UỶ BAN NHÂN DÂN THÀNH PHỐ HỒ CHÍ MINH

TRƯỜNG ĐẠI HỌC SÀI GÒN

KHOA CÔNG NGHỆ THÔNG TIN

BÁO CÁO MÔN KIỂM THỬ PHẦN MỀM

TÊN ĐỀ TÀI: KIỂM THỬ PHẦN MỀM QUẢN LÝ SIÊU THỊ MINI

THÀNH VIÊN

HUỲNH TRUNG KIÊN - 3120560049

LÊ DƯ GIÁP HÀO - 3120410152

NGUYỄN TẤN KIỆT - 3120410272

NGUYỄN TRƯƠNG TẤN LỘC - 3120410293

NGÔ VĂN TÍN - 3120410534

GIẢNG VIÊN HƯỚNG DẪN: VŨ THỊ HẠNH

TP. HỒ CHÍ MINH, 11/2022

MỤC LỤC

LỜI CẢM ƠN

Nhóm em xin gửi lời cảm ơn chân thành và sự tri ân sâu sắc đối với cô Vũ Thị Hạnh, giảng viên khoa Công nghệ thông tin – Trường Đại học Sài Gòn, cô đã nhiệt tình giảng dạy và hướng dẫn nhóm em hoàn thành tốt báo cáo cuối kỳ.

Trong quá trình làm bài báo cáo, khó tránh khỏi sai sót, rất mong cô bỏ qua, do trình độ lý luận cũng như kinh nghiệm thực tiễn, chuyên sâu còn hạn chế nên bài báo cáo không thể tránh khỏi những thiếu sót, nhóm em rất mong nhận được ý kiến đóng góp từ cô để nhóm em học thêm được nhiều kinh nghiệm và sẽ hoàn thành tốt hơn bài báo cáo tốt nghiệp sắp tới.

Nhóm em trân thành cảm ơn cô!

DANH MỤC CÁC THUẬT NGỮ VÀ CHỮ VIẾT TẮT

Thuật ngữ / Chữ viết tắt	Ý nghĩa
Test case	Là một tập hợp các hành động được thực thi để xác minh một function, một hệ thống phần mềm có hoạt động đúng hay không
CSDL	Cơ sở dữ liệu, nơi lưu trữ liệu cho toàn hệ thống.
CTKM	Chương trình khuyến mãi
NCC	Nhà cung cấp

I. GIỚI THIỆU CHUNG

1. Giới thiệu dự án

1.1. Giới thiệu

Tên dự án – Quản lý hệ thống siêu thị mini.

Một siêu thị mini A đang kinh doanh và họ đang cần số hóa mô hình kinh doanh của mình nên đã quyết định xây dựng một hệ thống quản lý siêu thị mini để giúp tiết kiệm chi phí quản lý, thời gian, cũng như hạn chế tối thiểu sai sót trong quá trình quản lý. Hệ thống bao gồm các chức năng như: quản lý bán hàng, quản lý nhập hàng, quản lý nhà cung ứng, quản lý tài khoản, quản lý nhân viên, quản lý thành viên, quản lý sản phẩm, loại sản phẩm, quản lý chương trình khuyên mãi, phiếu giảm giá, quản lý doanh thu, thống kê.

Hệ thống sẽ có 2 phân quyền tài khoản truy cập, cụ thể là là phân quyền giành cho admin và staff. Khi đăng nhập hệ thống sẽ kiểm tra tài khoản đăng nhập là của admin hay của staff. Tài khoản của staff (Nhân viên) chỉ được cấp chức năng như: bán hàng; thay đổi thông tin tài khoản; nhập phiếu giảm giá khi thanh toán; đăng ký thành viên cho khách hàng. Trong khi ngoài các chức năng tương tự như staff thì tài khoản của admin sẽ được cấp thêm nhiều quyền nữa như: bán hàng, nhập hàng, quản lý nhà cung ứng, quản lý tài khoản, quản lý nhân viên, quản lý sản phẩm, loại sản phẩm, quản lý thành viên, quản lý chương trình khuyến mãi, phiếu giảm giá, quản lý doanh thu, thống kê.

1.2. Công cụ và ngôn ngữ sử dụng

- Ngôn ngữ: java, ngôn ngữ truy vấn SQL.
- Công cụ: Netbeans, Swing (được tích hợp vào netbeans), Xampp, excel

2. Mô tả dự án

2.1. Mô tả chức năng hệ thống

Hệ thống bao gồm những chức năng:

- Chức năng đăng nhập: Mỗi nhân viên và cửa hàng trưởng của cửa hàng sẽ được cấp một tài khoản để truy cập vào hệ thống. Mỗi tài khoản ứng với

từng phân quyền khác nhau sẽ có những chức năng khác nhau (cửa hàng trưởng sẽ có nhiều quyền hơn nhân viên bán hàng).

- ❖ Chức năng tìm kiếm, filter theo giá, ngày, số lượng,…
- ❖ Quản lý bán hàng (cửa hàng trưởng / nhân viên bán hàng): nhập chọn sản phẩm mà khách hàng mua, nhập mã giảm giá hoặc thẻ thành viên nếu khách hàng có và yêu cầu xác nhận, xuất hóa đơn bao gồm chi tiết sản phẩm và giá (giá trước và sau khi nhập mã giám giá hoặc trừ điểm tích lũy).
- ❖ Chức năng đăng ký thành viên (cửa hàng trưởng / nhân viên bán hàng): đăng ký thẻ thành viên cho khách hàng và thực hiện tính điểm tích lũy (khách hàng nhập mã thành viên sau khi thanh toán sẽ được tích lũy điểm tương ứng sau mỗi lần thanh toán, cụ thể là 1% được hoàn vào điểm tích lũy).
- ❖ Sửa thông tin tài khoản (cửa hàng trưởng / nhân viên bán hàng): sửa thông tin tài khoản (họ, tên, tài khoản , mật khẩu).
- ❖ Quản lý nhập hàng: chọn sản phẩm thuộc nhà cung cấp tương ứng vào danh sách nhập hàng, sau đó xác nhận nhập. Những sản phẩm sau khi nhập sẽ được cập nhật vào danh sách sản phẩm.
- ❖ Quản lý nhà cung cấp (cửa hàng trưởng): thực hiện 4 chức năng cơ bản là: create, read, update, delete nhà cung cấp. Đồng thời create, read, update, delete thông tin những sản phẩm mà nhà cung cấp đấy phân phối.
- ❖ Quản lý loại sản phẩm (cửa hàng trưởng): thực hiện 4 chức năng cơ bản là: create; read; update; delete loại sản phẩm.
- ❖ Quản lý sản phẩm (cửa hàng trưởng): thực hiện 4 chức năng cơ bản là: create, read, update, delete (sản phẩm sau khi được thêm chỉ có số lượng là 0, nếu muốn cập nhật số lượng sản phẩm thì phải tiến hành nhập hàng với số lượng tương ứng).
- ❖ Quản lý tài khoản (cửa hàng trưởng): thực hiện 4 chức năng cơ bản là: create, read, update, delete tài khoản. Đồng thời được quyền cấp quyền cho tài khoản.

- ❖ Quản lý nhân viên (cửa hàng trưởng): thực hiện 4 chức năng cơ bản là: create, read, update, delete tài khoản. Cấp tài khoản cụ thể cho nhân viên.
- ❖ Quản lý chương trình khuyến mãi (cửa hàng trưởng): thực hiện 4 chức năng cơ bản là: create, read, update, delete chương trình khuyến mãi. Đồng thời thêm, xóa, sửa thông tin chi tiết của chương trình khuyến mãi (chương trình khuyến mãi đó có những sản phẩm nào được giảm giá, thời gian bắt đầu và thời gian kết thúc của chương trình khuyến mãi,). Áp dụng và ngưng chương trình khuyến mãi trước thời hạn.
- ❖ Quản lý phiếu giảm giá (cửa hàng trưởng): thực hiện 4 chức năng cơ bản là: create, read, update, delete phiếu giảm giá.
- ❖ Quản lý thành viên (cửa hàng trưởng): thực hiện chức năng: đọc, sửa, xóa thông tin thành viên.
- ❖ Chức năng thống kê (cửa hàng trưởng): thống kê tổng quát, thống kê doanh thu (hóa đơn thanh toán), thống kê nhập hàng (đơn nhập hàng), xuất thông tin ra file excel, pdf.

