TRƯỜNG ĐẠI HỌC KINH TẾ

**KHOA THỐNG KÊ – TIN HỌC**

--🙢🕮🙠--



**BÁO CÁO CHUYÊN ĐỀ**

**THỰC TẬP TỐT NGHIỆP**

**TÊN ĐỀ TÀI**

**THỰC HIỆN KIỂM THỬ**

**HỆ THỐNG BÁN CAMERA**

**<Đơn vị thực tập: SmartDev LLC>**

SVTH: Phan Thị Đài

Lớp: 44K14

GVHD: ThS. Cao Thị Nhâm

**Đà Nẵng, 2/2022**

**LỜI CẢM ƠN**

**LỜI CAM ĐOAN**

Em xin cam đoan báo cáo “Thực hiện kiểm thử hệ thống bán camera” là kết quả nghiên cứu độc lập của em trong thời gian thực tập tại công ty SmartDev cùng sự hướng dẫn tận tình của giáo viên hướng dẫn: Ths. Cao Thị Nhâm.

Nội dung dự án là trung thực, ngoài một số định nghĩa ở phần lý thuyết thì tất cả hoàn toàn được thực hiện trên cơ sở khoá học của Công ty SmartDev, không sao chép bất kỳ nguồn nào khác. Nếu có bất kỳ hành vi sao chép nào hoặc vấn đề xảy ra chúng em xin được chịu hoàn toàn trách nhiệm trước bộ môn, khoa và nhà trường về sự cam đoan này.

## **MỤC LỤC**

[**MỤC LỤC** iv](#_Toc96720727)

[**DANH MỤC HÌNH ẢNH** vi](#_Toc96720728)

[**DANH MỤC BẢNG BIỂU** vii](#_Toc96720729)

[**CHƯƠNG I. MỞ ĐẦU** 1](#_Toc96720730)

[**CHƯƠNG II. GIỚI THIỆU CÔNG TY THỰC TẬP** 3](#_Toc96720731)

[**CHƯƠNG III. CƠ SỞ LÝ THUYẾT** 4](#_Toc96720732)

[**3.1. Kiểm thử thủ công** 4](#_Toc96720733)

[**3.1.1. Các loại kiểm thử:** 4](#_Toc96720734)

[**3.1.2. Các mức độ kiểm thử:** 4](#_Toc96720735)

[**3.2. Kiểm thử tự động** 5](#_Toc96720736)

[**3.2.1. Mục đích của kiểm thử tự động** 6](#_Toc96720737)

[**3.2.2. Kiểm thử tự động khi nào?** 6](#_Toc96720738)

[**3.2.3. Một số công cụ kiểm thử tự động** 6](#_Toc96720739)

[**3.3. Quy trình kiểm thử** 7](#_Toc96720740)

[**3.3.1. Test life cycle trong mô hình Scrum** 7](#_Toc96720741)

[**3.3.2. Thiết kế test case** 7](#_Toc96720742)

[**3.3.3. Test Estimation** 8](#_Toc96720743)

[**3.3.4. Viết test case** 8](#_Toc96720744)

[**3.3.5. Thực thi test case** 8](#_Toc96720745)

[**3.3.6. Bug report** 8](#_Toc96720746)

[**CHƯƠNG IV. TOOLS AUTOMATION TESTING** 9](#_Toc96720747)

[**4.1. Selenium** 9](#_Toc96720748)

[**4.2. Cucumber** 9](#_Toc96720749)

[**4.2. Serenity** 9](#_Toc96720750)

[**CHƯƠNG V. PHÂN TÍCH HỆ THỐNG** 10](#_Toc96720751)

[**5.1. Tổng quan về hệ thống** 10](#_Toc96720752)

[**5.2. Giao diện hệ thống** 10](#_Toc96720753)

[**5.3. User story** 10](#_Toc96720754)

[**5.4. Use case** 10](#_Toc96720755)

[**5.5. Mô tả chức năng** 10](#_Toc96720756)

[**5.6. Flowchart** 10](#_Toc96720757)

