BỘ CÔNG THƯƠNG

**TRƯỜNG ĐẠI HỌC CÔNG NGHIỆP HÀ NỘI**

**KHOA CÔNG NGHỆ THÔNG TIN**

**======\*\*\*======**

****

**BÀI TẬP LỚN**

**KIỂM THỬ PHẦN MỀM**

**QUẢN LÝ KHO**

|  |  |
| --- | --- |
| Giáo viên: | TS Nguyễn Hoàng Tú |
| Lớp: | ĐH–KTPM3–K11 |
| Nhóm: |  |
| Thành viên: | Nguyễn Văn Nam |
|  | Nguyễn Văn Hào  Lê Mạnh Hùng |

Hà nội, Năm 2019

LỜI MỞ ĐẦU

Ngày nay, các ngành công nghiệp khoa học, kỹ thuật ngày càng phát triển như vũ bão trong đó khoa học máy tính cũng phát triển mạnh mẽ và xâm nhập vào mọi lĩnh vực đời sống xã hội… Từ việc xây dựng các chương trình khoa học kỹ thuật lớn đến công việc quản ký thường nhật, máy tính giúp chúng ta cập nhật, tìm kiếm và lưu trữ thông tin một cách tốt nhất. Nhưng các phần mềm hỗ trợ cho việc sử dụng máy tính, hay các chương trình quản lý hầu hết của nước ngoài, điều này gây rất nhiều trở ngại đối với người sử dụng máy tính không biết hoặc biết ít tiếng Anh. Vì vậy cần có các phần mềm, Website sử dụng tiếng Việt sẽ giúp người sử dụng có hiệu quả hơn trong công việc.

Thực tế đã cho thấy nhiều doanh nghiệp từ lớn đến nhỏ đã áp dụng có hiệu quả các sản phẩm tin học. Nhưng một bước cũng khá quan trọng trong quá trình phát triển phần mềm đó là kiểm thử phần mềm. Đây là bước quan trọng đóng góp không nhỏ vào sự thành công của phần mềm. Vì vậy hôm nay nhóm chúng em xin được trình bày bài tập lớn với chủ đề kiểm thử phần mềm “Quản lý kho”.

Tuy nhiên tong quá trình học tập còn gặp nhiều khó khăn, tài liệu học tập còn thiếu thốn và nhận thức còn nhiều hạn chế, cho nên trong quá trình thực hiện thao tác không tránh khỏi những thiếu sót, rất mong nhận được sự chỉ bảo góp ý, bổ sung của thầy, cô giáo và đặc biệt là thầy Nguyễn Hoàng Tú đã hướng dẫn chỉ bảo để nhóm em hoàn thành phần bài tập lớn này.

Bảng phân việc

|  |  |
| --- | --- |
| **Tên thành viên** | **Công việc cụ thể** |
| Nguyễn Văn Nam | Tìm hiểu công cụ NUnit, NUnitForms.  Phân tích, lập kế hoạch, viết script, thực hiện, báo cáo kết quả test chức năng “Tính doanh thu tháng”. |
| Lê Mạnh Hùng | Tìm hiểu công cụ NUnit, NUnitForms.  Phân tích, lập kế hoạch, viết script, thực hiện, báo cáo kết quả test chức năng “Thanh toán hóa đơn”. |
| Nguyễn Văn Hào | Tìm hiểu công cụ NUnit, NUnitForms.  Phân tích, lập kế hoạch, viết script, thực hiện, báo cáo kết quả test chức năng “Thêm sản phẩm”. |

MỤC LỤC

[Chương 1. Tìm hiểu công cụ NUnit 5](#_Toc6757784)

[1.1 Tóm tắt lịch sử 5](#_Toc6757785)

[1.2 Đặc điểm 5](#_Toc6757786)

[1.3 Cài đặt và sử dụng công cụ NUnit 5](#_Toc6757787)

[Chương 2. Giới thiệu phần mềm “Quản lý kho” 6](#_Toc6757788)

[2.1 Yêu cầu đề tài 6](#_Toc6757789)

[2.1.1 Lý do chọn đề tài 6](#_Toc6757790)

[2.1.2 Mục đích nghiên cứu 6](#_Toc6757791)

[2.1.3 Bài toán kiểm thử 6](#_Toc6757792)

[2.2 Phân tích thiết kế hệ thống 6](#_Toc6757793)

[2.2.1 Đặc tả hệ thống 6](#_Toc6757794)

[2.2.2 Thiết kế hệ thống 7](#_Toc6757795)

[2.2.2.1 Danh sách các tác nhân 7](#_Toc6757796)

[2.2.2.2 Danh sách các ca sử dụng 7](#_Toc6757797)

[2.2.2.3 Đặc tả các ca sử dụng 7](#_Toc6757798)

[2.2.2.4 Biểu đồ. 11](#_Toc6757799)

[Chương 3. phần mềm “Quản lý kho” 32](#_Toc6757800)

[3.1 Lập kế hoạch kiểm thử 32](#_Toc6757801)

[3.1.1 Nguyễn Văn Nam – Kiểm thử chức năng “Tính doanh thu tháng” 32](#_Toc6757802)

[3.1.1.1 Mục tiêu của lập kế hoạch kiểm thử 32](#_Toc6757803)

[3.1.1.2 Nhiệm vụ của lập kế hoạch kiểm thử 33](#_Toc6757804)

[3.1.2 Nguyễn Văn Hào – Kiểm thử chức năng “Thêm sản phẩm” 33](#_Toc6757805)

[3.1.2.1 Mục tiêu của lập kế hoạch kiểm thử 33](#_Toc6757806)

[3.1.2.2 Nhiệm vụ của lập kế hoạch kiểm thử 34](#_Toc6757807)

[3.1.3 Lê Mạnh Hùng – Kiểm thử chức năng “Thanh toán hóa đơn” 35](#_Toc6757808)

[3.1.3.1 Mục tiêu của lập kế hoạch kiểm thử 35](#_Toc6757809)

[3.1.3.2 Nhiệm vụ của lập kế hoạch kiểm thử 35](#_Toc6757810)

[3.2 Thực hiện kiểm thử 36](#_Toc6757811)

[3.2.1 Nguyễn Văn Nam - Tính doanh thu tháng (V0.1) 36](#_Toc6757812)

[3.2.1.1 Phân tích thiết kế kểm thử 36](#_Toc6757813)

[3.2.1.2 Thực hiện kiểm thử 41](#_Toc6757814)

[3.2.1.3 Báo cáo kết quả kiểm thử 44](#_Toc6757815)

[3.2.2 Nguyễn Văn Nam - Tính doanh thu tháng (V0.11) 45](#_Toc6757816)

[3.2.2.1 Phân tích thiết kế kểm thử 45](#_Toc6757817)

[3.2.2.2 Thực hiện kiểm thử 50](#_Toc6757818)

[3.2.2.3 Báo cáo kết quả kiểm thử 53](#_Toc6757819)

[3.2.3 Nguyễn Văn Hào – Thêm sản phẩm (V0.1) 55](#_Toc6757820)

[3.2.3.1 Phân tích thiết kế kiểm thử 55](#_Toc6757821)

[3.2.3.2 Thực hiện kiểm thử 61](#_Toc6757822)

[3.2.3.3 Báo cáo kết quả kiểm thử 67](#_Toc6757823)

[3.2.4 Nguyên Văn Hào – Thêm sản phẩm(V0.11) 67](#_Toc6757824)

[3.2.4.1 Phân tích thiết kế kiểm thử 67](#_Toc6757825)

[3.2.4.2 Thực hiện kiểm thử 74](#_Toc6757826)

[3.2.4.3 Báo cáo kết quả kiểm thử 80](#_Toc6757827)

[3.2.5 Lê Mạnh Hùng – Thanh toán hóa đơn (V0.1) 81](#_Toc6757828)

[3.2.5.1 Phân tích thiết kế kểm thử 81](#_Toc6757829)

[3.2.5.2 Thực hiện kiểm thử 85](#_Toc6757830)

[3.2.5.3 Báo cáo kết quả kiểm thử 89](#_Toc6757831)

[3.2.6 Lê Mạnh Hùng – Thanh toán hóa đơn (V0.11) 90](#_Toc6757832)

[3.2.6.1 Phân tích thiết kế kểm thử 90](#_Toc6757833)

[3.2.6.2 Thực hiện kiểm thử 98](#_Toc6757834)

[3.2.6.3 Báo cáo kết quả kiểm thử 102](#_Toc6757835)

[Phụ lục 104](#_Toc6757836)

[Tên phụ lục 1 104](#_Toc6757837)

[Tên phụ lục 2 104](#_Toc6757838)

[Tài liệu tham khảo 105](#_Toc6757839)

# Tìm hiểu công cụ NUnit

## Tóm tắt lịch sử

NUnit là framework có lịch sử gần 30 năm, vì vậy chúng đã có từ trước .NET. NUnit khởi đầu là một cổng từ [JUnit](https://stackify.com/junit-5/) của Java, nhưng cuối cùng các tác giả đã biến nó thành thành một framework phục vụ C # nhiều hơn. Vì vậy, công cụ này có lịch sử phong phú về unit test đằng sau nó, nhưng nó với thích hợp nhất với C#.

## Đặc điểm

Do lịch sử độc lập của nó, NUnit cũng có sự khác biệt trong việc tương tác độc đáo với các công cụ khác, chẳng hạn như các nền tảng xây dựng không phải của Microsoft và các trình chạy thử nghiệm tùy chỉnh. Trên hết, NUnit cũng nổi tiếng về việc chạy thử nhanh và nó cũng có một số tính năng bổ sung thú vị, bao gồm các chú thích kiểm tra cho phép dễ dàng xác định nhiều đầu vào cho một thử nghiệm nhất định.

Nhược điểm chính ở đây là nó không tích hợp vào Visual Studio theo cách mà MSTest làm. Sử dụng nó có nghĩa là thực hiện thêm công việc và cài đặt các công cụ bổ sung, bất kể các tác giả của các công cụ đó có thể thực hiện quy trình dễ dàng như thế nào.

## Cài đặt và sử dụng công cụ NUnit

Cài đặt:

1. Truy cập đượng link: <https://nunit.org/download/> và chọn phiên bản muốn cài đặt. (Ở đây nhóm sử dụng phiên bản NUnit 2.7.0)
2. Giải nén file vừa tải được.
3. Mở Microsoft Visual Studio và vào project bạn muốn test.
4. Khi ở trong project bạn kích vào tag “Solution”, kích phải vào phần “References”, chọn “Add Reference”, kích vào tag “Browser”, vào thư mục bạn vừa giải nén được “chọn NUnitFramework.dll”, OK.

Sử dụng:

1. Tạo một class library mới với tên “MyTest” (Tên tự chọn).
2. Import thư viện NUnit và thư viện NUnitForm (bạn tải và cài đặt tương tự như với NUnit, ở đây nhóm sử dụng phiên bản 1.3.1 để tiện cho việc làm bài tập lớn).
3. Sử dụng các Annotation bạn đã tìm hiểu trong NUnit để viết một unit test.
4. Run test, nếu không có lỗi gì xảy ra thì mở NUnitFrom lên và chọn đường dẫn tới unit test bạn vừa tạo, kích “Run” trên NUnitForm. OK.

Video hướng dẫn cụ thể:

# Giới thiệu phần mềm “Quản lý kho”

## 2.1 Yêu cầu đề tài

### Lý do chọn đề tài

Lĩnh vực mà nhóm chọn để thực hiện đề tài lần này chính là lĩnh vực quản lý với các cửa hàng kinh doanh thời trang vừa và nhỏ. Đây là lĩnh vực rất tiềm năng, bởi các shop thời trang hiện nay phát triển rất nhanh và đều có nhu cầu rất lớn đối với các sản phẩm hỗ trợ quản lý như này.

Khảo sát trên địa bàn Hà Nội cho thấy nhiều cửa hàng từ lớn đến nhỏ đã áp dụng có hiệu quả các sản phẩm tin học. Tại các cửa hàng ở nhiều nơi đã và đang áp dụng chương trình quản lý vào quản ký công việc quản lý kinh doanh, nhằm tạo ra nhiều cơ hội để nâng cao hiệu quả công việc và giúp cho những khách hàng có được điều kiện tốt nhất để có được trải nghiệm tốt và sản phẩm mà họ mong muốn và giảm dần những việc thủ công trước đây. Do vậy trong đợt báo cáo này em chọn đề tài: “Quản lý kho cho shop thời trang Genviet” để củng cố kiến thức đã học trong trường, nâng cao kỹ năng làm bài cũng như tiếp cận trực tiếp với thực tế phục vụ công tác sau này.

### Mục đích nghiên cứu

Tìm hiểu thêm được nhiều thông tin phục vụ việc học hiện thời, ngoài ra biết được quy trình xây dựng, phát triển phần mềm. Từ khâu khảo sát, phỏng vấn, lập kế hoạch, tìm nguồn nhân lực, phân công công việc, xây dựng phần mềm cho đến việc kiểm thử phần mềm và bàn giao cho khách hàng. Đề tài này cho chúng em cơ hội ma sát với thực tế bên ngoài, đây là bài học quý giá.

### Bài toán kiểm thử

Trong quá trình xây dựng và phát triển phần mềm chắc chắn không thể tránh được việc gặp những lỗi về hệ thống hay giao diện. Do vậy việc kiểm thử là vô cung cần thiết để đảm bảo chất lượng phần mềm bàn giao tới tay khách hàng. Bài toán kiểm thử đặt ra ở đây là làm sao để có thể khắc phục tối đa những lỗi mà chương trình gặp phải, muốn như thế thì người kiểm thử cần thu thập nhiều thông tin, hiểu biết về hệ thống và công cụ kiểm thử. Nên yêu cầu đặt ra là cần hiểu rõ phần mềm mà mình đang xây dựng, phát triển và phải có hiểu biết và ứng dụng công cụ kiểm thử mình định sử dụng, ngoài ra có thể tìm hiểu thêm một vài công cụ kiểm thử hiệu năng khác.

## Phân tích thiết kế hệ thống

### Đặc tả hệ thống

Hệ thống phần mềm quản lý kho cho phép người dùng đăng nhập vào hệ thống để thêm, sửa, xóa các sản phẩm trong kho, tìm kiếm sản phẩm trong kho, tạo hóa đơn mới cho khách hàng và cho phép xem danh sach, xem chi tiết, tìm kiếm hóa đơn đã thanh toán và tính doanh thu từng ngày, từng tháng, các hóa đơn đã chọn.

### Thiết kế hệ thống

#### Danh sách các tác nhân

* Người dùng (Người quản lý kho).
* Máy in.
* Cơ sở dữ liệu.

#### Danh sách các ca sử dụng

* Đăng nhập
* Quản lý sản phẩm
* Quản lý hóa đơn xuất
* Tinh doanh thu

#### Đặc tả các ca sử dụng

##### **Mô tả use case “Tính doanh thu”.**

**Mô tả vắn tắt use case:** Use case này cho phép người dùng tính doanh thu theo ngày, tháng.

* **Luồng sự kiện:**
* **Luồng cơ bản:**

1. Use case này bắt đầu khi người dùng kích nút  trên menu quản trị chính. Hệ thống sẽ hiển thị ra một form chứa danh sách các hóa đơn xuất và hai ô thông tin doanh thu tháng, ngày.
2. Tính doanh thu ngày: Người dùng chọn một ngày từ ô tính doanh thu ngày rồi nhấn nút . Hệ thống sẽ hiển thị kết quả trong phần kết quả của ô tính doanh thu ngày.
3. Tính doanh thu tháng: Người dùng nhập một tháng vào ô text (Yêu cầu đúng định dạng: MM/yyyy) tính doanh thu tháng rồi nhấn nút . Hệ thống sẽ hiển thị kết quả trong phần kết quả của ô tính doanh thu tháng và hiển thị một thông báo cho người dùng biết chức năng có thành công hay không.
4. Use case kết thúc khi người dùng nhấn “X” trên menu quản trị chính.

* **Luồng rẽ nhánh:**

1. Không kết nối được với CSDL: Tại bất kỳ thời điểm nào trong quá trình thực hiện use case nếu không kết nối được với CSDL thì hệ thống sẽ hiển thị một thông báo lỗi và use case kết thúc.
2. Hủy bỏ: Tại bất kỳ bước nào trong luồng cơ bản nếu người dùng nhấn vào một mục khác trên menu chính của người quản trị thì use case sẽ kết thúc.

* **Các yêu cầu đặc biệt:** Không có.
* **Tiền điều kiện:**

Người dùng đã có một tài khoản trong hệ thống.

* **Hậu điều kiện:**

Người dùng cần nhấn  (Reset) để có thể tính với các phép chọn khác.

* **Điểm mở rộng:**

Không có.

##### **Mô tả use case “Đăng nhập”.**

**Mô tả vắn tắt use case** Mô tả vắn tắt use case: Use case này cho phép người dùng đăng nhập vào phần mềm.

**Luồng sự kiện:**

* **Luồng cơ bản:**

1. Use case này bắt đầu khi người dùng kích vào nút Login.Hệ thống sẽ hiển thị form menu chính, từ đó người dùng có thể sử dụng được các chức năng của phần mềm. Use case kết thúc khi người dùng tắt form “Đăng Nhập”.

* **Luồng rẽ nhánh:**

2. Không kết nối được với CSDL: Tại bất kỳ thời điểm nào trong quá trình thực hiện

use case nếu không kết nối được với CSDL thì hệ thống sẽ hiển thị một thông

báo lỗi và use case kết thúc.

3. Nếu người dùng nhập sai tài khoản hoặc mật khẩu trong CSDL hệ thống sẽ hiển

thị một thông báo lỗi và yêu cầu người dùng phải nhập đúng tài khoản và mật

* **Các yêu cầu đặc biệt:** Không có.
* **Tiền điều kiện:**

Người dùng phải đang ở trong form đăng nhập..

* **Hậu điều kiện:**

Không có.

* **Điểm mở rộng:**

Không có.

##### **Mô tả use case “Quản lý hóa đơn xuất”.**

* **Luồng sự kiện:**
* **Luồng cơ bản:**

1. Use case này bắt đầu khi người dùng kích vào nút “Quản lý hóa đơn xuất” trên form chính. Hệ thống sẽ truy xuất vào cở sở dữ liệu và hiển thị ra một form mới chứa danh sách các hóa đơn với thông tin như mã hóa đơn, người mua, ngày, tổng tiền và các nút chức năng xóa hóa đơn đã chọn, xem chi tiết hóa đơn đã chọn, thêm hóa đơn xuất mới.
2. Xem chi tiết hóa đơn xuất:
3. Khi người dùng kích chọn một hóa đơn trong danh sách các hóa đơn xuất, rồi kích vào nút “Xem chi tiết”. Hệ thống sẽ truy xuất vào cơ sở dữ liệu và hiển thị ra thông tin chi tiết của hóa đơn xuất đó như mã hóa đơn, tổng tiền, ngày xuất, người mua và các mặt hàng chi tiết lên một form khác.
4. Người dùng kích vào nút “X”. Hệ thống sẽ quay trở lại menu quản lý hóa đơn xuất.
5. Xóa hóa đơn: Người dùng nhấn chọn một hóa đơn trong danh sách hóa đơn xuất và nhấn nút “Xóa hóa đơn”. Hệ thống sẽ xóa thông tin bản ghi hóa đơn xuất đó trong cơ sở dữ liệu và cập nhật hiển thị lại danh sách hóa đơn xuất.
6. Thêm hóa đơn xuất: Kích chọn  trên form chính. Hệ thống sẽ truy xuất vào cơ sở dữ liệu và hiển thị thông tin mặc định như mã hóa đơn, ngày và hai bảng ( một bảng chứa các sản phẩm trong kho và thông tin chi tiết của chúng , bảng phía trên chứa danh sách sản phẩm hiện thời người dùng đang chọn ), các nút chức năng  ( “Thêm sản phẩm vào danh sách chọn”),  (“Xóa sản phẩm khỏi danh sách chọn”) ,  (“Tính tổng tiền hóa đơn”),  (“In hóa đơn”).

* Người dùng chọn sản phẩm ở bảng danh sách sản phẩm rồi kích chọn , hệ thống thêm sản phẩm được chọn lên danh sách chọn.
* Người dùng nhập số lượng cho từng sản phẩm trong danh sách chọn, người dùng kích chọn  ( Yêu cầu danh sách chọn có ít nhất 1 sản phẩm). Hệ thống sẽ kiểm tra điều kiện của số lượng bán ( Số lượng bán là số nguyên không âm và không lớn hơn số lượng trong kho ) nếu thỏa mã hệ thống sẽ truy xuất vào cơ sở dữ liệu thêm mới thông tin hóa đơn, chi tiết hóa đơn , cập nhật số lượng sản phẩm trong kho và in ra thông báo “SUCCESS”. Rồi mở quyền (“In hóa đơn”).

1. Use case kết thúc khi người dùng nhấn “Home” trên menu quản trị chính.

* **Luồng rẽ nhánh:**

1. Không kết nối được với CSDL: Tại bất kỳ thời điểm nào trong quá trình thực hiện use case nếu không kết nối được với CSDL thì hệ thống sẽ hiển thị một thông báo lỗi và use case kết thúc.
2. Hủy bỏ: Tại bất kỳ bước nào trong luồng cơ bản nếu người dùng nhấn vào một mục khác trên menu chính của người quản trị thì use case sẽ kết thúc.

* **Các yêu cầu đặc biệt:** Không có.
* **Tiền điều kiện:**

Người dùng đã có một tài khoản trong hệ thống.

* **Hậu điều kiện:**

Không có.

* **Điểm mở rộng:**

Không có.

##### **Mô tả use case “Quản lý sản phẩm”.**

**Mô tả vắn tắt use case**: Use case này cho phép người dùng quản lí các sản phẩm.

**Luồng sự kiện:**

* **Luồng cơ bản:**

1. Use case này bắt đầu khi người dùng kích vào nút “Quản Lý Sản Phẩm”.

Hệ thống sẽ hiển thị ra form “Quản Lý Sản Phẩm”, từ đây người dùng có thể “THÊM”,”SỬA” hoặc “XÓA” các sản phẩm trong kho. Use case kết thúc khi người dùng tắt form “Quản Lý Sản Phẩm”.

* Chức năng Thêm sản phẩm: Chức năng này bắt đầu khi người dùng kích chọn nútC:\Users\LENOVO\Desktop\icon\add5.png trên form chính. Hệ thống sẽ hiển thị ra form Thêm sản phẩm. Người dùng nhập các thông tin Tên sản phẩm ([1-50] ký tự), màu sắc ([1-20] ký tự), kích thước ([1-3] ký tự), nhà sản xuất [1-30] ký tự), số lượng trong kho (Z, [0-10000]), giá (Z,[0-2000000]) và mô tả ([1-200] ký tự). Người dùng kích chọn nút C:\Users\LENOVO\Desktop\icon\save.png để lưu lại. Hệ thống kiểm tra dữ liệu nhập vào, lưu lại và hiển thị thông báo.
* Chức năng Sửa sản phẩm:Chức năng này bắt đầu khi người dùng kích chọn sản phẩm cần sửa trên danh sách sản phẩm. Hệ thống sẽ hiển thị thông tin sản phẩm trên các trường tương ứng trong form chính. Người dùng Sửa các thông tin Tên sản phẩm ([1-50] ký tự), màu sắc ([1-20] ký tự), kích thước ([1-3] ký tự), nhà sản xuất [1-30] ký tự), số lượng trong kho (Z, [0-10000]), giá (Z,[0-2000000]) và mô tả ([1-200] ký tự). Người dùng kích chọn nút C:\Users\LENOVO\Desktop\icon\update.pngđể lưu lại. Hệ thống sẽ kiểm tra dữ liệu, lưu lại và đưa ra thông báo kết quả.
* Chức năng xóa sản phẩm: Chức năng này bắt đầu khi người dùng kích chọn sản phẩm cần xóa trên danh sách. Người dùng kích chọn nút C:\Users\LENOVO\Desktop\icon\delete1.pngđể xóa sản phẩm đã chọn. Hệ thống hiển thị thông báo xác nhận xóa. Người dùng kích chọn nút “Yes” để xóa. Hệ thống sẽ xóa sản phẩm trong dữ liệu.
* **Luồng rẽ nhánh:**

2. Không kết nối được với CSDL: Tại bất kỳ thời điểm nào trong quá trình thực hiện

use case nếu không kết nối được với CSDL thì hệ thống sẽ hiển thị một thông

báo lỗi và use case kết thúc.

* **Các yêu cầu đặc biệt:** Không có.
* **Tiền điều kiện:**

Người dùng phải đang ở trong form quản lí sản phẩm..

* **Hậu điều kiện:**

Không có.

* **Điểm mở rộng:**

Không có.