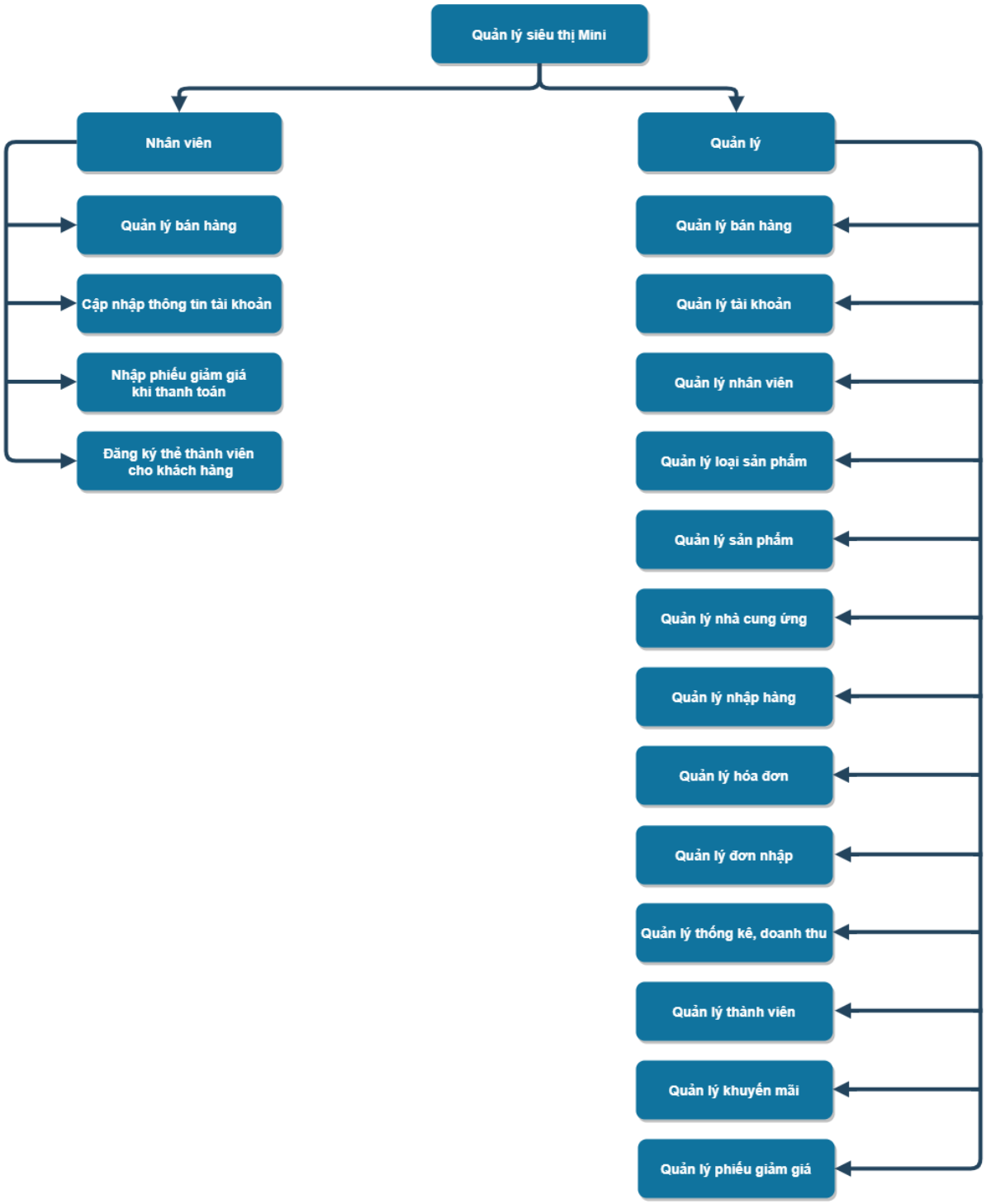
2.2. Cấu trúc tổ chức

- ❖ Tổ chức theo phân quyền (bộ phận): hệ thống được tổ chức theo 2 phân quyền chức nằng chính là cửa hàng trưởng và nhân viên bán hàng. Với từng phân quyền khác nhau sẽ có từng chức năng riêng biệt khác nhau hoặc cả hai.
- ❖ Tổ chức theo mô hình chức năng: hệ thống được tổ chức theo mô hình 3 lớp (three - layer). Với mỗi lớp thực hiện một nhiệm vụ khác nhau, cụ thể:
    - o Lớp GUI (Graphical User Interface) là lớp có nhiệm vụ giao tiếp với người dùng, thực hiện các công việc như nhập liệu, hiển thị dữ liệu, kiểm tra tính đúng đắn dữ liệu trước khi gọi lớp Business Logic Layer (BLL).
    - o Lớp BLL (Business Logic Layer) có nhiệm vụ là nơi đáp ứng các yêu cầu thao tác dữ liệu của GUI layer, xử lý chính nguồn dữ liệu từ Presentation Layer trước khi truyền xuống Data Access Layer và lưu

xuống hệ quản trị CSDL. Đồng thời còn là nơi đáp ứng các yêu cầu thao tác dữ liệu của GUI layer, xử lý chính nguồn dữ liệu từ được nhập vào trước khi truyền xuống Data Access Layer và lưu xuống hệ quản trị CSDL.

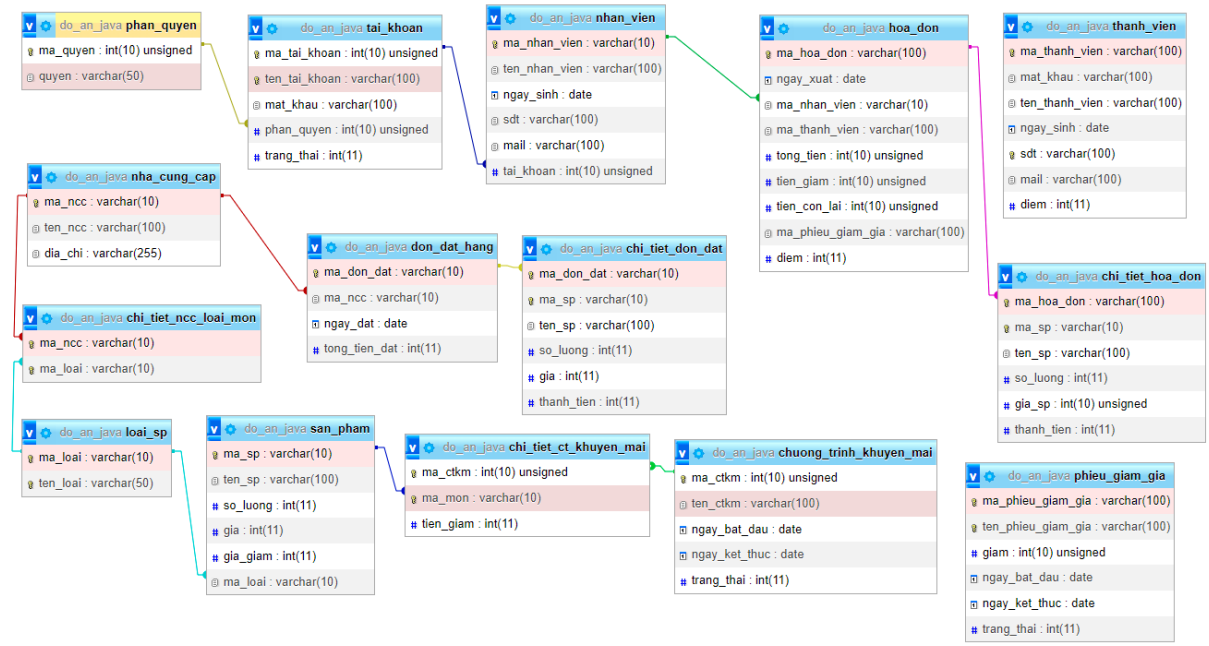
- Lớp DAL (Data Access Layer) có nhiệm vụ giao tiếp với hệ quản trị CSDL như thực hiện các công việc liên quan đến lưu trữ và truy vấn dữ liệu (tìm kiếm, thêm, xóa, sửa,…).

