**BỘ CÔNG THƯƠNG**

**TRƯỜNG ĐẠI HỌC CÔNG NGHIỆP THỰC PHẨM**

** THÀNH PHỐ HỒ CHÍ MINH**

**Kiểm định chất lượng phần mền: website bán mỹ phẩm**

**SINH VIÊN THỰC HIỆN:**

1. 2001190791 – Từ Huệ Sơn
2. 2001190248 – Trần Thành Tâm
3. 2001190794 – Lê Đức Tài
4. 2001190876 – Huỳnh Mỹ Trân
5. 2001190866 – Lư Phước Toàn
6. 2001190520 – Nguyễn Trung Hậu

**GVHD: Trần Văn Thọ**

TP. HỒ CHÍ MINH, tháng 03 năm 2022

**Mục Lục**

[**LỜI MỞ ĐẦU** 1](#_Toc100524322)

[**Danh mục hình** 2](#_Toc100524323)

[**Danh mục bảng** 3](#_Toc100524324)

[**CHƯƠNG 1: TỔNG QUAN VỀ KIỂM THỬ PHẦN MỀM** 4](#_Toc100524325)

[**1.1. Các khái niệm cơ bản về kiểm thử phần mềm** 4](#_Toc100524326)

[**1.1.1. Kiểm thử phần mềm là gì?** 4](#_Toc100524327)

[**1.1.2. Các phương pháp kiểm thử** 4](#_Toc100524328)

[**1.1.3. Các chiến lược kiểm thử** 5](#_Toc100524329)

[**1.1.4. Các cấp độ kiểm thử phần mềm** 7](#_Toc100524330)

[**1.1.5. Các phương pháp kiểm thử con người** 13](#_Toc100524331)

[**1.2. Nguyên tắc kiểm thử phần mềm** 14](#_Toc100524332)

[**CHƯƠNG 2: THIẾT KẾ TEST PLAN** 15](#_Toc100524333)

[**2.1. Giới thiệu chung** 15](#_Toc100524334)

[**2.1.1. Tổng quan** 15](#_Toc100524335)

[**2.1.2. Phạm vi** 15](#_Toc100524336)

[**2.1.3. Các định nghĩa và các từ viết tắt được dùng** 15](#_Toc100524337)

[**2.1.4. Những người sử dụng tài liệu này** 16](#_Toc100524338)

[**2.2. Các yêu cầu kiểm thử** 16](#_Toc100524339)

[**2.2.1. Yêu cầu chức năng** 16](#_Toc100524340)

[**2.2.2. Yêu cầu phi chức năng** 16](#_Toc100524341)

[**2.2.3. Công cụ kiểm thử** 16](#_Toc100524342)

[**2.3. Các tiêu chí chấp nhận sản phẩm** 17](#_Toc100524343)

[**2.3.1. Tiêu chí tạm dừng** 17](#_Toc100524344)

[**2.3.2. Tiêu chí hoàn thành** 17](#_Toc100524345)

[**2.4. Các chiến lược kiểm thử** 17](#_Toc100524346)

[**2.5. Nguồn lực cho kiểm thử** 18](#_Toc100524347)

[**2.5.1. Nguồn nhân lực: Vai trò và trách nhiệm** 18](#_Toc100524348)

[**2.5.2. Nguồn lực hệ thống** 18](#_Toc100524349)

[**2.5.3. Môi trường test** 19](#_Toc100524350)

[**2.6. Các mốc kiểm thử** 19](#_Toc100524351)

[**2.7. Deliverables of test: Test Plan, Test Case, Test Reports** 20](#_Toc100524352)

# **LỜI MỞ ĐẦU**

Trong thời buổi công nghệ thông tin hiện nay có mặt khắp mọi lĩnh vực, nhu cầu sử dụng các phần mềm để thực hiện công việc được hiệu quả, nhanh chóng ngày càng tăng cao. Việc đảm bảo chất lượng phần mềm ngày càng quan trọng. Bên cạnh đó, để đáp ứng nhu cầu chia sẻ thông tin, cũng như truyền đạt thông tin một cách nhanh chóng thì website là phương tiện có khả năng đáp ứng nhu cầu đó.

Ngày nay, các website được phát triển một cách mạnh mẽ, phổ biến và nhanh chóng. Tuy nhiên với sự phát triển vượt bậc thì cũng có không ít các trở ngại dẫn đến việc website không hoạt động một cách hiệu quả. Do đó, cần thiết phải kiểm thử và đảm bảo chất lượng của website.

Với sự hướng dẫn của thầy, chúng em thực hiện đề tài “Kiểm định chất lượng phần mền: website bán mỹ phẩm” và thực hiện trên website đã được xây dựng nhưng chưa hoàn thiện, cũng như gặp nhiều thiếu sót trong quá trình phát triển sản phẩm. Do hạn chế về mặt kiến thức cho nên không thể tránh khỏi sai sót trong quá trình làm báo cáo, rất mong được sự giúp đỡ và chỉ dạy của thầy và các bạn.

# **Danh mục hình**

# **Danh mục bảng**

# **CHƯƠNG 1: TỔNG QUAN VỀ KIỂM THỬ PHẦN MỀM**

## **1.1. Các khái niệm cơ bản về kiểm thử phần mềm**

### **1.1.1. Kiểm thử phần mềm là gì?**

Kiểm thử phần mềm là quá trình khảo sát một hệ thống hay thành phần dưới những điều kiện xác định, quan sát và ghi lại các kết quả, và đánh giá một khía cạnh nào đó của hệ thống hay thành phần đó. (Theo Bảng chú giải thuật ngữ chuẩn IEEE của Thuật ngữ kỹ nghệ phần mềm- IEEE Standard Glossary of Software Engineering Terminology).