#### Biểu đồ.

##### **Phân tích chức năng Xem danh sách hóa đơn**

* **Biểu đồ trình tự**

****

*Hình 2.1 Biểu đồ trình tự chức năng Xem danh sách hóa đơn*

* **Biểu đồ lớp phân tích**

****

*Hình 2.2 Biểu đồ lớp phân tích chức năng Xem danh sách hóa đơn*

* **Biểu đồ lớp màn hình**

****

*Hình 2.3 Biểu đồ lơp màn hình chức năng Xem danh sách hóa đơn*

* **Biểu đồ cộng tác của các màn hình**



*Hình 2.4 Biểu đồ cộng tác chức năng Xem danh sách hóa đơn*

##### **Phân tích chức năng Xem chi tiết hóa đơn**

* **Biểu đồ trình tự**

****

*Hình 2.5 Biểu đồ trình tự chức năng Xem chi tiết hóa đơn*

* **Biểu đồ lớp phân tích**

****

*Hình 2.6 Biểu đồ lớp phân tích chức năng Xem chi tiết hóa đơn*

* **Biểu đồ lớp màn hình**

****

*Hình 2.7 Biểu đồ lớp màn hình chức năng Xem chi tiết hóa đơn*

* **Biểu đồ cộng tác của các màn hình**



*Hình 2.8 Biểu đồ cộng tác của các màn hình chức năng Xem chi tiết hóa đơn*

##### **Phân tích chức năng Xóa hóa đơn**

* **Biểu đồ trình tự**

****

*Hình 2.9 Biểu đồ trình tự chức năng Xóa hóa đơn*

* **Biểu đồ lớp phân tích**

****

*Hình 2.10 Biểu đồ lớp phân tích chức năng Xóa hóa đơn*

* **Biểu đồ lớp màn hình**

****

*Hình 2.11 Biểu đồ lớp màn hình chức năng Xóa hóa đơn*

* **Biểu đồ cộng tác của các màn hình**



*Hình 2.12 Biểu đồ cộng tác của các màn hình chức năng Xóa hóa đơn*

##### **Phân tích chức năng Tính doanh thu ngày**

* **Biểu đồ trình tự**

****

*Hình 2.13 Biểu đồ trình tự chức năng Tính doanh thu ngày*

* **Biểu đồ lớp phân tích**

****

*Hình 2.14 Biểu đồ lớp phân tích chức năng Tính doanh thu ngày*

* **Biểu đồ lớp màn hình**

****

*Hình 2.15 Biểu đồ lớp màn hình chức năng Tính doanh thu ngày*

* **Biểu đồ cộng tác của các màn hình**



*Hình 2.16 Biểu đồ cộng tác của các màn hình chức năng Tính doanh thu ngày*

##### **Phân tích chức năng Tính doanh thu tháng**

* **Biểu đồ trình tự**

****

*Hình 2.17 Biểu đồ trình tự chức năng Tính doanh thu tháng*

* **Biểu đồ lớp phân tích**

****

*Hình 2.18 Biểu đồ lớp phân tích chức năng Tính doanh thu tháng*

* **Biểu đồ lớp màn hình**

****

*Hình 2.19 Biểu đồ lớp màn hình chức năng Tính doanh thu tháng*

* **Biểu đồ cộng tác của các màn hình**



*Hình 2.20 Biểu đồ cộng tác của các màn hình chức năng Tính doanh thu tháng*

##### **Phân tích chức năng Nhập hóa đơn**

* **Biểu đồ trình tự**

**** *Hình 2.21 Biểu đồ trình tự chức năng Nhập hóa đơn*

* **Biểu đồ lớp phân tích**

****

*Hình 2.22 Biểu đồ lớp phân tích chức năng Nhập hóa đơn*

* **Biểu đồ lớp màn hình**

****

*Hình 2.23 Biểu đồ lớp màn hình chức năng Nhập hóa đơn*

* **Biểu đồ cộng tác của các màn hình**

****

*Hình 2.24 Biểu đồ cộng tác của các màn hình chức năng Nhập hóa đơn*

##### **Phân tích chức năng Thanh Toán**

* **Biểu đồ trình tự**

****

*Hình 2.25 Biểu đồ trình tự chức năng Thanh toán*

* **Biểu đồ lớp phân tích**

****

*Hình 2.26 Biểu đồ lớp phân tích chức năng Nhập hóa đơn*

##### **Phân tích chức năng Thêm (Thêm mới hóa đơn)**

* **Biểu đồ trình tự**

****

*Hình 2.27 Biểu đồ trình tự chức năng Thêm (Thêm mới hóa đơn)*

* **Biểu đồ lớp phân tích**



*Hình 2.28 Biểu đồ lớp phân tích chức năng Thêm (Thêm mới hóa đơn)*

##### **Phân tích chức năng Xóa ( Thêm mới hóa đơn)**

* **Biểu đồ trình tự**

****

*Hình 2.29 Biểu đồ trình tự chức năng Xóa (Thêm mới hóa đơn)*

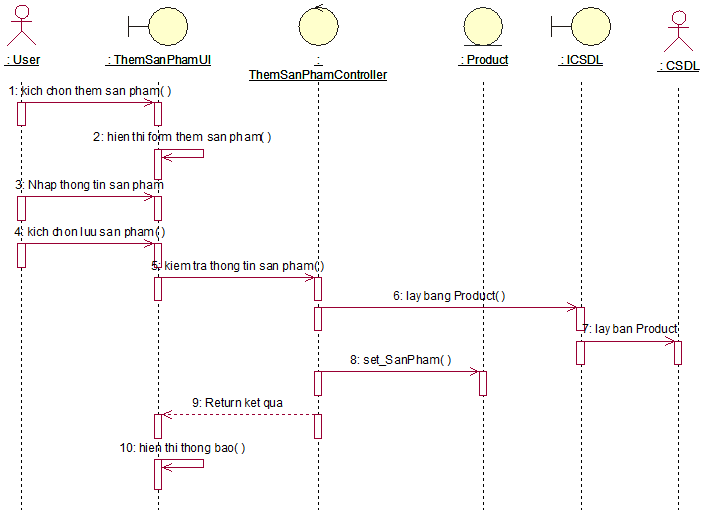
* **Biểu đồ lớp phân tích**

****

*Hình 2.30 Biểu đồ lớp phân tích chức năng Xóa (Thêm mới hóa đơn)*

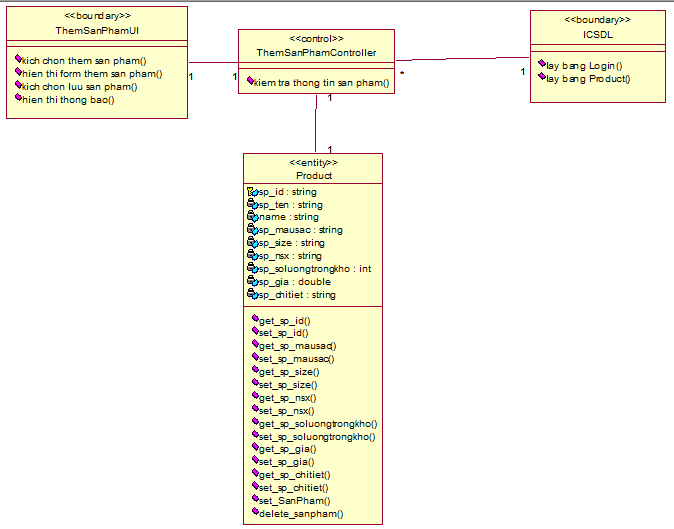
##### Phân tích chức năng “Thêm sản phẩm”

* Biểu đồ trình tự



*Hình 2.31 Biểu trình tự chức năng Thêm sản phẩm*

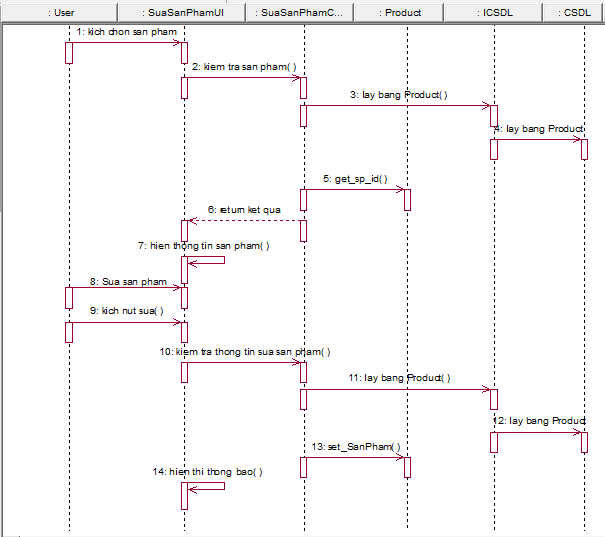
* Biểu đồ lớp phân tích



*Hình 2.32 Biểu đồ lớp phân tích chức năng Thêm sản phẩm*

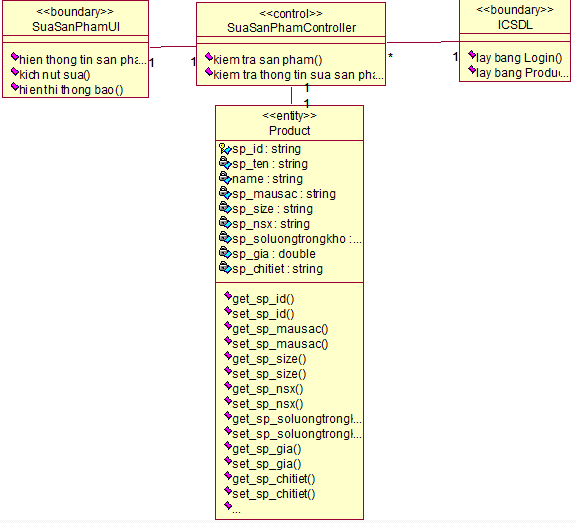
##### Phân tích chức năng “Sửa sản phẩm”

* Biểu đồ trình tự



*Hình 2.33 Biểu đồ trình tự chức năng Sửa sản phẩm*

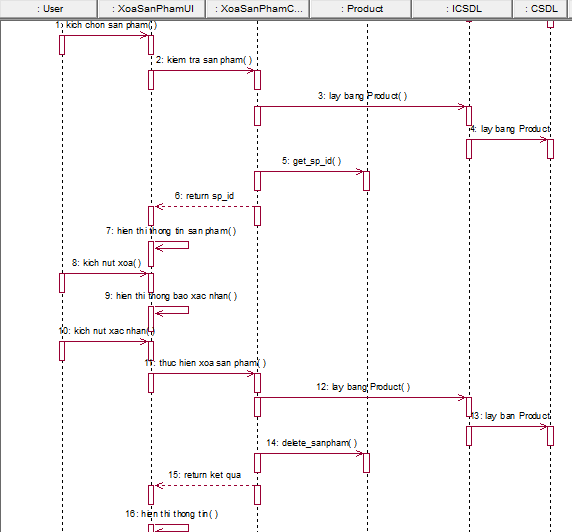
* Biểu đồ lớp phân tích



*Hình 2.34 Biểu đồ lớp phân tích chức năng Sửa sản phẩm*

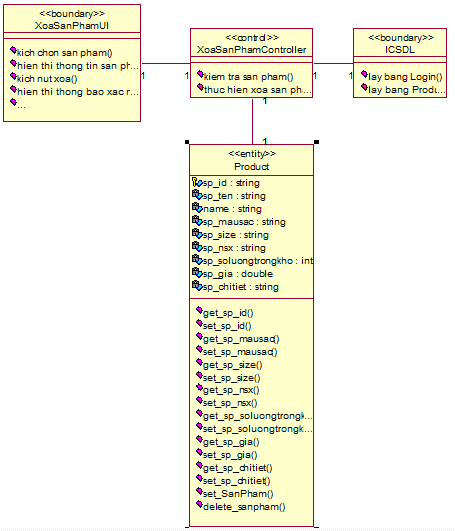
##### Phân tích chức năng “Xóa sản phẩm”

* Biểu đồ trình tự



*Hình 2.35 Biểu đồ trình tự chức năng Xóa sản phẩm*

* Biểu đồ lớp phân tích



*Hình 2.36 Biểu đồ lớp phân tích chức năng Xóa sản phẩm*

# Phần mềm “Quản lý kho”

## Lập kế hoạch kiểm thử

Kế hoạch kiểm thử là tài liệu mô tả về phạm vi, cách tiếp cận, các nguồn lực và kế hoạch thực hiện kiểm thử.

* Mục tiêu của lập kế hoạch kiểm thử:
* Thiết lập được mục tiêu ngắn hạn và dài hạn của việc kiểm thử.
* Nhận biết được các rủi ro có thể xảy ra.
* Xác định được cách tiếp cận và kế hoạch cho việc kiểm thử.
* Nhiệm vụ chính của hoạt động lập kế hoạch kiểm thử:
* Xác định được phạm vi kiểm thử, các rủi ro có thể xảy ra, xác định được mục tiêu kiểm thử để đảm bảo đo đạc được chất lượng phần mềm.
* Xác định cách tiếp cận việc kiểm thử, các kĩ thuật được sử dụng, độ bao phủ kiểm thử cho phép.
* Thực thi theo chính sách và chiến lược kiểm thử.
* Xác định nguồn lực, chi phí kiểm thử cần thiết.
* Lên kế hoạch cho hoạt động phân tích, thiết kế, thực thi kiểm thử.
* Xác định tiêu chí kết thúc kiểm thử.
* Liệt kê những kết quả, tài kiệu có được sau khi thực hiện kiểm thử.

### Nguyễn Văn Nam – Kiểm thử chức năng “Tính doanh thu tháng”

#### Mục tiêu của lập kế hoạch kiểm thử

* Mục tiêu ngắn hạn: Kiểm tra được ngay lập tức, liệu hoạt động của chức năng tính doanh thu này có hoạt động ổn định và theo bản thiết kế hay không.
* Mục tiêu dài hạn: Đảm bảo chức năng hoạt động ổn định và chính xác với nhiều trường hợp khác nhau. Vì chức năng này là một trong những chức năng có vai trò quan trọng đối với tính chính xác của hệ thống.
* Rủi ro có thể xảy ra: Đầu vào của chức năng có thể không phù hợp với tham số của chức năng. Hay đầu ra của chức năng không được như mong muốn tức là có vấn đề trong mã nguồn của chức năng.
* Cách tiếp cận và kế hoạch cho kiểm thử:
* Cách tiếp cận kiểm thử: Kiểm thử BigBang. Vì kiểm thử big bang kiểm thử những module nhỏ thành các module hoàn chỉnh, mà hệ thống này tương đối nhỏ và đã hoàn thành cấu trúc. Nên cách tiếp này là phù hợp với cách kiểm thử của chức năng này.
* Kế hoạch cho việc kiểm thử: Như đã trình bày ở cách tiếp cận kiểm thử thì việc kiểm thử sẽ thực hiện khi module chức năng đã được hoàn thành và tích hợp. Và kiểm thử module này cho đến khi được như mong muốn.
* Mục tiêu kiểm thử: Đảm bảo chức năng tính doanh thu tháng hoạt động ổn định, đúng như bản thiết kế.

#### Nhiệm vụ của lập kế hoạch kiểm thử

* Xác định phạm vi kiểm thử, các rủi ro có thể xảy ra, xác định được mục tiêu kiểm thử để đảm bảo đo đạc được chất lượng phần mềm.
* Xác định phạm vi kiểm thử: bao gồm việc kiểm tra mã, thực hiện các mã trong môi trường và điều kiện khác nhau, và việc kiểm thử các khía cạnh của mã nguồn chức năng tính doanh thu ngày: nó có làm đúng nhiệm vụ của nó hay không, và nó có làm những gì cần phải làm hay không.
* Rủi ro có thể xảy ra: đầu vào không đúng như yêu cầu của chức năng dẫn tới chức năng không thể hoạt động, hay mã nguồn của chức năng không đúng như yêu cầu của bản thiết kế dẫn tới chức năng không hoạt động đúng chức năng của nó.
* Xác định cách tiếp cận việc kiểm thử, các kĩ thuật được sử dụng, độ bao phủ kiểm thử cho phép.
* Cách tiếp cận: Như đã trình bày ở trên thì phương pháp tiếp cận hợp lý nhất với chức năng tính doanh thu ngày này là từ dưới lên.
* Các kĩ thuật sử dụng: kĩ thuật đánh giá rủi ro, phân tích rủi ro, xác định cách tiếp cận, xác định mục tiêu…
* Độ bao phủ kiểm thử cho phép: độ bao phủ kiểm thử phải đạt ít nhất 85%.
* Thực thi theo chính sách và chiến lược kiểm thử:
* Thực thi theo chính sách: Đảm bảo nguyên tắc kiểm thử hết với các giá trị biên, giá trị thỏa mãn, mỗi đường thực thi đêu được kiểm thử ít nhất một lần,... và hoàn thành tất cả các mục tiêu nêu trên.
* Chiến lược kiểm thử: Xác định chức năng sẽ được kiểm thử ở mức kiểm thử đơn vị, kiểm thử tích hợp.
* Xác định nguồn lức, chi phí kiểm thử cần thiết: Nguồn lực ở đây chính là người phát triển phần mềm nên không cần thêm bất kỳ chi phí nào.
* Xác định tiêu chí kết thúc kiểm thử: Chức năng hoạt động tốt, ổn định như bản thiết kế ban đầu và yêu cầu của khách hàng.
* Bảng kế hoạch kiểm thử:

|  |  |  |  |
| --- | --- | --- | --- |
| STT | Task | Time | Date |
| 1 | Phân tích chức năng tính doanh thu theo tháng | 19h | 01/04/2019 – 02/24/2019 |
| 2 | Thu thập tài liệu, tìm hiểu unit test với NUnit và công cụ hỗ trợ NUnitForms | 48h | 02/04/2019 – 04/04/2019 |
| 3 | Phân tích test case, viết test script | 48h | 04/04/2019 – 06/04/2019 |
| 4 | Phân tích kết quả test | 8h | 06/04/2019 – 07/04/2019 |
| 5 | Quay video, viết báo cáo | 8h | 07/04/2019 – 08/04/2019 |

### Nguyễn Văn Hào – Kiểm thử chức năng “Thêm sản phẩm”

#### Mục tiêu của lập kế hoạch kiểm thử

* Mục tiêu ngắn hạn: Phát hiện các lỗi của hệ thống, đồng thời cố gắng sửa các lỗi đó.
* Mục tiêu dài hạn: Đảm bảo mô đun chạy đúng theo đặc tả, ổn định trong thời gian dài, tỷ lệ xuất hiện lỗi là thấp nhất.
* Xác định cách tiếp cận và kế hoạch cho kiểm thử
* Cách tiếp cận: Big bang: Tất cả các thành phần được tích hợp cùng một lúc, sau đó tiến hành kiểm thử. Bởi đây là phần mềm có quy mô nhỏ, hệ thống đơn giản,tất cả các mô đun đã được hoàn thiện
* Thực hiện kiểm thử: Sử dụng phương pháp tiếp cận big bang, bởi vậy sau khi các thành phần đã tích hợp, hoàn thiện. Ta thực hiện kiểm thử mà không cần phải xây dựng các stubs hay drives như phương pháp tiếp cận tăng dần.

#### Nhiệm vụ của lập kế hoạch kiểm thử

1. Xác định được phạm vi kiểm thử, các rủi ro có thể xảy ra:

* Xác định phạm vi kiểm thử: bao gồm việc kiểm tra mã, thực hiện các mã trong môi trường và điều kiện khác nhau, và việc kiểm thử các khía cạnh của mã nguồn chức năng tính doanh thu ngày: nó có làm đúng nhiệm vụ của nó hay không, và nó có làm những gì cần phải làm hay không.
* Rủi ro có thể xảy ra: đầu vào không đúng như yêu cầu của chức năng dẫn tới chức năng không thể hoạt động, hay mã nguồn của chức năng không đúng như yêu cầu của bản thiết kế dẫn tới chức năng không hoạt động đúng chức năng của nó.

1. Xác định cách tiếp cận việc kiểm thử:

* Cách tiếp cận: Như đã trình bày ở trên thì phương pháp tiếp cận Big Bang
* Các kĩ thuật sử dụng: kĩ thuật đánh giá rủi ro, phân tích rủi ro, xác định cách tiếp cận, xác định mục tiêu…
* Độ bao phủ kiểm thử cho phép: độ bao phủ kiểm thử phải đạt ít nhất 90%.

1. Thực thi theo chính sách và chiến lược kiểm thử:

* Chiến lược kiểm thử: Xác định chức năng sẽ được kiểm thử ở mức kiểm thử đơn vị, kiểm thử tích hợp.

1. Xác định nguồn lực kiểm thử cần thiết:
   * Nguồn lực ở đây chính là người phát triển phần mềm nên không cần thêm bất kỳ chi phí nào.
   * Công cụ hỗ trợ Nunit và các framework của Nunit, excel…
2. Lên kế hoạch cho hoạt động phân tích, thiết kế, thực thi…

* Bảng kế hoạch kiểm thử:

|  |  |  |  |
| --- | --- | --- | --- |
| STT | Task | Time | Date |
| 1 | Phân tích chức năng thêm sản phẩm | 8h | 01/04/2019 |
| 2 | Thu thập tài liệu, tìm hiểu unit test với NUnit và công cụ hỗ trợ NUnitForms | 48h | 02/04/2019 – 04/04/2019 |
| 3 | Phân tích test case, viết test script | 72h | 04/04/2019 – 07/04/2019 |
| 4 | Phân tích kết quả test | 8h | 08/04/2019 |
| 5 | Quay video, viết báo cáo | 10h | 08/04/2019 |

1. Xác định tiêu chí kết thúc kiểm thử

Độ bao phủ của ca kiểm thử đạt trên 90% và số test case (pass) trên 90%.

### Lê Mạnh Hùng – Kiểm thử chức năng “Thanh toán hóa đơn”

#### Mục tiêu của lập kế hoạch kiểm thử

* Mục tiêu ngắn hạn: Tìm các bug phát sinh do người lập trình tạo ra khi code.
* Mục tiêu dài hạn: Đảm bảo rằng kết quả cuối cùng đáp ứng các yêu cầu kinh doanh và người sử dụng.
* Rủi ro có thể xảy ra: Đầu vào của số lượng bán không phù hợp với tham số của chức năng thanh toán. Hay đầu ra của chức năng của không được như mong đợi.
* Cách tiếp cận và kế hoạch kiểm thử :
  + Cách tiếp cận kiểm thử : Kiểm thử từ dưới lên(Bottom - up). Trong phương pháp này, kiểm thử được thực hiện từ dưới lên. Cụ thể là, các module ở lớp dưới cùng (module lá) sẽ được tích hợp và kiểm thử trước (các module này không gọi module nào khác) và sau đó đến lần lượt là các module trực tiếp gọi 1 hay nhiều module vừa được kiểm thử được tích hợp vào và test. Những module chưa có hay chưa được phát triển sẽ được thay thế bởi các drivers.
  + Kế hoạch cho việc kiểm thử : Thực hiện xây dựng testcase theo phương pháp Bottom – Up, viết drivers hoặc stub để phục vụ cho việc kiểm thử.

#### Nhiệm vụ của lập kế hoạch kiểm thử

1. Xác định được phạm vi kiểm thử, các rủi ro có thể xảy ra:

* Xác định phạm vi kiểm thử: bao gồm việc kiểm tra mã, thực hiện các mã trong môi trường và điều kiện khác nhau, và việc kiểm thử các khía cạnh của mã nguồn chức năng tính doanh thu ngày: nó có làm đúng nhiệm vụ của nó hay không, và nó có làm những gì cần phải làm hay không.
* Rủi ro có thể xảy ra: Đầu vào của số lượng bán không phù hợp với tham số của chức năng thanh toán. Hay đầu ra của chức năng của không được như mong đợi dẫn tới việc chức năng không thực hiện đúng yêu cầu như được thiết kế của nó.

1. Xác định cách tiếp cận việc kiểm thử:

* Cách tiếp cận: Cách tiếp cận phù hợp nhất là kiểm thử từ dưới lên.
* Kỹ thuật: Duyệt danh sách sản phẩm chọn trong hóa đơn, kiểm tra điều kiện số lượng bán phù hợp. Kiểm tra chức năng thanh toán có hoạt động đúng thiết kế hay không. Tổng tiền trên lý thuyết và tổng tiền chức năng chạy phải bằng nhau.
* Độ bao phủ kiểm thử cho phép: độ bao phủ kiểm thử phải đạt ít nhất 90%.

1. Thực thi theo chính sách và chiến lược kiểm thử:

* Chiến lược kiểm thử: Xác định chức năng sẽ được kiểm thử ở mức kiểm thử đơn vị, kiểm thử tích hợp.

1. Xác định nguồn lực kiểm thử cần thiết:
   * Nguồn lực ở đây chính là người phát triển phần mềm nên không cần thêm bất kỳ chi phí nào.
   * Công cụ hỗ trợ Nunit và các framework của Nunit 2.7.0, NunitForm v2 alpha5 ,excel…
2. Lên kế hoạch cho hoạt động phân tích, thiết kế, thực thi…

* Bảng kế hoạch kiểm thử:

|  |  |  |  |
| --- | --- | --- | --- |
| STT | Task | Time | Date |
| 1 | Phân tích chức năng thanh toán hóa đơn | 9h | 01/04/2019 |
| 2 | Thu thập tài liệu, tìm hiểu unit test với NUnit và công cụ hỗ trợ NUnitForms | 48h | 02/04/2019 – 04/04/2019 |
| 3 | Phân tích test case, viết test script | 72h | 04/04/2019 – 07/04/2019 |
| 4 | Phân tích kết quả test | 10h | 08/04/2019 |
| 5 | Quay video, viết báo cáo | 12h | 08/04/2019 |

1. Xác định tiêu chí kết thúc kiểm thử

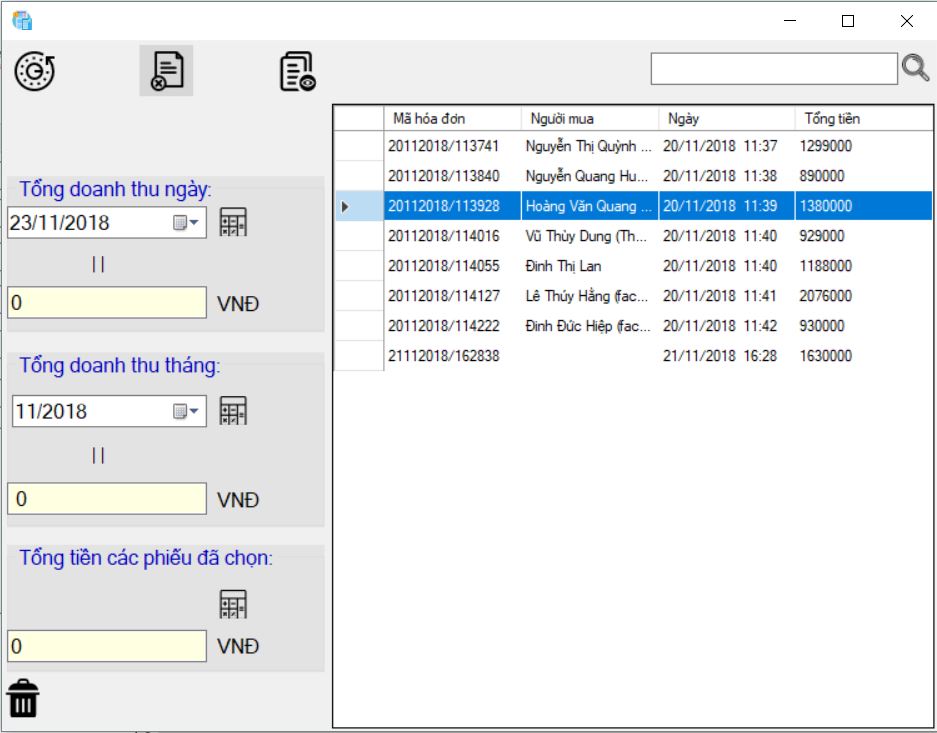
Độ bao phủ của ca kiểm thử đạt trên 90% và số testcase đạt mức mong đợi trên 90%.