[**5.7. Phân tích yêu cầu** 10](#_Toc96720758)

[**CHƯƠNG VI. THỰC HIỆN KIỂM THỬ** 11](#_Toc96720759)

[**6.1. Kế hoạch kiểm thử** 11](#_Toc96720760)

[**6.2. UI Check list** 11](#_Toc96720761)

[**6.3. Viết test case** 11](#_Toc96720762)

[**6.4. Viết test script trong automation test** 11](#_Toc96720763)

[**6.5. Thực thi test case** 11](#_Toc96720764)

[**6.6. Test report** 11](#_Toc96720765)

[**CHƯƠNG VII. KẾT QUẢ** 12](#_Toc96720766)

[**KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN** 13](#_Toc96720767)

[**TÀI LIỆU THAM KHẢO** 14](#_Toc96720768)

## **DANH MỤC HÌNH ẢNH**

## **DANH MỤC BẢNG BIỂU**

## **CHƯƠNG I. MỞ ĐẦU**

1. **Mục tiêu nghiên cứu của đề tài**

Đề tài này sẽ mô phỏng lại quy trình kiểm thử cho một dự án trong mô hình Scrum. Dự án này yêu cầu cả về manual testing và automation testing.

1. **Nhiệm vụ của đề tài**

* Áp dụng những kiến thức về kiểm thử và các kỹ thuật thiết kế test cases vào trong một dự án.
* Thực hiện kiểm thử tự động cho một số chức năng trong hệ thống bán camera.

1. **Phương pháp nghiên cứu**

* Tham gia khóa training về manual testing và automation testing của công ty SmartDev.
* Tham gia vào dự án thực tế của công ty để hiểu hơn về quy trình kiểm thử trong mô hình Scrum.
* Tham khảo các tài liệu về kiểm thử tự động.
* Thực hiện kiểm thử trên hệ thống bán camera do sinh viên của khoa lập trình.

1. **Đối tượng và phạm vi nghiên cứu**

* Manual testing
* Automation testing
* Hệ thống bán camera

1. **Kết cấu của đề tài**

Đề tài được tổ chức gồm phần mở đầu, 4 chương nội dung và phần kết luận:

* **Chương 1**: Mở đầu
* **Chương 2**: Giới thiệu công ty
* **Chương 3**: Cơ sở lý thuyết
* **Chương 4**: Tools automation testing
* **Chương 5**: Phân tích hệ thống
* **Chương 6**: Thực hiện kiểm thử
* **Chương 7**: Kết quả
* Kết luận và hướng phát triển

## **CHƯƠNG II. GIỚI THIỆU CÔNG TY THỰC TẬP**

**Tên công ty:** SmartDev LLC

**Địa chỉ:** 81 Quang Trung, Thạch Thang, Hải Châu, Đà Nẵng, Việt Nam

**Tóm tắt:**

SmartDev là một Trung tâm Phát triển Offshore (ODC) có trụ sở tại Thành phố Đà Nẵng, Việt Nam.

Là công ty con thuộc sở hữu hoàn toàn của Verysell Technologies SA (Nyon, Thụy Sĩ) và hoạt động như một trung tâm phát triển phần mềm nội bộ từ những ngày đầu. Sau khi phát triển nhanh chóng tại khu vực Đông Nam Á đang phát triển bùng nổ vào năm 2015, SmartDev cùng với việc phát triển các sản phẩm cho công ty mẹ của mình đã bắt đầu cung cấp dịch vụ cho khách hàng, đưa các ý tưởng Startups và SMBs trở thành các sản phẩm vận hành thành công.

Mô hình kinh doanh và sản xuất phần mềm tại SmartDev đã được thiết lập theo các tiêu chuẩn quốc tế tốt nhất bởi chuyên gia đào tạo AGILE người Thụy Sĩ và Giám đốc điều hành đầu tiên của SmartDev - Alexandre Cuva.