Kiểm thử phần mềm là quá trình thực thi một chương trình với mục đích tìm lỗi. (Theo “The Art of Software Testing” – Nghệ thuật kiểm thử phần mềm). Kiểm thử phần mềm là hoạt động khảo sát thực tiễn sản phẩm hay dịch vụ phần mềm trong đúng môi trường chúng dự định sẽ được triển khai nhằm cung cấp cho người có lợi ích liên quan những thông tin về chất lượng của sản phẩm hay dịch vụ phần mềm ấy. Mục đích của kiểm thử phần mềm là tìm ra các lỗi hay khiếm khuyết phần mềm nhằm đảm bảo hiệu quả hoạt động tối ưu của phần mềm trong nhiều ngành khác nhau. (Theo Bách khoa toàn thư mở Wikipedia).

Có thể định nghĩa một cách dễ hiểu như sau: Kiểm thử phần mềm là một tiến trình hay một tập hợp các tiến trình được thiết kế để đảm bảo mã hóa máy tính thực hiện theo cái mà chúng đã được thiết kế để làm, và không thực hiện bất cứ thứ gì không mong muốn. Đây là một pha quan trọng trong quá trình phát triển hệ thống, giúp cho người xây dựng hệ thống và khách hàng thấy được hệ thống mới đã đáp ứng yêu cầu đặt ra hay chưa?

### **1.1.2. Các phương pháp kiểm thử**

Có 2 phương pháp kiểm thử chính là: Kiểm thử tĩnh và Kiểm thử động.

***1.1.2.1. Kiểm thử tĩnh – Static testing***

Là phương pháp thử phần mềm đòi hỏi phải duyệt lại các yêu cầu và các đặc tả bằng tay, thông qua việc sử dụng giấy, bút để kiểm tra logic, lần từng chi tiết mà không cần chạy chương trình. Kiểu kiểm thử này thường được sử dụng bởi chuyên viên thiết kế người mà viết mã lệnh một mình.

Kiểm thử tĩnh cũng có thể được tự động hóa. Nó sẽ thực hiện kiểm tra toàn bộ bao gồm các chương trình được phân tích bởi một trình thông dịch hoặc biên dịch mà xác nhận tính hợp lệ về cú pháp của chương trình.

***1.1.2.2. Kiểm thử động – Dynamic testing***

Là phương pháp thử phần mềm thông qua việc dùng máy chạy chương trình để điều tra trạng thái tác động của chương trình. Đó là kiểm thử dựa trên các ca kiểm thử xác định bằng sự thực hiện của đối tượng kiểm thử hay chạy các chương trình. Kiểm thử động kiểm tra cách thức hoạt động của mã lệnh, tức là kiểm tra sự phản ứng vật lý từ hệ thống tới các biến luôn thay đổi theo thời gian. Trong kiểm thử động, phần mềm phải thực sự được biên dịch và chạy. Kiểm thử động thực sự bao gồm làm việc với phần mềm, nhập các giá trị đầu vào và kiểm tra xem liệu đầu ra có như mong muốn hay không. Các phương pháp kiểm thử động gồm có kiểm thử Unit – Unit Tests, Kiểm thử tích hợp – Intergration Tests, Kiểm thử hệ thống – System Tests, và Kiểm thử chấp nhận sản phẩm – Acceptance Tests

### **1.1.3. Các chiến lược kiểm thử**

Ba trong số những chiến lược kiểm thử thông dụng nhất bao gồm: Kiểm thử hộp đen, Kiểm thử hộp trắng, và Kiểm thử hộp xám.

***1.1.3.1. Kiểm thử hộp đen – Black box testing***

Một trong những chiến lược kiểm thử quan trọng là kiểm thử hộp đen, hướng dữ liệu, hay hướng vào/ra. Kiểm thử hộp đen xem chương trình như là một “hộp đen”. Mục đích của bạn là hoàn toàn không quan tâm về cách cư xử và cấu trúc bên trong của chương trình. Thay vào đó, tập trung vào tìm các trường hợp mà chương trình không thực hiện theo các đặc tả của nó.

Theo hướng tiếp cận này, dữ liệu kiểm tra được lấy chỉ từ các đặc tả.

**Các phương pháp kiểm thử hộp đen:**

* Phân lớp tương đương – Equivalence partitioning.
* Phân tích giá trị biên – Boundary value analysis.
* Kiểm thử mọi cặp – All-pairs testing.
* Kiểm thử fuzz – Fuzz testing.
* Kiểm thử dựa trên mô hình – Model-based testing.
* Ma trận dấu vết – Traceability matrix.
* Kiểm thử thăm dò – Exploratory testing.
* Kiểm thử dựa trên đặc tả – Specification-base testing.

Kiểm thử dựa trên đặc tả tập trung vào kiểm tra tính thiết thực của phần mềm theo những yêu cầu thích hợp. Do đó, kiểm thử viên nhập dữ liệu vào, và chỉ thấy dữ liệu ra từ đối tượng kiểm thử. Mức kiểm thử này thường yêu cầu các ca kiểm thử triệt để được cung cấp cho kiểm thử viên mà khi đó có thể xác minh là đối với dữ liệu đầu vào đã cho, giá trị đầu ra (hay cách thức hoạt động) có giống với giá trị mong muốn đã được xác định trong ca kiểm thử đó hay không. Kiểm thử dựa trên đặc tả là cần thiết, nhưng không đủ để ngăn chặn những rủi ro chắc chắn.

