TRƯỜNG ĐẠI HỌC KINH TẾ

**KHOA THỐNG KÊ – TIN HỌC**

–––––––––––––––––––––––––––––––

**BÁO CÁO THỰC TẬP NGHỀ NGHIỆP**

**KIỂM THỬ WEBSITE BÁN HÀNG PRODUCT STORE**

**Sinh viên thực hiện:** Trương Thị Kiều Diểm

**Lớp:**  44K14

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

**Đà Nẵng, 2021**

**LỜI CẢM ƠN**

**LỜI CAM ĐOAN**

Quy định:

Soạn thảo trên trang A4 (trang dọc, lề trái: 3.5cm; trên, phải, dưới: 2.5 cm), Font Times New Roman, canh đều 2 bên, size 13, cách dòng 1.5, cách đoạn trên 6pt, cách đoạn dưới 3pt; hình và bảng soạn theo caption, chèn trích dẫn chéo (Cross-reference) cho bảng và hình; các danh mục hình, bảng, mục lục làm tự động; đánh số trang như file mẫu (bìa không có số trang, danh mục + mục lục số trang theo i, ii, iii…, nội dung chính theo 1,2,3…)

# 

[LỜI MỞ ĐẦU 1](#_Toc76756984)

[CHƯƠNG 1: TỔNG QUAN VỀ LÝ THUYẾT 2](#_Toc76756985)

[1.1. Kiểm thử phần mềm và mục đích KTPM 2](#_Toc76756986)

[1.1.1. Kiểm thử phần mềm 2](#_Toc76756987)

[1.1.2. Mục đích 2](#_Toc76756988)

[1.2. Quy trình kiếm thử phần mềm 2](#_Toc76756989)

[1.2.1. Giai đoạn 1: Phân tích yêu cầu 3](#_Toc76756990)

[1.2.2. Giai đoạn 2: Lập kế hoạch kiểm thử 3](#_Toc76756991)

[1.2.3. Giai đoạn 3: Thiết kế tài liệu kiểm thử 4](#_Toc76756992)

[1.2.4. Giai đoạn 4: Thiết lập môi trường kiểm thử 4](#_Toc76756993)

[1.2.5. Giai đoạn 5: Thực hiện kiểm thử 5](#_Toc76756994)

[1.2.6. Giai đoạn 6: Kết thúc quy trình kiểm thử 5](#_Toc76756995)

[1.3. Các mức độ kiểm thử phần mềm 6](#_Toc76756996)

[1.3.1. Kiểm thử mức đơn vị (Unit testing) 6](#_Toc76756997)

[1.3.2. Kiểm thử tích hợp (Integration testing) 6](#_Toc76756998)

[1.3.3. Kiểm thử hệ thống (System testing) 6](#_Toc76756999)

[1.3.4. Kiểm thử chấp nhận (Acceptance testing) 6](#_Toc76757000)

[1.4. Các kỹ thuật kiểm thử phần mềm 7](#_Toc76757001)

[1.4.1. Kiểm thử hộp trắng 7](#_Toc76757002)

[1.4.2. Kiểm thử hộp đen 7](#_Toc76757003)

[1.4.3. Kiểm thử hộp xám 7](#_Toc76757004)

[1.5. Test case và kỹ thuật thiết kế test case 7](#_Toc76757005)

[1.5.1. Test case là gì 7](#_Toc76757006)

[1.5.2. Cấu trúc của test case 7](#_Toc76757007)

[1.5.3. Các kỹ thuật thiết kế test case 8](#_Toc76757008)

[1.6. Bug và quy trình quản lý bug 8](#_Toc76757009)

[1.6.1. Bug là gì và phân loại bug 8](#_Toc76757010)

[1.6.2. Cấu trúc report bug 9](#_Toc76757011)

[1.6.3. Quy trình quản lý lỗi 10](#_Toc76757012)

[1.6.4. Vòng đời của lỗi 10](#_Toc76757013)

[CHƯƠNG 2. TỔNG QUAN VỀ HỆ THỐNG CẤN TEST 11](#_Toc76757014)

[2.1. Mô tả website 11](#_Toc76757015)

[2.2. Mô tả về chức năng kiểm thử của website 11](#_Toc76757016)