## Thực hiện kiểm thử

### Nguyễn Văn Nam - Tính doanh thu tháng (V0.1)

#### Phân tích thiết kế kểm thử

* Giao diện chức năng :



*Hình 3.1 Giao diện tình doanh thu*

* Mã nguồn của chức năng

public Int32 TotalMonth(String s) Begin

{

Int32 tong = 0; **1**

if (isTruePattern(s) == false) **2**

{

MessageBox.Show("ERRO PARTTERN"); **3**

}

else

**5**

**9**

**4**

{

for (Int32 i = 0; i < dgvDSHD.RowCount; i++)

{

String temp = dgvDSHD.Rows[i].Cells[2].Value.ToString(); **6**

String cpr = temp.Substring(3, 7);

if (s.Equals(cpr)) **7**

{

tong += Convert.ToInt32(dgvDSHD.Rows[i].Cells[3].Value.ToString()); **8**

}

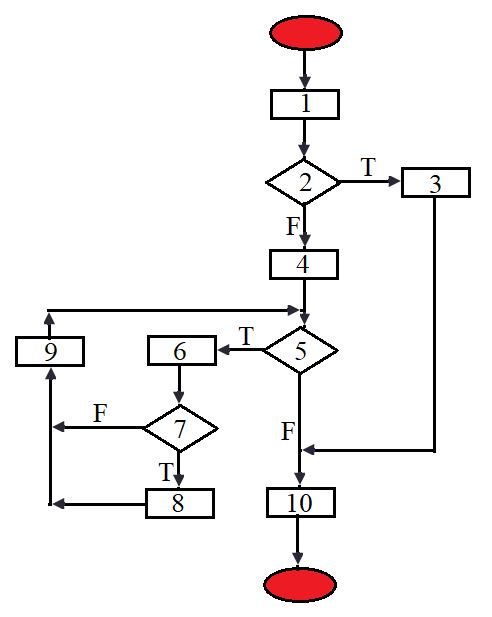
}

}

return tong; **10**

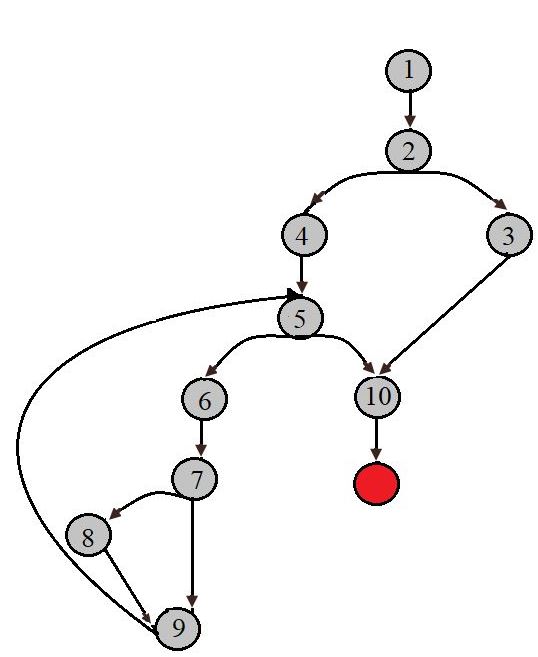
} End

* Lưu đồ giải thuật hoặc sơ đồ đồ thị dòng
* Lưu đồ giải thuật



*Hình 3.1 Lưu đồ thuật giải chức năng tính doanh thu tháng*

* Sơ đồ đồ thị dòng



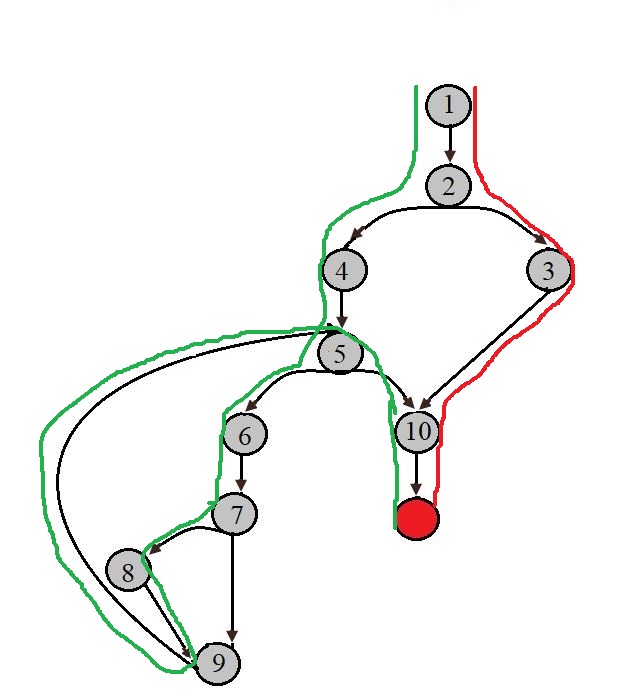
*Hình 3.2 Sơ đồ đồ thị dòng từ lưu đồ thuật giải ở trên*

* Độ phức tạp chu trình

C = 3 + 1 = 4

1. 1, 2, 3, 10.
2. 1, 2, 4, 5, 10.
3. 1, 2, 4, 5, 6, 7, 8, 9, 5, 10.
4. 1, 2, 4, 5, 6, 7, 9, 5, 10.

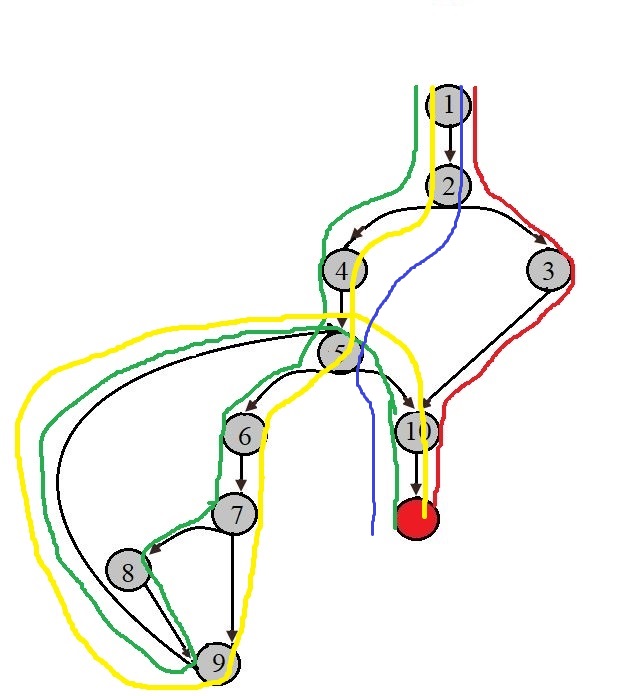
* Các cấp bao phủ kiểm thử
* Bao phủ cấp 1 : Các câu lệnh được thực thi ít nhất 1 lần. Với chức năng trên thì chỉ cần 2 test case để đạt được 100% bao phủ cấp 1. Chính là 2 đường thi hành chính đã thông báo ở trên.



*Hình 3.3 Sơ đồ thể hiện đường đi của bài toán trong bao phủ cấp 1*

* Bao phủ cấp 2 : Bao phủ nhánh

Với 2 test case xác định trong phần trước thì ta chỉ đạt được 1/2 = 50% độ bao phủ nhánh, ta phải thêm hai test case để độ bao phủ nhánh đạt 100%.



*Hình 3.4 Sơ đồ thể hiện đường đi của bài toán trong bao phủ cấp 2*

* Thiết kế test case :
  + Mô tả điều kiện :
* Bắt buộc phải nhập.
* Chuỗi nhập theo định dạng : MM/yyyy
* Phải đúng 7 ký tự.
  + Xây dựng test case :
* Áp dụng phương pháp phân tích giá trị biên, phân vùng tương đương.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Đầu vào | Vùng hợp lệ | Ký hiệu đánh dấu | Vùng không hợp lệ | Đánh dấu |
| Chuỗi tháng năm | Đúng định dạng 7 ký tự | H1 | >7 | K1 |
| <7 | K2 |
| Bỏ trống | K3 |
| Nhập sai định dạng | K4 |

*Bảng 3.1 Bảng phân vùng tương đương.*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| TCs | Input | Output | | Cover |
| Return | Message |
| 1 | 12/2018 | 111010000 | SUITABLE FORMAT | H1 |
| 2 | 1212/2019 | 0 | ERRO PATTERN | K1 |
| 3 | /2018 | 0 | ERRO PATTERN | K2 |
| 4 |  | 0 | ERRO PATTERN | K3 |
| 5 | 1a%2 19 | 0 | ERRO PATTERN | K4 |
| 6 | Oct/2018 | 0 | ERRO PATTERN | K4 |
| 7 | \_1/2017 | 0 | ERRO PATTERN | K4 |
| 8 | 2019/08 | 0 | ERRO PATTERN | K4 |
| 9 | 1/2015 | 0 | ERRO PATTERN | K4 |
| 10 | 13/2018 | 0 | ERRO PATTERN | K4 |
| 11 | 00/2019 | 0 | ERRO PATTERN | K4 |
| 12 | $#/#$%^ | 0 | ERRO PATTERN | K4 |

*Bảng 3.2 Ca kiểm thử.*

* TestcaseTinhDoanhThuThang.xlsx

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Tcs | Tên test case | Input | Output |  |
|  |  |  | Return | Messsage |
| Tc1 | Tính doanh thu tháng | 12/2018 | 111010000 | SUITABLE FORMAT |
| Tc2 | Tính doanh thu tháng | 1212/2019 | 0 | ERRO PATTERN |
| Tc3 | Tính doanh thu tháng | /2018 | 0 | ERRO PATTERN |
| Tc4 | Tính doanh thu tháng |  | 0 | ERRO PATTERN |
| Tc5 | Tính doanh thu tháng | 1a%2 19 | 0 | ERRO PATTERN |
| Tc6 | Tính doanh thu tháng | Oct/2018 | 0 | ERRO PATTERN |
| Tc7 | Tính doanh thu tháng | \_1/2017 | 0 | ERRO PATTERN |
| Tc8 | Tính doanh thu tháng | 2019/08 | 0 | ERRO PATTERN |
| Tc9 | Tính doanh thu tháng | 1/2015 | 0 | ERRO PATTERN |
| Tc10 | Tính doanh thu tháng | 13/2018 | 0 | ERRO PATTERN |
| Tc11 | Tính doanh thu tháng | 00/2019 | 0 | ERRO PATTERN |
| Tc12 | Tính doanh thu tháng | $#/#$%^ | 0 | ERRO PATTERN |

*Bảng 3.4 Datatest sau khi phân tích test case.*

#### Thực hiện kiểm thử

1. Cài đặt:

B1: Truy cập trang web NUnit và NUnit để tải bản NUnitForm mới nhất và bản NUnit 2.7.0 (Nhóm em chọn bản 2.7.0 vì bản này có hỗ trợ GUI giúp cho việc run nhiều test case một lúc thuận tiện hơn).

B2: Giải nén hai file vừa tải và cài đặt bản NUnitForm, còn bản NUnit 2.7.0 thì không cần. Cài đặt NUnitForm như cài đặt một phần mềm bình thường.

1. Cầu hình:

B1: Vào project, sau đó tạo một class library tên Test (nguyenvannam186) (Tên tùy ý). B2: Giờ ta chuột phải vào Reference trong class library (nguyenvannam186) 🡪 chọn Add Reference 🡪 chọn Brows 🡪 chọn browse… 🡪 Tìm đến nơi vừa cài đặt bản NUnitFrom (thường là: C:\Program Files (x86)\NUnitForms) 🡪 chọn file NUnitForms.dll, sau đó chúng ta tìm đến folder NUnit vừa giải nén và add file unit.framework.dll

1. Chuẩn bị test:

B1: Sau khi phân tích test case ở trên thì chúng ta cần tạo ra một file excel để chứa các dữ liệu để phục vụ test được thuận tiện và mang tính tự động cao hơn.

B2: Để việc test được tự động thì việc dữ liệu được tự động đưa vào test case và việc ghi lại kết quả của test case là vô cung cần thiết. Nên ta tạo một class đọc ghi file trong class library Test (nguyenvannam186) với tên DocGhiFileExcel.cs (tên tùy ý).

B3: Xây dựng mã nguồn của class DocGhiFileExcel vừa tạo. Dưới đây là mã của class DocGhiFileExcel kèm chú thích.

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Data.OleDb; //Thư viện hỗ trợ đọc ghi file excel

using System.Data;

using System.Windows.Forms;

using System.IO;

using System.Runtime.InteropServices; //Thư viện hỗ trợ đọc ghi file Excel, cái này cần add reference vào vì nó không có sẵn trong C#

using Excel = Microsoft.Office.Interop.Excel; // Đối tượng thay thế cho tên thư viện

using System.Collections.ObjectModel;

class DocGhiFileExcel

{

public static Excel.Application xlApp;

public static Excel.Workbook xlWorkbook; //Sử dụng Workbook cho file đuôi

.xlsx và HWorkbook đối với file đuôi .slx

public static Excel.\_Worksheet xlWorksheet; //Quản lý các sheet trong excel

public static Excel.Range xlRange; //Quản lý các thành phần trong 1 sheet

public static void open(String fileName)

{

//Tạo đối tượng COM và mở kết nối tới các đối tượng cần dùng để đọc dữ

Liệu

xlApp = new Excel.Application();

xlWorkbook = xlApp.Workbooks.Open(fileName);

xlWorksheet = xlWorkbook.Sheets[1];

xlRange = xlWorksheet.UsedRange;

}

public static void close()

{

//Đóng toàn bộ

GC.Collect();

GC.WaitForPendingFinalizers();

Marshal.ReleaseComObject(xlRange);

Marshal.ReleaseComObject(xlWorksheet);

//Đóng và thu hồi kết nối

xlWorkbook.Save();

xlWorkbook.Close();

Marshal.ReleaseComObject(xlWorkbook);

//Thoát và thu hồi kết nối

xlApp.Quit();

Marshal.ReleaseComObject(xlApp);

}

//Phương thức này được dùng để tạo ra một danh sách các mảng đối tượng một

Chiều, mỗi một mảng là một dòng trong bảng tính Excel.

public static Collection<Object[]> getExcelFile(String fileName)

{

open(fileName);

int rowCount = xlRange.Rows.Count;

int colCount = 5;

int j = 3; //Vì mình chỉ đọc dữ liệu từ cột số 3 đến cột số 5

Object[] obj;

Collection<Object[]> list = new Collection<object[]>();

for (int i = 3; i <= rowCount; i++)

{

// Đọc từng thành phần trong từng dòng của file Excel

String thang = xlRange.Cells[i, j++].Value2.ToString();

Int32 tien = Convert.ToInt32(xlRange.Cells[i, j++].Value2.ToString());

String mess = xlRange.Cells[i, j++].Value2.ToString();

//Tạo đối tượng lưu các thành phần vừa đọc được vào một mảng đối tượng chiều

obj = new Object[] {thang, tien, mess};

list.Add(obj); //Thêm đối tượng vừa tạo được vào danh sách để có thể

truyền vào TestCaseSource bên test.

if (j > colCount)

j = 3;

}

close();//Đóng kết nối file Excel

return list;

}

//Phương thức này sử dụng để ghi dữ liệu vào file Excel theo từng thành phần

public static void setExcelFile(int i, int j, String str, String fileName)

{

open(fileName);

xlRange.Cells[i, j].value2 = str;

close();

}

}

1. Viết Test:

Chúng ta tạo một class tên myTest.cs (tên tùy ý), và mã bên trong class myTest.cs như sau:

using GUI;

using System;

using NUnit.Extensions.Forms; //Sử dụng để viết test, nhất thiết phải có

using NUnit.Framework; //Cần có để sử dụng các annotation trong NUnit

using System.Windows.Forms;

using System.Runtime.InteropServices;

using Excel = Microsoft.Office.Interop.Excel;

using System.Collections.Generic;

using System.Collections.ObjectModel;

namespace Test\_NguyenVanNam\_186\_

{

[TestFixture] //Annotation của NUnit, báo cho biết ở đây chứa các test case

public class MyTest

{

public int row = 3;

public static Collection<Object[]> data()

{

//Lấy về danh sách các mảng đối tượng một chiều để đọc thành data test

return DocGhiFileExcel.getExcelFile(@"F:\C#\quanlykho\DataTest\TestcaseTinhDoanhThuThang.xlsx");

}

//Đọc dữ liệu từ danh sách vừa thu được thành đầu vào cho các test case

[TestCaseSource("data"), Test]

public void testTinh(String thang, Int32 expectedValue, String mess)

{

//Tạo đối tượng của form để có thể quản lý các thành phần của form

danhsachhoadon ds = new danhsachhoadon();

ds.Show();//Khời động form

//Dùng để lấy ra thông tin từ MessageBox để so sánh

String actMsg = "";

ModalFormTester msgBox = new ModalFormTester();

msgBox.ExpectModal("NOTE", delegate

{

MessageBoxTester msg = new MessageBoxTester("NOTE");

actMsg = msg.Text;

msg.ClickOk();//Phát hiện ra MessageBox cần chứa thông tin cần

tìm rối click OK

});

if (expectedValue == ds.TotalMonth(thang) && mess.Equals(actMsg))

{

//So sánh kết quả mong đợi và thực rồi ghi vào file Excel

DocGhiFileExcel.setExcelFile(row++, 6, "Pass", @"F:\C#\quanlykho\DataTest\TestcaseTinhDoanhThuThang.xlsx");

}

else

{

DocGhiFileExcel.setExcelFile(row++, 6, "Fail", @"F:\C#\quanlykho\DataTest\TestcaseTinhDoanhThuThang.xlsx");

}

//đảm bảo việc so sánh cả hai kết quả để đảm bảo không bỏ sót điều kiện sánh của test case

if(expectedValue == ds.TotalMonth(thang))

{

//So sánh kết quả thu được và mong chờ bằng phương thức Assert.Areequal và cho biết kết quả của test case là pass hay fail

Assert.AreEqual(mess, actMsg);

}

else

{

Assert.AreEqual(expectedValue, ds.TotalMonth(thang));

}

ds.Close();

}

}

}

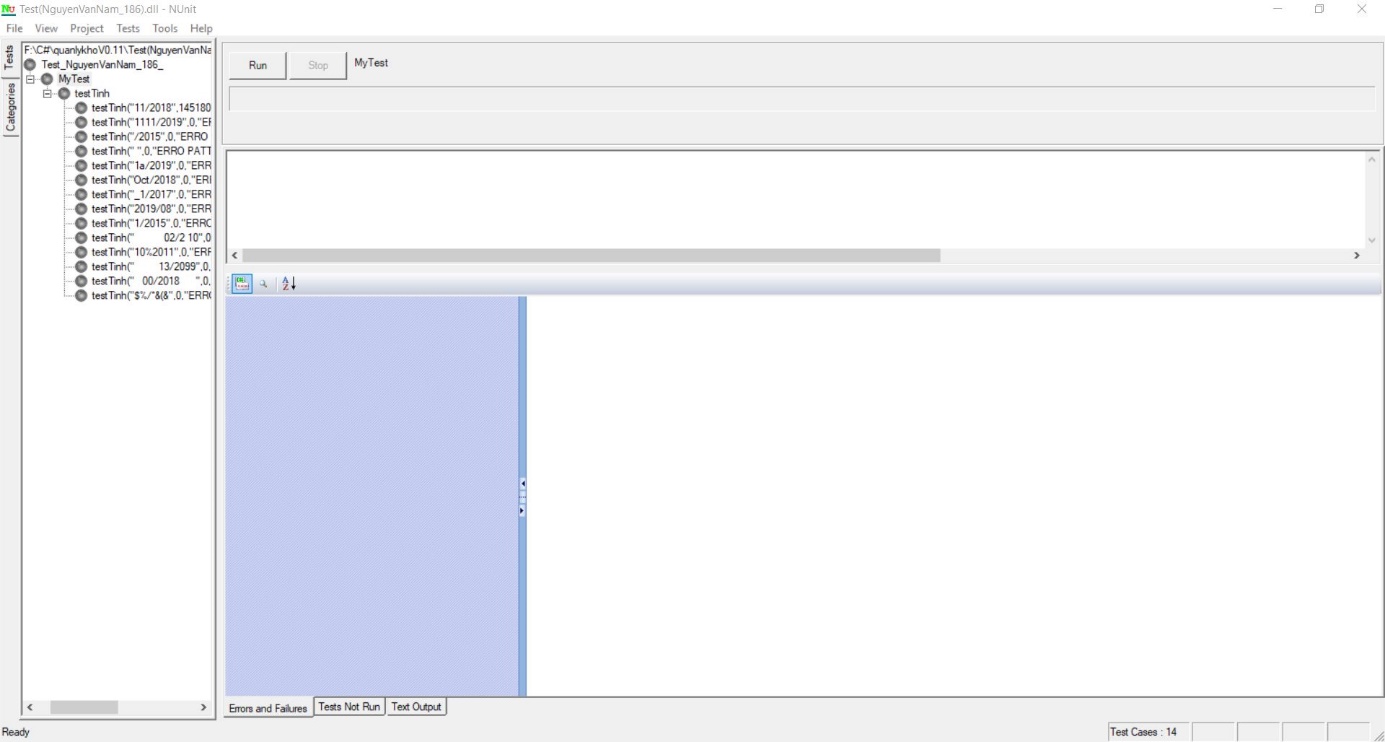
1. Run test:

B1: Chuột phải trên nội dung của class myTest.cs 🡪 kích vào nút Run test.

B2: Mở file NUnit 2.7.0 đã giải nén ban đầu rồi chọn nunit.exe.

B3: Trong giao diện chính của NUnit bạn chọn File 🡪 Open Project 🡪 Chọn đến class library Test (nguyenvannam186) đã tạo từ ban đầu 🡪 bin 🡪 myTest 🡪 OK.

B4: Nếu thành công thì giao diện NUnit sẽ hiển thị lên như sau:



B5: Nhấn Run trên giao diện và xem kết quả. Thanks.

#### Báo cáo kết quả kiểm thử

Báo cáo kết quả kiểm thử:

***Round 1:***

Đã test: 100%

Thành công: 83, 3%

Pending: 16, 7%

***Round 2:***

Đã test: 100%

Thành công: 83, 3%

Pending: 16, 7%

|  |  |  |
| --- | --- | --- |
|  | ROUND1 | ROUND2 |
| Total | 12 | 12 |
| OK | 10 | 10 |
| NG | 2 | 2 |
| NT | 0 | 0 |
| Not Yet Test | 0 | 0 |

Bảng 3.5 Kết quả test chức năng tính doanh thu tháng.

### Nguyễn Văn Nam - Tính doanh thu tháng (V0.11)

#### Phân tích thiết kế kểm thử

Ở phiên bản này thì chức năng “Tính doanh thu theo tháng” sẽ được cải tiến mã nguồn (thêm điều kiện) để đảm bảo rằng chức năng sẽ hoạt động tốt ở nhiều tình huống khác nhau.