**Ưu, nhược điểm:**

Kiểm thử hộp đen không có mối liên quan nào tới mã lệnh, và kiểm thử viên chỉ rất đơn giản tâm niệm là: một mã lệnh phải có lỗi. Sử dụng nguyên tắc “ Hãy đòi hỏi và bạn sẽ được nhận”, những kiểm thử viên hộp đen tìm ra lỗi mà những lập trình viên đã không tìm ra. Nhưng mặt khác, người ta cũng nói kiểm thử hộp đen “giống như là đi trong bóng tối mà không có đèn vậy”, bởi vì kiểm thử viên không biết các phần mềm được kiểm tra thực sự được xây dựng như thế nào. Đó là lý do mà có nhiều trường hợp mà một kiểm thử viên hộp đen viết rất nhiều ca kiểm thử để kiểm tra một thứ gì đó mà đáng lẽ có thể chỉ cần kiểm tra bằng 1 ca kiểm thử duy nhất, và/hoặc một số phần của chương trình không được kiểm tra chút nào.

Do vậy, kiểm thử hộp đen có ưu điểm của “một sự đánh giá khách quan”, mặt khác nó lại có nhược điểm của “thăm dò mù”.

***1.1.3.2. Kiểm thử hộp trắng – White box testing***

Là một chiến lược kiểm thử khác, trái ngược hoàn toàn với kiểm thử hộp đen, kiểm thử hộp trắng hay kiểm thử hướng logic cho phép bạn khảo sát cấu trúc bên trong của chương trình. Chiến lược này xuất phát từ dữ liệu kiểm thử bằng sự kiểm thử tính logic của chương trình. Kiểm thử viên sẽ truy cập vào cấu trúc dữ liệu và giải thuật bên trong chương trình (và cả mã lệnh thực hiện chúng).

**Các phương pháp kiểm thử hộp trắng**

* Kiểm thử giao diện lập trình ứng dụng - API testing (application programming interface): là phương pháp kiểm thử của ứng dụng sử dụng các API công khai và riêng tư.
* Bao phủ mã lệnh – Code coverage: tạo các kiểm tra để đáp ứng một số tiêu chuẩn về bao phủ mã lệnh.
* Các phương pháp gán lỗi – Fault injection.
* Các phương pháp kiểm thử hoán chuyển – Mutation testing methods.
* Kiểm thử tĩnh – Static testing: kiểm thử hộp trắng bao gồm mọi kiểm thử tĩnh.

Phương pháp kiểm thử hộp trắng cũng có thể được sử dụng để đánh giá sự hoàn thành của một bộ kiểm thử mà được tạo cùng với các phương pháp kiểm thử hộp đen. Điều này cho phép các nhóm phần mềm khảo sát các phần của 1 hệ thống ít khi được kiểm tra và đảm bảo rằng những điểm chức năng quan trọng nhất đã được kiểm tra.

***1.1.3.3. Kiểm thử hộp xám – Gray box testing***

Kiểm thử hộp xám đòi hỏi phải có sự truy cập tới cấu trúc dữ liệu và giải thuật bên trong cho những mục đích thiết kế các ca kiểm thử, nhưng là kiểm thử ở mức người sử dụng hay mức hộp đen. Việc thao tác tới dữ liệu đầu vào và định dạng dữ liệu đầu ra là không rõ ràng, giống như một chiếc “hộp xám”, bởi vì đầu vào và đầu ra rõ ràng là ở bên ngoài “hộp đen” mà chúng ta vẫn gọi về hệ thống được kiểm tra. Sự khác biệt này đặc biệt quan trọng khi quản lý kiểm thử tích hợp – Intergartion testing giữa 2 mođun mã lệnh được viết bởi hai chuyên viên thiết kế khác nhau, trong đó chỉ giao diện là được đưa ra để kiểm thử. Kiểm thử hộp xám có thể cũng bao gồm cả thiết kế đối chiếu để quyết định, ví dụ: giá trị biên hay thông báo lỗi.

### **1.1.4. Các cấp độ kiểm thử phần mềm**

Kiểm thử phần mềm gồm có các cấp độ: Kiểm thử đơn vị, Kiểm thử tích hợp, Kiểm thử hệ thống và Kiểm thử chấp nhận sản phẩm.

***1.1.4.1. Kiểm thử đơn vị - Unit Test***

Một đơn vị là một thành phần phần mềm nhỏ nhất mà ta có thể kiểm thử được. Ví dụ: các hàm (Function), thủ tục (Procedure), lớp (Class) hay phương thức (Method) đều có thể được xem là Unit.

Vì Unit được chọn để kiểm tra thường có kích thước nhỏ và chức năng hoạt động đơn giản, chúng ta không khó khăn gì trong việc tổ chức kiểm thử, ghi nhận và phân tích kết quả kiểm thử. Nếu phát hiện lỗi, việc xác định nguyên nhân và khắc phục cũng tương đối dễ dàng vì chỉ khoanh vùng trong một đơn thể Unit đang kiểm tra. Một nguyên lý đúc kết từ thực tiễn: thời gian tốn cho Unit Test sẽ được đền bù bằng việc tiết kiệm rất nhiều thời gian và chi phí cho việc kiểm thử và sửa lỗi ở các mức kiểm thử sau đó.