2.3. Mô hình BFD



Hình 2.3. Mô hình BFD

2.4. Mô hình dữ liệu quan hệ



Hình 2.4. Mô hình dữ liệu quan hệ

II. Lập kế hoạch

1. Mục đích của việc lập kế hoạch

- ❖ Xác định những thông tin dự án và các phần dự án cần được kiểm thử.
- ❖ Nhận dạng các chiến lược được dùng để kiểm tra và đảm bảo rằng sản phẩm thỏa mãn đặc tả thiết kế phần mềm và các yêu cầu khác về phần mềm.
- ❖ Định nghĩa các mục tiêu và phạm vi của nỗ lực kiểm thử
- ❖ Nhận dạng phương pháp luận mà đội kiểm thử sẽ dùng để thực hiện công việc kiểm thử.
- ❖ Nhận dạng phần cứng, phần mềm và các tiện ích cần cho kiểm thử
- ❖ Nhận dạng các tính chất và các chức năng sẽ được kiểm thử
- ❖ Xác định các hệ số rủi ro gây nguy hại cho việc kiểm thử
- ❖ Lập lịch kiểm thử và phân phối công việc cho mỗi thành viên tham gia.

2. Phạm vi test (test scope)

⇨ Chỉ bao gồm các chức năng và phân hệ chức năng.

3. Cách tiếp cận và chiến lược sử dụng

- ❖ Nhóm quyết định sử dụng chiến lượt kiểm thử phổ biến hiện nay là kiểm thử chức năng (functional tesring).
- ❖ Trong quá trình kiểm thử có sử dụng 2 phương pháp kiểm thử lần lượt là kiểm thử hộp trắng (white-box testing) và kiểm thử hộp đen (black-box testing).
- ❖ Với phương pháp kiểm thử hộp trắng (white-box testing) nhóm sử dụng kỹ thuật kiểm thử dòng điều khiển để tiến hành kiểm thử cấu trúc một chức năng của hệ thống (kiểm thử chức năng, phân hệ chức năng).
- ❖ Với phương pháp kiểm thử hộp đen (black-box testing) nhóm sử dụng kỹ thuật bảng quyết định để thiết kế các test-case dùng để kiểm thử chức năng của phần mềm mà không cần đến mã nguồn của phần mềm.
- ❖ Việc kiểm thử hoàn toàn bằng thủ công thông qua những test case trong quá trình thiết kế mà không thông qua bất cứ công cụ automatic test nào.

4. Test tool

ID	Hoạt động	Tools	Version
1	Quản lý hoạt động kiểm thử	Word	Microsoft app for enterprise, 2016, 2010
2	Vẽ và quản lý bảng quyết định	Excel	Microsoft app for enterprise, 2016, 2010
3	Vẽ CFG	Drawio	14.6.13
4	Quản lý mã nguồn + Chạy phần mềm + Test	Netbeans IDE	13, 14, 15

5. Rủi ro và khó khăn

- ❖ Việc kiểm soát tốt công cụ automatic testing để tiến hành test gặp nhiều khó khăn.
- ❖ Vì chức năng của hệ thống phần mềm là tương đối lớn nên việc test toàn bộ chức năng hệ thống tốn không ít thời gian và công sức.

- ❖ Trong quá trình test sẽ phát sinh hiểu sai luồng chức năng nên việc test sẽ gặp nhiều khó khăn.
- ❖ Việc một số chức năng lập đi lặp lại một vài nghiệp vụ như thêm, xóa, sửa, load khiến cho việc test qua những chức năng có nghiệp vụ này mất nhiều thời gian trong khi chỉ cần kiểm thử 1 chức năng có vài nghiệp vụ tương tự và bỏ qua những chức năng có nghiệp vụ đó.

III. Phân tích, thiết kế và triển khai test case

- ❖ Tiến hành phân tích từng chức năng một để thiết kế từng test case

1. Đăng nhập

Đặc tả: Để truy cập vào hệ thống, nhân viên cần phải có một tài khoản để đăng nhập vào hệ thống, tài khoản đó sẽ được quản lý hoặc cửa hàng trưởng cấp cho.

1.1. Kiểm thử hộp đen

- ❖ Thành phần: bao gồm 2 Textfields là tài khoản và mật khẩu, 2 button là đăng nhập và thoát.

Tên trường	Bắt buộc	Yêu cầu	Ghi chú
Tài khoản	Có	Không được bỏ trống	
Mật khẩu	Có	Không được bỏ trống	

- ❖ Các trường hợp kiểm thử (bảng quyết định): BangQuyetDinh_DangNhap_WhiteBoxTesting.xlsx
- ❖ Test case:

ID	Test step	Test data	Expected results	A result	Comment
1	- Người dùng focus vào	Tài khoản: admin Mật khẩu: admin	Truy cập thành công vào hệ thống,	Pass	

	Textfield tài khoản, sau đó nhập thông tin tài khoản. - Người dùng focus vào Textfield mật khẩu, sau đó nhập thông tin mật khẩu. - Người dùng click vào button đăng nhập.		giao diện bán hàng xuất hiện		
2		Tài khoản: admin Mật khẩu: 1	Truy cập thành công vào hệ thống, giao diện bán hàng xuất hiện	Fail	
3		Tài khoản: admin Mật khẩu: 1	Truy cập thất bại, hệ thống thông báo sai tài khoản hoặc mật khẩu	Pass	

❖ Test report

Số lượng test case	3
Số test case Pass	2
Số test case Fail	1
Số test case Error	0
Số test case chưa thực hiện	0
Tỉ lệ thành công	66.67%