[CHƯƠNG 3. THIẾT KẾ TEST CASE VÀ THỰC HIỆN KIỂM THỬ 12](#_Toc76757017)

[3.1. Test case 12](#_Toc76757018)

[3.2. Thực hiện kiểm thử 12](#_Toc76757019)

[3.3. Report bug 12](#_Toc76757020)

[KẾT LUẬN 13](#_Toc76757021)

[TÀI LIỆU THAM KHẢO 14](#_Toc76757022)

DANH MỤC HÌNH ẢNH

**DANH MỤC BẢNG BIỂU**

**DANH MỤC TỪ VIẾT TẮT**

# LỜI MỞ ĐẦU

1. **Mục tiêu của đề tài**

* Nghiên cứu về kiểm thử phần mềm và các kỹ thuật kiểm thử phần mềm.
* Thực hiện Manual test để kiểm thử chức năng Cart của website Product Store.

1. **Nhiệm vụ của đề tài**

* Nghiên cứu về kiểm thử phần mềm và các kỹ thuật kiểm thử phần mềm.
* Thiết kế test case để chuẩn bị kiểm thử.
* Thực hiện kiểm thử chức năng Cart của website Product Store.

1. **Kết cấu của đề tài**

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

* **Chương 1**: Cơ sở lý thuyết
* **Chương 2**: Tổng quan về hệ thống cần test
* **Chương 3**: Thiết kế test case và thực hiện test

Kết luận và hướng phát triển

# CHƯƠNG 1: TỔNG QUAN VỀ LÝ THUYẾT

## Kiểm thử phần mềm và mục đích KTPM

### Kiểm thử phần mềm

* Kiểm thử phần mềm là phương pháp kiểm tra sản phẩm phần mềm thực tế tạo ra có phù hợp với mong đợi và phù hợp với các yêu cầu của khách hàng hay không. Đảm bảo rằng sản phẩm không có bất kỳ lỗi nào khi bàn giao cho khách hàng.
* Có hai loại kiểm thử phần mềm: Kiểm thử phần mềm thủ công (Manual Test) và Kiểm thử phần mềm tự động (Automation Test).

### Mục đích

* KTPM để tìm ra các lỗi xảy ra trong quá trình phát triển phần mềm. Sản phẩm cuối cùng đưa ra không có bất kỳ lỗi nào 🡪 duy trì lòng trung thành của khách hàng và lặp lại các hoạt động kinh doanh.
* KTPM để kiểm tra hiệu suất, hiệu quả của sản phẩm phần mềm, phát hiện lỗi sớm sẽ bớt tốn kém về chi phí.
* KTPM để đảm bảo chất lượng phần mềm. Tránh việc các sản phẩm đưa ra cho khách hàng có chất lượng kém 🡪 sản phẩm không được chấp nhận 🡪 gây tổn thất cho doanh nghiệp.

## Quy trình kiếm thử phần mềm

Software Testing Life Cycle (STLC) là quy trình kiểm thử được thực hiện theo hệ thống và có kế hoạch rõ ràng nhằm đảm bảo đáp ứng về mục tiêu chất lượng phần mềm. Quy trình kiểm thử phần mềm gồm 6 giai đoạn:

### Giai đoạn 1: Phân tích yêu cầu

* Đầu vào: Tài liệu đặc tả yêu cầu, Tài liệu thiêt kế phần mềm, Tài liệu khách hàng yêu cầu về các tiêu chí của sản phẩm.
* Hoạt động:
  + Phân tích yêu cầu là giai đoạn đầu tiên của Quy trình kiểm thử phần mềm.
  + Đọc hiểu và nghiên cứu yêu cầu của tài liệu đặc tả để nắm bắt yêu cầu mà dự án đưa ra gồm: Yêu cầu kiểm thử chức năng/ phi chức năng nào?
  + Đề xuất các vấn đề cần giải quyết với BA, PM (nếu có).
* Đầu ra: Tài liệu phân tích rủi ro của việc kiểm thử phần mềm, Tài liệu báo cáo tính khả thi, Tài liệu chứa các câu hỏi và câu trả lời liên quan đến nghiệp vụ hệ thống.