Unit Test thường do lập trình viên thực hiện. Công đoạn này cần được thực hiện càng sớm càng tốt trong giai đoạn viết code và xuyên suốt chu kỳ phát triển phần mềm. Thông thường, Unit Test đòi hỏi kiểm thử viên có kiến thức về thiết kế và code của chương trình. Mục đích của Unit Test là bảo đảm thông tin được xử lý và xuất (khỏi Unit) là chính xác, trong mối tương quan với dữ liệu nhập và chức năng của Unit. Điều này thường đòi hỏi tất cả các nhánh bên trong Unit đều phải được kiểm tra để phát hiện nhánh phát sinh lỗi. Một nhánh thường là một chuỗi các lệnh được thực thi trong một Unit. Ví dụ: chuỗi các lệnh sau điều kiện If và nằm giữa then ... else là một nhánh. Thực tế việc chọn lựa các nhánh để đơn giản hóa việc kiểm thử và quét hết Unit đòi hỏi phải có kỹ thuật, đôi khi phải dùng thuật toán để chọn lựa.

Cùng với các mục kiểm thử khác, Unit Test cũng đòi hỏi phải chuẩn bị trước các ca kiểm thử (Test case) hoặc kịch bản kiểm thử (Test script), trong đó chỉ định rõ dữ liệu đầu vào, các bước thực hiện và dữ liệu đầu ra mong muốn. Các Test case và Test script này nên được giữ lại để tái sử dụng.

***1.1.4.2. Kiểm thử tích hợp – Intergration Test***

Integration test kết hợp các thành phần của một ứng dụng và kiểm thử như một ứng dụng đã hoàn thành. Trong khi Unit Test kiểm tra các thành phần và Unit riêng lẻ thì Intgration Test kết hợp chúng lại với nhau và kiểm tra sự giao tiếp giữa chúng.

Hai mục tiêu chính của Integration Test:

* Phát hiện lỗi giao tiếp xảy ra giữa các Unit.
* Tích hợp các Unit đơn lẻ thành các hệ thống nhỏ (Subsystem) và cuối cùng là nguyên hệ thống hoàn chỉnh (System) chuẩn bị cho kiểm thử ở mức hệ thống (System Test).

Trong Unit Test, lập trình viên cố gắng phát hiện lỗi liên quan đến chức năng và cấu trúc nội tại của Unit. Có một số phép kiểm thử đơn giản trên giao tiếp giữa Unit với các thành phần liên quan khác, tuy nhiên mọi giao tiếp liên quan đến Unit chỉ thật sự được kiểm tra đầy đủ khi các Unit tích hợp với nhau trong khi thực hiện Integration Test.

Trừ một số ít ngoại lệ, Integration Test chỉ nên thực hiện trên những Unit đã được kiểm tra cẩn thận trước đó bằng Unit Test, và tất cả các lỗi mức Unit đã được sửa chữa. Một số người hiểu sai rằng Unit một khi đã qua giai đoạn Unit Test với các giao tiếp giả lập thì không cần phải thực hiện Integration Test nữa. Thực tế việc tích hợp giữa các Unit dẫn đến những tình huống hoàn toàn khác. Một chiến lược cần quan tâm trong Integration Test là nên tích hợp dần từng Unit. Một Unit tại một thời điểm được tích hợp vào một nhóm các Unit khác đã tích hợp trước đó và đã hoàn tất các đợt Integration Test trước đó. Lúc này, ta chỉ cần kiểm thử giao tiếp của Unit mới thêm vào với hệ thống các Unit đã tích hợp trước đó, điều này sẽ làm cho số lượng ca kiểm thử giảm đi rất nhiều, và sai sót sẽ giảm đáng kể.

Có 4 loại kiểm thử trong Integration Test:

* **Kiểm thử cấu trúc (Structure Test):** Tương tự White Box Test, kiểm thử cấu trúc nhằm bảo đảm các thành phần bên trong của một chương trình chạy đúng và chú trọng đến hoạt động của các thành phần cấu trúc nội tại của chương trình chẳng hạn các câu lệnh và nhánh bên trong.
* **Kiểm thử chức năng (Functional Test)**: Tương tự Black Box Test, kiểm thử chức năng chỉ chú trọng đến chức năng của chương trình, mà khôngquan tâm đến cấu trúc bên trong, chỉ khảo sát chức năng của chương trìnhtheo yêu cầu kỹ thuật.
* **Kiểm thử hiệu năng (Performance Test):** Kiểm thử việc vận hành của hệ thống.
* **Kiểm thử khả năng chịu tải (Stress Test):** Kiểm thử các giới hạn của hệ thống.

***1.1.4.3. Kiểm thử hệ thống – System Test***

Mục đích System Test là kiểm thử thiết kế và toàn bộ hệ thống (sau khi tích hợp) có thỏa mãn yêu cầu đặt ra hay không.

System Test bắt đầu khi tất cả các bộ phận của phần mềm đã được tích hợp thành công. Thông thường loại kiểm thử này tốn rất nhiều công sức và thời gian. Trong nhiều trường hợp, việc kiểm thử đòi hỏi một số thiết bị phụ trợ, phần mềm hoặc phần cứng đặc thù, đặc biệt là các ứng dụng thời gian thực, hệ thống phân bố, hoặc hệ thống nhúng. Ở mức độ hệ thống, người kiểm thử cũng tìm kiếm các lỗi, nhưng trọng tâm là đánh giá về hoạt động, thao tác, sự tin cậy và các yêu cầu khác liên quan đến chất lượng của toàn hệ thống.

Điểm khác nhau then chốt giữa Integration Test và System Test là System Test chú trọng các hành vi và lỗi trên toàn hệ thống, còn Integration Test chú trọng sự giao tiếp giữa các đơn thể hoặc đối tượng khi chúng làm việc cùng nhau. Thông thường ta phải thực hiện Unit Test và Integration Test để bảo đảm mọi Unit và sự tương tác giữa chúng hoạt động chính xác trước khi thực hiện System Test.