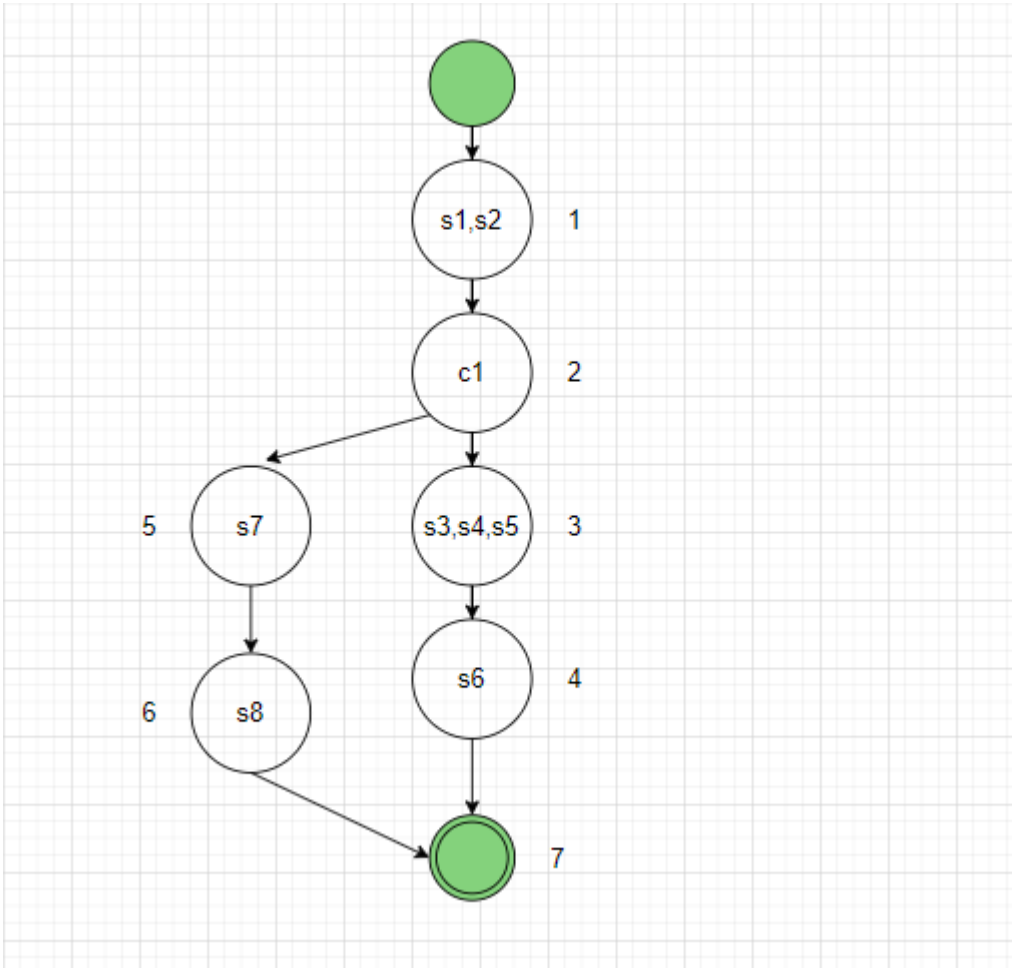
1.2. Kiểm thử hộp trắng

❖ Mã nguồn

```
    private void btnLoginActionPerformed(java.awt.event.ActionEvent evt) {
/*s1*/  String username = txtUsername.getText();
/*s2*/  String password = String.valueOf(txtPassword.getPassword());

/*c1*/  if (taiKhoanBLL.login(username, password) == true) {
/*s3*/      int ma_tk = taiKhoanBLL.getMaTkAfterLogin(username, password);
/*s4*/      int quyen = taiKhoanBLL.getTkByMaTK(ma_tk).getPhan_quyen();
/*s5*/      String ma_nv = new NhanVienBLL().getMaNvByMaTk(ma_tk);

/*s6*/      new ManageForm(ma_nv,quyen,ma_tk).setVisible(true);
        } else {
/*s7*/      JOptionPane.showMessageDialog(this, "Sai tài khoản hoặc mật khẩu", "Lỗi", JOptionPane.ERROR_MESSAGE);
/*s8*/      return;
        }
    }
```

Hình 1.2.1 Mã nguồn chức năng đăng nhập.

❖ Đồ thị dòng điều khiển cơ bản



Hình 1.2.2 Đồ thị dòng điều khiển chức năng đăng nhập.

⇨ Độ phức tạp Cyclomatic M = N + 1, với N là số nút điều kiện rẽ nhánh nhị phân.

⇨ M = 1 + 1 = 2

⇨ Có 2 đường độc lập tuyến tính cơ bản là:

- 1 – 2 – 3 – 4 – 7
- 1 – 2 – 5 – 6 – 7

❖ Test case:

ID	Basis path	Data	Expected results
1	1-2-3-4-7	Tài khoản: admin Mật khẩu: admin (Tài khoản và mật khẩu đúng)	Chuyển hướng làm việc sang FormManage.
2	1-2-5-6-7	Tài khoản: admin Mật khẩu: 1 (Tài khoản và mật khẩu sai)	Dialog hiện lên thông báo người dùng nhập sai tài khoản hoặc mật khẩu.

❖ Test report

Số lượng test case	2
Số test case Pass	1
Số test case Fail	1
Số test case Error	0
Số test case chưa thực hiện	0
Tỉ lệ thành công	50%

2. Bán hàng

Đặc tả : Nhân viên bán hàng sẽ dựa theo những vật phẩm khách hàng đã chọn để tiến hành thanh toán cho khách hàng. Khách hàng được quyền sử dụng voucher hoặc đăng ký thành viên để đổi điểm.

2.1. Kiểm thử hộp đen

❖ Thành phần: 1 Combobox dựa trên phân loại sản phẩm để người bán hành dễ dàng tìm kiếm sản phẩm, 1 Button thêm sản phẩm, 1 Button xóa sản phẩm, 1 Button apply voucher, 1 Button xác nhận mã thành viên, 1 Button đổi điểm thành viên , 1 Button xem thông tin thẻ thành viên, 1 Button thanh toán, 1 Button xem lại các tác nhân giảm giá, 1 TextField tổng tiền, 1 TextField số tiền giảm, 1 TextField Phải thanh toán (đã giảm giá), 1 TextField Voucher, 1 TextField Mã thành viên, 1 Combo box bước nhảy, 1 Spinner điểm.

Tên trường	Bắt buộc	Yêu cầu	Khác
Các sản phẩm đã mua	Có	Sản phẩm vẫn còn trong kho	
Tổng tiền	Có	Giá trị -=0	
Giảm	Có	Giá trị -=0	
Phải thanh toán	Có	Giá trị -=0	
Mã Voucher	Không	Đã tồn tại trong DB	
Mã thành viên	Không	Đã tồn tài trong DB	
Mật khẩu thành viên	Có	Đã nhập mã thành viên	
Điểm thành viên	Không	Thành viên phải có đủ số điểm	

❖ Các trường hợp kiểm thử (bảng quyết định):
BangQuyetDinh_BanHang_WhiteBoxTesting.xlsx

❖ Test case

ID	Test Step	Test data	Expected Result	A Result
1	- Chọn những sản phẩm với số lượng do khách hàng muốn thanh toán. - Nhập voucher (nếu có) - Nhập mã thành viên (nếu có) -Nhập mật khẩu thành viên -Nhập điểm thành viên muốn quy đổi	Chọn sản phẩm với số lượng hợp lệ (còn trong kho) Voucher: PGG01 Mã Thành Viên:TV01 Mật khẩu Thành Viên :1 Điểm:20	Thanh toán thành công với tổng số tiền đã giảm so với giá gốc (giảm được 2 lần voucher và điểm thành viên)	Pass
2		Chọn sản phẩm với số lượng KHÔNG hợp lệ (không còn trong kho) Voucher: PGG01 Mã Thành Viên:TV01 Mật khẩu Thành Viên :1 Điểm:20	Xuất hiện thông báo đã hết hàng	Pass
3		Chọn sản phẩm với số lượng hợp lệ (còn trong kho) Voucher:PGG03 (QUÁ THỜI HẠN) Mã Thành Viên:TV01 Mật khẩu Thành Viên :1	Thanh toán thành công với tổng số tiền giảm thông qua voucher và điểm thành viên	Fail

		Điểm:20		
4		Chọn sản phẩm với số lượng hợp lệ (còn trong kho) Voucher:PGG03 (QUÁ THỜI HẠN) Mã Thành Viên:TV01 Mật khẩu Thành Viên :1 Điểm:20	Thanh toán thành công với tổng số tiền giảm thông qua điểm thành viên	Pass
5		Chọn sản phẩm với số lượng hợp lệ (còn trong kho) Voucher:PGG01 Mã Thành Viên:TV01 Mật khẩu Thành Viên :2 Điểm:20	Thanh toán thành công với tổng số tiền giảm thông qua voucher và điểm thành viên	Fail
6		Chọn sản phẩm với số lượng hợp lệ (còn trong kho) Voucher:PGG01 Mã Thành Viên:TV01 Mật khẩu Thành Viên :2 Điểm:20	Xuất hiện thông báo sai Mật khẩu thành viên	Pass

7		Chọn sản phẩm với số lượng hợp lệ (còn trong kho) Voucher:PGG01 Mã Thành Viên:DV03 (Không tồn tại) Mật khẩu Thành Viên :1 Điểm:20	Thanh toán thành công với tổng số tiền giảm thông qua voucher và điểm thành viên	Fail
8		Chọn sản phẩm với số lượng hợp lệ (còn trong kho) Voucher:PGG01 Mã Thành Viên:TV01 Mật khẩu Thành Viên :1 Điểm:100000	Thanh toán thành công với tổng số tiền giảm thông qua voucher và điểm thành viên	Fail
9		Chọn sản phẩm với số lượng hợp lệ (còn trong kho) Voucher:PGG01 Mã Thành Viên:TV01 Mật khẩu Thành Viên :1 Điểm:100000	Xuất hiện thông báo điểm không đủ	Pass

❖ Test report

Số lượng test case	9

Số test case Pass	5
Số test case Fail	4
Số test case Error	0
Số test case chưa thực hiện	0
Tỉ lệ thành công	55.56%

