên.
* Nếu sử dụng các bài kiểm tra giống nhau nhiều lần, hiệu quả của nó trong việc tìm kiếm các sai sót sẽ giảm dần theo thời gian.
  + - 1. **Testing is context dependent** (**Kiểm thử phụ thuộc vào ngữ cảnh**)
* Phải tiếp cận kiểm thử theo nhiều ngữ cảnh khác nhau, kiểm thử ứng dụng web và ứng dụng di động bằng cách sử dụng chiến lược kiểm thử giống nhau là sai, mỗi loại sẽ có chiến lược để kiểm thử.
  + - 1. **Absence-of-errors is a fallacy (Không có lỗi - Sai lầm)**
* Việc không tìm thấy lỗi trên sản phẩm không đồng nghĩa với việc sản phẩm đã sẵn sàng để tung ra thị trường.
* Việc không tìm thấy lỗi cũng có thể là do bộ trường hợp kiểm thử được tạo ra chỉ nhằm kiểm tra những tính năng được làm đúng theo yêu cầu thay vì nhằm tìm kiếm lỗi mới
  + 1. **Test level, Test type, Test method**
       1. **Test level**



* + - * 1. **Test Unit (Component)**
* Test unit Là một cấp độ kiểm thử phần mềm trong đó các đơn vị / thành phần riêng lẻ của một phần mềm được kiểm tra. Mục đích là để xác nhận rằng mỗi đơn vị của phần mềm hoạt động như được thiết kế.
* Test unit có thể được thực hiện tách biệt với phần còn lại của hệ thống tùy thuộc vào mô hình vòng đời phát triển được chọn cho Ứng dụng đặc biệt.
* Thường thì dev thực hiện
  + - * 1. **Test Integration (Kiểm thử tích hợp)**
* Kiểm thử tích hợp là giai đoạn trong đó các mô-đun phần mềm riêng lẻ được kết hợp và kiểm tra như một nhóm để xác minh hệ thống tích hợp đã sẵn sàng cho việc kiểm tra hệ thống.
* Nó xảy ra sau khi kiểm tra đơn vị và trước khi kiểm tra xác nhận.
* Kiểm thử tích hơp được thực hiện bởi người kiểm tra tích hợp hoặc nhóm kiểm tra cụ thể.
* Việc Integration testing là cần thiết mặc dù đã unit test nhưng các lỗi vẫn có thể tồn tại
* Kiểm thử tích hợp khác với các trường hợp kiểm tra khác, nó tập trung chủ yếu vào các giao diện & lưu lượng dữ liệu / thông tin giữa các module. Ưu tiên được trao cho các liên kết tích hợp chứ không phải là các đơn vị chức năng.
  + - * 1. **Test System (Kiểm thử hệ thống)**
* Kiểm tra hệ thống được tiến hành trên một hệ thống hoàn chỉnh, tích hợp
* Hệ thống đánh giá sự tuân thủ của hệ thống đối với các yêu cầu.
* Kiểm tra hệ thống nằm trong phạm vi kiểm tra hộp đen và như vậy, không yêu cầu kiến thức về thiết kế bên trong của mã hoặc logic.
* Kiểm tra hệ thống thường là kiểm tra cuối cùng để xác minh rằng hệ thống được cung cấp đáp ứng các đặc điểm kỹ thuật và mục đích của nó.
* Kiểm thử hệ thống nên điều tra cả các yêu cầu chức năng và không chức năng của thử nghiệm.
* Kiểm thử hệ thống làm giảm sự cố và bảo trì sau khi triển khai.
  + - * 1. **Test Acceptance (Kiểm thử chấp nhận)**
* Sau khi kiểm tra hệ thống đã sửa chữa tất cả hoặc hầu hết các lỗi, hệ thống sẽ được giao cho người dùng hoặc khách hàng để kiểm tra chấp nhận.
* Thử nghiệm chấp nhận là một thử nghiệm được tiến hành để xác định xem các yêu cầu của một đặc điểm kỹ thuật hoặc hợp đồng được đáp ứng trước khi giao hàng.
* Kiểm tra chấp nhận về cơ bản được thực hiện bởi người dùng hoặc khách hàng mặc dù các bên liên quan khác cũng có thể tham gia.
  + - 1. **Test type**
         1. **Functional Testing (Kiểm thử chức năng)**