Ở đây cung cấp các giải pháp thông minh để giải quyết các thách thức trong kinh doanh và tạo ra giá trị cho khách hàng. Smartdev kết hợp niềm đam mê và chuyên môn để nhân viên và khách hàng có thể cung cấp và trải nghiệm các dự án trên toàn cầu.

## **CHƯƠNG III. CƠ SỞ LÝ THUYẾT**

### **3.1. Kiểm thử thủ công**

Kiểm thử thủ công là kiểu kiểm thử nguyên thủy nhất trong các loại kiểm thử, giúp tìm ra lỗi trong hệ thống phần mềm. Đây là loại kiểm thử mà tester phải tự làm tất cả mọi việc bằng tay, từ việc viết test case cho đến việc thực thi những test case đó. Bất kì ứng dụng nào cũng đều phải được kiểm tra một cách thủ công trước khi có thể thực hiện test tự động. Kiểm thử tự động đòi hỏi nhiều năng suất và thời gian hơn nhưng nó là cần thiết để kiểm tra tính khả thi của tự động hóa. Và một lý do khiến kiểm thử thủ công trở nên bắt buộc là vì không phải bất kỳ sản phẩm nào cũng có thể thực hiện kiểm thử tự động.

#### **3.1.1. Các loại kiểm thử:**

- Kiểm thử chức năng: Kiểm thử chức năng chú trọng đến các chức năng của phần mềm đã đáp ứng yêu cầu hay chưa. Các test case được viết và thực thi dựa trên bản đặc tả yêu cầu của hệ thống.

- Kiểm thử cấu trúc: Tập trung vào việc kiểm thử những gì đang diễn ra ở bên trong phần mềm. Yêu cầu tester phải có hiểu biết về quá trình xây dựng và phát triển của phần mềm này.

- Kiểm thử hiệu năng: Trọng tâm của kiểm thử hiệu năng là thời gian phản hồi (thời gian phản hồi của hệ thống là bao nhiêu, nhanh hay chậm), khả năng mở rộng (xác định tải người dùng tối đa mà hệ thống có thể xử lý), tính ổn định (xác định xem hệ thống có ổn định dưới các tải khác nhau hay không).

#### **3.1.2. Các mức độ kiểm thử:**

- Unit Testing (Kiểm thử đơn vị):

Mục đích: Kiểm thử đơn vị dùng để xác nhận mỗi thành phần của phần mềm thực hiện đúng với thiết kế ban đầu.

Người thực hiện: Kiểm thử đơn vị thường do lập trình viên thực hiện.

Phương pháp: Kiểm thử đơn vị thường sử dụng cả phương pháp kiểm thử hộp đen (Black-box testing) và kiểm thử hộp trắng (White-box testing) để thực hiện.

- Integration Testing (Kiểm thử tích hợp):

Mục đích: Đảm bảo rằng các chức năng (module) khi tích hợp lại với nhau thì chạy đúng theo yêu cầu ban đầu đặt ra. Là bước chuẩn bị quan trọng để tiến hành kiểm thử ở mức hệ thống (System Testing). Kiểm thử tích hợp được thực hiện sau khi kiểm thử đơn vị và trước khi kiểm thử hệ thống.

Người thực hiện: Integration testing được thực hiện bởi một tester hoặc một nhóm các tester trong dự án. Lập trình viên không tham gia vào giai đoạn kiểm thử này.

Phương pháp: Kiểm thử hộp đen (Black-box testing)

- System Testing (Kiểm thử hệ thống):

Mục đích: System Testing là thực hiện kiểm thử một hệ thống đã được tích hợp hoàn chỉnh để xác minh rằng nó đúng yêu cầu của phần mềm.

Người thực hiện: Kiểm thử hệ thống thường được thực hiện bởi tester. Lập trình viên không tham gia vào giai đoạn kiểm thử này.

Phương pháp: Kiểm thử hộp đen (Black-box testing)

- Acceptance Testing (Kiểm thử chấp nhận):

Mục đích: Đảm bảo phần mềm đáp ứng đúng yêu cầu của khách hàng. Sản phẩm nhận được sự chấp nhận từ khách hàng/người dùng cuối cùng (end-user).