2.2. Kiểm thử hộp trắng

❖ Mã nguồn

- Thêm sản phẩm vào giỏ

```
private void btnAddFoodActionPerformed(java.awt.event.ActionEvent evt) {

    int select = tblFoodManage.getSelectedRow();
    if(select >= 0){
        int soLuongMua = (int) spnCount.getValue();
        String maMon  = (String) tblFoodManage.getValueAt( row: select,  column: 0);
        if(model_1.getRowCount() == 0){
            MonAnDTO monAnDTO = new MonAnBLL().getFoodByMaMon(maMon);
            model_1.addRow(new Object[]{
                monAnDTO.getMa_mon(),
                monAnDTO.getTen_mon(),
                monAnDTO.getGia(),
                soLuongMua,
                (int)(monAnDTO.getGia()*soLuongMua)
            });
            model_1.fireTableDataChanged();

            tinhtong();
            setGiaThanhToan();
        }
        else{
            for(int i = 0 ; i < model_1.getRowCount() ; i++){
                if(model_1.getValueAt( row: i,  column: 0).equals( obj: maMon)){
                    int soluong = ((int) model_1.getValueAt( row: i, column: 3)+ soLuongMua);
                    int thanhtien = (int)(model_1.getValueAt( row: i, column: 2))*soluong;
                    model_1.setValueAt( aValue: soluong,  row: i, column: 3);
                    model_1.setValueAt( aValue: thanhtien,  row: i, column: 4);
```

Hình 2.2.1. Mã nguồn thêm sản phẩm vào giỏ.

```
                tinhtong();
                setGiaThanhToan();

                int soLuongConLai = ((int)model.getValueAt( row: select, column: 3)-(int)model_1.getValueAt( row: i, column: 3));
                int soLuongTru = (int)model_1.getValueAt( row: i, column: 3);
                System.out.println( x: soLuongConLai);
                SpinnerModel spinnerModel = new SpinnerNumberModel(
                    soLuongConLai == 0 ? 0 : 1,
                    0-soLuongTru,
                     maximum: soLuongConLai,
                     stepSize: 1);
                spnCount.setModel( model: spinnerModel);
                if((int)model_1.getValueAt( row: i, column: 3) == 0){
                    model_1.removeRow( row: i);
                }
                return;
            }
        }
        MonAnDTO monAnDTO = new MonAnBLL().getFoodByMaMon(maMon);
        model_1.addRow(new Object[]{
            monAnDTO.getMa_mon(),
            monAnDTO.getTen_mon(),
            monAnDTO.getGia(),
            soLuongMua,
            (int)(soLuongMua*monAnDTO.getGia())
        });
        model_1.fireTableDataChanged();
```

Hình 2.2.2. Mã nguồn thêm sản phẩm vào giỏ(tt).

```
            tinhtong();
            setGiaThanhToan();
        }
        for(int i = 0; i < model_1.getRowCount(); i++){
            if(model_1.getValueAt( row: i, column: 0).equals( obj: maMon)){
                int soLuongConLai = ((int)model.getValueAt( row: select, column: 3)-(int)model_1.getValueAt( row: i, column: 3));
                int soLuongTru = (int)model_1.getValueAt( row: i, column: 3);
                System.out.println( x: soLuongConLai);
                SpinnerModel spinnerModel = new SpinnerNumberModel(
                    soLuongConLai == 0 ? 0 : 1,
                    0-soLuongTru,
                     maximum: soLuongConLai,
                     stepSize: 1);
                spnCount.setModel( model: spinnerModel);
                if((int)model_1.getValueAt( row: i, column: 3) == 0){
                    model_1.removeRow( row: i);
                }
            }
        }
    }

}
```

Hình 2.2.3. Mã nguồn thêm sản phẩm vào giỏ(tt).

- Xóa sản phẩm khỏi giỏ

```
private void btnXoaMonActionPerformed(java.awt.event.ActionEvent evt) {
    int click = tblInfobill.getSelectedRow();
    if(click >= 0){
        String ma_sp = String.valueOf( obj: tblInfobill.getValueAt( row: click, column: 0));
        int totalPrice = Integer.parseInt( s: lblTotalPrice.getText());
        int giaSpXoa = (int) tblInfobill.getValueAt( row: click, column: 2);
        int gia_giam = Integer.parseInt( s: lblGiam.getText());
        lblTotalPrice.setText( text: String.valueOf((totalPrice-giaSpXoa)));
        lblTotalPriceAfter.setText( text: String.valueOf(totalPrice-giaSpXoa-gia_giam));
        model_1.removeRow( row: click);

    }
}
```

Hình 2.2.4. Mã nguồn xóa sản phẩm khỏi giỏ.

- Xác nhận mã giảm giá

```
private void btnXacNhanMaActionPerformed(java.awt.event.ActionEvent evt) {
    String ten_giam_gia = txtMaGiamGia.getText();
    if(tblInfobill.getRowCount() > 0){
        if(ma_phieu_giam != null){
            JOptionPane.showMessageDialog( parentComponent: this, message: "Chỉ được sử dụng 1 Voucher cho một hóa đơn", title: "Lỗi", messageType: JOptionPane.ERROR_MESSA
            return;
        }
        PhieuGiamGiaDTO phieuGiamGiaDTO = new PhieuGiamGiaBLL().getPGGbyTenPGG( ten_PGG: ten_giam_gia);
          System.out.println(phieuGiamGiaDTO);
        if(phieuGiamGiaDTO != null){
            if(new PhieuGiamGiaBLL().checkDate( x: phieuGiamGiaDTO)){
                if(phieuGiamGiaDTO.getTrang_thai() == 1){
                    int kq = JOptionPane.showConfirmDialog( parentComponent: this, message: "Mỗi mã chỉ sử dụng được một lần \nBạn có thật sự muốn sử dụng mã này",
                      System.out.println(kq);
                    if(kq == 0){
                      System.out.println(phieuGiamGiaDTO);
                    giam_theo_ma = phieuGiamGiaDTO.getGiam();
                    setGiaThanhToan();
                    ma_phieu_giam = phieuGiamGiaDTO.getMa_phieu_giam_gia();
                    new PhieuGiamGiaBLL().updateTrangThai(ma_phieu_giam);
                    }
                }
                else{
                    JOptionPane.showMessageDialog( parentComponent: this, message: "Voucher đã được sử dụng", title: "Lỗi", messageType: JOptionPane.ERROR_MESSAGE);
                }
            }
            else{
```

Hình 2.2.5. Mã nguồn xác nhận mã giảm giá.

```
                JOptionPane.showMessageDialog( parentComponent: this, message: "Phiếu đã hết hạn sử dụng", title: "Lỗi", messageType: JOptionPane.ERROR_MESSAGE);
                return;
            }
        }
        else{
            JOptionPane.showMessageDialog( parentComponent: this, message: "Không tồn tại hoặc sai Voucher", title: "Lỗi", messageType: JOptionPane.ERROR_MESSAGE);
            return;
        }
    }
    else{
        JOptionPane.showMessageDialog( parentComponent: this, message: "Chưa có sản phẩm nào được chọn", title: "Lỗi", messageType: JOptionPane.ERROR_MESSAGE);
        return;
    }
}
```

Hình 2.2.6. Mã nguồn xác nhận mã giảm giá(tt).

- Xác nhận thành viên

```
private void btnXacNhanTVActionPerformed(java.awt.event.ActionEvent evt) {
    String ma_tv = txtMaTV.getText();
    if(new ThanhVienBLL().checkMaThanhVien( ma_thanh_vien: ma_tv)){
        String mat_khau = JOptionPane.showInputDialog( parentComponent: this, message: "Mã xác nhận (mật khẩu):", title: "Yêu cầu nhập mật khẩu", messageType: JOptionPane.
        if(new ThanhVienBLL().checkMatKhau(mat_khau)){

            //lưu ma thành viên vào piến static
            ma_thanh_vien = ma_tv;
            txtMaTV.setText( t: ma_thanh_vien);

            //form thanh vien
            new ThanhVienForm(ma_thanh_vien).setVisible( b: true);

            //thiết lập điểm theo mã thành viên
            ThanhVienDTO thanhVienDTO = new ThanhVienBLL().getTvByMaTV(ma_thanh_vien);
            diem =thanhVienDTO.getPoint();
            int buocNhay = Integer.parseInt((String)cbxbuocNhay.getSelectedItem());
            SpinnerModel spinnerModel = new SpinnerNumberModel(
                    value: 0, minimum: 0, maximum: diem, stepSize: buocNhay
            );
            spnDiem.setModel( model: spinnerModel);
        }
        else{
            JOptionPane.showMessageDialog( parentComponent: this, message: "Sai mật khẩu", title: "Sai thông tin", messageType: JOptionPane.ERROR_MESSAGE);
        }
    }
    else{
        JOptionPane.showMessageDialog( parentComponent: this, message: "Mã thành viên không đúng",  title: "Sai thông tin", messageType: JOptionPane.ERROR_MESSAGE);
    }
```

Hình 2.2.7. Mã nguồn xác nhận thành viên.