Sau khi hoàn thành Integration Test, một hệ thống phần mềm đã được hình thành cùng với các thành phần đã được kiểm tra đầy đủ. Tại thời điểm này, lập trình viên hoặc kiểm thử viên bắt đầu kiểm thử phần mềm như một hệ thống hoàn chỉnh. Việc lập kế hoạch cho System Test nên bắt đầu từ giai đoạn hình thành và phân tích các yêu cầu.

System Test kiểm thử cả các hành vi chức năng của phần mềm lẫn các yêu cầu về chất lượng như độ tin cậy, tính tiện lợi khi sử dụng, hiệu năng và bảo mật. Mức kiểm thử này đặc biệt thích hợp cho việc phát hiện lỗi giao tiếp với phần mềm hoặc phần cứng bên ngoài, chẳng hạn các lỗi "tắc nghẽn" (deadlock) hoặc chiếm dụng bộ nhớ. Sau giai đoạn System Test, phần mềm thường đã sẵn sàng cho khách hàng hoặc người dùng cuối cùng kiểm thử chấp nhận sản phẩm (Acceptance Test) hoặc dùng thử (Alpha/Beta Test).

Đòi hỏi nhiều công sức, thời gian và tính chính xác, khách quan, System Test thường được thực hiện bởi một nhóm kiểm thử viên hoàn toàn độc lập với nhóm phát triển dự án. Bản thân System Test lại gồm nhiều loại kiểm thử khác nhau, phổ biến nhất gồm:

* **Kiểm thử chức năng (Functional Test):** Bảo đảm các hành vi của hệ thống thỏa mãn đúng yêu cầu thiết kế.
* **Kiểm thử hiệu năng (Performance Test):** Bảo đảm tối ưu việc phân bổ tài nguyên hệ thống (ví dụ bộ nhớ) nhằm đạt các chỉ tiêu như thời gian xử lý hay đáp ứng câu truy vấn, ...
* **Kiểm thử khả năng chịu tải (Stress Test hay Load Test):** Bảo đảm hệ thống vận hành đúng dưới áp lực cao (ví dụ nhiều người truy xuất cùng lúc). Stress Test tập trung vào các trạng thái tới hạn, các "điểm chết", các tình huống bất thường như đang giao dịch thì ngắt kết nối (xuất hiện nhiều trong kiểm tra thiết bị như POS, ATM...)...
* **Kiểm thử cấu hình (Configuration Test).**
* **Kiểm thử bảo mật (Security Test):** Bảo đảm tính toàn vẹn, bảo mật của dữ liệu và của hệ thống.
* **Kiểm thử khả năng phục hồi (Recovery Test):** Bảo đảm hệ thống có khả năng khôi phục trạng thái ổn định trước đó trong tình huống mất tài nguyên hoặc dữ liệu; đặc biệt quan trọng đối với các hệ thống giao dịch như ngân hàng trực tuyến...

Nhìn từ quan điểm người dùng, các cấp độ kiểm thử trên rất quan trọng: Chúng bảo đảm hệ thống đủ khả năng làm việc trong môi trường thực.

Lưu ý là không nhất thiết phải thực hiện tất cả các loại kiểm thử nêu trên. Tùy yêu cầu và đặc trưng của từng hệ thống, tuỳ khả năng và thời gian cho phép của dự án, khi lập kế hoạch, người Quản lý dự án sẽ quyết định áp dụng những loại kiểm thử nào.

***1.1.4.4. Kiểm thử chấp nhận sản phẩm – Acceptance Test***

Thông thường, sau giai đoạn System Test là Acceptance Test, được khách hàng thực hiện (hoặc ủy quyền cho một nhóm thứ ba thực hiện). Mục đích của Acceptance Test là để chứng minh phần mềm thỏa mãn tất cả yêu cầu của khách hàng và khách hàng chấp nhận sản phẩm (và trả tiền thanh toán hợp đồng).

Acceptance Test có ý nghĩa hết sức quan trọng, mặc dù trong hầu hết mọi trường hợp, các phép kiểm thử của System Test và Acceptance Test gần như tương tự, nhưng bản chất và cách thức thực hiện lại rất khác biệt.

Đối với những sản phẩm dành bán rộng rãi trên thị trường cho nhiều người sử dụng, thông thường sẽ thông qua hai loại kiểm thử gọi là kiểm thử Alpha – Alpha Test và kiểm thử Beta – Beta Test. Với Alpha Test, người dùng kiểm thử phần mềm ngay tại nơi phát triển phần mềm, lập trình viên sẽ ghi nhận các lỗi hoặc phản hồi, và lên kế hoạch sửa chữa. Với Beta Test, phần mềm sẽ được gửi tới cho người dùng để kiểm thử ngay trong môi trường thực, lỗi hoặc phản hồi cũng sẽ gửi ngược lại cho lập trình viên để sửa chữa.

Thực tế cho thấy, nếu khách hàng không quan tâm và không tham gia vào quá trình phát triển phần mềm thì kết quả Acceptance Test sẽ sai lệch rất lớn, mặc dù phần mềm đã trải qua tất cả các kiểm thử trước đó. Sự sai lệch này liên quan đến việc hiểu sai yêu cầu cũng như sự mong chờ của khách hàng. Ví dụ đôi khi một phần mềm xuất sắc vượt qua các phép kiểm thử về chức năng thực hiện bởi nhóm thực hiện dự án, nhưng khách hàng khi kiểm thử sau cùng vẫn thất vọng vì bố cục màn hình nghèo nàn, thao tác không tự nhiên, không theo tập quán sử dụng của khách hàng v.v...

Gắn liền với giai đoạn Acceptance Test thường là một nhóm những dịch vụ và tài liệu đi kèm, phổ biến như hướng dẫn cài đặt, sử dụng v.v... Tất cả tài liệu đi kèm phải được cập nhật và kiểm thử chặt chẽ.