Mã nguồn ban đầu:

if (isTruePattern(s) == false)

{

MessageBox.Show("ERRO PATTERN", "NOTE");

}

Mã nguồn sau khi chỉnh sửa:

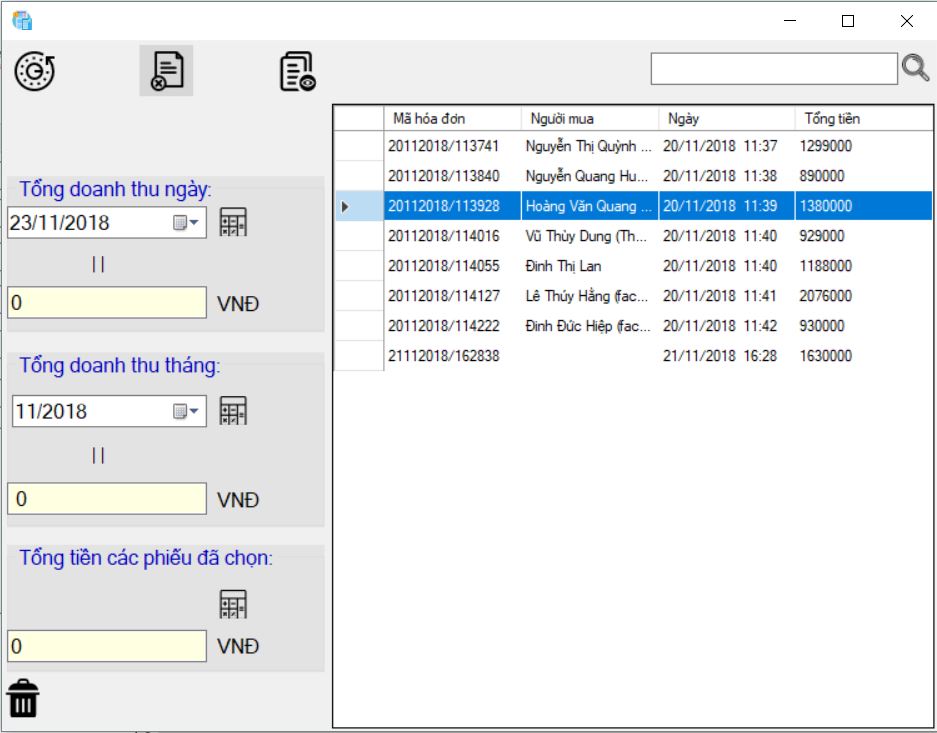
if (isTruePattern(s) == false || s.Length != 7)

{

MessageBox.Show("ERRO PATTERN", "NOTE");

}

* Giao diện chức năng :



*Hình 3.5 Giao diện tình doanh thu V0.11*

* Mã nguồn của chức năng

public Int32 TotalMonth(String s) Begin

{

s = s.Trim();

Int32 tong = 0; **1**

if (isTruePattern(s) == false || s.Length != 7) **2**

{

MessageBox.Show("ERRO PARTTERN"); **3**

}

else

**5**

**9**

**4**

{

for (Int32 i = 0; i < dgvDSHD.RowCount; i++)

{

String temp = dgvDSHD.Rows[i].Cells[2].Value.ToString(); **6**

String cpr = temp.Substring(3, 7);

if (s.Equals(cpr)) **7**

{

tong += Convert.ToInt32(dgvDSHD.Rows[i].Cells[3].Value.ToString()); **8**

}

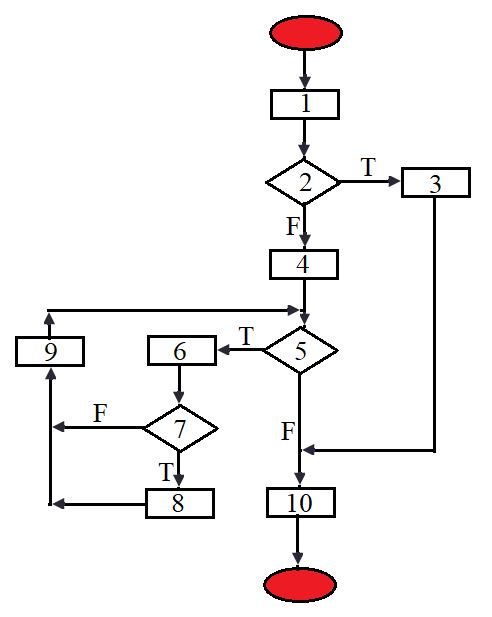
}

}

return tong; **10**

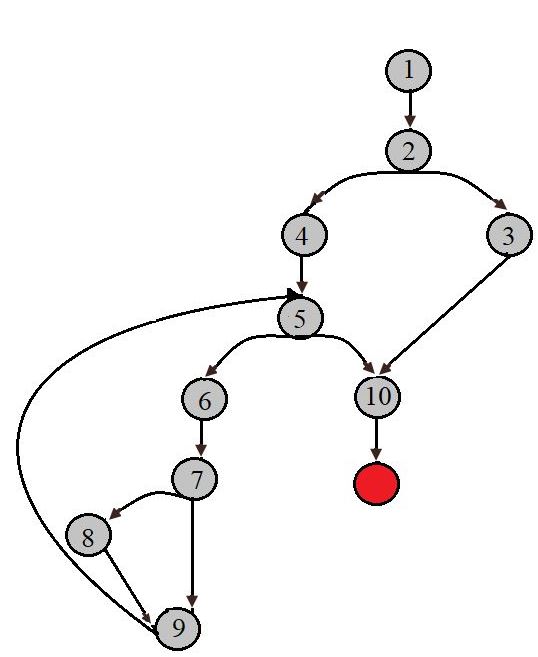
} End

* Lưu đồ giải thuật hoặc sơ đồ đồ thị dòng
* Lưu đồ giải thuật



*Hình 3.6 Lưu đồ thuật giải chức năng tính doanh thu tháng V0.11*

* Sơ đồ đồ thị dòng



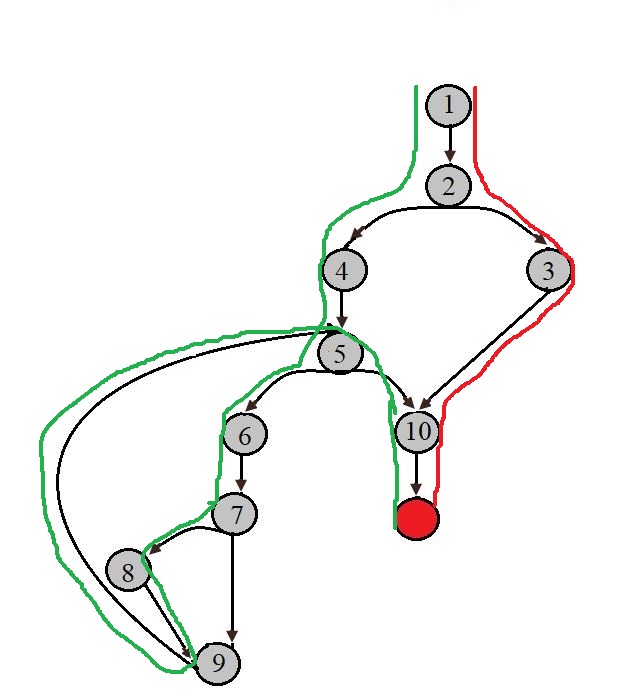
*Hình 3.7 Sơ đồ đồ thị dòng từ lưu đồ thuật giải ở trên*

* Độ phức tạp chu trình

C = 3 + 1 = 4

1. 1, 2, 3, 10.
2. 1, 2, 4, 5, 10.
3. 1, 2, 4, 5, 6, 7, 8, 9, 5, 10.
4. 1, 2, 4, 5, 6, 7, 9, 5, 10.

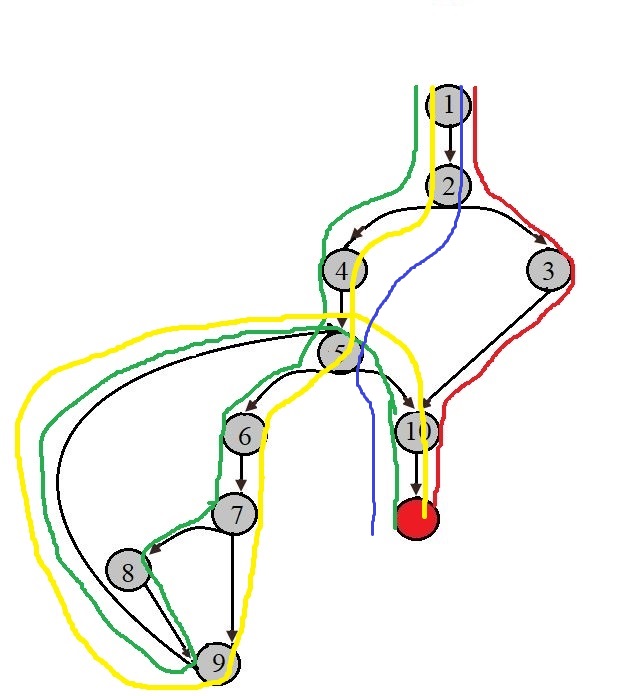
* Các cấp bao phủ kiểm thử
* Bao phủ cấp 1 : Các câu lệnh được thực thi ít nhất 1 lần. Với chức năng trên thì chỉ cần 2 test case để đạt được 100% bao phủ cấp 1. Chính là 2 đường thi hành chính đã thông báo ở trên.



*Hình 3.8 Sơ đồ thể hiện đường đi của bài toán trong bao phủ cấp 1*

* Bao phủ cấp 2 : Bao phủ nhánh

Với 2 test case xác định trong phần trước thì ta chỉ đạt được 1/2 = 50% độ bao phủ nhánh, ta phải thêm hai test case để độ bao phủ nhánh đạt 100%.



*Hình 3.9 Sơ đồ thể hiện đường đi của bài toán trong bao phủ cấp 2*

* Bao phủ cấp 3 : Bao phủ điều kiện

|  |  |  |
| --- | --- | --- |
| Predicate | True | False |
| isTruePattern(s) == false | TC1 :  totalMonth(’’1212/2018’’)  return 0 ; | TC2 :  totalMonth(‘’10/2018’’)  return Tong ; |
| s.Length != 7 | TC3 :  totalMonth(‘’10aaaaaaaa’’)  return 0 ; | TC4 :  totalMonth(‘’abc4567’’)  return 0 ; |
| i < dgvDSHD.RowCount | Continue ; | Return Tong ; |
| s.Equals(cpr) | Tong += Tong ;Continue; | Continue ; |

*Bảng 3.6 Bảng phân tích độ bao phủ cấp 3*

* Thiết kế test case :
  + Mô tả điều kiện :
* Bắt buộc phải nhập.
* Chuỗi nhập theo định dạng : MM/yyyy
* Phải đúng 7 ký tự.
  + Xây dựng test case :
* Áp dụng phương pháp phân tích giá trị biên, phân vùng tương đương.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Đầu vào | Vùng hợp lệ | Ký hiệu đánh dấu | Vùng không hợp lệ | Đánh dấu |
| Chuỗi tháng năm | Đúng định dạng 7 ký tự | H1 | >7 | K1 |
| <7 | K2 |
| Bỏ trống | K3 |
| Nhập sai định dạng | K4 |

*Bảng 3.7 Bảng phân vùng tương đương.*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| TCs | Input | Output | | Cover |
| Return | Message |
| 1 | 11/2018 | 14518000 | SUITABLE FORMAT | H1 |
| 2 | 1111/2019 | 0 | ERRO PATTERN | K1 |
| 3 | /2015 | 0 | ERRO PATTERN | K2 |
| 4 |  | 0 | ERRO PATTERN | K3 |
| 5 | 1a/2019 | 0 | ERRO PATTERN | K4 |
| 6 | Oct/2018 | 0 | ERRO PATTERN | K4 |
| 7 | \_1/2017 | 0 | ERRO PATTERN | K4 |
| 8 | 2019/08 | 0 | ERRO PATTERN | K4 |
| 9 | 1/2015 | 0 | ERRO PATTERN | K4 |
| 10 | 02/2 10 | 0 | ERRO PATTERN | K4 |
| 11 | 10%2011 | 0 | ERRO PATTERN | K4 |
| 12 | 13/2099 | 0 | ERRO PATTERN | K4 |
| 13 | 00/2018 | 0 | ERRO PATTERN | K4 |
| 14 | $%/\*&(& | 0 | ERRO PATTERN | K4 |

*Bảng 3.8 Ca kiểm thử.*

* TestcaseTinhDoanhThuThang.xlsx

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Tcs | Tên test case | Input | Output |  |
|  |  |  | Return | Messsage |
| Tc1 | Tính doanh thu tháng | 11/2018 | 14518000 | SUITABLE FORMAT |
| Tc2 | Tính doanh thu tháng | 1111/2019 | 0 | ERRO PATTERN |
| Tc3 | Tính doanh thu tháng | /2015 | 0 | ERRO PATTERN |
| Tc4 | Tính doanh thu tháng |  | 0 | ERRO PATTERN |
| Tc5 | Tính doanh thu tháng | 1a/2019 | 0 | ERRO PATTERN |
| Tc6 | Tính doanh thu tháng | Oct/2018 | 0 | ERRO PATTERN |
| Tc7 | Tính doanh thu tháng | \_1/2017 | 0 | ERRO PATTERN |
| Tc8 | Tính doanh thu tháng | 2019/08 | 0 | ERRO PATTERN |
| Tc9 | Tính doanh thu tháng | 1/2015 | 0 | ERRO PATTERN |
| Tc10 | Tính doanh thu tháng | 02/2 10 | 0 | ERRO PATTERN |
| Tc11 | Tính doanh thu tháng | 10%2011 | 0 | ERRO PATTERN |
| Tc12 | Tính doanh thu tháng | 13/2099 | 0 | ERRO PATTERN |
| Tc13 | Tính doanh thu tháng | 00/2018 | 0 | ERRO PATTERN |
| Tc14 | Tính doanh thu tháng | $%/\*&(& | 0 | ERRO PATTERN |

*Bảng 3.9 Datatest sau khi phân tích test case.*

#### Thực hiện kiểm thử

1. Cài đặt:

B1: Truy cập trang web NUnit và NUnit để tải bản NUnitForm mới nhất và bản NUnit 2.7.0 (Nhóm em chọn bản 2.7.0 vì bản này có hỗ trợ GUI giúp cho việc run nhiều test case một lúc thuận tiện hơn).

B2: Giải nén hai file vừa tải và cài đặt bản NUnitForm, còn bản NUnit 2.7.0 thì không cần. Cài đặt NUnitForm như cài đặt một phần mềm bình thường.

1. Cầu hình:

B1: Vào project, sau đó tạo một class library tên Test (nguyenvannam186) (Tên tùy ý). B2: Giờ ta chuột phải vào Reference trong class library (nguyenvannam186) 🡪 chọn Add Reference 🡪 chọn Brows 🡪 chọn browse… 🡪 Tìm đến nơi vừa cài đặt bản NUnitFrom (thường là: C:\Program Files (x86)\NUnitForms) 🡪 chọn file NUnitForms.dll, sau đó chúng ta tìm đến folder NUnit vừa giải nén và add file unit.framework.dll

1. Chuẩn bị test:

B1: Sau khi phân tích test case ở trên thì chúng ta cần tạo ra một file excel để chứa các dữ liệu để phục vụ test được thuận tiện và mang tính tự động cao hơn.

B2: Để việc test được tự động thì việc dữ liệu được tự động đưa vào test case và việc ghi lại kết quả của test case là vô cung cần thiết. Nên ta tạo một class đọc ghi file trong class library Test (nguyenvannam186) với tên DocGhiFileExcel.cs (tên tùy ý).

B3: Xây dựng mã nguồn của class DocGhiFileExcel vừa tạo. Dưới đây là mã của class DocGhiFileExcel kèm chú thích.

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Data.OleDb; //Thư viện hỗ trợ đọc ghi file excel

using System.Data;

using System.Windows.Forms;

using System.IO;

using System.Runtime.InteropServices; //Thư viện hỗ trợ đọc ghi file Excel, cái này cần add reference vào vì nó không có sẵn trong C#

using Excel = Microsoft.Office.Interop.Excel; // Đối tượng thay thế cho tên thư viện

using System.Collections.ObjectModel;

class DocGhiFileExcel

{

public static Excel.Application xlApp;

public static Excel.Workbook xlWorkbook; //Sử dụng Workbook cho file đuôi

.xlsx và HWorkbook đối với file đuôi .slx

public static Excel.\_Worksheet xlWorksheet; //Quản lý các sheet trong excel

public static Excel.Range xlRange; //Quản lý các thành phần trong 1 sheet

public static void open(String fileName)

{

//Tạo đối tượng COM và mở kết nối tới các đối tượng cần dùng để đọc dữ

Liệu

xlApp = new Excel.Application();

xlWorkbook = xlApp.Workbooks.Open(fileName);

xlWorksheet = xlWorkbook.Sheets[1];

xlRange = xlWorksheet.UsedRange;

}

public static void close()

{

//Đóng toàn bộ

GC.Collect();

GC.WaitForPendingFinalizers();

Marshal.ReleaseComObject(xlRange);

Marshal.ReleaseComObject(xlWorksheet);

//Đóng và thu hồi kết nối

xlWorkbook.Save();

xlWorkbook.Close();

Marshal.ReleaseComObject(xlWorkbook);

//Thoát và thu hồi kết nối

xlApp.Quit();

Marshal.ReleaseComObject(xlApp);

}

//Phương thức này được dùng để tạo ra một danh sách các mảng đối tượng một

Chiều, mỗi một mảng là một dòng trong bảng tính Excel.

public static Collection<Object[]> getExcelFile(String fileName)

{

open(fileName);

int rowCount = xlRange.Rows.Count;

int colCount = 5;

int j = 3; //Vì mình chỉ đọc dữ liệu từ cột số 3 đến cột số 5

Object[] obj;

Collection<Object[]> list = new Collection<object[]>();

for (int i = 3; i <= rowCount; i++)

{

// Đọc từng thành phần trong từng dòng của file Excel

String thang = xlRange.Cells[i, j++].Value2.ToString();

Int32 tien = Convert.ToInt32(xlRange.Cells[i, j++].Value2.ToString());

String mess = xlRange.Cells[i, j++].Value2.ToString();

//Tạo đối tượng lưu các thành phần vừa đọc được vào một mảng đối tượng chiều

obj = new Object[] {thang, tien, mess};

list.Add(obj); //Thêm đối tượng vừa tạo được vào danh sách để có thể

truyền vào TestCaseSource bên test.

if (j > colCount)

j = 3;

}

close();//Đóng kết nối file Excel

return list;

}

//Phương thức này sử dụng để ghi dữ liệu vào file Excel theo từng thành phần

public static void setExcelFile(int i, int j, String str, String fileName)

{

open(fileName);

xlRange.Cells[i, j].value2 = str;

close();

}

}

1. Chúng ta tạo một class tên myTest.cs (tên tùy ý), và mã bên trong class myTest.cs như sau:

using GUI;

using System;

using NUnit.Extensions.Forms; //Sử dụng để viết test, nhất thiết phải có

using NUnit.Framework; //Cần có để sử dụng các annotation trong NUnit

using System.Windows.Forms;

using System.Runtime.InteropServices;

using Excel = Microsoft.Office.Interop.Excel;

using System.Collections.Generic;

using System.Collections.ObjectModel;

namespace Test\_NguyenVanNam\_186\_

{

[TestFixture] //Annotation của NUnit, báo cho biết ở đây chứa các test case

public class MyTest

{

public int row = 3;

public static Collection<Object[]> data()

{

//Lấy về danh sách các mảng đối tượng một chiều để đọc thành data test

return DocGhiFileExcel.getExcelFile(@"F:\C#\quanlykhoV0.11\DataTest\TestcaseTinhDoanhThuThang.xlsx");

}

//Đọc dữ liệu từ danh sách vừa thu được thành đầu vào cho các test case

[TestCaseSource("data"), Test]

public void testTinh(String thang, Int32 expectedValue, String mess)

{

//Tạo đối tượng của form để có thể quản lý các thành phần của form

danhsachhoadon ds = new danhsachhoadon();

ds.Show();//Khời động form

//Dùng để lấy ra thông tin từ MessageBox để so sánh

String actMsg = "";

ModalFormTester msgBox = new ModalFormTester();

msgBox.ExpectModal("NOTE", delegate

{

MessageBoxTester msg = new MessageBoxTester("NOTE");

actMsg = msg.Text;

msg.ClickOk();//Phát hiện ra MessageBox cần chứa thông tin cần

tìm rối click OK

});

if (expectedValue == ds.TotalMonth(thang) && mess.Equals(actMsg))

{

//So sánh kết quả mong đợi và thực rồi ghi vào file Excel

DocGhiFileExcel.setExcelFile(row++, 6, "Pass", @"F:\C#\quanlykhoV0.11\DataTest\TestcaseTinhDoanhThuThang.xlsx");

}

else

{

DocGhiFileExcel.setExcelFile(row++, 6, "Fail", @"F:\C#\quanlykhoV0.11\DataTest\TestcaseTinhDoanhThuThang.xlsx");

}

//đảm bảo việc so sánh cả hai kết quả để đảm bảo không bỏ sót điều kiện sánh của test case

if(expectedValue == ds.TotalMonth(thang))

{

//So sánh kết quả thu được và mong chờ bằng phương thức Assert.Areequal và cho biết kết quả của test case là pass hay fail

Assert.AreEqual(mess, actMsg);

}

else

{

Assert.AreEqual(expectedValue, ds.TotalMonth(thang));

}

ds.Close();

}

}

}

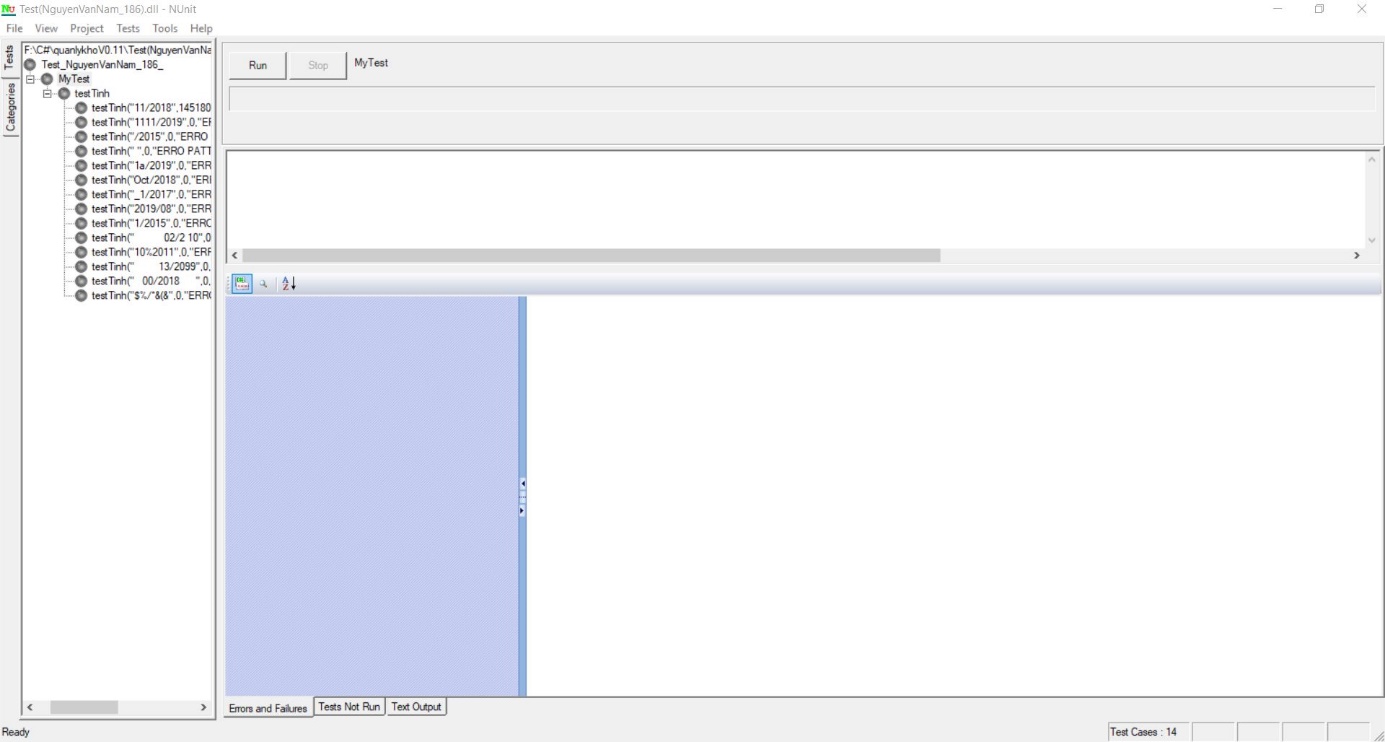
1. Run test:

B1: Chuột phải trên nội dung của class myTest.cs 🡪 kích vào nút Run test.

B2: Mở file NUnit 2.7.0 đã giải nén ban đầu rồi chọn nunit.exe.

B3: Trong giao diện chính của NUnit bạn chọn File 🡪 Open Project 🡪 Chọn đến class library Test (nguyenvannam186) đã tạo từ ban đầu 🡪 bin 🡪 myTest 🡪 OK.

B4: Nếu thành công thì giao diện NUnit sẽ hiển thị lên như sau:



B5: Nhấn Run trên giao diện và xem kết quả. Thanks.

#### Báo cáo kết quả kiểm thử

Báo cáo kết quả kiểm thử:

***Round 1:***

Đã test: 100%

Thành công: 92, 8%

Pending: 7, 2 %

***Round 2:***

Đã test: 100%

Thành công: 92, 8%

Pending: 7, 2%

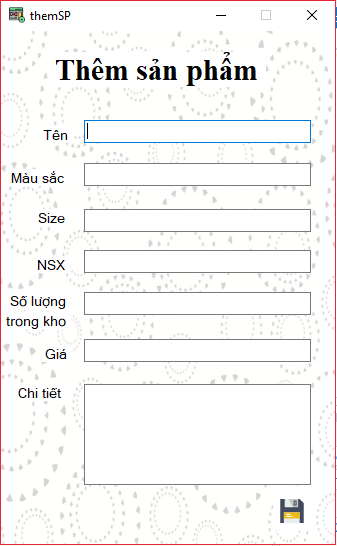
|  |  |  |
| --- | --- | --- |
|  | ROUND1 | ROUND2 |
| Total | 14 | 14 |
| OK | 13 | 13 |
| NG | 1 | 1 |
| NT | 0 | 0 |
| Not Yet Test | 0 | 0 |

Bảng 3.9 Kết quả test chức năng tính doanh thu tháng.