- Thanh toán

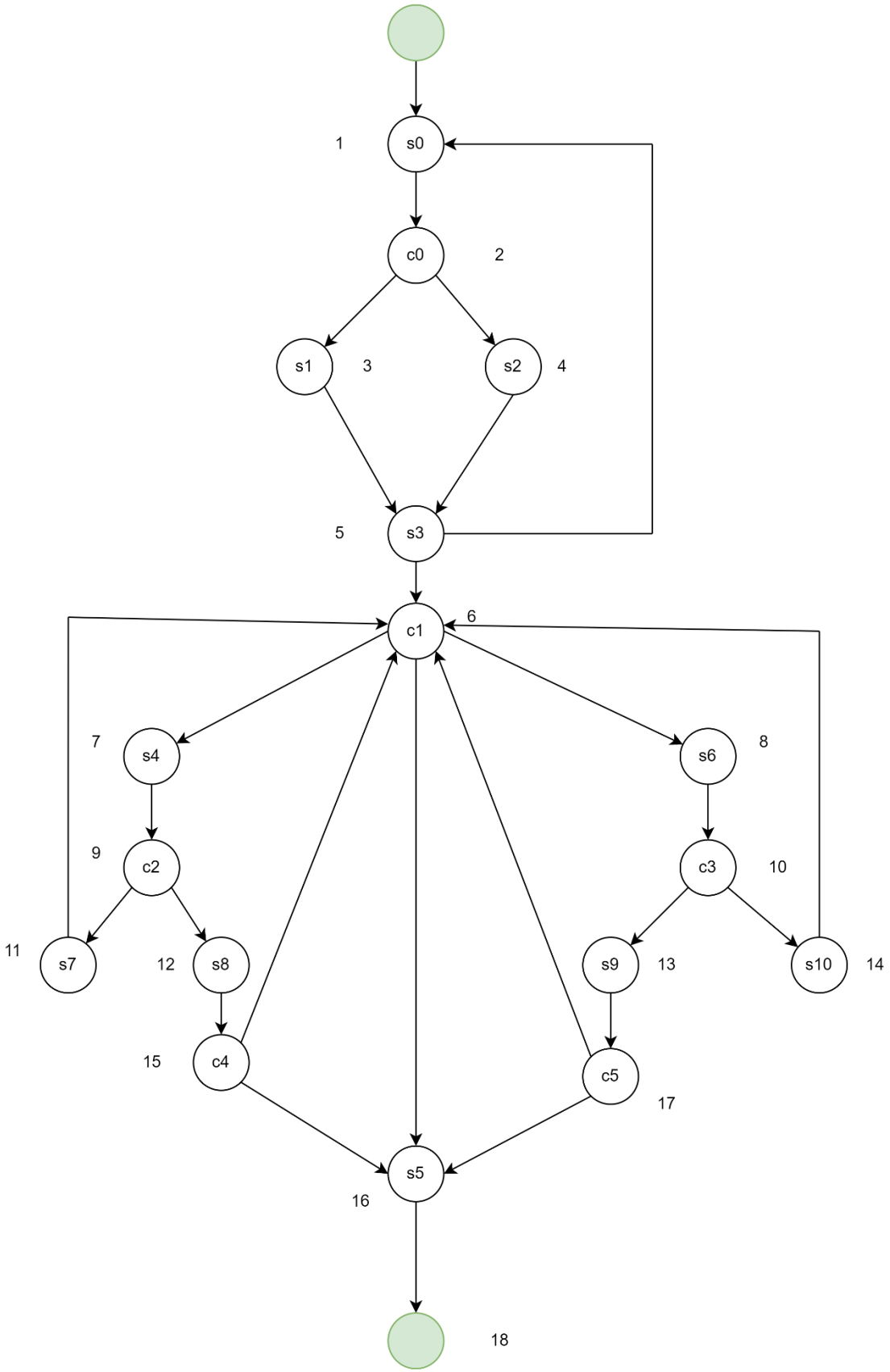
```
private void btnThanhToanActionPerformed(java.awt.event.ActionEvent evt) {
    int kq = JOptionPane.showConfirmDialog( parentComponent: this, message: "Xác nhận thanh toán", title: "Thông báo",  optionType: JOptionPane.YES_OPTIO
    if(kq ==0){
        HoaDonBLL hdbll = new HoaDonBLL();
        CTHoaDonBLL cthdbll = new CTHoaDonBLL();

        long millis=System.currentTimeMillis();
        Date date = new Date( date: millis);

        String ma_hd = "HD"+hdbll.MaMoi();
        String[] hd = {
            ma_hd,//arr[0]
            ma_nv,//arr[1]
            ma_thanh_vien,//arr[2],
            lblTotalPrice.getText(),//arr[3]
            lblGiam.getText(),//arr[4]
            lblTotalPriceAfter.getText(),//arr[5]
            ma_phieu_giam,//arr[6]
            String.valueOf( i: giam_theo_diem),//arr[7]
            date.toString(),//arr[8]
        };
        hdbll.CreateHD( str: hd);
        cthdbll.taoCTHoaDonToHoaDon( model: model_1, ma_hd);

        int tong_tien = Integer.parseInt( s: lblTotalPrice.getText());
        if(ma_thanh_vien != null){
            if(tong_tien > 10000){
                String []thanhVien = {
```

Hình 2.2.8. Mã nguồn thanh toán.

```
                    ma_thanh_vien,
                    String.valueOf(tong_tien*0.1)
                };
                new  ThanhVienBLL().capNhatDiemTichLuy( str: thanhVien);
            }
        }
        JOptionPane.showMessageDialog( parentComponent: this,("Thông tin hóa đơn \nMã hóa đơn: "+ma_hd+"\n"
                + "Mã thành viên: "+ma_thanh_vien+"\n"
                + "Nhân viên thanh toán: "+ma_nv+"\n"
                + "Ngày: "+date.toString()+"\n"
                + "Tổng tiền: "+lblTotalPrice.getText()+" Đ\n"
                + "Tiền giảm: "+lblGiam.getText()+" Đ\n"
                + "Phải thanh toán: "+lblTotalPriceAfter.getText()+" Đ\n"
                + "Điểm tích lũy được: "+(int)(tong_tien*0.1)), title: "Thông tin hóa đơn", messageType: JOptionPane.INFORMATION_MESSAGE);

        sauKhiThanhToan();
    }

}
```

Hình 2.2.9. Mã nguồn thanh toán(tt).

❖ Đồ thị dòng điều khiển cơ bản



Hình 2.2.10. Đồ thị dòng điều khiển cơ bản phân hệ chức năng mua hàng.

❖ Chú thích

Trạng thái	Điều kiện
s0:chọn sản phẩm	c0:chọn thêm vào hoặc xóa sản phẩm khỏi giỏ hàng
s1: thêm sản phẩm	c1 :nhập voucher hoặc thẻ thành viên (nếu nhập voucher rồi thì có thể nhập thẻ thành viên và ngược lại) hoặc thanh toán
s2:xóa sản phẩm	c2: nhập voucher thành công hay thất bại
s3:giỏ hàng sau khi thêm hoặc xóa=-tổng tiền	c3:nhập thẻ thành viên thành công hay thất bại
s4: nhập voucher	c4: thanh toán hoặc nhập tiếp khuyến mãi còn lại (thẻ thành viên)
s5 :thanh toán	c5:thanh toán hoặc nhập tiếp khuyến mãi còn lại (voucher)
s6:nhập thẻ thành viên	
s7:nhập voucher thất bại	
s8:nhập voucher thành công	
s9:nhập thẻ thành viên thành công	
s10: nhập thẻ thành viên thất bại	

⇨ Độ phức tạp Cyclomatic M = N + 1 (Số cạnh – số nút + 2), với N là số nút điều kiện rẽ nhánh nhị phân.