***1.1.4.5. Một số cấp độ kiểm thử khác***

Ngoài các cấp độ trên, còn một số cấp độ kiểm thử khác như:

**Kiểm thử hồi quy – Regression Testing:**

Theo chuẩn IEEE610.12-90, kiểm thử hồi quy là “sự kiểm tra lại có lựa chọn của một hệ thống hay thành phần để xác minh là những sự thay đổi không gây ra những hậu quả không mong muốn...”. Trên thực tế, quan niệm này là chỉ ra rằng phần mềm mà đã qua được các kiểm tra trước đó vẫn có thể được kiểm tra lại. Beizer định nghĩa đó là sự lặp lại các kiểm tra để chỉ ra rằng cách hoạt động của phần mềm không bị thay đổi, ngoại trừ tới mức như yêu cầu. Hiển nhiên là sự thỏa hiệp phải được thực hiện giữa sự đảm bảo được đưa ra bởi kiểm thử hồi quy mỗi lần thực hiện một sự thay đổi và những tài nguyên được yêu cầu thực hiện điều đó.

**Kiểm thử tính đúng – Correctness testing:**

Tính đúng đắn là yêu cầu tối thiểu của phần mềm, là mục đích chủ yếu của kiểm thử. Kiểm thử tính đúng sẽ cần một kiểu người đáng tin nào đó, để chỉ ra cách hoạt động đúng đắn từ cách hoạt động sai lầm. Kiểm thử viên có thể biết hoặc không biết các chi tiết bên trong của các mođun phần mềm được kiểm thử, ví dụ luồng điều khiển, luồng dữ liệu, v.v .... Do đó, hoặc là quan điểm hộp trắng, hoặc là quan điểm hộp đen có thể được thực hiện trong kiểm thử phần mềm.

### **1.1.5. Các phương pháp kiểm thử con người**

Hai phương pháp kiểm thử con người chủ yếu là Code Inspections và Walkthroughs. Hai phương pháp này bao gồm một nhóm người đọc và kiểm tra theo mã lệnh của chương trình. Mục tiêu của chúng là để tìm ra lỗi mà không gỡ lỗi.

Một Inspection hay Walkthrough là 1 sự cải tiến của phương pháp kiểm tra mà lập trình viên đọc chương trình của họ trước khi kiểm thử nó. Inspections và Walkthroughs hiệu quả hơn là bởi vì những người khác sẽ kiểm thử chương trình tốt hơn chính tác giả của chương trình đó.

Inspections/Walkthroughs và kiểm thử bằng máy tính bổ sung cho nhau. Hiệu quả tìm lỗi sẽ kém đi nếu thiếu đi 1 trong 2 phương pháp. Và đối với việc sửa đổi chương trình cũng nên sử dụng các phương pháp kiểm thử này cũng như các kỹ thuật kiểm thử hồi quy.

***1.1.5.1. Tổng duyệt – Walkthrough***

Walkthrough là một thuật ngữ mô tả sự xem xét kỹ lưỡng của một quá trình ở mức trừu tượng trong đó nhà thiết kế hay lập trình viên lãnh đạo các thành viên trong nhóm và những người có quan tâm khác thông qua một sản phẩm phần mềm, và những người tham gia đặt câu hỏi, và ghi chú những lỗi có thể có, sự vi phạm các chuẩn phát triển và các vấn đề khác. Walkthrough mã lệnh là 1 tập các thủ tục và các công nghệ dò lỗi cho việc đọc nhóm mã lệnh. Trong một Walkthrough, nhóm các nhà phát triển – với 3 hoặc 4 thành viên là tốt nhất – thực hiện xét duyệt lại. Chỉ 1 trong các thành viên là tác giả của chương trình.

Một ưu điểm khác của walkthroughs, hiệu quả trong chi phí gỡ lỗi, là 1 thực tế mà khi một lỗi được tìm thấy, nó thường được định vị chính xác trong mã lệnh. Thêm vào đó, phương pháp này thường tìm ra 1 tập các lỗi, cho phép sau đó các lỗi đó được sửa tất cả với nhau. Mặt khác, kiểm thử dựa trên máy tính, chỉ tìm ra triệu chứng của lỗi (chương trình không kết thúc hoặc đưa ra kết quả vô nghĩa), và các lỗi thường được tìm ra và sửa lần lượt từng lỗi một.

***1.1.5.2. Thanh tra mã nguồn – Code Inspection***

Thanh tra mã nguồn là 1 tập hợp các thủ tục và các kỹ thuật dò lỗi cho việc đọc các nhóm mã lệnh. Một nhóm kiểm duyệt thường gồm 4 người. Một trong số đó đóng vai trò là người điều tiết – một lập trình viên lão luyện và không được là tác giả của chương trình và phải không quen với các chi tiết của chương trình. Người điều tiết có nhiệm vụ: phân phối nguyên liệu và lập lịch cho các buổi kiểm duyệt, chỉ đạo phiên làm việc, ghi lại tất cả các lỗi được tìm thấy và đảm bảo là các lỗi sau đó được sửa. Thành viên thứ hai là một lập trình viên. Các thành viên còn lại trong nhóm thường là nhà thiết kế của chương trình (nếu nhà thiết kế khác lập trình viên) và một chuyên viên kiểm thử.