### Nguyễn Văn Hào – Thêm sản phẩm (V0.1)

#### Phân tích thiết kế kiểm thử

* Giao diện



Hình *3.2.3.1a: Form Thêm sản phẩm*

* Mã nguồn chức năng

if (txtten.Text.Length < 1 || txtten.Text.Length > 50)

1

{

2

MessageBox.Show("Add fail", "NOTE",MessageBoxButtons.OK,MessageBoxIcon.Information);

txtten.Focus();

}

else

3

if (txtmausac.Text.Length < 1 || txtmausac.Text.Length > 20)

{

4

MessageBox.Show("Add fail", "NOTE",MessageBoxButtons.OK,MessageBoxIcon.Information);

txtmausac.Focus();

}

else

5

if (txtsize.Text.Length < 1 || txtsize.Text.Length > 3)

{

MessageBox.Show("Add fail", "NOTE",MessageBoxButtons.OK,MessageBoxIcon.Information);

6

txtsize.Focus();

}

else

7

if (txtnsx.Text.Length < 1 || txtnsx.Text.Length > 30)

{

MessageBox.Show("Add fail", "NOTE",MessageBoxButtons.OK,MessageBoxIcon.Information);

8

txtnsx.Focus();

}

else

9

if (Int32.Parse(txtsoluongtrongkho.Text) < 0 ||

Int32.Parse(txtsoluongtrongkho.Text) > 10000)

{

10

MessageBox.Show("Add fail", "NOTE",MessageBoxButtons.OK,MessageBoxIcon.Information);

txtsoluongtrongkho.Focus();

}

else

11

if (Double.Parse(txtgia.Text) > 2000000)

{

12

MessageBox.Show("Add fail", "NOTE",MessageBoxButtons.OK,MessageBoxIcon.Information);

txtgia.Focus();

}

else

13

if (txtchitiet.Text.Length < 1 || txtchitiet.Text.Length > 200)

{

14

MessageBox.Show("Add fail", "NOTE",MessageBoxButtons.OK,MessageBoxIcon.Information);

txtchitiet.Focus();

}

else

{

Int32 soluong = Convert.ToInt32(txtsoluongtrongkho.Text);

Int32 giaSP = Convert.ToInt32(txtgia.Text);

sp\_DTO sp = new sp\_DTO(txtten.Text, txtmausac.Text, txtsize.Text, txtnsx.Text, soluong, giaSP, txtchitiet.Text);

15

sp\_BUS.themSP(sp);

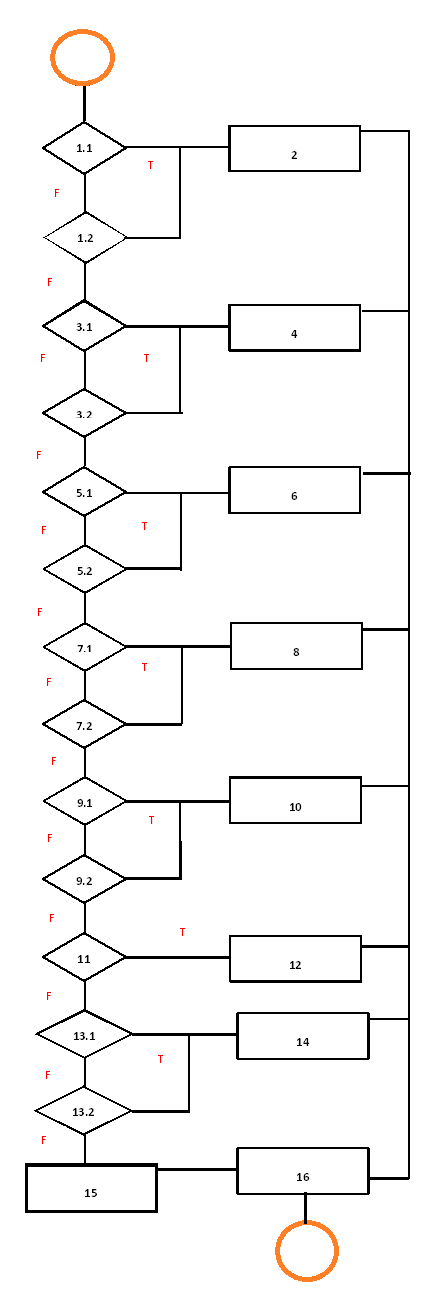
MessageBox.Show("Add successful", "NOTE",MessageBoxButtons.OK,MessageBoxIcon.Information); //Form1.frm.Form1\_Load(sender, e);

}

16

Return;

* Lưu đồ



*Hình 3.2.3.1b : Lưu đồ*

Độ phức tạp của chu trình : C = P + 1 = 14 ( P : số nút quyết định)

C = N – P + 2 = 14 (N : Số nhánh, P : Số nút) C = 36 – 24 + 2 = 14

1. 1.1-1.2-3.1-3.2-5.1-5.2-7.1-7.2-9.1-9.2-11-13.1-13.2-15-16
2. 1.1-2-16
3. 1.1-1.2-2-16
4. 1.1-1.2-3.1-4-16
5. 1.1-1.2-3.1-3.2-4-16
6. 1.1-1.2-3.1-3.2-5.1-6-16
7. 1.1-1.2-3.1-3.2-5.1-5.2-6-16
8. 1.1-1.2-3.1-3.2-5.1-5.2-7.1-8-16
9. 1.1-1.2-3.1-3.2-5.1-5.2-7.1-7.2-8-16
10. 1.1-1.2-3.1-3.2-5.1-5.2-7.1-7.2-9.1-10-16
11. 1.1-1.2-3.1-3.2-5.1-5.2-7.1-7.2-9.1-9.2-10-16
12. 1.1-1.2-3.1-3.2-5.1-5.2-7.1-7.2-9.1-9.2-11-12-16
13. 1.1-1.2-3.1-3.2-5.1-5.2-7.1-7.2-9.1-9.2-11-13.1-14-16
14. 1.1-1.2-3.1-3.2-5.1-5.2-7.1-7.2-9.1-9.2-11-13.1-13.2-14-16

* Độ bao phủ cấp 4
* **Thiết kế test case**
* Mô tả điều kiện

-Tên sản phẩm: Số lượng ký tự [1-50]

-Màu sắc: Số lượng ký tự [1-20]

-Size: Số lượng ký tự [1-3]

-NSX: Số lượng ký tự[1-30]

-Số lượng trong kho: số nguyên, [0-10000]

-Giá: Số thực, [0-2000000]

- Chi tiết của sản phẩm: số lượng ký tự [1-200]

* **Xây dựng test case**

Áp dụng phương pháp phân tích giá trị biên và phân vùng tương đương

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Đầu vào | Vùng hợp lệ | Ký hiệu đánh dấu | Vùng không hợp lệ | Đánh dấu |
| Tên sản phẩm | [1-50] | H1 | Trống | K1 |
| >50 | K2 |
| Màu sắc | [1-20] | H2 | Trống | K3 |
| > 20 | K4 |
| Size | [1-3] | H3 | Trống | K5 |
| >3 | K6 |
| Nhà sản xuất | [1-30] | H4 | Trống | K7 |
| >30 | K8 |
| Số lượng trong kho | [0-10000] | H5 | Ký tự đặc biệt | K9 |
| Số thực | K10 |
| Trống | K11 |
| <0 | K12 |
| >10000 | K13 |
| Giá | [0-2000000] | H6 | Ký tự đặc biệt | K14 |
| <0 | K15 |
| Trống | K16 |
| >2000000 | K17 |
| Chi tiết sản phẩm | [1-200] | H7 | Trống | K18 |
| >200 | K19 |

*Bảng 3.2.3.1a: phân vùng tương đương cho chức năng Thêm sản phẩm*

* **Thiết kế ca kiểm thử**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| TCs | INPUT | | | | | | | OUTPUT | Cover |
| Tên sản phẩm | Màu sắc | Size | Nhà sản xuất | Số lượng trong kho | Giá | Chi tiết sản phẩm |
| 1 |  | AA | M | AA | 10 | 150000 | None | Add fail | K1,H2,H3,H4,H5,H6,H7 |
| 2 | A | AA | M | AA | 10 | 150000 | None | Add successful | H1,H2,H3,H4,H5,H6,H7 |
| 3 | AA | AA | M | AA | 10 | 150000 | None | Add successful | H1,H2,H3,H4,H5,H6,H7 |
| 4 | (49 ký tự) | AA | M | AA | 10 | 150000 | None | Add successful | H1,H2,H3,H4,H5,H6,H7 |
| 5 | (50Ký tự) | AA | M | AA | 10 | 150000 | None | Add successful | H1,H2,H3,H4,H5,H6,H7 |
| 6 | (51 ký tự) | AA | M | AA | 10 | 150000 | None | Add fail | K2,H2,H3,H4,H5,H6,H7 |
| 7 | A |  | M | AA | 10 | 150000 | None | Add fail | H1,K3,H3,H4,H5,H6,H7 |
| 8 | A | A | M | AA | 10 | 150000 | None | Add successful | H1,H2,H3,H4,H5,H6,H7 |
| 9 | A | AA | M | AA | 10 | 150000 | None | Add successful | H1,H2,H3,H4,H5,H6,H7 |
| 10 | A | (19 ký tự) | M | AA | 10 | 150000 | None | Add successful | H1,H2,H3,H4,H5,H6,H7 |
| 11 | A | (20 ký tự) | M | AA | 10 | 150000 | None | Add successful | H1,H2,H3,H4,H5,H6,H7 |
| 12 | A | (21 ký tự) | M | AA | 10 | 150000 | None | Add fail | H1,K4,H3,H4,H5,H6,H8 |
| 13 | A | AA |  | AA | 10 | 150000 | None | Add fail | H1,H2,K5,H4,H5,H6,H7 |
| 14 | A | AA | M | AA | 10 | 150000 | None | Add successful | H1,H2,H3,H4,H5,H6,H7 |
| 15 | A | AA | MM | AA | 10 | 150000 | None | Add successful | H1,H2,H3,H4,H5,H6,H7 |
| 16 | A | AA | MMM | AA | 10 | 150000 | None | Add successful | H1,H2,H3,H4,H5,H6,H7 |
| 17 | A | AA | MMMM | AA | 10 | 150000 | None | Add fail | H1,H2,K6,H4,H5,H6,H7 |
| 18 | A | AA | M |  | 10 | 150000 | None | Add fail | H1,H2,H3,K7,H5,H6,H7 |
| 19 | A | AA | M | A | 10 | 150000 | None | Add successful | H1,H2,H3,H4,H5,H6,H7 |
| 20 | A | AA | M | AA | 10 | 150000 | None | Add successful | H1,H2,H3,H4,H5,H6,H7 |
| 21 | A | AA | M | (29 ký tự) | 10 | 150000 | None | Add successful | H1,H2,H3,H4,H5,H6,H7 |
| 22 | A | AA | M | (30 ký tự) | 10 | 150000 | None | Add successful | H1,H2,H3,H4,H5,H6,H7 |
| 23 | A | AA | M | (31 ký tự) | 10 | 150000 | None | Add fail | H1,H2,H3,K8,H5,H6,H7 |
| 24 | A | AA | M | AA | -1 | 150000 | None | Add fail | H1,H2,H3,H4,K12,H6,H7 |
| 25 | A | AA | M | AA | 0 | 150000 | None | Add successful | H1,H2,H3,H4,H5,H6,H7 |
| 26 | A | AA | M | AA | 1 | 150000 | None | Add successful | H1,H2,H3,H4,H5,H6,H7 |
| 27 | A | AA | M | AA | 9999 | 150000 | None | Add successful | H1,H2,H3,H4,H5,H6,H7 |
| 28 | A | AA | M | AA | 10000 | 150000 | None | Add successful | H1,H2,H3,H4,H5,H6,H7 |
| 29 | A | AA | M | AA | 10001 | 150000 | None | Add fail | H1,H2,H3,H4,K13,H6,H7 |
| 30 | A | AA | M | AA | 1.5 | 150000 | None | Add fail | H1,H2,H3,H4,K10,H6,H7 |
| 31 | A | AA | M | AA | a | 150000 | None | Add fail | H1,H2,H3,H4,K9,H6,H7 |
| 32 | A | AA | M | AA |  | 150000 | None | Add fail | H1,H2,H3,H4,K11,H6,H7 |
| 33 | A | AA | M | AA | 10 | -1 | None | Add fail | H1,H2,H3,H4,H5,K15,H7 |
| 34 | A | AA | M | AA | 10 | 0 | None | Add successful | H1,H2,H3,H4,H5,H6,H7 |
| 35 | A | AA | M | AA | 10 | 1 | None | Add successful | H1,H2,H3,H4,H5,H6,H7 |
| 36 | A | AA | M | AA | 10 | 1999999 | None | Add successful | H1,H2,H3,H4,H5,H6,H7 |
| 37 | A | AA | M | AA | 10 | 2000000 | None | Add successful | H1,H2,H3,H4,H5,H6,H7 |
| 38 | A | AA | M | AA | 10 | 2000001 | None | Add fail | H1,H2,H3,H4,H5,K17,H7 |
| 39 | A | AA | M | AA | 10 | a | None | Add fail | H1,H2,H3,H4,H5,K14,H7 |
| 40 | A | AA | M | AA | 10 |  | None | Add fail | H1,H2,H3,H4,H5,K16,H7 |
| 41 | A | AA | M | AA | 10 | 150000 | A | Add successful | H1,H2,H3,H4,H5,H6,H7 |
| 42 | A | AA | M | AA | 10 | 150000 | AA | Add successful | H1,H2,H3,H4,H5,H6,H7 |
| 43 | A | AA | M | AA | 10 | 150000 | (199 ký tự) | Add successful | H1,H2,H3,H4,H5,H6,H7 |
| 44 | A | AA | M | AA | 10 | 150000 | (200 ký tự) | Add successful | H1,H2,H3,H4,H5,H6,H7 |
| 45 | A | AA | M | AA | 10 | 150000 | (201 ký tự) | Add fail | H1,H2,H3,H4,H5,H6,K19 |
| 46 | A | AA | M | AA | 11 | 150000 |  | Add fail | H1,H2,H3,H4,H5,H6,K18 |

*Bảng 3.2.3.1b: Data Test chức năng Thêm sản phẩm*

#### Thực hiện kiểm thử

B1: Tạo Unit Test Project tên ‘TestThemSanPham’ trong Solution ‘Kho’.

B2: ADD References các framework:

* Nunit.framework
* Nunitforms
* GUI (Project kiểm thử)
* System.Windows.Forms

Các framework hỗ trợ đọc ghi file:

* LinqToExcel
* Microsoft.Office.Interop.Excel.dll
* Microsoft.Office.Interop.Excel
* System

B3:

* Tạo class objectData.cs có chức năng lưu trữ dữ liệu từ excel

using LinqToExcel.Attributes;

namespace TestThemSanPham\_NguyenVanHao\_253\_

{

class objectData

{

[ExcelColumn("Tên")]

public string ten { get; set; }

[ExcelColumn("Màu sắc")]

public string mau { get; set; }

[ExcelColumn("Size")]

public string size { get; set; }

[ExcelColumn("Nhà sản xuất")]

public string nsx { get; set; }

[ExcelColumn("Số lượng")]

public string soluong { get; set; }

[ExcelColumn("Giá")]

public string gia { get; set; }

[ExcelColumn("Mô tả")]

public string mota { get; set; }

[ExcelColumn("MessageBox")]

public string msg { get; set; }

}

}

* Tạo class DocFile.cs để đọc dữ liệu từ file excel

using LinqToExcel;//Thư viện

using System.Linq;//hỗ trợ ghi file excel

using System;

using System.Collections.ObjectModel;

namespace TestThemSanPham\_NguyenVanHao\_253\_

{

class DocFile

{

//Phương thức

public static Collection<object[]> Doc(String fileName)

{

//Lấy file excel

var excel = new ExcelQueryFactory(fileName);

//Lấy toàn bộ dữ liệu từ Sheet 1, đối tượng lưu trữ objectData

var data = from hv in excel.Worksheet<objectData>("Sheet1")

select hv;

//Danh sách lưu trữ liệu

Collection<object[]> rs = new Collection<object[]>();

//Duyệt dữ liệu từ data

foreach (objectData item in data)

{

object[] rsItem = new object[] { item.ten, item.mau, item.size, item.nsx, item.soluong, item.gia, item.mota, item.msg };

rs.Add(rsItem);

}

//Trả lại Data Test

return rs;

}

}

}

* Tạo class GhiFile.cs để ghi dư liệu vào file excel

using System;

using System.Runtime.InteropServices; //Thư viện hỗ trợ đọc ghi file Excel, cái này cần add reference vào vì nó không có sẵn trong C#

using Excel = Microsoft.Office.Interop.Excel; // Đối tượng thay thế cho tên thư viện

using System.Collections.ObjectModel;

class GhiFile

{

public static Excel.Application xlApp;

public static Excel.Workbook xlWorkbook; //Sử dụng Workbook cho file đuôi

//.xlsx và HWorkbook đối với file đuôi.slx

public static Excel.\_Worksheet xlWorksheet; //Quản lý các sheet trong excel

public static Excel.Range xlRange; //Quản lý các thành phần trong 1 sheet

public static void open(String fileName)

{

//Tạo đối tượng COM và mở kết nối tới các đối tượng cần dùng để đọc dữ

//Liệu

xlApp = new Excel.Application();

xlWorkbook = xlApp.Workbooks.Open(fileName);

xlWorksheet = xlWorkbook.Sheets[1];

xlRange = xlWorksheet.UsedRange;

}

public static void close()

{

//Đóng toàn bộ

GC.Collect();

GC.WaitForPendingFinalizers();

Marshal.ReleaseComObject(xlRange);

Marshal.ReleaseComObject(xlWorksheet);

//Đóng và thu hồi kết nối

xlWorkbook.Save();

xlWorkbook.Close();

Marshal.ReleaseComObject(xlWorkbook);

//Thoát và thu hồi kết nối

xlApp.Quit();

Marshal.ReleaseComObject(xlApp);

}

//Phương thức này sử dụng để ghi dữ liệu vào file Excel theo từng thành phần

public static void Ghi(int i, int j, String str, String fileName)

{

open(fileName);

xlRange.Cells[i, j].value2 = str;

close();

}

}

B5: Tạo class TestVersion0\_1.cs trong ‘TestThemSanPham’

using System;

using NUnit.Framework;

using NUnit.Extensions.Forms;

using System.Collections.ObjectModel;

using GUI;

namespace TestThemSanPham\_NguyenVanHao\_253\_

{

[TestFixture]

public class TestVersion0\_1

{

//Lấy data test

static Collection<object[]> DataTest()

{

return DocFile.Doc(@"D:\kiem thu\BAO\_CAO\_BTL\_KIEM\_THU\quanlykho\DataTest\TestCaseThemSanPham.xlsx");

}

//Dòng bắt đầu ghi

public int row1 = 2;

//Test case với bộ DataTest cùng các tham số tương ứng các cột trong Datatest

[TestCaseSource("DataTest")]

public void Round1(string ten, string mau, string size, string nsx, string soluong, string gia, string Mota, string msg)

{

//Sử dụng try-catch

//Khi chức năng xảy ra bất cứ lỗi nào -> Test case tương ứng với Fail

try

{

//Khởi tạo và hiển thị form

themSP frm = new themSP();

frm.Show();

//Lấy thông báo và đóng messageBox

string message = "";

ModalFormTester mdf = new ModalFormTester();

mdf.ExpectModal("NOTE", delegate

{

MessageBoxTester msgt = new MessageBoxTester("NOTE");

message = msgt.Text;

msgt.ClickOk();

});

//Khởi tạo nút bấm

ButtonTester btn = new ButtonTester("btnluu");

//Đưa dữ liệu từ DataTest vào các trường TextBox

frm.set\_txtten(ten);

frm.set\_txtmausac(mau);

frm.set\_txtsize(size);

frm.set\_txtnsx(nsx);

frm.set\_txtsoluongtrongkho(soluong);

frm.set\_txtgia(gia);

frm.set\_txtchitiet(Mota);

//Chạy nút Thêm

btn[0].Click();

//Kiểm tra kết quả mong đợi và kết quả thực tế- [Mong đợi: msg, Thực tế: message]

if (msg.Equals(message))

{

//Ghi kết quả vào file excel

GhiFile.Ghi(row1++, 9, "Pass", @"D:\kiem thu\BAO\_CAO\_BTL\_KIEM\_THU\quanlykho\DataTest\TestCaseThemSanPham.xlsx");

Assert.AreEqual(msg, message);

}

else

if (!msg.Equals(message))

{

//Ném một ngoại lệ khi Kết quả mong đợi và thực tế khác nhau

Assert.AreEqual(msg, message);

}

//Đóng form

frm.Close();

}

//Xử lý ngoại lệ

catch (Exception ex)

{

//Ngoại lệ khi kết quả mong đợi và thực tế khác nhau <=> Fail

//Hoặc khi form xảy ra lỗi <=> Fail

GhiFile.Ghi(row1++, 9, "Fail", @"D:\kiem thu\BAO\_CAO\_BTL\_KIEM\_THU\quanlykho\DataTest\TestCaseThemSanPham.xlsx");

//Trả lại kết quả của test case

Assert.Fail();

}

}

public int row = 2;

//Test case với bộ DataTest cùng các tham số tương ứng các cột trong Datatest

[TestCaseSource("DataTest")]

public void Round2(string ten, string mau, string size, string nsx, string soluong, string gia, string Mota, string msg)

{

//Sử dụng try-catch

//Khi chức năng xảy ra bất cứ lỗi nào -> Test case tương ứng với Fail

try

{

//Khởi tạo và hiển thị form

themSP frm = new themSP();

frm.Show();

//Lấy thông báo và đóng messageBox

string message = "";

ModalFormTester mdf = new ModalFormTester();

mdf.ExpectModal("NOTE", delegate

{

MessageBoxTester msgt = new MessageBoxTester("NOTE");

message = msgt.Text;

msgt.ClickOk();

});

//Khởi tạo nút bấm

ButtonTester btn = new ButtonTester("btnluu");

//Đưa dữ liệu từ DataTest vào các trường TextBox

frm.set\_txtten(ten);

frm.set\_txtmausac(mau);

frm.set\_txtsize(size);

frm.set\_txtnsx(nsx);

frm.set\_txtsoluongtrongkho(soluong);

frm.set\_txtgia(gia);

frm.set\_txtchitiet(Mota);

//Chạy nút Thêm

btn[0].Click();

//Kiểm tra kết quả mong đợi và kết quả thực tế- [Mong đợi: msg, Thực tế: message]

if (msg.Equals(message))

{

//Ghi kết quả vào file excel

GhiFile.Ghi(row++, 10, "Pass", @"D:\kiem thu\BAO\_CAO\_BTL\_KIEM\_THU\quanlykho\DataTest\TestCaseThemSanPham.xlsx");

Assert.AreEqual(msg, message);

}

else

if (!msg.Equals(message))

{

//Ném một ngoại lệ khi Kết quả mong đợi và thực tế khác nhau

Assert.AreEqual(msg, message);

}

//Đóng form

frm.Close();

}

//Xử lý ngoại lệ

catch (Exception ex)

{

//Ngoại lệ khi kết quả mong đợi và thực tế khác nhau <=> Fail

//Hoặc khi form xảy ra lỗi <=> Fail

GhiFile.Ghi(row++, 10, "Fail", @"D:\kiem thu\BAO\_CAO\_BTL\_KIEM\_THU\quanlykho\DataTest\TestCaseThemSanPham.xlsx");

//Trả lại kết quả của test case

Assert.Fail();

}

}

//Hiển thị kết quả trong excel

[Test]

public void zResult()

{

System.Diagnostics.Process.Start(@"D:\kiem thu\BAO\_CAO\_BTL\_KIEM\_THU\quanlykho\DataTest\TestCaseThemSanPham.xlsx");

Assert.Pass();

}

}

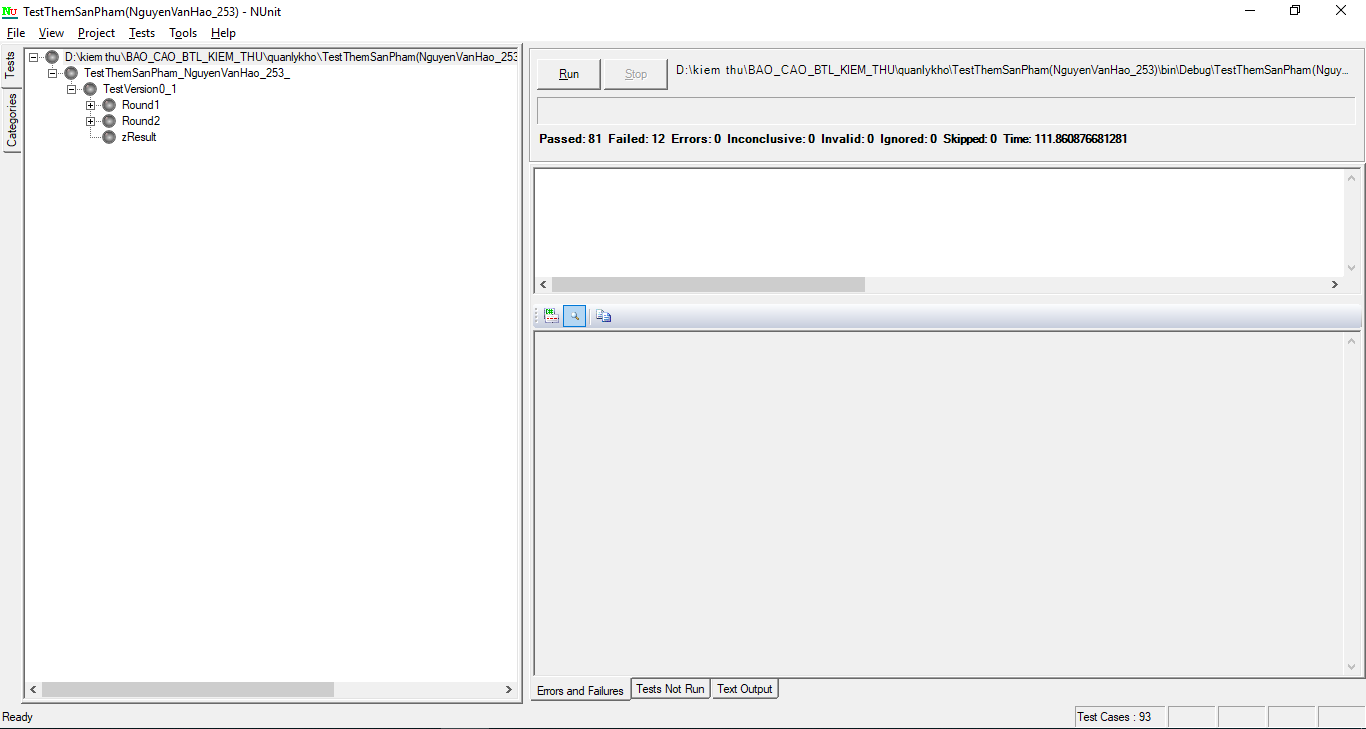
}

B6: Chuột phải chọn Run Test

B7: Mở nunit-x86.exe

* Chọn open project
* Chọn project ‘Kho’
* Chọn thư mục TestThemSanPham(NguyenVanHao\_253)
* Chọn thư mục bin
* Chọn thư mục debug
* Chọn TestThemSanPham(NguyenVanHao\_253).dll và Open

B8: bấm Run



*Hình 3.2.3.2: Công cụ Nunit*

#### Báo cáo kết quả kiểm thử

Báo cáo kết quả kiểm thử:

***Round 1:***

Đã test: 100%

Thành công: 87%

Pending: 13%

***Round 2:***

Đã test: 100%

Thành công: 87%

Pending: 13%

|  |  |  |
| --- | --- | --- |
|  | ROUND1 | ROUND2 |
| Total | 46 | 46 |
| OK | 40 | 40 |
| NG | 6 | 6 |
| NT | 0 | 0 |
| Not Yet Test | 0 | 0 |

*Bảng 3.2.3.3: Kết quả*

### Nguyên Văn Hào – Thêm sản phẩm(V0.11)

#### Phân tích thiết kế kiểm thử

Phiên bản V0.11 là phiên bản fix 1 số lỗi ở phiên bản V0.1 nhằm tăng độ tin cậy của phần mềm (Tăng độ % pass).