### **3.2. Kiểm thử tự động**

Kiểm thử tự động là quá trình sử dụng sự hỗ trợ của tool, script và phần mềm để thực hiện các trường hợp kiểm thử. Tester sẽ viết các tập lệnh và dùng một phần mềm để thực thi các bước của test case thay vì tự ngồi làm như kiểm thử thủ công. Test report cũng sẽ được xuất ra khi tất cả các test case đã chạy xong.

#### **3.2.1. Mục đích của kiểm thử tự động**

- Giảm bớt công sức và tiết kiệm thời gian thực hiện quá trình kiểm thử

- Tăng độ tin cậy.

- Giảm chi phí cho tổng quá trình kiểm thử

#### **3.2.2. Kiểm thử tự động khi nào?**

- Không đủ tài nguyên: Khi số lượng test case quá nhiều mà tester không thể hoàn tất trong thời gian cụ thể.

- Kiểm tra hồi quy: Lặp lại kiểm tra các chức năng cũ liệu chúng có hoạt động bình thường không khi nâng cấp hoặc bảo trì phần mềm.

- Kiểm tra khả năng vận hành phần mềm trong môi trường đặc biệt (Đo tốc độ trung bình xử lý một yêu cầu của Web server, xác định cấu hình máy thấp nhất mà phần mềm vẫn có thể hoạt động tốt).