## **1.2. Nguyên tắc kiểm thử phần mềm**

Để kiểm thử đạt hiệu quả thì khi tiến hành kiểm thử phần mềm cần phải tuân thủ một số quy tắc sau:

**Quy tắc 1**: Một phần quan trọng của 1 ca kiểm thử là định nghĩa của đầu ra hay kết quả mong muốn.

**Quy tắc 2**: Lập trình viên nên tránh tự kiểm tra chương trình của mình.

**Quy tắc 3**: Nhóm lập trình không nên kiểm thử chương trình của chính họ.

**Quy tắc 4**: Kiểm tra thấu đáo mọi kết quả của mỗi kiểm tra.

**Quy tắc 5**: Các ca kiểm thử phải được viết cho các trạng thái đầu vào không hợp lệ và không mong muốn, cũng như cho các đầu vào hợp lệ và mong muốn.

**Quy tắc 6**: Khảo sát 1 chương trình để xem liệu chương trình có thực hiện cái mà nó cần thực hiện chỉ là 1 phần, phần còn lại là xem liệu chương trình có thực hiện cái mà nó không cần phải thực hiện hay không.

**Quy tắc 7**: Tránh các ca kiểm thử bâng quơ trừ khi chương trình thực sự là 1 chương trình bâng quơ.

**Quy tắc 8**: Không dự kiến kết quả của kiểm thử theo giả thiết ngầm là không tìm thấy lỗi.

**Quy tắc 9**: Xác suất tồn tại lỗi trong 1 đoạn chương trình là tương ứng với số lỗi đã tìm thấy trong đoạn đó.

**Quy tắc 10**: Kiểm thử là 1 nhiệm vụ cực kỳ sáng tạo và có tính thử thách trí tuệ.

# **CHƯƠNG 2: THIẾT KẾ TEST PLAN**

## **2.1. Giới thiệu chung**

### **2.1.1. Tổng quan**

<https://thegioiskinfood.com/> là một website dành cho tất cả mọi người, với nhiều mặt hàng mỹ phẩm khác nhau giúp khách hàng nhất là phái đẹp có thể mua sắm, tham khảo và đặt hàng các loại mỹ phẩm để chăm sóc da và làm đẹp bản thân.

### **2.1.2. Phạm vi**

Tài liệu kế hoạch kiểm thử này được áp dụng cho việc kiểm thử những chức năng

của website <https://thegioiskinfood.com/>

Các dịch vụ, chức năng cần kiểm tra:

* Tìm kiếm sản phẩm.
* Đăng nhập.
* Đặt hàng.
* Thêm sản phẩm mới vào giỏ hàng.
* Thanh toán sản phẩm ở giỏ hàng.

### **2.1.3. Các định nghĩa và các từ viết tắt được dùng**

Các định nghĩa

|  |  |
| --- | --- |
| **Thuật Ngữ** | **Định Nghĩa** |
| Comment | Là nơi khách hàng đánh giá về sản phẩm đang được bán. |
| Cart | Là nơi khách thêm sản phẩm muốn mua và đăt hàng. |
| Product portfolio | Là danh mục sản phẩm để phân chia thành từng loại mỹ phẩm. |

Các từ viết tắt

|  |  |
| --- | --- |
| **Từ viết tắt** | **Mô tả** |
| IE11 | Trình duyệt Web Windows Internet Explorer 11 |

### **2.1.4. Những người sử dụng tài liệu này**

Tài liệu này sử dụng cho Test Manager, Test Designer, Tester.

## **2.2. Các yêu cầu kiểm thử**

### **2.2.1. Yêu cầu chức năng**

Tìm kiếm: kiểm tra chức năng tìm kiếm cơ bản:

* Tên sản phẩm

Kiểm tra chức năng xem thông tin sản phẩm gồm các thông tin:

* Tên sản phẩm
* Giá
* Nhà sản xuất

Kiểm tra chức năng thanh toán ở giỏ hàng

* Nhập xuất hóa đơn.
* Thông tin hóa đơn.
* Tính tiền dư cho khách hàng.

Tài khoản

* Kiểm tra chức năng đăng ký, đăng nhập vào hệ thống

### **2.2.2. Yêu cầu phi chức năng**

* Tốc độ xử lí của web.
* Giao điện thân thiện với người dùng.
* Thông tin hiển thị dễ nhìn.

### **2.2.3. Công cụ kiểm thử**

|  |  |  |  |
| --- | --- | --- | --- |
| **Hoạt động** | **Công cụ** | **Nhà cung cấp** | **Phiên bản** |
| Quản lý Test Case | Microsoft Office Excel | Microsoft | 2010 |
| Quản lý Configuration | Microsoft Office Word | Microsoft | 2010 |
| Defect Tracking | Microsoft Office Excel | Microsoft | 2010 |
| Quản lý tiến độ kiểm thử | Microsoft Project Professional | Microsoft | 2010 |

## **2.3. Các tiêu chí chấp nhận sản phẩm**

### **2.3.1. Tiêu chí tạm dừng**

Nếu các thành viên trong nhóm báo cáo rằng có 40% trường hợp thử nghiệm thất bại, tạm dừng thử nghiệm cho đến khi nhóm phát triển sửa chữa tất cả các trường hợp thất bại.

### **2.3.2. Tiêu chí hoàn thành**

Chỉ định các tiêu chí biểu thị hoàn thành thành công giai đoạn thử nghiệm

* Tốc độ chạy bắt buộc là 100% trừ khi có lý do rõ ràng.
* Tỷ lệ vượt qua là 80%, đạt được tỷ lệ vượt qua là bắt buộc.