⇨ M = 26 - 19 + 2 = 9

⇨ Có 9 đường độc lập tuyến tính cơ bản là:

- 1-2-3-5-6-16-18
- 1-2-3-5-1-2-4-5-6-16-18

- 1-2-3-5-6-7-9-12-15-6-8-10-13-17-16-18
- 1-2-3-5-6-7-9-11-6-8-10-13-17-16-18
- 1-2-3-5-6-8-10-13-17-6-7-9-12-15-16-18
- 1-2-3-5-6-8-10-14-6-7-9-12-15-16-18
- 1-2-4-5-6-7-9-11-6-8-10-13-17-16-18
- 1-2-4-5-6-8-10-13-17-6-7-9-12-15-16-18
- 1-2-4-5-6-8-10-14-6-7-9-12-15-16-18

❖ Test report

Số lượng test case	9
Số test case Pass	9
Số test case Fail	0
Số test case Error	0
Số test case chưa thực hiện	0
Tỉ lệ thành công	100%

3. Nhập hàng

Đặc tả: Chọn nhà cung cấp phù hợp, và tìm kiếm sản phẩm cần nhập (tìm kiếm theo mã món hoặc tên món hoặc loại món hoặc giá) và nhập số lượng sản phẩm muốn nhập để tiến hành nhập hàng.

3.1. Kiểm thử hộp đen

❖ Thành phần : 1 Combobox Nhà cung cấp , 1 Combobox loại sản phẩm , 1 TextField Số lượng, 1 Button thanh toán.

Tên Trường	Bắt buộc	Yêu cầu	Khác

Nhà cung cấp	Có	Click vào để chọn nhà cung cấp sản phẩm phù hợp	
Chọn sản phẩm	Có	Thông qua quy trình tìm kiếm để chọn lựa sản phẩm	
Số lượng sản phẩm	Có	Số lượng sản phẩm mà người quản lý đặt sẽ được cập nhật vào số lượng sản phẩm còn tồn trong kho	

- Các trường hợp kiểm thử (bảng quyết định):
  BangQuyetDinh_NhapHang_WhiteBoxTesting.xlsx (Sheet 3)
- Test case

ID	Test step	Test data	Expected Result	Result
1	- Tìm kiếm và chọn sản phẩm - Chọn nhà cung cấp - Nhập số lượng sản phẩm	- Chọn sản phẩm đã tìm kiếm: cà phê Việt - Nhà cung cấp:Công ty TNHH 1 - Số lượng :10	Nhập hàng thành công	Pass
2		- Chọn sản phẩm đã tìm kiếm: cà phê việt - Nhà cung cấp:Công ty TNHH 1 - Số lượng :-1	Nhập hàng thất bại	Pass

❖ Test report

Số lượng test case	2
Số test case Pass	2
Số test case Fail	0
Số test case Error	0
Số test case chưa thực hiện	0
Tỉ lệ thành công	100%

3.2. Kiểm thử hộp trắng

❖ Mã nguồn

- Thêm món

```
    private void btnDatActionPerformed(java.awt.event.ActionEvent evt) {

        int select = tblFood.getSelectedRow();
        if(select >= 0){
            int soLuongMua = (int) spnCount.getValue();
            String maMon  = (String) tblFood.getValueAt(select, 0);
            if(modelCTDN.getRowCount() == 0){
                MonAnDTO monAnDTO = new MonAnBLL().getFoodByMaMon(maMon);
                modelCTDN.addRow(new Object[]{
                    monAnDTO.getMa_mon(),
                    monAnDTO.getTen_mon(),
                    monAnDTO.getGia(),
                    soLuongMua,
                    (int)(monAnDTO.getGia()*soLuongMua)
                });
                modelCTDN.fireTableDataChanged();

                tinhtong();
            }
            else{
                for(int i = 0 ; i < modelCTDN.getRowCount() ; i++){
                    if(modelCTDN.getValueAt(i, 0).equals(maMon)){
                        int soluong = ((int) modelCTDN.getValueAt(i,3)+ soLuongMua);
                        int thanhtien = (int)(modelCTDN.getValueAt(i,2))*soluong;
                        modelCTDN.setValueAt(soluong, i,3);
                        modelCTDN.setValueAt(thanhtien, i,4);
                        modelCTDN.fireTableDataChanged();

                        tinhtong();

                        int soLuongTru = (int)modelCTDN.getValueAt(i,3);
//                          System.out.println(soLuongConLai);
                        SpinnerModel spinnerModel = new SpinnerNumberModel(
                            1,
                            0-soLuongTru,
                            100,
                            1);
                        spnCount.setModel(spinnerModel);
                        if((int)modelCTDN.getValueAt(i,3) == 0){
                            modelCTDN.removeRow(i);
                        }
                        return;
                    }
                }
                MonAnDTO monAnDTO = new MonAnBLL().getFoodByMaMon(maMon);
```

Hình 3.2.1. Mã nguồn chức năng thêm sản phẩm đặt hàng.

- Xóa món

```
private void btnXoaMonDNActionPerformed(java.awt.event.ActionEvent evt) {
    int click = tblCTDonNhap.getSelectedRow();
    if(click >= 0){
        int thanhTien = (int)modelCTDN.getValueAt(click,4);
        int totalPrice = Integer.parseInt(lblTotalPrice.getText());
        lblTotalPrice.setText(String.valueOf(totalPrice-thanhTien));
        modelCTDN.removeRow(click);
    }
}
```

Hình 3.2.2. Mã nguồn chức năng xóa sản phẩm đặt hàng.

- Chọn nhà cung cấp

```
private void cbxNccActionPerformed(java.awt.event.ActionEvent evt) {

    if(modelCTDN.getRowCount() == 0){
        String ten_ncc = String.valueOf(cbxNcc.getSelectedItem());
        ncc_current = ten_ncc;
        String ma_ncc = nhaCungCapBLL.getMaNccByTenNcc(ten_ncc);
        modelCbxLmOfDN = new DefaultComboBoxModel();
        cTNhaCungCapLoaiMonBLL.setTenLoaiToComboboxByMaNcc(modelCbxLmOfDN, ma_ncc);
        cbxCategory.setModel(modelCbxLmOfDN);
        return;
    }
    System.out.println(ncc_current);
    if(!String.valueOf(cbxNcc.getSelectedItem()).equals(ncc_current)){
        JOptionPane.showMessageDialog(this,"Một đơn nhập chỉ có thể nhập ở 1 nhà cung cấp","Lỗi",JOptionPane.ERROR_MESSAGE);
        cbxNcc.setSelectedItem(ncc_current);
        return;
    }

}
```

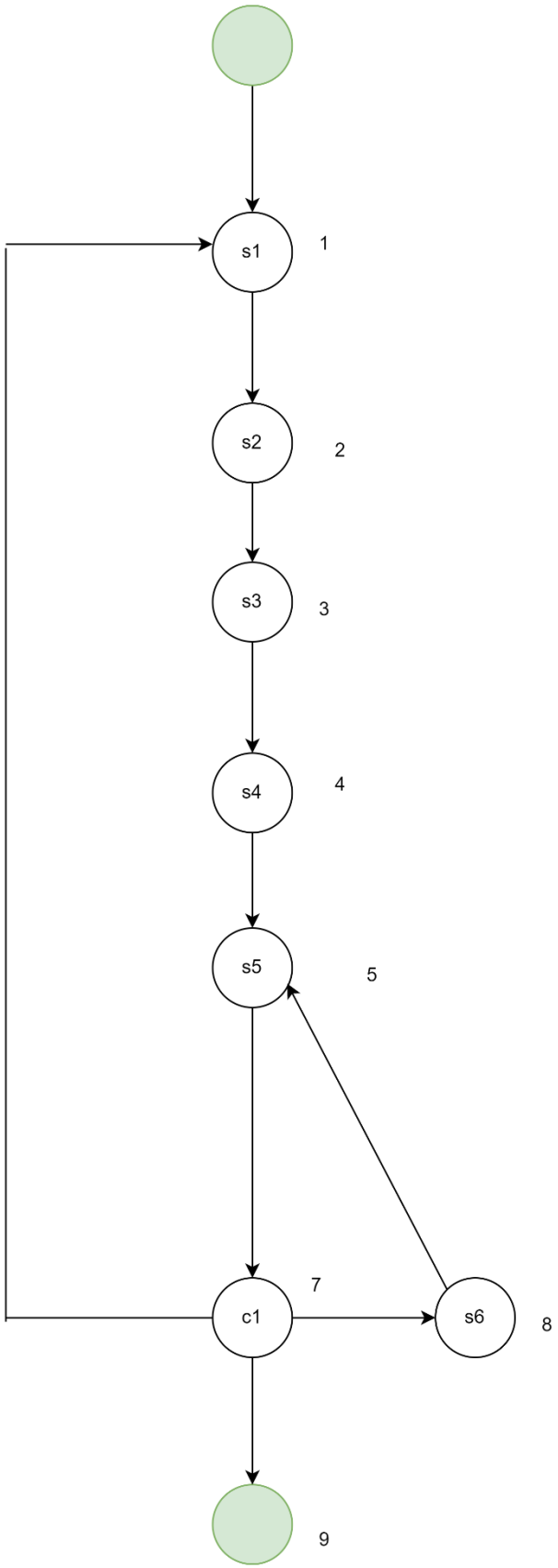
Hình 3.2.3. Mã nguồn chức năng chọn nhà cung cấp đặt hàng.