* Kiểm tra chức năng đề cập đến các hoạt động xác minh một hành động cụ thể hoặc chức năng của dòng code.
* Các bài kiểm tra chức năng có xu hướng trả lời câu hỏi "người dùng có thể làm điều này không" hoặc "tính năng cụ thể này có hoạt động không."
  + - * 1. **Non Functional Testing (Kiểm thử phi chức năng)**
* Kiểm thử phi chức năng đề cập đến các khía cạnh của phần mềm có thể không có liên quan đến một chức năng cụ thể hoặc hành động của người dùng, chẳng hạn như khả năng mở rộng hoặc cải tiến, hành vi dưới các ràng buộc hoặc bảo mật nhất định.
  + - * 1. **Structural testing (Kiểm thử hộp trắng)**
* Kiểm tra cấu trúc (hộp trắng) có thể được thực hiện ở tất cả các cấp độ kiểm tra.
* Kiểm thử cấu trúc có thể dựa trên kiến trúc của hệ thống, chẳng hạn như cấu trúc phân cấp đang gọi.
* Các phương pháp tiếp cận kiểm tra cấu trúc cũng có thể được áp dụng ở cấp độ hệ thống, tích hợp hệ thống hoặc kiểm tra chấp nhận (ví dụ: đối với các mô hình kinh doanh hoặc cấu trúc menu).
  + - * 1. **Testing related to changes (Kiểm tra xác nhận và Kiểm tra hồi quy)**
* Kiểm tra xác nhận: Sau khi phát hiện và sửa lỗi, phần mềm nên được kiểm tra lại để xác nhận rằng lỗi ban đầu đã được loại bỏ thành công.
* Kiểm tra hồi quy: Việc kiểm tra lặp lại một chương trình đã được kiểm tra, sau khi sửa đổi, để phát hiện ra bất kỳ khiếm khuyết nào được đưa vào hoặc phát hiện ra do các thay đổi.
* Kiểm thử hồi quy có thể được thực hiện ở tất cả các cấp độ kiểm tra và áp dụng cho kiểm thử chức năng, phi chức năng và cấu trúc.
  + - 1. **Test method**
         1. **White box testing (Kiểm thử hộp trắng)**
* Kiểm thử hộp trắng là cuộc kiểm tra chi tiết về logic bên trong và cấu trúc của code.
* Các phương pháp: kiểm tra đơn vị, tích hợp , hệ thống nhưng kiểm tra đơn vị là chủ yếu
* Ưu điểm của kiểm thử hôp trắng
* Tối ưu hóa mã nguồn bằng cách tìm ra các lỗi ẩn.
* Dễ dàng thực hiện tự động nhưng cũng kỹ lưỡng hơn
* Và việc kiểm tra có thể bắt đầu sớm ngay cả khi GUI không khả dụng.
* Nhược điểm của kiểm thử hộp trắng
* Thực hiện khá phức tạp và tốn kém chi phí.
* Đòi hỏi người kiểm thử phải thực sự chuyên nghiệp và hiểu biết về lập trình.
* Tốn rất nhiều thời gian.
  + - * 1. **Black box testing** **(Kiểm thử hộp đen)**
* Kiểm thử hộp đen là à phương pháp test dựa trên đầu vào và đầu ra của chương trình để test mà không quan tâm tới code bên trong được viết ra sao.
* Các phương pháp: kiểm tra tích hợp , hệ thống, chấp nhận.
* Ưu điểm của kiểm thử hộp đen
* Tester không cần biết quá nhiều về kiến thức lập trình, quy trình code
* Các trường hợp thử nghiệm có thể được thiết kế ngay sau khi các thông số kỹ thuật hoàn tất.
* Các thử nghiệm được thực hiện theo quan điểm của người dùng và sẽ giúp chỉ ra sự khác biệt trong các thông số kỹ thuật.
* Nhược điểm của kiểm thử hộp đen
* Không có thông số kỹ thuật rõ ràng, đó là tình trạng trong nhiều dự án, các trường hợp kiểm thử sẽ khó thiết kế.
* Có nhiều trường hợp khi một tester viết rất nhiều trường hợp test để kiểm tra một số thứ có thể chỉ được test bằng một trường hợp test và/hoặc một vài phần cuối cùng không được test hết.
  + - * 1. **Gray box testing (Kiểm thử hộp xám)**