### Giai đoạn 2: Lập kế hoạch kiểm thử

* Đầu vào: Các tài liệu đặc tả đã được cập nhật, tài liệu báo cáo tính khả thi, phân tích rủi ro của Kiểm thử phần mềm.
* Hoạt động:
  + Test leader sẽ lập kế hoạch kiểm thử cho cả team
    - Xác định phạm vi dự án: Thời gian thực hiện, những công việc cho từng khoảng thời gian nhất định 🡪 Lịch trình thực hiện công việc.
    - Xác định phương án tiếp cận: Thời gian cho phép test, yêu cầu về chất lượng từ phía khách hàng (cao/ thấp), công nghệ/ nền tảng được sử dụng để phát triển sản phẩm 🡪 đưa ra những phương án và kế hoạch phù hợp với tiến độ bàn giao sản phẩm.
  + Xác định nguồn lực:
    - Con người: bao nhiêu người tham gia dự án, ai test task nào.
    - Thiết bị: số lượng server, máy tính, điện thoại.
  + Lên kế hoạch kiểm thử: bản kế hoạch kiểm thử
    - Liệt kê các chức năng cần kiểm thử.
    - Để thực hiện test chức năng này thì cần làm công việc gì, thời gian bao lâu, ai thực hiện.
    - Xác định điều kiện bắt đầu: những điều kiện tối thiểu để bắt đầu hoạt động kiểm thử cho từng chức năng.
    - Xác định điều kiện kết thúc: Khi có những điều kiện nào thì kết thúc kiểm thử.
* Đầu ra: Test plan, test estimation, test schedule.

### Giai đoạn 3: Thiết kế tài liệu kiểm thử

* Đầu vào: test plan, test estimation, tài liệu đặc tả yêu cầu đã được cập nhật.
* Hoạt động:
  + Review tài liệu: review lại các công việc cần làm, các chức năng cần test, chức năng không cần test...
  + Viết test case: test case được viết dựa vào kế hoạch đã đưa ra, đánh giá mức độ ưu tiên của từng test case.
  + Chuẩn bị dữ liệu kiểm thử: test data, test automation script.
  + Review test case: các thành viên và test leader review lại test case để tạo bổ sung, tránh sai xót.
* Đầu ra: test design, test case, test data, test automation script.

### Giai đoạn 4: Thiết lập môi trường kiểm thử

* Đầu vào: test plan, test data, test automation script, smoke test case.
* Hoạt động:
  + Môi trường kiểm thử sẽ được quyết định dựa trên những yêu cầu của khách hàng, đặc thù của sản phẩm (server/ client/ network...).
  + Chuẩn bị test case để kiểm tra môi trường cài đặt đã sẵn sàng cho việc kiểm thử chưa 🡪 thực thi smoke test.
* Đầu ra: môi trường cài đặt theo đúng yêu cầu, kết quả của smoke test.

### Giai đoạn 5: Thực hiện kiểm thử

* Đầu vào: test plan, test design, test case, test data, test automation script.
* Hoạt động:
  + Thực hiện các test case theo độ ưu tiên.
  + Báo cáo bug lên tool quản lý bug và theo dõi trạng thai đến khi sửa lỗi thành công.
  + Thực hiện retest để xác nhận các bug được fix và regression test khi có sự thay đổi liên quan.
  + Phân tích tiến độ phù hợp với tiến độ dự án đưa ra.
  + Report cho PM và khách hàng về tình hình dự án: đã kiểm thử những chức năng nào, còn chức năng nào chưa kiểm thử, tiến độ công việc hằng này.
* Đầu ra: test result, danh sách các bug đã phát hiện.

### Giai đoạn 6: Kết thúc quy trình kiểm thử

* Đầu vào: tất cả các tài liệu trong suốt quá trình kiểm thử dự án bao gồm: tài liệu phân tích đặc tả yêu cầu, test plan, test results, danh sách các bug đã phát hiện.
* Hoạt động:
  + Thực hiện tổng kết báo cáo về việc thực thi test case, bao nhiêu test case pass/ fail, bao nhiêu test case đã được fix, mức độ nghiêm trọng, bao nhiêu lỗi ở các chức năng nào, chức năng nào đã hoàn thành/ chưa hoàn thành.
  + Đánh giá các tiêu chí hoàn thành: chất lượng, chi phí, thời gian.
* Đầu ra: Test report, Test result.