* Mã nguồn gây lỗi

//Fail, chưa kiểm tra số nguyên, ký tự đặc biệt, rỗng.

else if (Int32.Parse(txtsoluongtrongkho.Text) < 0 || Int32.Parse(txtsoluongtrongkho.Text) > 10000)

{

MessageBox.Show("Add fail", "NOTE", MessageBoxButtons.OK, MessageBoxIcon.Information);

txtsoluongtrongkho.Focus();

}

//Fail, chưa kiểm tra rỗng, ký tự đặc biệt, giá < 1

else

if (Double.Parse(txtgia.Text) > 2000000)

{

MessageBox.Show("Add fail", "NOTE", MessageBoxButtons.OK, MessageBoxIcon.Information);

txtgia.Focus();

}

* Chỉnh sửa lại mã nguồn

//Chưa kiểm tra được lỗi ký tự đặc biệt

else if (txtsoluongtrongkho.Text == null || Double.Parse(txtsoluongtrongkho.Text) < 0 || Double.Parse(txtsoluongtrongkho.Text) > 10000 ||

(Double.Parse(txtsoluongtrongkho.Text) - (int)Double.Parse(txtsoluongtrongkho.Text)) != 0)

{

MessageBox.Show("Add fail", "NOTE", MessageBoxButtons.OK, MessageBoxIcon.Information);

txtsoluongtrongkho.Focus();

}

//Chưa kiểm tra được lỗi ký tự đặc biệt

else

if (txtgia.Text.Length == 0|| Double.Parse(txtgia.Text) > 2000000 || Double.Parse(txtgia.Text) < 0

)

{

MessageBox.Show("Add fail", "NOTE", MessageBoxButtons.OK, MessageBoxIcon.Information);

txtgia.Focus();

}

if (txtten.Text.Length < 1 || txtten.Text.Length > 50)

1

{

2

MessageBox.Show("Add fail", "NOTE",MessageBoxButtons.OK,MessageBoxIcon.Information);

txtten.Focus();

}

else

3

if (txtmausac.Text.Length < 1 || txtmausac.Text.Length > 20)

{

4

MessageBox.Show("Add fail", "NOTE",MessageBoxButtons.OK,MessageBoxIcon.Information);

txtmausac.Focus();

}

else

5

if (txtsize.Text.Length < 1 || txtsize.Text.Length > 3)

{

MessageBox.Show("Add fail", "NOTE",MessageBoxButtons.OK,MessageBoxIcon.Information);

6

txtsize.Focus();

}

else

7

if (txtnsx.Text.Length < 1 || txtnsx.Text.Length > 30)

{

MessageBox.Show("Add fail", "NOTE",MessageBoxButtons.OK,MessageBoxIcon.Information);

8

txtnsx.Focus();

}

else

9

if (txtsoluongtrongkho.Text == null || Double.Parse(txtsoluongtrongkho.Text) < 0 || Double.Parse(txtsoluongtrongkho.Text) > 10000 ||

(Double.Parse(txtsoluongtrongkho.Text) - (int)Double.Parse(txtsoluongtrongkho.Text)) != 0)

{

10

MessageBox.Show("Add fail", "NOTE",MessageBoxButtons.OK,MessageBoxIcon.Information);

txtsoluongtrongkho.Focus();

}

else

if (txtgia.Text.Length == 0 || Double.Parse(txtgia.Text) > 2000000 || Double.Parse(txtgia.Text) < 0)

11

{

12

MessageBox.Show("Add fail", "NOTE",MessageBoxButtons.OK,MessageBoxIcon.Information);

txtgia.Focus();

}

else

13

if (txtchitiet.Text.Length < 1 || txtchitiet.Text.Length > 200)

{

14

MessageBox.Show("Add fail", "NOTE",MessageBoxButtons.OK,MessageBoxIcon.Information);

txtchitiet.Focus();

}

else

{

Int32 soluong = Convert.ToInt32(txtsoluongtrongkho.Text);

Int32 giaSP = Convert.ToInt32(txtgia.Text);

sp\_DTO sp = new sp\_DTO(txtten.Text, txtmausac.Text, txtsize.Text, txtnsx.Text, soluong, giaSP, txtchitiet.Text);

15

sp\_BUS.themSP(sp);

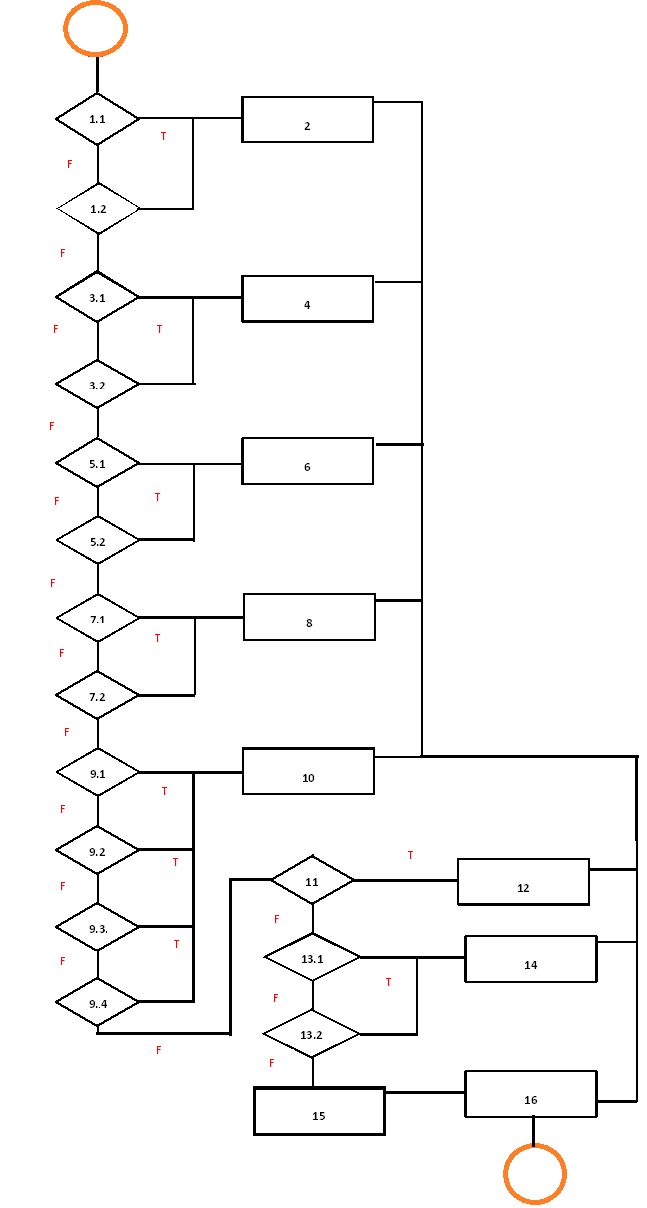
MessageBox.Show("Add successful", "NOTE",MessageBoxButtons.OK,MessageBoxIcon.Information); //Form1.frm.Form1\_Load(sender, e);

}

16

Return;

* Lưu đồ



*Hình 3.2.4.1: Lưu đồ*

Độ phức tạp của chu trình : C = P + 1 = 16 ( P : số nút quyết định) 15

C = N – P + 2 = 16 (N : Số nhánh, P : Số nút) 40-26+2

* 1. 1.1-1.2-3.1-3.2-5.1-5.2-7.1-7.2-9.1-9.2-11-13.1-13.2-15-16
  2. 1.1-2-16
  3. 1.1-1.2-2-16
  4. 1.1-1.2-3.1-4-16
  5. 1.1-1.2-3.1-3.2-4-16
  6. 1.1-1.2-3.1-3.2-5.1-6-16
  7. 1.1-1.2-3.1-3.2-5.1-5.2-6-16
  8. 1.1-1.2-3.1-3.2-5.1-5.2-7.1-8-16
  9. 1.1-1.2-3.1-3.2-5.1-5.2-7.1-7.2-8-16
  10. 1.1-1.2-3.1-3.2-5.1-5.2-7.1-7.2-9.1-10-16
  11. 1.1-1.2-3.1-3.2-5.1-5.2-7.1-7.2-9.1-9.2-10-16
  12. 1.1-1.2-3.1-3.2-5.1-5.2-7.1-7.2-9.1-9.2-9.3-10-16
  13. 1.1-1.2-3.1-3.2-5.1-5.2-7.1-7.2-9.1-9.2-9.3-9.4-10-16
  14. 1.1-1.2-3.1-3.2-5.1-5.2-7.1-7.2-9.1-9.2-9.3-9.4 -11-12-16
  15. 1.1-1.2-3.1-3.2-5.1-5.2-7.1-7.2-9.1-9.2-9.3-9.4-11-13.1-14-16
  16. 1.1-1.2-3.1-3.2-5.1-5.2-7.1-7.2-9.1-9.2-9.3-9.4-11-13.1-13.2-14-16
* Độ bao phủ cấp 4
* **Thiết kế test case**
* Mô tả điều kiện

-Tên sản phẩm: Số lượng ký tự [1-50]

-Màu sắc: Số lượng ký tự [1-20]

-Size: Số lượng ký tự [1-3]

-NSX: Số lượng ký tự[1-30]

-Số lượng trong kho: số nguyên, [0-10000]

-Giá: Số thực, [0-2000000]

- Chi tiết của sản phẩm: số lượng ký tự [1-200]

* **Xây dựng test case**

Áp dụng phương pháp phân tích giá trị biên và phân vùng tương đương

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Đầu vào | Vùng hợp lệ | Ký hiệu đánh dấu | Vùng không hợp lệ | Đánh dấu |
| Tên sản phẩm | [1-50] | H1 | Trống | K1 |
| >50 | K2 |
| Màu sắc | [1-20] | H2 | Trống | K3 |
| > 20 | K4 |
| Size | [1-3] | H3 | Trống | K5 |
| >3 | K6 |
| Nhà sản xuất | [1-30] | H4 | Trống | K7 |
| >30 | K8 |
| Số lượng trong kho | [0-10000] | H5 | Ký tự đặc biệt | K9 |
| Số thực | K10 |
| Trống | K11 |
| <0 | K12 |
| >10000 | K13 |
| Giá | [0-2000000] | H6 | Ký tự đặc biệt | K14 |
| <0 | K15 |
| Trống | K16 |
| >2000000 | K17 |
| Chi tiết sản phẩm | [1-200] | H7 | Trống | K18 |
| >200 | K19 |

*Bảng 3.2.4.1a: phân vùng tương đương cho chức năng Thêm sản phẩm*

* **Thiết kế ca kiểm thử**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| TCs | INPUT | | | | | | | OUTPUT | Cover |
| Tên sản phẩm | Màu sắc | Size | Nhà sản xuất | Số lượng trong kho | Giá | Chi tiết sản phẩm |
| 1 |  | AA | M | AA | 10 | 150000 | None | Add fail | K1,H2,H3,H4,H5,H6,H7 |
| 2 | A | AA | M | AA | 10 | 150000 | None | Add successful | H1,H2,H3,H4,H5,H6,H7 |
| 3 | AA | AA | M | AA | 10 | 150000 | None | Add successful | H1,H2,H3,H4,H5,H6,H7 |
| 4 | (49 ký tự) | AA | M | AA | 10 | 150000 | None | Add successful | H1,H2,H3,H4,H5,H6,H7 |
| 5 | (50Ký tự) | AA | M | AA | 10 | 150000 | None | Add successful | H1,H2,H3,H4,H5,H6,H7 |
| 6 | (51 ký tự) | AA | M | AA | 10 | 150000 | None | Add fail | K2,H2,H3,H4,H5,H6,H7 |
| 7 | A |  | M | AA | 10 | 150000 | None | Add fail | H1,K3,H3,H4,H5,H6,H7 |
| 8 | A | A | M | AA | 10 | 150000 | None | Add successful | H1,H2,H3,H4,H5,H6,H7 |
| 9 | A | AA | M | AA | 10 | 150000 | None | Add successful | H1,H2,H3,H4,H5,H6,H7 |
| 10 | A | (19 ký tự) | M | AA | 10 | 150000 | None | Add successful | H1,H2,H3,H4,H5,H6,H7 |
| 11 | A | (20 ký tự) | M | AA | 10 | 150000 | None | Add successful | H1,H2,H3,H4,H5,H6,H7 |
| 12 | A | (21 ký tự) | M | AA | 10 | 150000 | None | Add fail | H1,K4,H3,H4,H5,H6,H8 |
| 13 | A | AA |  | AA | 10 | 150000 | None | Add fail | H1,H2,K5,H4,H5,H6,H7 |
| 14 | A | AA | M | AA | 10 | 150000 | None | Add successful | H1,H2,H3,H4,H5,H6,H7 |
| 15 | A | AA | MM | AA | 10 | 150000 | None | Add successful | H1,H2,H3,H4,H5,H6,H7 |
| 16 | A | AA | MMM | AA | 10 | 150000 | None | Add successful | H1,H2,H3,H4,H5,H6,H7 |
| 17 | A | AA | MMMM | AA | 10 | 150000 | None | Add fail | H1,H2,K6,H4,H5,H6,H7 |
| 18 | A | AA | M |  | 10 | 150000 | None | Add fail | H1,H2,H3,K7,H5,H6,H7 |
| 19 | A | AA | M | A | 10 | 150000 | None | Add successful | H1,H2,H3,H4,H5,H6,H7 |
| 20 | A | AA | M | AA | 10 | 150000 | None | Add successful | H1,H2,H3,H4,H5,H6,H7 |
| 21 | A | AA | M | (29 ký tự) | 10 | 150000 | None | Add successful | H1,H2,H3,H4,H5,H6,H7 |
| 22 | A | AA | M | (30 ký tự) | 10 | 150000 | None | Add successful | H1,H2,H3,H4,H5,H6,H7 |
| 23 | A | AA | M | (31 ký tự) | 10 | 150000 | None | Add fail | H1,H2,H3,K8,H5,H6,H7 |
| 24 | A | AA | M | AA | -1 | 150000 | None | Add fail | H1,H2,H3,H4,K12,H6,H7 |
| 25 | A | AA | M | AA | 0 | 150000 | None | Add successful | H1,H2,H3,H4,H5,H6,H7 |
| 26 | A | AA | M | AA | 1 | 150000 | None | Add successful | H1,H2,H3,H4,H5,H6,H7 |
| 27 | A | AA | M | AA | 9999 | 150000 | None | Add successful | H1,H2,H3,H4,H5,H6,H7 |
| 28 | A | AA | M | AA | 10000 | 150000 | None | Add successful | H1,H2,H3,H4,H5,H6,H7 |
| 29 | A | AA | M | AA | 10001 | 150000 | None | Add fail | H1,H2,H3,H4,K13,H6,H7 |
| 30 | A | AA | M | AA | 1.5 | 150000 | None | Add fail | H1,H2,H3,H4,K10,H6,H7 |
| 31 | A | AA | M | AA | a | 150000 | None | Add fail | H1,H2,H3,H4,K9,H6,H7 |
| 32 | A | AA | M | AA |  | 150000 | None | Add fail | H1,H2,H3,H4,K11,H6,H7 |
| 33 | A | AA | M | AA | 10 | -1 | None | Add fail | H1,H2,H3,H4,H5,K15,H7 |
| 34 | A | AA | M | AA | 10 | 0 | None | Add successful | H1,H2,H3,H4,H5,H6,H7 |
| 35 | A | AA | M | AA | 10 | 1 | None | Add successful | H1,H2,H3,H4,H5,H6,H7 |
| 36 | A | AA | M | AA | 10 | 1999999 | None | Add successful | H1,H2,H3,H4,H5,H6,H7 |
| 37 | A | AA | M | AA | 10 | 2000000 | None | Add successful | H1,H2,H3,H4,H5,H6,H7 |
| 38 | A | AA | M | AA | 10 | 2000001 | None | Add fail | H1,H2,H3,H4,H5,K17,H7 |
| 39 | A | AA | M | AA | 10 | a | None | Add fail | H1,H2,H3,H4,H5,K14,H7 |
| 40 | A | AA | M | AA | 10 |  | None | Add fail | H1,H2,H3,H4,H5,K16,H7 |
| 41 | A | AA | M | AA | 10 | 150000 | A | Add successful | H1,H2,H3,H4,H5,H6,H7 |
| 42 | A | AA | M | AA | 10 | 150000 | AA | Add successful | H1,H2,H3,H4,H5,H6,H7 |
| 43 | A | AA | M | AA | 10 | 150000 | (199 ký tự) | Add successful | H1,H2,H3,H4,H5,H6,H7 |
| 44 | A | AA | M | AA | 10 | 150000 | (200 ký tự) | Add successful | H1,H2,H3,H4,H5,H6,H7 |
| 45 | A | AA | M | AA | 10 | 150000 | (201 ký tự) | Add fail | H1,H2,H3,H4,H5,H6,K19 |
| 46 | A | AA | M | AA | 11 | 150000 |  | Add fail | H1,H2,H3,H4,H5,H6,K18 |

*Bảng 3.2.4.1b: Dữ liệu kiểm thử*

#### Thực hiện kiểm thử

B1: Tạo Unit Test Project tên ‘TestThemSanPham’ trong Solution ‘Kho’.

B2: ADD References các framework:

* Nunit.framework
* Nunitforms
* GUI (Project kiểm thử)
* System.Windows.Forms

Các framework hỗ trợ đọc ghi file:

* LinqToExcel
* Microsoft.Office.Interop.Excel.dll
* Microsoft.Office.Interop.Excel
* System

B3:

* Tạo class objectData.cs có chức năng lưu trữ dữ liệu từ excel

using LinqToExcel.Attributes;

namespace TestThemSanPham\_NguyenVanHao\_253\_

{

class objectData

{

[ExcelColumn("Tên")]

public string ten { get; set; }

[ExcelColumn("Màu sắc")]

public string mau { get; set; }

[ExcelColumn("Size")]

public string size { get; set; }

[ExcelColumn("Nhà sản xuất")]

public string nsx { get; set; }

[ExcelColumn("Số lượng")]

public string soluong { get; set; }

[ExcelColumn("Giá")]

public string gia { get; set; }

[ExcelColumn("Mô tả")]

public string mota { get; set; }

[ExcelColumn("MessageBox")]

public string msg { get; set; }

}

}

* Tạo class DocFile.cs để đọc dữ liệu từ file excel

using LinqToExcel;//Thư viện

using System.Linq;//hỗ trợ ghi file excel

using System;

using System.Collections.ObjectModel;

namespace TestThemSanPham\_NguyenVanHao\_253\_

{

class DocFile

{

//Phương thức

public static Collection<object[]> Doc(String fileName)

{

//Lấy file excel

var excel = new ExcelQueryFactory(fileName);

//Lấy toàn bộ dữ liệu từ Sheet 1, đối tượng lưu trữ objectData

var data = from hv in excel.Worksheet<objectData>("Sheet1")

select hv;

//Danh sách lưu trữ liệu

Collection<object[]> rs = new Collection<object[]>();

//Duyệt dữ liệu từ data

foreach (objectData item in data)

{

object[] rsItem = new object[] { item.ten, item.mau, item.size, item.nsx, item.soluong, item.gia, item.mota, item.msg };

rs.Add(rsItem);

}

//Trả lại Data Test

return rs;

}

}

}

* Tạo class GhiFile.cs để ghi dư liệu vào file excel

using System;

using System.Runtime.InteropServices; //Thư viện hỗ trợ đọc ghi file Excel, cái này cần add reference vào vì nó không có sẵn trong C#

using Excel = Microsoft.Office.Interop.Excel; // Đối tượng thay thế cho tên thư viện

using System.Collections.ObjectModel;

class GhiFile

{

public static Excel.Application xlApp;

public static Excel.Workbook xlWorkbook; //Sử dụng Workbook cho file đuôi

//.xlsx và HWorkbook đối với file đuôi.slx

public static Excel.\_Worksheet xlWorksheet; //Quản lý các sheet trong excel

public static Excel.Range xlRange; //Quản lý các thành phần trong 1 sheet

public static void open(String fileName)

{

//Tạo đối tượng COM và mở kết nối tới các đối tượng cần dùng để đọc dữ

//Liệu

xlApp = new Excel.Application();

xlWorkbook = xlApp.Workbooks.Open(fileName);

xlWorksheet = xlWorkbook.Sheets[1];

xlRange = xlWorksheet.UsedRange;

}

public static void close()

{

//Đóng toàn bộ

GC.Collect();

GC.WaitForPendingFinalizers();

Marshal.ReleaseComObject(xlRange);

Marshal.ReleaseComObject(xlWorksheet);

//Đóng và thu hồi kết nối

xlWorkbook.Save();

xlWorkbook.Close();

Marshal.ReleaseComObject(xlWorkbook);

//Thoát và thu hồi kết nối

xlApp.Quit();

Marshal.ReleaseComObject(xlApp);

}

//Phương thức này sử dụng để ghi dữ liệu vào file Excel theo từng thành phần

public static void Ghi(int i, int j, String str, String fileName)

{

open(fileName);

xlRange.Cells[i, j].value2 = str;

close();

}

}

B5: Tạo class TestVersion0\_1.cs trong ‘TestThemSanPham’

using System;

using NUnit.Framework;

using NUnit.Extensions.Forms;

using System.Collections.ObjectModel;

using GUI;

namespace TestThemSanPham\_NguyenVanHao\_253\_

{

[TestFixture]

public class TestVersion0\_1

{

//Lấy data test

static Collection<object[]> DataTest()

{

return DocFile.Doc(@"D:\kiem thu\BAO\_CAO\_BTL\_KIEM\_THU\quanlykho\DataTest\TestCaseThemSanPham1.xlsx");

}

//Dòng bắt đầu ghi

public int row1 = 2;

//Test case với bộ DataTest cùng các tham số tương ứng các cột trong Datatest

[TestCaseSource("DataTest")]

public void Round1(string ten, string mau, string size, string nsx, string soluong, string gia, string Mota, string msg)

{

//Sử dụng try-catch

//Khi chức năng xảy ra bất cứ lỗi nào -> Test case tương ứng với Fail

try

{

//Khởi tạo và hiển thị form

themSP frm = new themSP();

frm.Show();

//Lấy thông báo và đóng messageBox

string message = "";

ModalFormTester mdf = new ModalFormTester();

mdf.ExpectModal("NOTE", delegate

{

MessageBoxTester msgt = new MessageBoxTester("NOTE");

message = msgt.Text;

msgt.ClickOk();

});

//Khởi tạo nút bấm

ButtonTester btn = new ButtonTester("btnluu");

//Đưa dữ liệu từ DataTest vào các trường TextBox

frm.set\_txtten(ten);

frm.set\_txtmausac(mau);

frm.set\_txtsize(size);

frm.set\_txtnsx(nsx);

frm.set\_txtsoluongtrongkho(soluong);

frm.set\_txtgia(gia);

frm.set\_txtchitiet(Mota);

//Chạy nút Thêm

btn[0].Click();

//Kiểm tra kết quả mong đợi và kết quả thực tế- [Mong đợi: msg, Thực tế: message]

if (msg.Equals(message))

{

//Ghi kết quả vào file excel

GhiFile.Ghi(row1++, 9, "Pass", @"D:\kiem thu\BAO\_CAO\_BTL\_KIEM\_THU\quanlykho\DataTest\TestCaseThemSanPham1.xlsx");

Assert.AreEqual(msg, message);

}

else

if (!msg.Equals(message))

{

//Ném một ngoại lệ khi Kết quả mong đợi và thực tế khác nhau

Assert.AreEqual(msg, message);

}

//Đóng form

frm.Close();

}

//Xử lý ngoại lệ

catch (Exception ex)

{

//Ngoại lệ khi kết quả mong đợi và thực tế khác nhau <=> Fail

//Hoặc khi form xảy ra lỗi <=> Fail

GhiFile.Ghi(row1++, 9, "Fail", @"D:\kiem thu\BAO\_CAO\_BTL\_KIEM\_THU\quanlykho\DataTest\TestCaseThemSanPham1.xlsx");