* Kiểm thử hộp xám có sự kết hợp giữa lợi ích của kiểm thử hộp đen và hộp trắng.
* Với kiểm thử hộp xám người kiểm thử cần phải có một phần sự hiểu biết về cấu trúc của hệ thống cũng như là quyền truy cập vào cơ sở dữ liệu.
* Ưu điểm của kiểm thử hộp xám
* Quan điểm kiểm thử của kiểm thử hộp xám là từ quan điểm của người dùng.
* Cung cấp các lợi ích của cả thử nghiệm hộp đen và hộp trắng cùng nhau.
* Sẽ dựa trên các đặc tả chức năng, mô tả của người dùng và sơ đồ kiến trúc hệ thống, từ đó xác nhận các yêu cầu ngay từ ban đầu.
* Việc kiểm tra sẽ tường minh vì sẽ có nhiều sự quan tâm giữa người kiểm thử phần mềm và người thiết kế hoặc kỹ sư.
* Nhược điểm của kiểm thử hộp xám
* Kiểm tra hộp xám cũng có thể mất nhiều thời gian để kiểm tra từng đường dẫn và đôi khi điều này là không thực tế.
* Rất khó để liên kết lỗi khi thực hiện kiểm tra hộp xám cho một ứng dụng có hệ thống phân tán.
* Thông thường sẽ dẫn đến phạm vi kiểm tra thấp hơn so với thực hiện kiểm tra hộp trắng và đen riêng biệt.
* Có thể không phù hợp để thử nghiệm một số loại chức năng.
  + 1. **Test design technique, Black box testing technique**
       1. **Test design technique**
* Thiết kế thử nghiệm là xác định một tập hợp các thử nghiệm từ tổng số tất cả các thử nghiệm có thể có cho một hệ thống nhất định.
* Chúng ta cần thiết kế kiểm thử trước khi tạo các trường hợp kiểm thử chi tiết và thực hiện kiểm thử để không bị bỏ xót các trường hợp kiểm thử nào.
* Có 2 loại kiểm thử:
* Static Testing (kiểm thử tĩnh)
* Kiểm thử tĩnh là việc kiểm tra từng phần của software chủ yếu dựa trên các tài liệu của phần mềm, hoặc tự phân tích các cú pháp của code để kiểm tra logic mà không cần phải chạy phần mềm trực tiếp.
* Kiểm thử tĩnh có thể được tiến hành ngay từ đầu của dự án ngay khi đặc tả yêu cầu sẵn sàng.
* Dynamic Testing (kiểm thử động)
* Kiểm thử động là phương pháp kiểm thử phần mềm thông qua việc dùng máy để chạy phần mềm nhằm điều tra các mã lệnh của phần mềm, nhập các giá trị đầu vào và kiểm tra xem giá trị đầu ra có như mong muốn hay không.
  + - 1. **Back box testing technique**
* Quivalence partitioning
* Boundary value analysis
* Decision table testing
* State transition testing
* Use case testing
  + 1. **Test plan, Test checklist**
       1. **Test plan**
* Test plan là là tài liệu mô tả các mục tiêu kiểm tra cần đạt được và phương tiện cũng như lịch trình để đạt được chúng, được tổ chức để phối hợp các hoạt động kiểm tra.
* Sau khi phân tích yêu cầu, Test plan là giai đoạn đầu tiên trong quá trình kiểm thử phần mềm
* Đầu vào của Test plan chứa tài liệu yêu cầu, tài liệu thiết kế, tiêu chí chấp nhận và ngày phát hành thông tin.
* Trước khi tạo một Test plan Tester phải nghiên cứu kỹ yêu cầu, ghi chú và nêu câu hỏi để đảm bảo rằng hiểu yêu cầu thông suốt.
  + - 1. **Test checklist**
* Test checklist Là một danh sách các đầu mục chức năng/ nghiệp vụ cần kiểm tra trong một thủ tục hay quy định nhất định. Nó giúp cho người kiểm thử nắm bao quát được tổng thể các chức năng trong 1 object & đánh giá được trường hợp pass hoặc fail.
* Từ bộ checklist sẽ được bóc tách chia thành nhiều ca kiểm thử (testcase).
* Trước khi thực hiện kiểm tra, QA cần tạo danh sách kiểm tra kiểm tra vì:

+ Để đảm bảo rằng tất cả các yêu cầu đều có trong danh sách kiểm tra thử nghiệm + Để tránh bỏ sót điểm xem thử nghiệm, ngăn chặn lỗi

+ Để dễ dàng theo dõi tiến độ thử nghiệm.

* Các bước tạo một Test checklist
* Bước 1: Nghiên cứu yêu cầu, thực hiện Q&A để làm rõ thông số kỹ thuật
* Bước 2: Tạo thiết kế thử nghiệm bằng các kỹ thuật thiết kế thử nghiệm
* Bước 3: Tạo danh sách kiểm tra Kiểm tra
  + 1. **Test case**
* Test Case là một tập tài liệu của điều kiện tiên quyết (điều kiện tiên quyết), thủ tục (đầu vào / hành động) và postconditions (kết quả mong đợi) mà một sử dụng thử nghiệm để xác định xem một hệ thống theo đáp ứng kiểm tra yêu cầu hoặc hoạt động chính xác.
* Sau khi lập kế hoạch kiểm thử và thiết kế kiểm thử, QA bắt đầu viết các trường hợp kiểm thử chi tiết.
* Đầu vào của giai đoạn này là các tài liệu yêu cầu cập nhật (SRS, Ca sử dụng, Chi tiết thiết kế ...)
* Cấu trúc của Test case

●Test case ID

● Priority: Đặt mức độ ưu tiên cho trường hợp thử nghiệm

○ High(P1): Chức năng chính

○ Medium(P2): Chức năng phụ

○ Low(P3): UI

● Prerequisites/Pre-condition: điều kiện nào cần thiết cần thực hiện trước khi thực hiện thử nghiệm

● Test view point: mục tiêu của trường hợp thử nghiệm

● Test data: dữ liệu để tiến hành thử nghiêm

● Steps : quy trình từng bước để thực hiện thử nghiệm

● Expected result: kết quả mong đợi khi thực hiện thử nghiệm

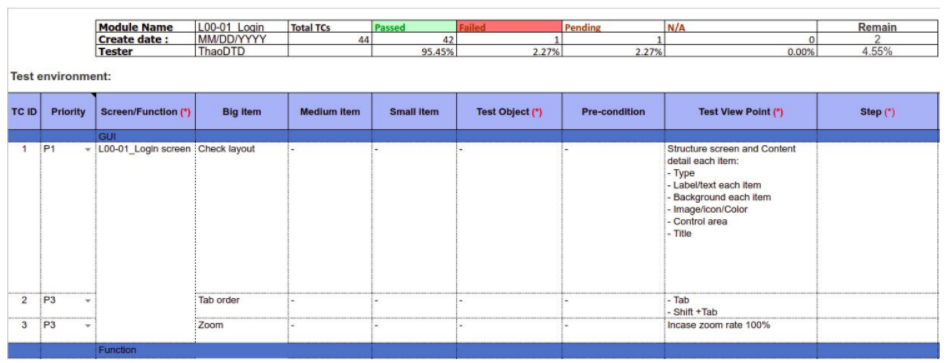
● Test Result: kết quả thử nghiệm (Pass/Fail/Pending/NA/Block)