## Các mức độ kiểm thử phần mềm

### Kiểm thử mức đơn vị (Unit testing)

Kiểm thử đơn vị là kiểm thử ở mức cơ bản nhất, kiểm tra từng module nhỏ trong hệ thống (hàm, chức năng, màn hình).

Được thực hiện bởi lập trình viên.

Mục đích để xác nhận rằng mỗi thành phần của hệ thống đúng với thiết kế.

### Kiểm thử tích hợp (Integration testing)

Kiểm thử tích hợp một nhóm các module riêng lẻ lại với nhau.

Được thực hiện bởi tester.

Mục đích để kiểm tra luồng dữ liệu từ module này tới module khác.

### Kiểm thử hệ thống (System testing)

Kiểm thử hệ thống được thực hiện trên một hệ thống hoàn chỉnh, tích hợp. Kiểm tra sự tương tác tổng thể của các thành phần bao gồm: tải, hiệu suất, kiểm tra bảo mật.

Mục đích để đánh giá hệ thống có tuân thủ theo các yêu cầu đưa ra hay không.

### Kiểm thử chấp nhận (Acceptance testing)

Kiểm thử chấp nhận nhằm kiểm thử hệ thống về khả năng chấp nhận được.

Kiểm thử chấp nhận giống kiểm thử hệ thống nhưng được thực hiện bởi khách hàng hoặc người dùng.

Mục đích để đánh giá sự tuân thủ của hệ thống với các yêu cầu nghiệp vụ và thẩm định sản phẩm đã có thể chấp nhận để bàn giao cho khách hàng hay chưa.

## Các kỹ thuật kiểm thử phần mềm

### Kiểm thử hộp trắng

Kiểm thử hộp trắng (White-box testing): là phương pháp kiểm thử cấu trúc bên trong của phần mềm, chỉ tập trung đến dữ liệu đầu vào và truy cập vào bên trong source code.

### Kiểm thử hộp đen

Kiểm thử hộp đen (Black-box testing): là phương pháp kiểm thử mà không cần biết được cấu tạo bên trong của phần mềm, chỉ tập trung vào kiểm thử các chức năng của phần mềm.

### Kiểm thử hộp xám

Kiểm thử hộp xám (Grey-box testing): là sự kết hợp giữa hai loại kiểm thử hộp trắng và kiểm thử hộp đen, kiểm thử hộp trắng cần sự hiểu biết sâu về bên trong hệ thống còn kiểm thử hộp đen thì sự hiểu biết này không quá cần thiết, nhưng kiểm thử hộp xám thì tester cần phải hiểu về cấu trúc bên trong của hệ thống cũng như quyền truy cập vào cơ sở dữ liệu.

## Test case và kỹ thuật thiết kế test case

### Test case là gì

Test case là tập hợp các hành động được thực hiện để xác minh một tính năng cụ thể hoặc một chức năng của ứng dụng phần mềm.

### Cấu trúc của test case

* Test case ID: là duy nhất và bắt buộc cho mỗi trường hợp kiểm thử.
* Pre-condition: là điều kiện cần thiết trong khi thực hiện kiểm thử.
* Module name: tên module thử nghiệm.
* Title/ Name: tiêu đề của trường hợp thử nghiệm.
* Description: mô tả ngắn gọn mục tiêu thử nghiệm.
* Step: liệt kê chi tiết các bước cần kiểm tra.
* Test data: dữ liệu thử nghiệm để làm dữ liệu đầu vào cho trường hợp thử nghiệm.
* Actual result: mô tả hành vi thực tế của hệ thống sau khi thực hiện thử nghiệm.
* Expected result: mô tả kết quả mong đợi của hệ thống sau khi thực hiện thử nghiệm.
* Status: nếu như Actual Result và Expected Result giống nhau thì đánh dấu pass, ngược lại thì Fail.
* Comment: nếu có câu hỏi hoặc các đề xuất liên quan sẽ được đính kèm ở đây.

Ngoài ra cấu trúc của một test case còn có:

* Test designed date: ngày thiết kế.
* Test executed by: tên của người test.
* Test execution date: ngày test.