## **2.4. Các chiến lược kiểm thử**

|  |  |
| --- | --- |
| Mục đích kiểm tra | Đảm bảo các chức năng được kiểm tra hoạt động chính xã theo đặc tả yêu cầu. |
| Kỹ thuật | Thực thi tất cả các trường hợp có thể có cho mỗi nhóm chức năng sử dụng dữ liệu hợp lệ để xác định:   * Kết quả mong đợi khi dữ liệu hợp lệ được sử dụng. * Cảnh báo phù hợp hiện ra khi dữ liệu không hợp lệ được sử dụng. |
| Tiêu chuẩn dùng | Tất cả các testcase đã được thiết kế đều được thực thi.  Tất cả các lỗi tìm thấy đều được ghi nhận lý do rõ ràng để có thể giúp cho người lập trình dễ dàng khắc phục. |
| Chịu trách nhiệm kiểm thử | Test Designer/Tester. |
| Các kiểm thử | Kiểm thử bằng tay thủ công, tuần tự theo các bước định nghĩa trong testcase. |
| Xử lý ngoại lệ | Liệt kê tất cả các vấn đề liên quan phát sinh trong quá trình thực thi kiểm thử. |

## **2.5. Nguồn lực cho kiểm thử**

### **2.5.1. Nguồn nhân lực: Vai trò và trách nhiệm**

|  |  |
| --- | --- |
| **Thành viên** | **Vai trò** |
| Từ Huệ Sơn | Test Manager / Test Designer / Tester: Lập kế hoạch kiểm thử, quản lý tiến độ hoạt động kiểm thử, thiết kế các testcase bổ sung và thực thi các testcase bổ sung |
| Lê Đức Tài | Test Designer / Tester: Thiết kế và viết các test case, thực thi các testcase cho chức năng đăng nhập tài khoản, xem lại Test Plan |
| Lư Phước Toàn | Test Designer / Tester: Thiết kế và viết các testcase bổ sung, thực thi các testcase bổ sung, đọc các tài liệu tham khảo để cung cấp thông tin cần thiết cho nhóm, xem lại Test Plan |
| Huỳnh Mỹ Trân | Test Designer / Tester: Thiết kế và viết các testcase bổ sung, thực thi các testcase bổ sung, đọc các tài liệu tham khảo để cung cấp thông tin cần thiết cho nhóm, xem lại Test Plan |
| Trần Thành Tâm | Test Designer / Tester: Thiết kế và viết các test case, thực thi các testcase cho 2 chức năng Đặt hàng và thanh toán hóa đơn đặt hàng |
| Nguyễn Trung Hậu | Test Designer / Tester: Thiết kế và viết các test case, thực thi các testcase cho chức năng Tìm kiếm |

### **2.5.2. Nguồn lực hệ thống**

Phần cứng

Máy tính cá nhân có kết nối mạng Internet

|  |  |  |  |
| --- | --- | --- | --- |
| **CPU** | **RAM** | **HDD** | **Architecture** |
| 2.2GHz Intel Core i7-8750H | 16 GB | 1TB | 64 bit |

Phần mềm

|  |  |  |
| --- | --- | --- |
| **Tên phần mềm** | **Phiển bản** | **Loại** |
| Mozilla Firefox | 98.0.2 | Trình duyệt Web |
| Google Chrome | 96 | Trình duyệt Web |
| Microsoft Windows 10 Single Language | 10 | Hệ điều hành |
| Adobe Flash Player plugin | 2022 | Tích hợp vào trình duyệt Web |

Máy tính cá nhân có kết nối mạng Internet để có thể truy cập vào trang web mỹ phẩm <https://thegioiskinfood.com/> bằng trình duyệt.

Các chức năng của trang web <https://thegioiskinfood.com/> được kiểm tra trên các trình duyệt Mozilla Firefox 98.0.2 và Google Chrome 96. Các trình duyệt này sử dụng phần mềm tích hợp Adobe Flash Player 2022 để chạy các sản phẩm.

Hệ điều hành được sử dụng là Microsoft Windows 10 Single Language.

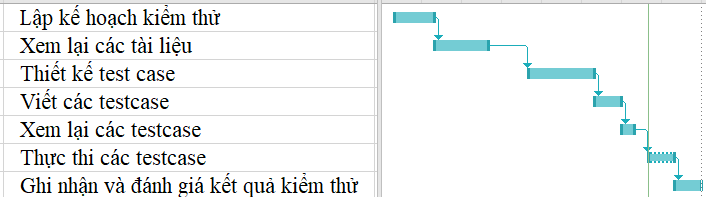
### **2.5.3. Môi trường test**

Test trên môi trường website

## **2.6. Các mốc kiểm thử**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Milestone** | **Deliverables** | **Duration (day)** | **Start Date** | **End Date** |
| Lập kế hoạch kiểm thử | Tài liệu Test Plan | 3 days | Tue 22/3/22 | Thu 24/3/22 |
| Xem lại các tài liệu | Tài liệu Test Plan | 3 days | Tue 22/3/22 | Thu 24/3/22 |
| Thiết kế test case | Tài liệu Test Case | 2 days | Fri 25/3/22 | Mon 28/3/22 |
| Viết các testcase | Tài liệu Testcase | 3 days | Fri 1/4/22 | Tue 5/4/22 |
| Xem lại các testcase | Tài liệu Testcase | 2 days | Wed 6/4/22 | Thu 7/4/22 |
| Thực thi các testcase | Tài liệu Testcase | 1 day | Fri 8/4/22 | Fri 8/4/22 |
| Ghi nhận và đánh giá kết quả kiểm thử | Tài liệu Testcase | 2 days | Sun 10/4/22 | Mon 11/4/22 |

Biểu đồ gantt



## **2.7. Deliverables of test: Test Plan, Test Case, Test Reports**

* Liệt kê tên sản phẩm cần bàn giao
* Ngày bàn giao
* Người bàn giao
* Bàn giao cho ai.