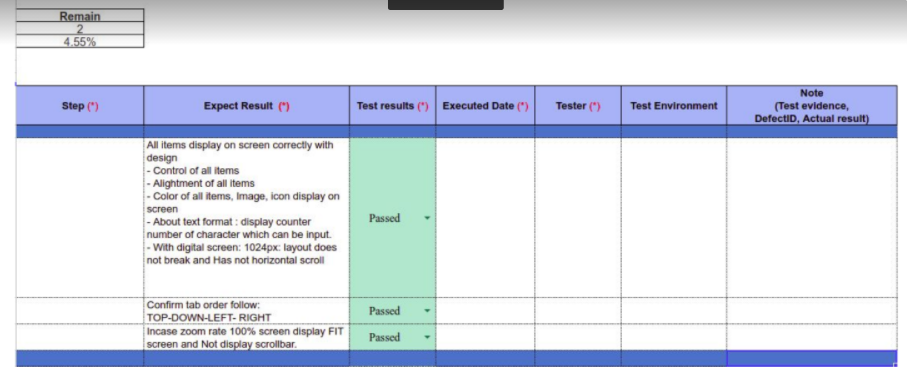
● Executed date: ngày thực hiện thử nghiệm

● Executed by: người thực hiện thử nghiệm

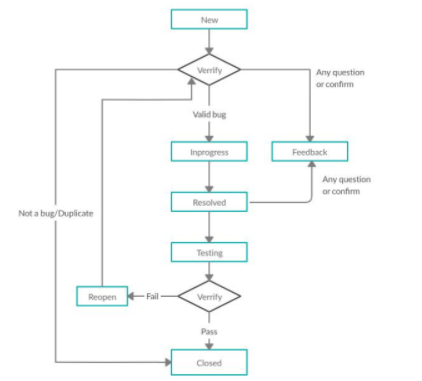
● Test environment: môi trường test

● Note





* + 1. **Bug report**
* BUG Phần mềm là một điều kiện trong một sản phẩm phần mềm không đáp ứng yêu cầu phần mềm (như đã nêu trong các thông số kỹ thuật yêu cầu) hoặc kỳ vọng của người dùng cuối (có thể không được chỉ định nhưng là hợp lý). Nói cách khác, khuyết tật là sự không hoàn hảo hoặc thiếu hụt trong sản phẩm làm việc mà sản phẩm đó không đáp ứng được các yêu cầu hoặc thông số kỹ thuật của nó.
* Vòng đời của Bug report bắt đầu với lỗi được phát hiện và báo cáo bởi người thử nghiệm và kết thúc sau khi đóng. Trong toàn bộ vòng đời của lỗi có các trạng thái khác nhau.



❖New: Khi một lỗi được báo cáo và đăng lần đầu. Trạng thái của nó là “New”.

❖Verify bug: Sau khi người kiểm tra đã báo cáo lỗi, nhóm trưởng của người kiểm tra xác nhận rằng lỗi đó là hợp lệ và nó được chỉ định cho nhà phát triển hoặc nhóm nhà phát triển thích hợp. Nếu lỗi đó không hợp lệ thì đóng lỗi.

❖Feedback: Nếu có bất kỳ câu hỏi hoặc bất kỳ vấn đề nào cần xác nhận, người kiểm tra hoặc nhà phát triển sẽ nêu phản hồi kèm theo nội dung sau đó chỉ định xác nhận.

❖Inprogress: Có nghĩa là nhà phát triển đã bắt đầu phân tích lỗi và cố gắng sửa chữa.

❖Resolved: Sau khi nhà phát triển thay đổi mã, sửa lỗi và triển khai, họ thay đổi trạng thái thành Đã giải quyết và nó có thể được chuyển cho nhóm QA để kiểm tra lại. Ở giai đoạn này, báo cáo lỗi đang chờ kiểm tra lại.