//Trả lại kết quả của test case

Assert.Fail();

}

}

public int row = 2;

//Test case với bộ DataTest cùng các tham số tương ứng các cột trong Datatest

[TestCaseSource("DataTest")]

public void Round2(string ten, string mau, string size, string nsx, string soluong, string gia, string Mota, string msg)

{

//Sử dụng try-catch

//Khi chức năng xảy ra bất cứ lỗi nào -> Test case tương ứng với Fail

try

{

//Khởi tạo và hiển thị form

themSP frm = new themSP();

frm.Show();

//Lấy thông báo và đóng messageBox

string message = "";

ModalFormTester mdf = new ModalFormTester();

mdf.ExpectModal("NOTE", delegate

{

MessageBoxTester msgt = new MessageBoxTester("NOTE");

message = msgt.Text;

msgt.ClickOk();

});

//Khởi tạo nút bấm

ButtonTester btn = new ButtonTester("btnluu");

//Đưa dữ liệu từ DataTest vào các trường TextBox

frm.set\_txtten(ten);

frm.set\_txtmausac(mau);

frm.set\_txtsize(size);

frm.set\_txtnsx(nsx);

frm.set\_txtsoluongtrongkho(soluong);

frm.set\_txtgia(gia);

frm.set\_txtchitiet(Mota);

//Chạy nút Thêm

btn[0].Click();

//Kiểm tra kết quả mong đợi và kết quả thực tế- [Mong đợi: msg, Thực tế: message]

if (msg.Equals(message))

{

//Ghi kết quả vào file excel

GhiFile.Ghi(row++, 10, "Pass", @"D:\kiem thu\BAO\_CAO\_BTL\_KIEM\_THU\quanlykho\DataTest\TestCaseThemSanPham1.xlsx");

Assert.AreEqual(msg, message);

}

else

if (!msg.Equals(message))

{

//Ném một ngoại lệ khi Kết quả mong đợi và thực tế khác nhau

Assert.AreEqual(msg, message);

}

//Đóng form

frm.Close();

}

//Xử lý ngoại lệ

catch (Exception ex)

{

//Ngoại lệ khi kết quả mong đợi và thực tế khác nhau <=> Fail

//Hoặc khi form xảy ra lỗi <=> Fail

GhiFile.Ghi(row++, 10, "Fail", @"D:\kiem thu\BAO\_CAO\_BTL\_KIEM\_THU\quanlykho\DataTest\TestCaseThemSanPham1.xlsx");

//Trả lại kết quả của test case

Assert.Fail();

}

}

//Hiển thị kết quả trong excel

[Test]

public void zResult()

{

System.Diagnostics.Process.Start(@"D:\kiem thu\BAO\_CAO\_BTL\_KIEM\_THU\quanlykho\DataTest\TestCaseThemSanPham1.xlsx");

Assert.Pass();

}

}

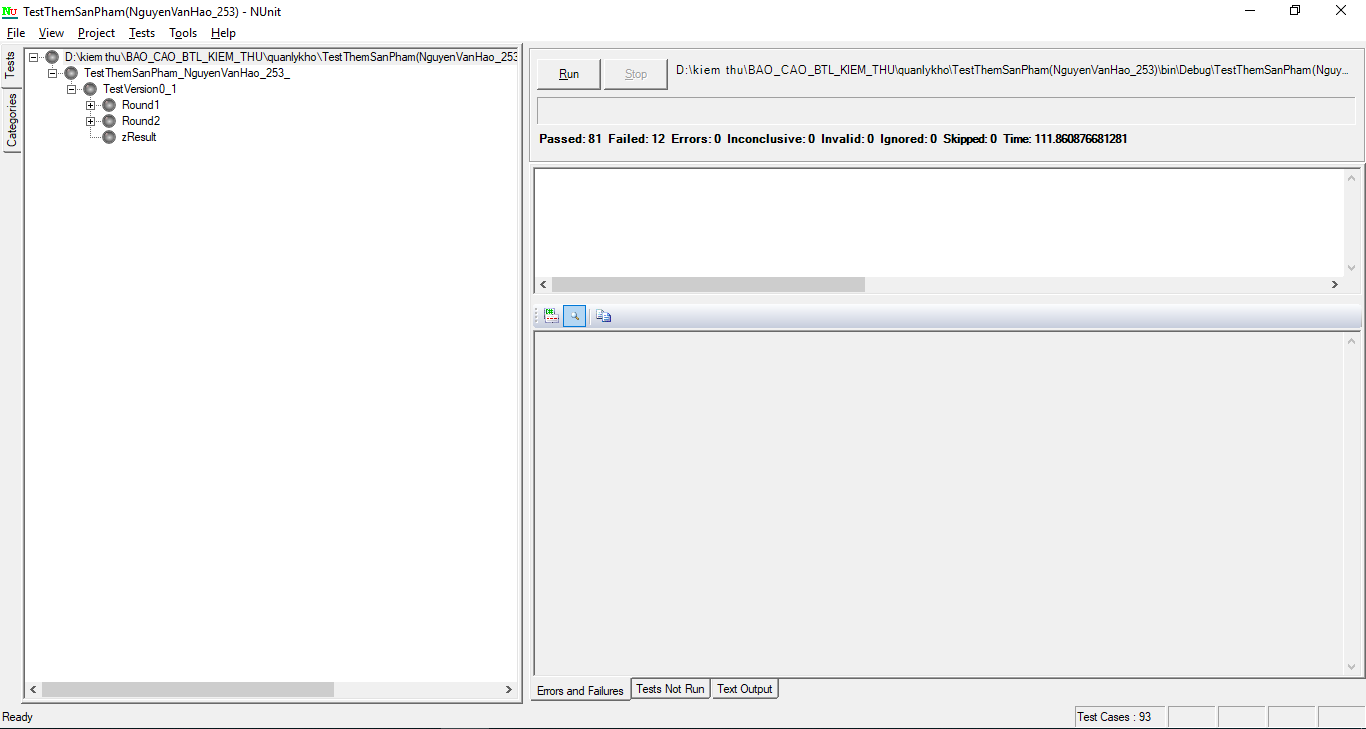
}

B6: Chuột phải chọn Run Test

B7: Mở nunit-x86.exe

* Chọn open project
* Chọn project ‘Kho’
* Chọn thư mục TestThemSanPham(NguyenVanHao\_253)
* Chọn thư mục bin
* Chọn thư mục debug
* Chọn TestThemSanPham(NguyenVanHao\_253).dll và Open

B8: bấm Run



*Hình 3.2.3.2: Công cụ Nunit*

#### Báo cáo kết quả kiểm thử

Báo cáo kết quả kiểm thử:

***Round 1:***

Đã test: 100%

Thành công: 95,7%

Pending: 4,3%

***Round 2:***

Đã test: 100%

Thành công: 95,7%

Pending: 4,3%

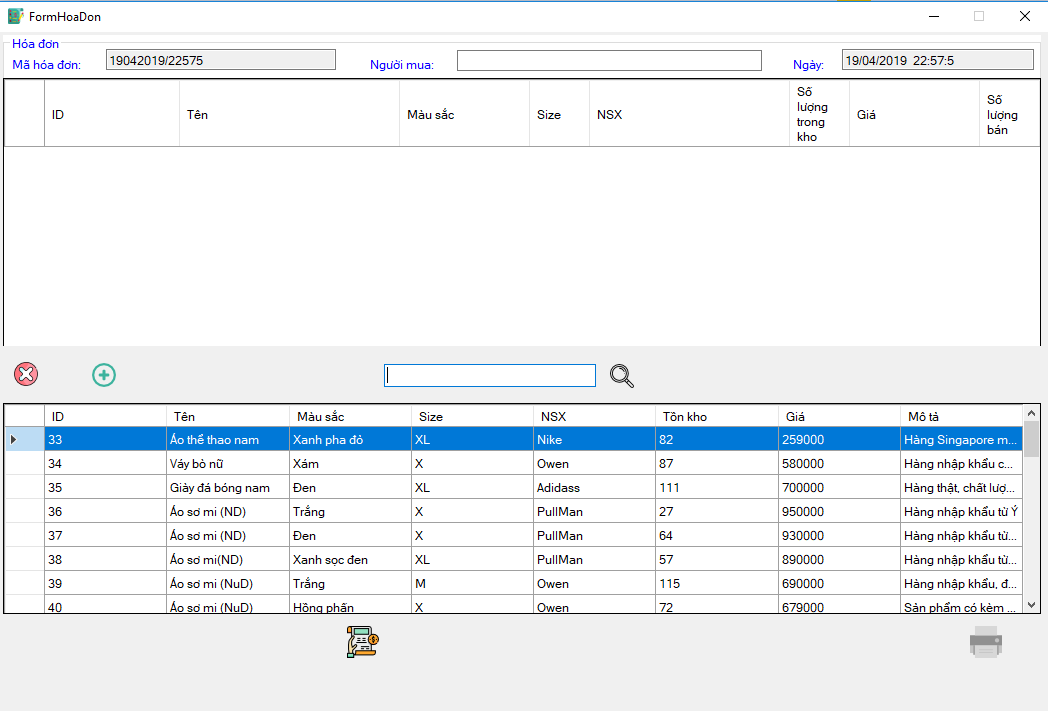
|  |  |  |
| --- | --- | --- |
|  | ROUND1 | ROUND2 |
| Total | 46 | 46 |
| OK | 44 | 44 |
| NG | 2 | 2 |
| NT | 0 | 0 |
| Not Yet Test | 0 | 0 |

*Bảng 3.2.3.3: Kết quả*

### Lê Mạnh Hùng – Thanh toán hóa đơn (V0.1)

#### Phân tích thiết kế kểm thử

* Giao diện chức năng :



*Hình 3.2.5.1: Giao diện chức năng*

* Mã nguồn của chức năng :

private void btnthanhtoan\_Click(object sender, EventArgs e) Begin

{

Boolean flag = true; 1

for (Int32 i = 0; i < dgvhoadon.RowCount; i++)

{

Int32 slBan;

try

{

Int32 slTrongKho = Convert.ToInt32(dgvhoadon.Rows[i].Cells["soluongtrongkho"].Value.ToString());

if (dgvhoadon.Rows[i].Cells["soluongban"].Value != null)

{

slBan = Convert.ToInt32(dgvhoadon.Rows[i].Cells["soluongban"].Value.ToString()); 2

if (slBan > slTrongKho || slBan < 0)

{

flag = false;

break;

}

}

else slBan = 0;

}

catch (FormatException)

{

flag = false;

}

}

if (flag == true)

{

tongtien = 0;

for (Int32 i = 0; i < dgvhoadon.RowCount; i++)

{

if(dgvhoadon.Rows[i].Cells["soluongban"].Value!=null)

tongtien += Convert.ToInt32(dgvhoadon.Rows[i].Cells["gia"].Value.ToString()) \* Convert.ToInt32(dgvhoadon.Rows[i].Cells["soluongban"].Value.ToString());

}

string tien = Convert.ToString(tongtien);

string tiendd = "";

Int32 sodauphay = (tien.Length - 1) / 3;

Int32 dem1 = 0;

Int32 dem2 = 0;

for (Int32 i = tien.Length - 1; i >= 0; i--)

{

if (dem1 < sodauphay)

{

tiendd += tien[i];

dem2++;

if (dem2 == 3)

{

tiendd += ",";

dem1++;

dem2 = 0;

}

}

else

{

tiendd += tien[i];

}

}

tiendd = ReverseString(tiendd); 3

txttongtien.Text = tiendd;

String mahoadon = txtmahoadon.Text;

string tennguoimua = txtnguoimua.Text;

string ngayhomnay = txtngay.Text;

for (Int32 i = 0; i < dgvhoadon.RowCount; i++)

{

Int32 idsp = Convert.ToInt32(dgvhoadon.Rows[i].Cells["ID"].Value.ToString());

string tensp = dgvhoadon.Rows[i].Cells["ten"].Value.ToString();

string mausacsp = dgvhoadon.Rows[i].Cells["mausac"].Value.ToString();

string sizesp = dgvhoadon.Rows[i].Cells["size"].Value.ToString();

string nsxsp = dgvhoadon.Rows[i].Cells["nsx"].Value.ToString();

Int32 soluongtrongkhosp = Convert.ToInt32(dgvhoadon.Rows[i].Cells["soluongtrongkho"].Value.ToString());

Int32 giasp = Convert.ToInt32(dgvhoadon.Rows[i].Cells["gia"].Value.ToString());

Int32 soluongbansp;

if (dgvhoadon.Rows[i].Cells["soluongban"].Value!=null)

{

soluongbansp = Convert.ToInt32(dgvhoadon.Rows[i].Cells["soluongban"].Value.ToString());

}

else {

soluongbansp = 0;

}

hoadon\_BUS.insertSPHoaDon(idsp, tensp, mausacsp, sizesp, nsxsp, giasp, soluongbansp, mahoadon);

Int32 soluongconlai = soluongtrongkhosp - soluongbansp;

sp\_BUS.truSoLuongSP(idsp, soluongconlai);

}

DShoadon\_BUS.themDSHD(mahoadon, tennguoimua, 3 ngayhomnay, tongtien);

MessageBox.Show("SUCCESS" ,"NOTE"); END

FormHoaDon\_Load(sender, e);

btnin.Enabled = true;

}

else

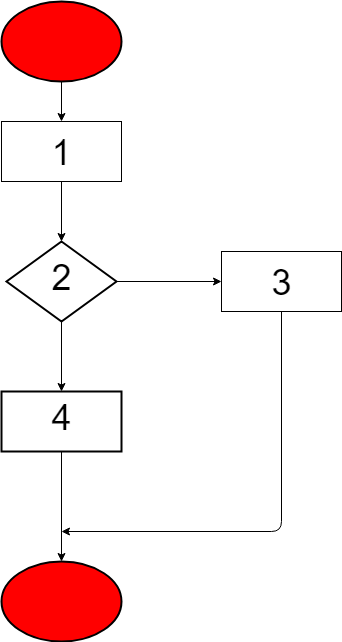
{ 4

MessageBox.Show("ERROR", "NOTE");

}

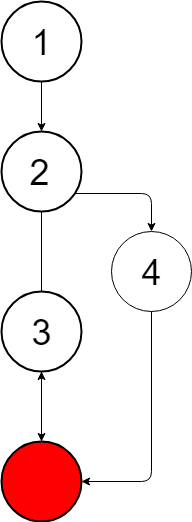
}

* Lưu đồ giải thuật hoặc sơ đồ đồ thị dòng
* Lưu đồ giải thuật



*Hình 3.2.5.1: Lưu đồ chức năng thanh toán hóa đơn*

* Sơ đồ đồ thị dòng



*Hình 3.2.5.3: Lưu đồ chức năng thanh toán hóa đơn*

* Độ phức tạp chu trình

C = 1 + 1 = 2

1. 1, 2, 4.
2. 1, 2, 3.

Độ bao phủ cấp 1

* Thiết kế test case :
  + Mô tả điều kiện :
* Số lượng phải nhập.
* Số lượng nhập vào x kiểu INT sao cho x>=0 hoặc x< Số lượng trong kho
  + Xây dựng test case :

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Đầu vào | Vùng hợp lệ | Ký hiệu đánh dấu | Vùng không hợp lệ | Đánh dấu |
| Sẩn phẩm đã có và số lượng nhập vào | Đúng định dạng  Không âm và nhỏ hơn số lượng trong kho, | H1 | Sai định dạng | K1 |
| < Số lượng trong kho | K2 |
| Bỏ trống | K3 |
| Số âm | K4 |

Bảng 3.1 Bảng phân vùng tương đương.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Ca | INPUT | | OUTPUT | | Bao phủ |
| ID Sản phẩm | Số lượng | Tổng giá | MESSAGE |  |
| 1 | 33,34,35 | 1,2,3 | 3519000 | SUCCESS | H1 |
| 2 | 35,36,37,38 | 1,,1,1 | 0 | ERROR | K3 |
| 3 | 33,,35 | 1,1,1 | 959000 | SUCCESS | H1 |
| 4 | 35,36,37 | 10,-1,2 | 0 | ERROR | K4 |
| 5 | 34,35,36 | a,b,0 | 0 | ERROR | K1 |
| 6 | 33,33,34 | 1,1,1 | 839000 | SUCCES | H1 |
| 7 | 33,34,35 | 1000,1,1 | 1280000 | ERROR | K2 |

Bảng 3.2 Ca kiểm thử.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| TestCase | Name | INPUT | | OUTPUT | |
| ID Sản phẩm | Số lượng | Tổng giá | MESSAGE |
| 1 | ThanhToan | 33,34,35 | 1,2,3 | 3519000 | SUCCESS |
| 2 | ThanhToan | 35,36,37,38 | 1,,1,1 | 0 | ERROR |
| 3 | ThanhToan | 33,,35 | 1,1,1 | 959000 | SUCCESS |
| 4 | ThanhToan | 35,36,37 | 10,-1,2 | 0 | ERROR |
| 5 | ThanhToan | 34,35,36 | a,b,0 | 0 | ERROR |
| 6 | ThanhToan | 33,33,34 | 1,1,11 | 839000 | SUCCESS |
| 7 | ThanhToan | 33,34,35 | 1000,1,1 | 1280000 | ERROR |

Bảng 3.3 Danh sách test case sau khi phân tích

#### Thực hiện kiểm thử

B1: Tải Nunit 2.7.0 và NunitForms v2.0 alpha5 rồi cài đặt.

B2: Tạo Unit Test Project tên ‘Test’ trong Solution ‘Kho’.

B3: Add References

* Nunit.framework
* Nunitforms
* GUI (Project kiểm thử)
* System.Windows.Forms
* Microsoft.Office.Interop.Excel

B4: Tạo file thao tác với file Excel và một số hàm hỗ trợ

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Data;

using System.IO;

using System.Runtime.InteropServices;

using Excel = Microsoft.Office.Interop.Excel;

using System.Collections.ObjectModel;

namespace Test

{

public class Class1

{

public static Excel.Application xlApp;

public static Excel.Workbook xlWorkbook;

public static Excel.\_Worksheet xlWorksheet;

public static Excel.Range xlRange;

public static void open(String fileName)

{

//Create COM Objects. Create a COM object for everything that is referenced

xlApp = new Excel.Application();

xlWorkbook = xlApp.Workbooks.Open(fileName);

xlWorksheet = xlWorkbook.Sheets[1];

xlRange = xlWorksheet.UsedRange;

}

public static void close()

{

//cleanup

GC.Collect();

GC.WaitForPendingFinalizers();

//release com objects to fully kill excel process from running in the background

Marshal.ReleaseComObject(xlRange);

Marshal.ReleaseComObject(xlWorksheet);

//close and release

xlWorkbook.Save();

xlWorkbook.Close();

Marshal.ReleaseComObject(xlWorkbook);

//quit and release

xlApp.Quit();

Marshal.ReleaseComObject(xlApp);

}

public static Collection<Object[]> getExcelFile(String fileName)

{

open(fileName);

int rowCount = xlRange.Rows.Count;

int colCount = 6;

int j = 3;

Object[] obj;

Collection<Object[]> list = new Collection<object[]>();

for (int i = 3; i <= rowCount; i++)

{

String idsp = xlRange.Cells[i, j++].Value2.ToString();

String soluong = xlRange.Cells[i, j++].Value2.ToString();

Int32 tien = Convert.ToInt32(xlRange.Cells[i, j++].Value2.ToString());

String mess = xlRange.Cells[i, j++].Value2.ToString();

obj = new Object[] { idsp, soluong, tien, mess };

list.Add(obj);

if (j > colCount)

j = 3;

}

close();

return list;

}

public static void setExcelFile(int i, int j, String str, String fileName)

{

open(fileName);

xlRange.Cells[i, j].value2 = str;

close();

}

public static String[] CatChuoi(String chuoi)

{

String []ds;

Char c = ',';

ds = chuoi.Split(c);

return ds;

}

public static bool timtrung(String[] a, int num\_elements, string value)

{

int i;

for (i= 0; i < num\_elements; i++)

{

if (a[i].Equals(value))

{

return true;

}

}

return false;

}

}

}

B5: Tạo file ‘Test’ trong Project Test chức mã nguồn phục vụ chạy

using GUI;

using System;

using NUnit.Extensions.Forms;

using NUnit.Framework;

using System.Windows.Forms;

using System.Runtime.InteropServices;

using Excel = Microsoft.Office.Interop.Excel;

using System.Collections.Generic;

using System.Collections.ObjectModel;

using System.Threading;

namespace Test

{

[TestFixture]

public class Test

{

public int row = 3;

public static Collection<Object[]> data()

{

return Class1.getExcelFile(@"F:\DT\testcase.xlsx");

}

[TestCaseSource("data")]

public void thanhToan(String idsp, String SoLuong, Int32 tien, String mess)

{

FormHoaDon fhd = new FormHoaDon();

fhd.Show();

string actmsg = "";

ModalFormTester msgbox = new ModalFormTester();

msgbox.ExpectModal("NOTE", delegate

{

MessageBoxTester msg = new MessageBoxTester("NOTE");

actmsg = msg.Text;

msg.ClickOk();

});

ControlTester t = new ControlTester("txtsearch", fhd);

ButtonTester b = new ButtonTester("btnthem", fhd);

string[] chuoiid;

chuoiid = Class1.CatChuoi(idsp);

for (int i = 0; i < chuoiid.Length; i++)

{

t[0].Click();

fhd.setText(chuoiid[i]);

b[0].Click();

t[0].Click();

fhd.setText("");

}

string[] chuoisl = Class1.CatChuoi(SoLuong);

try

{

for (int i = 0; i < chuoisl.Length; i++)

{

fhd.setSoLuong(i, 7, chuoisl[i]);

}

}

catch (Exception)

{

}

ButtonTester btnadd = new ButtonTester("btnthanhtoan", fhd);

btnadd[0].Click();

string error = "ERROR";

if ( fhd.tongtien==tien && mess.Equals(actmsg))

{

Class1.setExcelFile(row++, 7, "Pass", @"F:\DT\testcase.xlsx");

}

else

{

if (mess.Equals(error) && fhd.tongtien == 0)

{

Class1.setExcelFile(row++, 7, "Pass", @"F:\DT\testcase.xlsx");

}

else

Class1.setExcelFile(row++, 7, "Fail", @"F:\DT\testcase.xlsx");

}

if (mess.Equals(actmsg))

{

if(mess.Equals(error)&& fhd.tongtien == 0)

{

Assert.Pass();

}

else

Assert.AreEqual(fhd.tongtien, tien);

}

else

{

Assert.AreEqual(mess, actmsg);

}

fhd.Close();

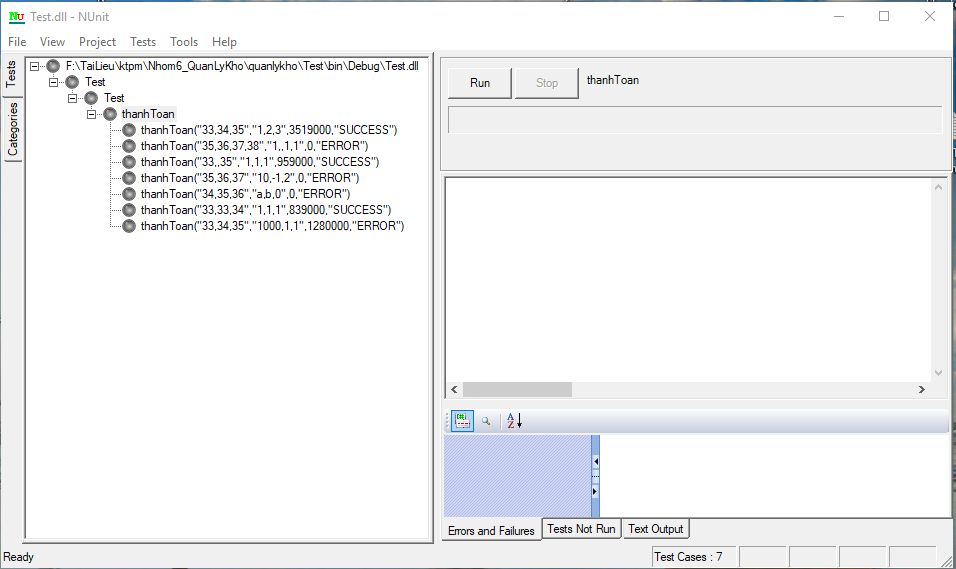
}

}

}

B6: Thực hiện chạy kiểm thử

* Chọn Test🡪 Run🡪 All tests.
* Mở file NUnit 2.7.0 đã giải nén ban đầu rồi chọn nunit.exe.
* Trong giao diện chính của NUnit bạn chọn File 🡪 Open Project 🡪 Chọn đến class Test.dll 🡪 OK.
* Nếu thành công thì giao diện NUnit sẽ hiển thị lên như sau:



*Hình 3.2.5.4: Công cụ Nunit 2.7.0*

* Chọn Run rồi kiểm tra kết quả.

#### Báo cáo kết quả kiểm thử

Báo cáo kết quả kiểm thử:

***Round 1:***

Đã test: 100%

Thành công: 85, 71%

Pending: 14, 29%

***Round 2:***

Đã test: 100%

Thành công: 85, 71%

Pending: 14, 29%

|  |  |  |
| --- | --- | --- |
|  | ROUND1 | ROUND2 |
| Total | 7 | 7 |
| OK | 6 | 6 |
| NG | 1 | 1 |
| NT | 0 | 0 |
| Not Yet Test | 0 | 0 |

Bảng 3.4 Kết quả test chức năng thanh toán hóa đơn

### Lê Mạnh Hùng – Thanh toán hóa đơn (V0.11)

#### Phân tích thiết kế kểm thử

Ở phiên bản này thì chức năng “Thanh toán hóa đơn” sẽ được cải tiến mã nguồn (thêm điều kiện bắt lỗi) để đảm bảo rằng chức năng sẽ hoạt động tốt ở nhiều tình huống khác nhau.