#### **3.2.3. Một số công cụ kiểm thử tự động**

- HP Quick Test Professional

- Selenium

- Appium

- Visual Studio Test Professional

- WATIR

- IBM Rational Functional Tester

- TestComplete

- Testing Anywhere

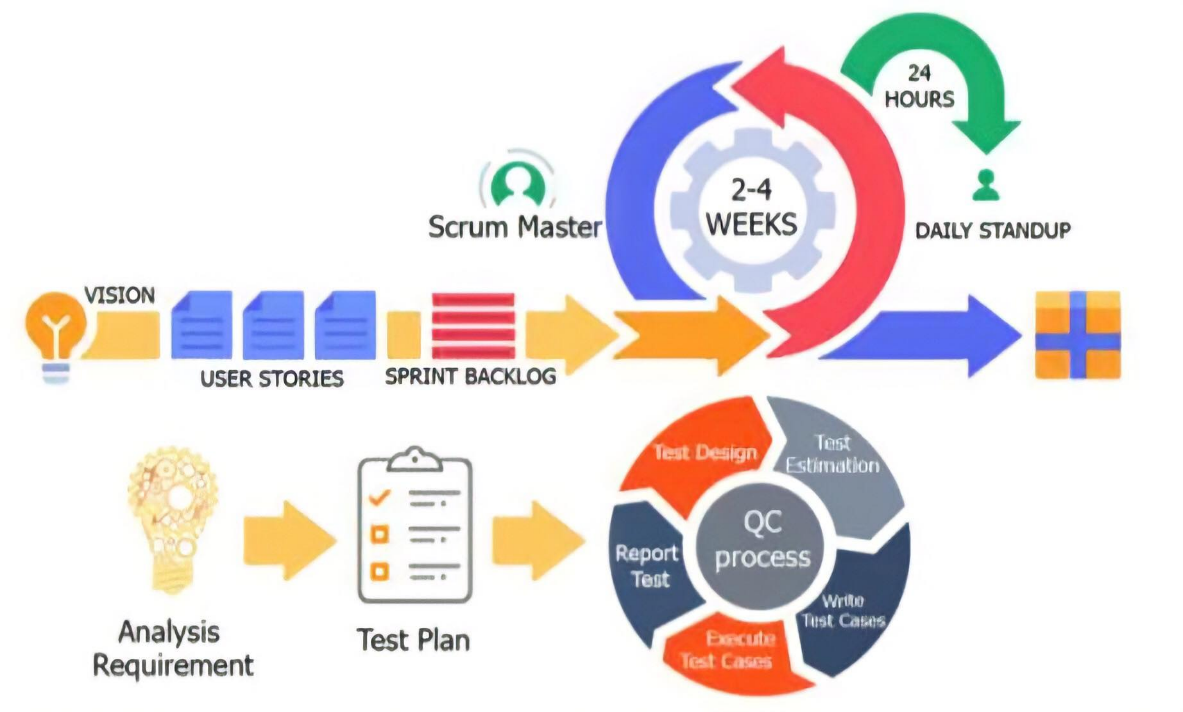
- WinRunner

- LaodRunner

- SilkTest

### **3.3. Quy trình kiểm thử**

#### **3.3.1. Test life cycle trong mô hình Scrum**



#### **3.3.2. Thiết kế test case**

##### ***3.3.2.1. WBS (Work Breakdown Structure)***

WBS là viết tắt của Work breakdown structure nghĩa cấu trúc phân chia công việc. WBS là khái niệm dùng để mô tả việc tách và chia nhỏ đối tượng công việc. Mục đích để dễ dàng, thuận tiện hơn trong việc triển khai và kiểm soát. Các việc nhỏ cần phải đảm bảo nằm trong phạm vi, giải quyết mục tiêu của công việc to.

Trong kiểm thử, tester có thể bẻ nhỏ user story để tiện cho việc viết test case cũng như quản lý chúng.

##### ***3.3.2.2. Boundary Value Analysis (BVA)***

Phân tích giá trị biên dựa trên việc thử nghiệm tại ranh giới giữa các phân vùng. Nó bao gồm các ranh giới tối đa, tối thiểu, bên trong hoặc bên ngoài, giá trị điển hình và giá trị lỗi.

- Nếu một điều kiện đầu vào bị hạn chế giữa các giá trị x và y, thì các trường hợp kiểm thử phải được thiết kế với các giá trị x và y cũng như các giá trị trên và dưới x, y.

- Ví dụ: Điều kiện đầu vào có giá trị từ 1 đến 10

🡪 Giá trị ranh giới 0,1,2 và 9,10,11

##### ***3.3.2.3. Equivalence Class Partitioning***

Phân vùng lớp tương đương cho phép bạn chia nhóm điều kiện thử nghiệm thành một phân vùng được coi là giống nhau. Phương pháp kiểm thử phần mềm này chia miền đầu vào của chương trình thành các lớp dữ liệu mà từ đó các trường hợp kiểm thử sẽ được thiết kế.

* Ví dụ: Điều kiện đầu vào có giá trị từ 1 đến 10

🡪Các lớp tương đương sẽ là:

Vô cùng đến 0 (không hợp lệ)

1 đến 10 (hợp lệ)

11 đến 19 (không hợp lệ)

##### ***3.3.2.4. Decision Table Based Testing***

##### ***3.3.2.5. Stage Transition***

##### ***3.3.2.6. Error Guessing***

##### ***3.3.2.7. End to End Testing***

#### **3.3.3. Test Estimation**

#### **3.3.4. Viết test case**

#### **3.3.5. Thực thi test case**

#### **3.3.6. Bug report**

## **CHƯƠNG IV. TOOLS AUTOMATION TESTING**

### **4.1. Selenium**

### **4.2. Cucumber**

### **4.2. Serenity**

## **CHƯƠNG V. PHÂN TÍCH HỆ THỐNG**

### **5.1. Tổng quan về hệ thống**

### **5.2. Giao diện hệ thống**

### **5.3. User story**

### **5.4. Use case**

### **5.5. Mô tả chức năng**

### **5.6. Flowchart**

### **5.7. Phân tích yêu cầu**

## **CHƯƠNG VI. THỰC HIỆN KIỂM THỬ**

### **6.1. Kế hoạch kiểm thử**

### **6.2. UI Check list**

### **6.3. Viết test case**

### **6.4. Viết test script trong automation test**

### **6.5. Thực thi test case**

### **6.6. Test report**

## **CHƯƠNG VII. KẾT QUẢ**

## **KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN**

## **TÀI LIỆU THAM KHẢO**