### Các kỹ thuật thiết kế test case

* Phân vùng giá trị biên: là kỹ thuật thiết kế thử nghiệm hộp đen, được sử dụng để tìm lỗi ở ranh giới của đầu vào thay vì tìm những lỗi đó ở trung tâm đầu vào. Chọn giá trị biến đầu vào ở mức: tối thiểu, ngay trên mức tối thiểu, ngay dưới mức tối thiểu, giá trị danh nghĩa, ngay dưới mức tối đa, tối đa và trên mức tối đa. Đối với mỗi phạm vi sẽ có hai ranh giới: ranh giới dưới (bắt đầu phạm vi) và ranh giới trên (cuối phạm vi), và các ranh giới là đầu và cuối của mỗi phân vùng hợp lệ.
* Phân vùng tương đương: dữ liệu đầu vào được chia thành các lớp dữ liệu tương đương khác nhau, thường là hợp lệ và không hợp lệ.
* Bảng quyết định: kỹ thuật kiểm thử phần mềm để kiểm tra hành vi của hệ thống các dữ liệu đầu vào khác nhau.
* Chuyển đổi trạng thái: được sử dụng để kiểm tra tính năng của 1 hệ thống được biểu diễn dưới dạng các trạng thái chuyển đổi thành một hệ thống khác.
* Use case: là kỹ thuật kiểm thử nhằm xác định các test case bao phủ toàn bộ hệ thống trên cơ sở từ điểm bắt đầu đến điểm kết thúc.
* Đoán lỗi: là kỹ thuật kiểm thử phần mềm dựa trên kinh nghiệm để đưa ra phỏng đoán về các lỗi có thể xảy ra của phần mềm.

## Bug và quy trình quản lý bug

### Bug là gì và phân loại bug

Bug là một vấn đề/ sự cố xảy ra trong phần mềm, các lỗi về code do các lập trình viên trong quá trình viết code.

Phân loại bug theo: mức độ nghiêm trọng, mức độ ưu tiên và theo loại (hình ảnh, nội dung, khả năng sử dụng)

1. Mức độ nghiêm trọng:

* Critical: ngăn chặn các chức năng chính của ứng dụng hoặc trang web.
* Major: ảnh hưởng đến trải nghiệm người dùng nhưng không ngăn chặn các chức năng chính.
* Medium: gây ra những hành vi không mong muốn, nhưng hệ thống vẫn hoạt động được.
* Low: tác động tối thiểu đến trải nghiệm người dùng.

1. Mức độ ưu tiên:

* High: phải được fix càng sớm càng tốt, ảnh hưởng nghiêm trọng tới hệ thống không thể sử dụng được cho đến khi lỗi này đươc khắc phục.
* Medium: cần được fix trong lần build tiếp theo.
* Low: không cần phải fix ngay, có thể fix ngay sau khi bug High và Medium đã fix xong.

1. Theo loại:

* Visual: người dùng có thể hoàn thành tác vụ, nhưng giao diện không được đẹp mắt, khung bố cục lộn xộn.
* Content: liên kết không truy cập được, thiếu hình ảnh.
* Usability: các cải tiến về chức năng và tính năng hiện có giúp cho sản phẩm dễ dàng sử dụng và trực quan hơn.

### Cấu trúc report bug

Bug ID: số nhận dạng duy nhất của bug.

Bug title: mô tả chi tiết về bug, gồm thông tin về các module của lỗi được tìm thấy 🡪 dễ dàng hình dung bug.

Step: mô tả chi tiết từng bước phát hiện ra bug 🡪 dễ dàng tái phát hiện bug.

Actual result: kết quả hiện tại của bug 🡪 hiểu rõ tình trạng bug như thế nào?

Expected result: kết quả mong đợi 🡪 mong đợi sau khi fix bug.

Evident: tài liệu tham khảo (ảnh chụp màn hình, record) 🡪 thể hiện bug một cách rõ ràng nhất.

Severity: mức độ nghiêm trọng 🡪 xác định mức độ ưu tiên để fix bug.

Priority: mức độ ưu tiên 🡪 xác định thứ tự để fix bug nghiêm trọng trước.

Detected by: tên tester phát hiện bug.