Mã nguồn ban đầu:

if (flag == true)

{

tongtien = 0;

for (Int32 i = 0; i < dgvhoadon.RowCount; i++)

{

if(dgvhoadon.Rows[i].Cells["soluongban"].Value!=null)

tongtien += Convert.ToInt32(dgvhoadon.Rows[i].Cells["gia"].Value.ToString()) \* Convert.ToInt32(dgvhoadon.Rows[i].Cells["soluongban"].Value.ToString());

}

string tien = Convert.ToString(tongtien);

string tiendd = "";

Int32 sodauphay = (tien.Length - 1) / 3;

Int32 dem1 = 0;

Int32 dem2 = 0;

for (Int32 i = tien.Length - 1; i >= 0; i--)

{

if (dem1 < sodauphay)

{

tiendd += tien[i];

dem2++;

if (dem2 == 3)

{

tiendd += ",";

dem1++;

dem2 = 0;

}

}

else

{

tiendd += tien[i];

}

}

tiendd = ReverseString(tiendd);

txttongtien.Text = tiendd;

String mahoadon = txtmahoadon.Text;

string tennguoimua = txtnguoimua.Text;

string ngayhomnay = txtngay.Text;

for (Int32 i = 0; i < dgvhoadon.RowCount; i++)

{

Int32 idsp = Convert.ToInt32(dgvhoadon.Rows[i].Cells["ID"].Value.ToString());

string tensp = dgvhoadon.Rows[i].Cells["ten"].Value.ToString();

string mausacsp = dgvhoadon.Rows[i].Cells["mausac"].Value.ToString();

string sizesp = dgvhoadon.Rows[i].Cells["size"].Value.ToString();

string nsxsp = dgvhoadon.Rows[i].Cells["nsx"].Value.ToString();

Int32 soluongtrongkhosp = Convert.ToInt32(dgvhoadon.Rows[i].Cells["soluongtrongkho"].Value.ToString());

Int32 giasp = Convert.ToInt32(dgvhoadon.Rows[i].Cells["gia"].Value.ToString());

Int32 soluongbansp;

if (dgvhoadon.Rows[i].Cells["soluongban"].Value!=null)

{

soluongbansp = Convert.ToInt32(dgvhoadon.Rows[i].Cells["soluongban"].Value.ToString());

}

else {

soluongbansp = 0;

}

hoadon\_BUS.insertSPHoaDon(idsp, tensp, mausacsp, sizesp, nsxsp, giasp, soluongbansp, mahoadon);

Int32 soluongconlai = soluongtrongkhosp - soluongbansp;

sp\_BUS.truSoLuongSP(idsp, soluongconlai);

}

DShoadon\_BUS.themDSHD(mahoadon, tennguoimua, ngayhomnay, tongtien);

MessageBox.Show("SUCCESS" ,"NOTE");

FormHoaDon\_Load(sender, e);

btnin.Enabled = true;

}

else

{

MessageBox.Show("ERROR", "NOTE");

}

}

Mã nguồn sau khi chỉnh sửa:

if (flag == true)

{

//tinh tong tien cua hoa don va luu lai hoa don vua nhap

tongtien = 0;

for (Int32 i = 0; i < dgvhoadon.RowCount; i++)

{

if (dgvhoadon.Rows[i].Cells["soluongban"].Value != null)

tongtien += Convert.ToInt32(dgvhoadon.Rows[i].Cells["gia"].Value.ToString()) \* Convert.ToInt32(dgvhoadon.Rows[i].Cells["soluongban"].Value.ToString());

}

//

//Định dạng lại text của tổng tiền

//

string tien = Convert.ToString(tongtien);

string tiendd = "";//Tien duoc dinh dang de hien thi trong o text (cho them dau ",")

Int32 sodauphay = (tien.Length - 1) / 3;//So dau phay trong chuoi

Int32 dem1 = 0;//dung de dem cac dau phay duoc chen vao trong chuoi.

Int32 dem2 = 0;//dùng để thực hiện vc cứ ba chữ số thì sẽ chèn một dấu phẩy.

for (Int32 i = tien.Length - 1; i >= 0; i--)

{

if (dem1 < sodauphay)

{

tiendd += tien[i];

dem2++;

if (dem2 == 3)

{

tiendd += ",";

dem1++;

dem2 = 0;

}

}

else

{

tiendd += tien[i];

}

}

tiendd = ReverseString(tiendd);

txttongtien.Text = tiendd;

// Dua cac thuoc tinh vao co so du lieu (vao bang trunggian, dshoadon)

String mahoadon = txtmahoadon.Text;

string tennguoimua = txtnguoimua.Text;

string ngayhomnay = txtngay.Text;

try

{

for (Int32 i = 0; i < dgvhoadon.RowCount; i++)

{

Int32 idsp = Convert.ToInt32(dgvhoadon.Rows[i].Cells["ID"].Value.ToString());

string tensp = dgvhoadon.Rows[i].Cells["ten"].Value.ToString();

string mausacsp = dgvhoadon.Rows[i].Cells["mausac"].Value.ToString();

string sizesp = dgvhoadon.Rows[i].Cells["size"].Value.ToString();

string nsxsp = dgvhoadon.Rows[i].Cells["nsx"].Value.ToString();

Int32 soluongtrongkhosp = Convert.ToInt32(dgvhoadon.Rows[i].Cells["soluongtrongkho"].Value.ToString());

Int32 giasp = Convert.ToInt32(dgvhoadon.Rows[i].Cells["gia"].Value.ToString());

Int32 soluongbansp;

soluongbansp = Convert.ToInt32(dgvhoadon.Rows[i].Cells["soluongban"].Value.ToString());

//Insert cac san pham vao bang trung gian

hoadon\_BUS.insertSPHoaDon(idsp, tensp, mausacsp, sizesp, nsxsp, giasp, soluongbansp, mahoadon);

//Cap nhat giam so luong cac san pham trong bang san pham

Int32 soluongconlai = soluongtrongkhosp - soluongbansp;

sp\_BUS.truSoLuongSP(idsp, soluongconlai); DShoadon\_BUS.themDSHD(mahoadon, tennguoimua, ngayhomnay, tongtien);

MessageBox.Show("SUCCESS", "NOTE");

FormHoaDon\_Load(sender, e);

btnin.Enabled = true;

}

}

catch (FormatException)

{

MessageBox.Show("ERROR", "NOTE");

}

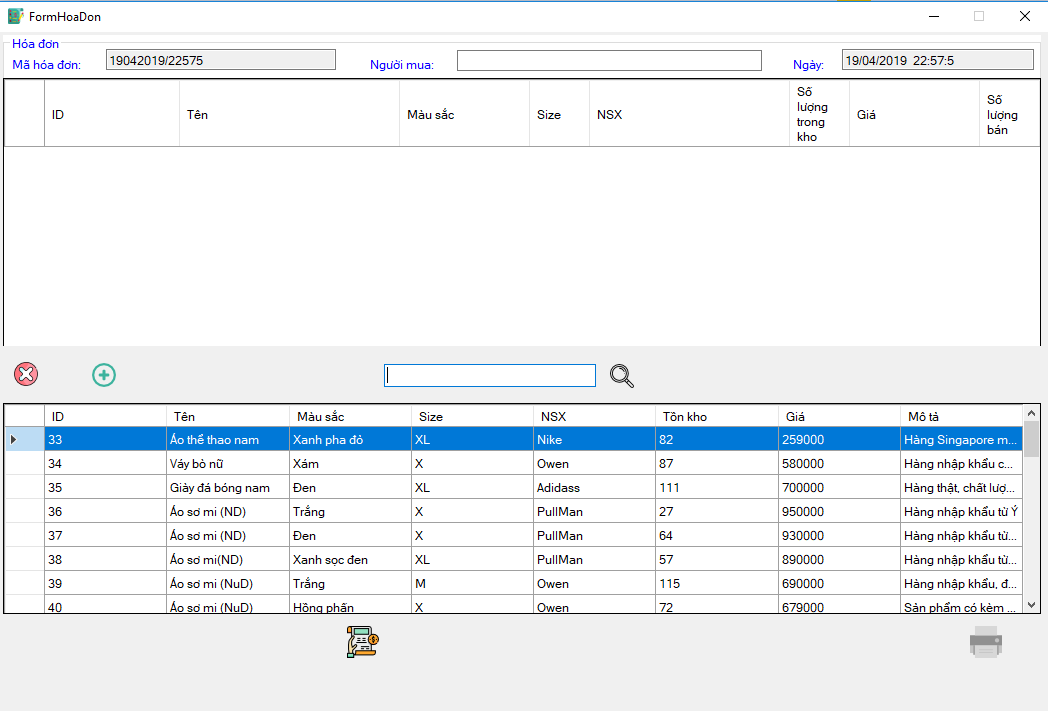
}

else {

MessageBox.Show("ERROR", "NOTE");

}

* Giao diện chức năng :



*Hình 3.2.6.1: Giao diện chức năng thanh toán hóa đơn*

* Mã nguồn của chức năng

private void btnthanhtoan\_Click(object sender, EventArgs e) Begin

{

Boolean flag = true; 1

for (Int32 i = 0; i < dgvhoadon.RowCount; i++)

{

Int32 slBan;

try

{

Int32 slTrongKho = Convert.ToInt32(dgvhoadon.Rows[i].Cells["soluongtrongkho"].Value.ToString());

if (dgvhoadon.Rows[i].Cells["soluongban"].Value != null)

{

slBan = Convert.ToInt32(dgvhoadon.Rows[i].Cells["soluongban"].Value.ToString()); 2

if (slBan > slTrongKho || slBan < 0)

{

flag = false;

break;

}

}

else slBan = 0;

}

catch (FormatException)

{

flag = false;

}

}

if (flag == true)

{

tongtien = 0;

for (Int32 i = 0; i < dgvhoadon.RowCount; i++)

{

if (dgvhoadon.Rows[i].Cells["soluongban"].Value != null)

tongtien += Convert.ToInt32(dgvhoadon.Rows[i].Cells["gia"].Value.ToString()) \* Convert.ToInt32(dgvhoadon.Rows[i].Cells["soluongban"].Value.ToString());

}

string tien = Convert.ToString(tongtien);

string tiendd = "";

Int32 sodauphay = (tien.Length - 1) / 3;

Int32 dem1 = 0;

Int32 dem2 = 0

for (Int32 i = tien.Length - 1; i >= 0; i--)

{

if (dem1 < sodauphay) 3

{

tiendd += tien[i];

dem2++;

if (dem2 == 3)

{

tiendd += ",";

dem1++;

dem2 = 0;

}

}

else

{

tiendd += tien[i];

}

}

tiendd = ReverseString(tiendd);

txttongtien.Text = tiendd;

String mahoadon = txtmahoadon.Text;

string tennguoimua = txtnguoimua.Text;

string ngayhomnay = txtngay.Text;

try

{

for (Int32 i = 0; i < dgvhoadon.RowCount; i++)

{

Int32 idsp = Convert.ToInt32(dgvhoadon.Rows[i].Cells["ID"].Value.ToString());

string tensp = dgvhoadon.Rows[i].Cells["ten"].Value.ToString();

string mausacsp = dgvhoadon.Rows[i].Cells["mausac"].Value.ToString();

string sizesp = dgvhoadon.Rows[i].Cells["size"].Value.ToString();

string nsxsp = dgvhoadon.Rows[i].Cells["nsx"].Value.ToString();

Int32 soluongtrongkhosp = Convert.ToInt32(dgvhoadon.Rows[i].Cells["soluongtrongkho"].Value.ToString());

Int32 giasp = Convert.ToInt32(dgvhoadon.Rows[i].Cells["gia"].Value.ToString()); 3

Int32 soluongbansp;

soluongbansp = Convert.ToInt32(dgvhoadon.Rows[i].Cells["soluongban"].Value.ToString());

hoadon\_BUS.insertSPHoaDon(idsp, tensp, mausacsp, sizesp, nsxsp, giasp, soluongbansp, mahoadon);

Int32 soluongconlai = soluongtrongkhosp - soluongbansp;

sp\_BUS.truSoLuongSP(idsp, soluongconlai); DShoadon\_BUS.themDSHD(mahoadon, tennguoimua, ngayhomnay, tongtien);

MessageBox.Show("SUCCESS", "NOTE");

FormHoaDon\_Load(sender, e);

btnin.Enabled = true;

}

}

catch (FormatException)

{

MessageBox.Show("ERROR", "NOTE");

}

}

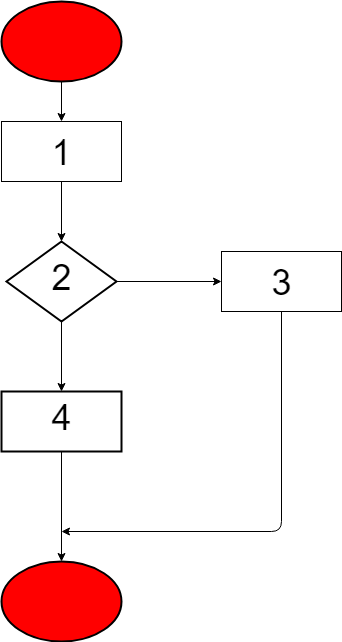
else {

MessageBox.Show("ERROR", "NOTE"); 4

}

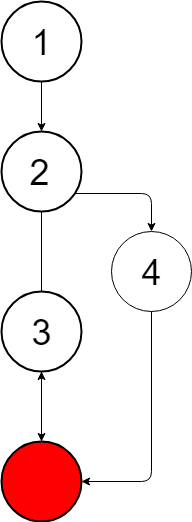
END

* Lưu đồ giải thuật hoặc sơ đồ đồ thị dòng
* Lưu đồ giải thuật



*Hình 3.2.6.2: Lưu đồ chức năng thanh toán hóa đơn*

* Sơ đồ đồ thị dòng



*Hình 3.2.3.2: Lưu đồ chức năng thanh toán hóa đơn*

* Độ phức tạp chu trình

C = 1 + 1 = 2

1. 1, 2, 4.
2. 1, 2, 3.

Độ bao phủ cấp 1

* Thiết kế test case :
  + Mô tả điều kiện :
* Số lượng phải nhập.
* Số lượng nhập vào x kiểu INT sao cho x>=0 hoặc x< Số lượng trong kho
  + Xây dựng test case :

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Đầu vào | Vùng hợp lệ | Ký hiệu đánh dấu | Vùng không hợp lệ | Đánh dấu |
| Sẩn phẩm đã có và số lượng nhập vào | Đúng định dạng  Không âm và nhỏ hơn số lượng trong kho, | H1 | Sai định dạng | K1 |
| < Số lượng trong kho | K2 |
| Bỏ trống | K3 |
| Số âm | K4 |

Bảng 3.5 Bảng phân vùng tương đương.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Ca | INPUT | | OUTPUT | | Bao phủ |
| ID Sản phẩm | Số lượng | Tổng giá | MESSAGE |  |
| 1 | 33,34,35 | 1,2,3 | 3519000 | SUCCESS | H1 |
| 2 | 35,36,37,38 | 1,,1,1 | 0 | ERROR | K3 |
| 3 | 33,,35 | 1,1,1 | 959000 | SUCCESS | H1 |
| 4 | 35,36,37 | 10,-1,2 | 0 | ERROR | K4 |
| 5 | 34,35,36 | a,b,0 | 0 | ERROR | K1 |
| 6 | 33,33,34 | 1,1,1 | 839000 | SUCCES | H1 |
| 7 | 33,34,35 | 1000,1,1 | 1280000 | ERROR | K2 |
| 8 | 33,34,35,36,37,38,39,40,41,42,43,45 | 1,1,1,1,1,1,1,1,1,1,1,1 | 8785000 | SUCCESS | H1 |
| 9 | 33 | 1,1,1,1 | 259000 | SUCCESS | H1 |
| 10 | 33,9,34 | 1,0,1 | 839000 | SUCCESS | H1 |

Bảng 3.6 Ca kiểm thử.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| TestCase | Name | INPUT | | OUTPUT | |
| ID Sản phẩm | Số lượng | Tổng giá | MESSAGE |
| 1 | ThanhToan | 33,34,35 | 1,2,3 | 3519000 | SUCCESS |
| 2 | ThanhToan | 35,36,37,38 | 1,,1,1 | 0 | ERROR |
| 3 | ThanhToan | 33,,35 | 1,1,1 | 959000 | SUCCESS |
| 4 | ThanhToan | 35,36,37 | 10,-1,2 | 0 | ERROR |
| 5 | ThanhToan | 34,35,36 | a,b,0 | 0 | ERROR |
| 6 | ThanhToan | 33,33,34 | 1,1,11 | 839000 | SUCCESS |
| 7 | ThanhToan | 33,34,35 | 1000,1,1 | 1280000 | ERROR |
| 8 | ThanhToan | 33,34,35,36,37,38,39,40,41,42,43,45 | 1,1,1,1,1,1,1,1,1,1,1,1 | 8785000 | SUCCESS |
| 9 | ThanhToan | 33 | 1,1,1,1 | 259000 | SUCCESS |
| 10 | ThanhToan | 33,9,34 | 1,0,1 | 839000 | SUCCÊSS |

*Bảng 3.7 Danh sách test case sau khi phân tích*

#### Thực hiện kiểm thử

B1: Tải Nunit 2.7.0 và NunitForms v2.0 alpha5 rồi cài đặt.

B2: Tạo Unit Test Project tên ‘Test’ trong Solution ‘Kho’.

B3: Add References

* Nunit.framework
* Nunitforms
* GUI (Project kiểm thử)
* System.Windows.Forms
* Microsoft.Office.Interop.Excel

B4: Tạo file thao tác với file Excel và một số hàm hỗ trợ

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Data;

using System.IO;

using System.Runtime.InteropServices;

using Excel = Microsoft.Office.Interop.Excel;

using System.Collections.ObjectModel;

namespace Test

{

public class Class1

{

public static Excel.Application xlApp;

public static Excel.Workbook xlWorkbook;

public static Excel.\_Worksheet xlWorksheet;

public static Excel.Range xlRange;

public static void open(String fileName)

{

//Create COM Objects. Create a COM object for everything that is referenced

xlApp = new Excel.Application();

xlWorkbook = xlApp.Workbooks.Open(fileName);

xlWorksheet = xlWorkbook.Sheets[1];

xlRange = xlWorksheet.UsedRange;

}

public static void close()

{

//cleanup

GC.Collect();

GC.WaitForPendingFinalizers();

//release com objects to fully kill excel process from running in the background

Marshal.ReleaseComObject(xlRange);

Marshal.ReleaseComObject(xlWorksheet);

//close and release

xlWorkbook.Save();

xlWorkbook.Close();

Marshal.ReleaseComObject(xlWorkbook);

//quit and release

xlApp.Quit();

Marshal.ReleaseComObject(xlApp);

}

public static Collection<Object[]> getExcelFile(String fileName)

{

open(fileName);

int rowCount = xlRange.Rows.Count;

int colCount = 6;

int j = 3;

Object[] obj;

Collection<Object[]> list = new Collection<object[]>();

for (int i = 3; i <= rowCount; i++)

{

String idsp = xlRange.Cells[i, j++].Value2.ToString();

String soluong = xlRange.Cells[i, j++].Value2.ToString();

Int32 tien = Convert.ToInt32(xlRange.Cells[i, j++].Value2.ToString());

String mess = xlRange.Cells[i, j++].Value2.ToString();

obj = new Object[] { idsp, soluong, tien, mess };

list.Add(obj);

if (j > colCount)

j = 3;

}

close();

return list;

}

public static void setExcelFile(int i, int j, String str, String fileName)

{

open(fileName);

xlRange.Cells[i, j].value2 = str;

close();

}

public static String[] CatChuoi(String chuoi)

{

String []ds;

Char c = ',';

ds = chuoi.Split(c);

return ds;

}

public static bool timtrung(String[] a, int num\_elements, string value)

{

int i;

for (i= 0; i < num\_elements; i++)

{

if (a[i].Equals(value))

{

return true;

}

}

return false;

}

}

}

B5: Tạo file ‘Test’ trong Project Test chức mã nguồn phục vụ chạy

using GUI;

using System;

using NUnit.Extensions.Forms;

using NUnit.Framework;

using System.Windows.Forms;

using System.Runtime.InteropServices;

using Excel = Microsoft.Office.Interop.Excel;

using System.Collections.Generic;

using System.Collections.ObjectModel;

using System.Threading;

namespace Test

{

[TestFixture]

public class Test

{

public int row = 3;

public static Collection<Object[]> data()

{

return Class1.getExcelFile(@"F:\DT\testcase.xlsx");

}

[TestCaseSource("data")]

public void thanhToan(String idsp, String SoLuong, Int32 tien, String mess)

{

FormHoaDon fhd = new FormHoaDon();

fhd.Show();

string actmsg = "";

ModalFormTester msgbox = new ModalFormTester();

msgbox.ExpectModal("NOTE", delegate

{

MessageBoxTester msg = new MessageBoxTester("NOTE");

actmsg = msg.Text;

msg.ClickOk();

});

ControlTester t = new ControlTester("txtsearch", fhd);

ButtonTester b = new ButtonTester("btnthem", fhd);

string[] chuoiid;

chuoiid = Class1.CatChuoi(idsp);

for (int i = 0; i < chuoiid.Length; i++)

{

t[0].Click();

fhd.setText(chuoiid[i]);

b[0].Click();

t[0].Click();

fhd.setText("");

}

string[] chuoisl = Class1.CatChuoi(SoLuong);

try

{

for (int i = 0; i < chuoisl.Length; i++)

{

fhd.setSoLuong(i, 7, chuoisl[i]);

}

}

catch (Exception)

{

}

ButtonTester btnadd = new ButtonTester("btnthanhtoan", fhd);

btnadd[0].Click();

string error = "ERROR";

if ( fhd.tongtien==tien && mess.Equals(actmsg))

{

Class1.setExcelFile(row++, 8, "Pass", @"F:\DT\testcase.xlsx");

}

else

{

if (mess.Equals(error) && fhd.tongtien == 0)

{

Class1.setExcelFile(row++, 8, "Pass", @"F:\DT\testcase.xlsx");

}

else

Class1.setExcelFile(row++, 8, "Fail", @"F:\DT\testcase.xlsx");

}

if (mess.Equals(actmsg))

{

if(mess.Equals(error)&& fhd.tongtien == 0)

{

Assert.Pass();

}

else

Assert.AreEqual(fhd.tongtien, tien);

}

else

{

Assert.AreEqual(mess, actmsg);

}

fhd.Close();

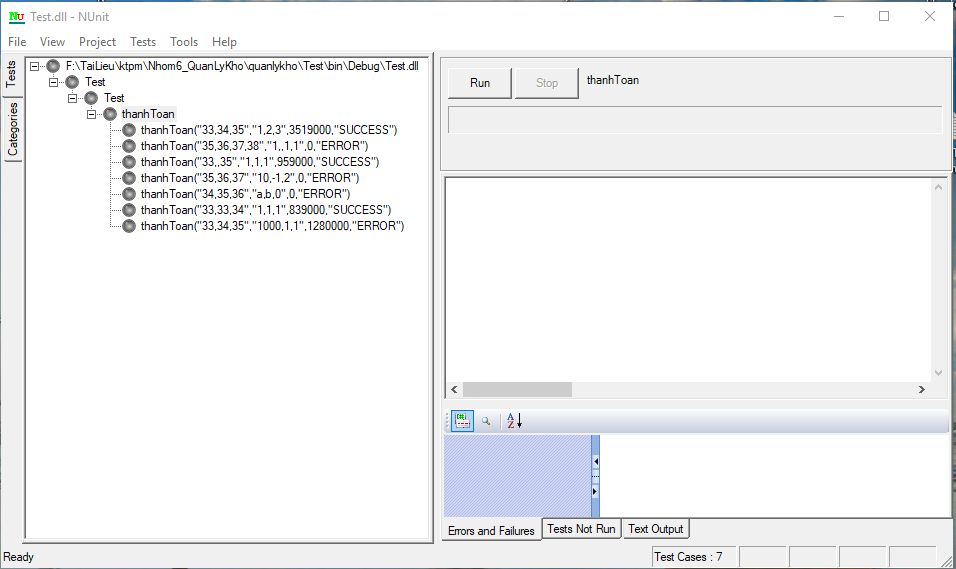
}

}

}

B6: Thực hiện chạy kiểm thử

* Chọn Test🡪 Run🡪 All tests.
* Mở file NUnit 2.7.0 đã giải nén ban đầu rồi chọn nunit.exe.
* Trong giao diện chính của NUnit bạn chọn File 🡪 Open Project 🡪 Chọn đến class Test.dll 🡪 OK.
* Nếu thành công thì giao diện NUnit sẽ hiển thị lên như sau:



*Hình 3.2.6.4: Công cụ Nunit 2.7.0*

* Chọn Run rồi kiểm tra kết quả.

#### Báo cáo kết quả kiểm thử

Báo cáo kết quả kiểm thử:

***Round 1:***

Đã test: 100%

Thành công: 90%

Pending: 10%

***Round 2:***

Đã test: 100%

Thành công: 90%

Pending: 10%

|  |  |  |
| --- | --- | --- |
|  | ROUND1 | ROUND2 |
| Total | 10 | 10 |
| OK | 9 | 9 |
| NG | 1 | 1 |
| NT | 0 | 0 |
| Not Yet Test | 0 | 0 |

Bảng 3.8 Kết quả test chức năng thanh toán hóa đơn

# Phụ lục

## Tên phụ lục 1

## Tên phụ lục 2

# Tài liệu tham khảo

[1] *Slide Kiểm thử phần mềm*  
TS Nguyễn Hoàng Tú

[2] *NUnit documentation [online]*

[3] *NUnitForms documentation [Online]*