❖Testing: Ở giai đoạn này, người kiểm tra kiểm tra các sửa đổi và kiểm tra lại thay đổi mà nhà phát triển đã thực hiện

❖Reopen: Trong trường hợp người kiểm tra đã kiểm tra lại và lỗi vẫn tồn tại, trạng thái lỗi sẽ trở thành "Reopen" và báo cáo lỗi sẽ đi qua vòng đời một lần nữa

❖Closed: Sau khi nhà phát triển sửa lỗi, họ sẽ gửi sản phẩm cho người thử nghiệm để kiểm tra lại. Nếu người kiểm tra quyết định rằng lỗi đã được sửa, họ sẽ thay đổi trạng thái báo cáo lỗi thành "Closed".

* + 1. **SQL**
* Các câu lệnh về ngôn ngữ thao tác dữ liệu (DML) như: Insert, Delete, Update
* Câu lệnh về ngôn ngữ truy vấn (DQL): Select
* Hỗ trợ truy vấn: distinct, top, as, identity
* Phép toán tập hợp: in, like, between, not in, exist
* Các hàm tổng nhóm: sum, max, min, avg, count
* Truy vấn đơn giản: select\*, select…where, select…order by, select… group by, …
* Truy vấn lồng: lồng các câu lệnh với nhau.
* Truy vấn liên bảng: Inner Join, Left Outer Join, Right Outer Join, Full Outer Join
  1. **Tổng quan về nghề nghiệp**
     1. **Mô tả vị trí việc làm QA**
        1. **Vị trí làm việc**
* Vị trí công việc : Nhân viên QA (Quality Assurance)
* QA là người chịu trách nhiệm đảm bảo chất lượng sản phẩm thông qua việc đưa ra quy trình làm việc giữa các bên liên quan.
  + - 1. **Mô tả công việc QA**
* Đề xuất, đưa ra quy trình phát triển (development process) sản phẩm phù hợp với yêu cầu cụ thể của từng dự án.
* Đưa ra những tài liệu, biểu mẫu, hướng dẫn để đảm bảo chất lượng của sản phẩm cho tất cả các bộ phận trong nhóm phát triển sản phẩm.
* Kiểm tra, kiểm toán việc thực thi quy trình của các bộ phận trong nhóm làm sản phẩm có đúng quy trình QA đã đề ra không.
* Nhắc nhở đội ngũ phát triển sản phẩm việc tuân thủ theo quy trình làm việc đã đưa ra.
* Điều chỉnh, thay đổi quy trình phù hợp với từng sản phẩm mà các team đang thực hiện.
  + - 1. **Các kỹ năng cần thiết cho QA**
* Hiểu sâu về kiến trúc hệ thống của phần mềm vì công việc của QA rộng hơn QC.
* Khả năng tổ chức, tư duy logic và có hệ thống.
* Kỹ năng phân tích, làm việc dựa trên số liệu tốt.
* Kiến thức rộng về các lĩnh vực của phần mềm mà các team đang thực hiện.
* Kỹ năng giao tiếp trong nội bộ team và các team khác. Mục đích: khai thác thông tin về sản phẩm, dự án và ứng dụng nó vào việc xây dựng hệ thống quy trình.
* Hiểu rõ về các chứng chỉ CMMI, ISO… trong phần mềm để xây dựng các quy trình chuẩn cho các team.
* Kỹ năng lập trình
* Có kỹ năng về ngôn ngữ như: Tiếng anh, tiếng nhật, …
  + 1. **Cơ hội nghề nghiệp, mức lương cho Fresher**
* Cơ hội nghề nghiệp của QA đang rất lớn, mức lương cao, nhân viên **QA** còn được hưởng các chế độ đãi ngộ hấp dẫn như thưởng hiệu quả công việc, thưởng sáng tạo, chế độ bảo hiểm, chăm sóc sức khỏe, …
* Mức lương cho Fresher từ 8-20tr tùy theo năng lực.

**CHƯƠNG 2:TỔNG QUAN VỀ HỆ THỐNG CẦN TEST**