### Quy trình quản lý lỗi

Quy trình quản lý lỗi là toàn bộ quá trình phát hiện, phân tích và sửa các lỗi đã được ghi nhận.

Quy trình quản lý lỗi gồm 6 giai đoạn:

* Phát hiện lỗi: tìm càng nhiều lỗi càng tốt, lỗi được phát hiện và chuyển sang trạng thái ghi nhận khi developer thừa nhận và chấp nhận lỗi đó.
* Phân loại lỗi: phân loại lỗi để sắp xếp mức độ ưu tiên, giúp developer sửa lỗi theo mức độ ưu tiên trước và sửa lỗi theo thứ tự giảm dần của mức độ ưu tiên.
* Sửa lỗi: là quá trình fix bug. Tester chỉ định lỗi cho developer 🡪 Lên lịch sửa lỗi theo mức độ ưu tiên 🡪 Lỗi đã được sửa 🡪 Báo cáo giải pháp sửa lỗi.
* Xác nhận: sau khi nhóm developer khắc phục và báo cáo lỗi 🡪 tester kiểm tra xem lỗi đã thực sự được giải quyết hay chưa.
* Kết thúc: sau khi tester xác nhận lỗi đó đã được khắc phục, lỗi được chuyển sang trạng thái **closed**.
* Báo cáo lỗi: là quá trình tester chuẩn bị báo cáo lỗi và gửi đến nhóm quản lý dự án về quá trình quản lý lỗi, tình trạng lỗi. Nhóm quản lý dự án sẽ kiểm tra và phản hồi báo cáo lỗi. Giúp dễ dàng trong việc quản lý lỗi, theo dõi và quản lý chi tiết các lỗi.

### Vòng đời của lỗi

* New: khi lỗi mới được phát hiện và ghi lại thì nó lại gán trạng thái “New”.
* Assigned: sau khi lỗi được ghi lại bởi tester, leader test sẽ phê duyệt lỗi và chỉ định lỗi này cho nhóm developer sửa lỗi.
* Open: developer sẽ phân tích lỗi và làm việc để sửa lỗi.
* Fixed: khi developer đã thực hiện sửa lỗi và xác minh đã sửa lỗi thì lỗi này sẽ chuyển sang trạng thái “fixed”.
* Pending retest: sau khi lỗi được khắc phục, trong thời gian tester kiểm tra lại lỗi thì được chuyển sang trạng thái “pending retest”.
* Retest: tester kiểm tra xem lỗi này đã được developer khắc phục hay chưa thì chuyển sang trạng thái “retest”.
* Verified: tester kiểm tra kiểm tra lại lỗi sau khi developer sửa lỗi, nếu không còn bất kỳ lỗi nào trong phần mềm nữa thì lỗi đó đã được sửa xong và chuyển sang trạng thái “verified”.
* Closed: nếu lỗi không còn trong phần mềm nữa thì chỉ định sang trạng thái “closed”.
* Re-open: nếu lỗi vẫn còn sau khi developer đã sửa lỗi, tester sẽ thay đổi trạng thái lỗi “re-open”, lỗi lặp lại một chu kỳ mới.
* Duplicate: nếu lỗi bị trùng lặp, giống nhau thì trạng thái lỗi “duplicate”.
* Rejected: nếu developer cho rằng lỗi không thật sự quan trọng thì sẽ chuyển sang trạng thái “rejected”.
* Deferred: nếu lỗi không được ưu tiên hàng đầu, và mong đợi sẽ sửa trong version tiếp theo của sản phẩm thì chuyển sang trạng thái “deferred”.
* Not a bug: nếu lỗi không ảnh hưởng đến chức năng của sản phẩm thì sẽ gán trạng thái “not a bug”.

# CHƯƠNG 2. TỔNG QUAN VỀ HỆ THỐNG CẤN TEST

## 2.1. Mô tả website

## 2.2. Mô tả về chức năng kiểm thử của website

# CHƯƠNG 3. THIẾT KẾ TEST CASE VÀ THỰC HIỆN KIỂM THỬ

## 3.1. Test case

## 3.2. Thực hiện kiểm thử

## 3.3. Report bug

# KẾT LUẬN

# TÀI LIỆU THAM KHẢO