- Chọn số lượng sản phẩm

```
private void tblFoodMouseClicked(java.awt.event.MouseEvent evt) {
    int select = tblFood.getSelectedRow();
    if(select >= 0){
        String ma_mon = (String) tblFood.getValueAt(select,0);
        for(int i = 0; i < modelCTDN.getRowCount();i++){
            if(modelCTDN.getValueAt(i,0).equals(ma_mon)){
                int soluong = (int) modelCTDN.getValueAt(i,3);
                SpinnerModel spinnerModel = new SpinnerNumberModel(
                    1, // initial value
                    0-soluong, // min
                    100, // max
                    1);// step
                spnCount.setModel(spinnerModel);
                return;
            }
        }
        SpinnerModel spinnerModel = new SpinnerNumberModel(
            1, // initial value
            1, // min
            100, // max
            1);// step
        spnCount.setModel(spinnerModel);
    }
}
```

Hình 3.2.4. Mã nguồn chức năng chọn số lượng sản phẩm đặt hàng.

❖ Đồ thị dòng điều khiển cơ bản



Hình 3.2.5. Đồ thị dòng điều khiển cơ bản.

❖ Chú thích

Trạng thái	Điều kiện
s1: Tìm kiếm sản phẩm	c1: Tiếp tục tìm kiếm sản phẩm để thêm vào danh sách nhập hoặc xóa sản phẩm khỏi danh sách nhập hoặc thanh toán
s2: chọn nhà cung cấp	
s3:nhập số lượng sản phẩm	
s4: thêm sản phẩm vào danh sách nhập	
s5: Danh sách nhập sau khi được tinh chỉnh (thêm hoặc xóa)	
s6: Xóa sản phẩm khỏi danh sách nhập	

- ⇨ Độ phức tạp Cyclomatic M = N + 1 (Số cạnh – số nút + 2), với N là số nút điều kiện rẽ nhánh nhị phân.
- ⇨ M = 10 - 9 + 2 = 3
- ⇨ Có 9 đường độc lập tuyến tính cơ bản là:
  - 1-2-3-4-5-7-9
  - 1-2-3-4-5-7-8-5-7-9
  - 1-2-3-4-5-7-1-2-3-4-5-7-9

❖ Test report

Số lượng test case	3
Số test case Pass	3
Số test case Fail	0
Số test case Error	0
Số test case chưa thực hiện	0
Tỉ lệ thành công	100%

4. Quản lý nhân viên

Đặc tả: Nhân viên muốn truy cập vào hệ thống cần phải có tài khoản do quản lý hoặc cửa hàng trưởng cấp cho, và các thông tin của tài khoản sẽ do cửa hàng trưởng quản lý.

4.1. Kiểm thử hộp đen

❖ Thành phần: 7 Textfield là tìm kiếm, (tìm kiếm theo ngày sinh) từ ngày, đến ngày, tên nhân viên, ngày sinh, số điện thoại và mail; 1 Label là mã nhân viên; 5 Button là tạo mã mới, thêm, sửa, xóa, load; 2 ComboBox là tìm kiếm theo tiêu chí nào và chọn tài khoản.

Tên trường	Bắt buộc	Yêu cầu	Ghi chú
Label Mã nhân viên	Có	Không được bỏ trống, Click vào button tạo mã mới để hệ thống tự tạo mới mã NV	Dữ liệu trong trường này dùng để tạo mới, sửa, hoặc xóa tài khoản NV.
TextField Tên nhân viên	Có	Không được bỏ trống	Dữ liệu trong trường này dùng để tạo mới, sửa, hoặc xóa tài khoản NV.
TextField Ngày sinh	Có	Không được bỏ trống	Dữ liệu trong trường này dùng để tạo mới, sửa, hoặc xóa tài khoản NV.

TextField Số điện thoại	Có	Không được bỏ trống	Dữ liệu trong trường này dùng để tạo mới, sửa, hoặc xóa CTKM.
TextField Mail	Có	Không được bỏ trống	Dữ liệu trong trường này dùng để tạo mới, sửa, hoặc xóa NV.
ComboBox Loại tài khoản	Có	Không được bỏ trống, click chọn loại tài khoản	Dữ liệu trong trường này dùng để tạo mới, sửa, hoặc xóa tài khoản.
ComboBox Loại tìm kiếm	Không	Chỉ Click chọn tiêu chí tìm kiếm khi cần tìm kiếm NV, sau khi nhập thông tin cần tìm kiếm nhấn button tìm kiếm	Tìm kiếm theo mã NV, tên NV…
TextField tìm kiếm từ ngày (date from)	Không	Nhập ngày bắt đầu tìm kiếm	Tìm kiếm NV theo ngày sinh
TextField tìm kiếm đến ngày (date to)	Không	Nhập ngày kết thúc tìm kiếm	Tìm kiếm NV theo ngày, sau khi nhập ngày tìm kiếm, hệ thống từ động lọc

			ra danh sách hợp lệ

❖ Các trường hợp kiểm thử (bảng quyết đinh):

BangQuyetDinh_QL_NhanVien_WhiteBoxTesting.xlsx

❖ Test case

ID	Test Step	Test data	Expected Result	A Result
1	-Nhập mã nhân viên -Nhập tên nhân viên -Nhập ngày sinh -Nhập số điện thoại -Nhập mail -Chọn loại tài khoản -Click vào nút thêm -Nhập mã nhân viên -Nhấn nút xóa. -Thực hiện tìm kiếm thông tin nhân viên muốn chỉnh sửa -Nhập thông tin muốn chỉnh sửa -Nhấn nút sửa.	Mã nhân viên: “NV_01” Tên nhân viên: “A” Ngày sinh: “10/11/2001” Số điện thoại: “01234” Mail: “abc@gmail” Loại tài khoản: “1”	Thêm thành công	Pass
2		Mã nhân viên: “NV_01” Tên nhân viên: “A” Ngày sinh: “10/11/2001” Số điện thoại: “” Mail: “abc@gmail” Loại tài khoản: “1” (Số điện thoại không hợp lệ)	Thêm không thành công	Fail
1		Mã nhân viên: “NV_02”	Xóa thành công thông tin nhân viên	Pass
2		Mã nhân viên: NV_99 (Mã nhân viên không tồn tại).	Xóa không thành công thông tin nhân viên	Fail

1		Mã nhân viên: “NV_01” Tên nhân viên: “A”(cũ) Ngày sinh: “10/11/2001” Số điện thoại: “01234” Mail: “abc@gmail” Loại tài khoản: “1” ↓ Mã nhân viên: “NV_01” Tên nhân viên: “B” Ngày sinh: “10/11/2001” Số điện thoại: “01234” Mail: “abc@gmail” Loại tài khoản: “1”	Sửa thành công thông tin nhân viên	Pass
2		Mã nhân viên: “NV_02” ↓ Mã nhân viên : “”(mã không hợp lệ)	Sửa không thành công thông tin nhân viên	Fail

❖ Test report

Số lượng test case	6
Số test case Pass	3
Số test case Fail	3
Số test case Error	0
Số test case chưa thực hiện	0
Tỉ lệ thành công	50%

4.2. Kiểm thử hộp trắng

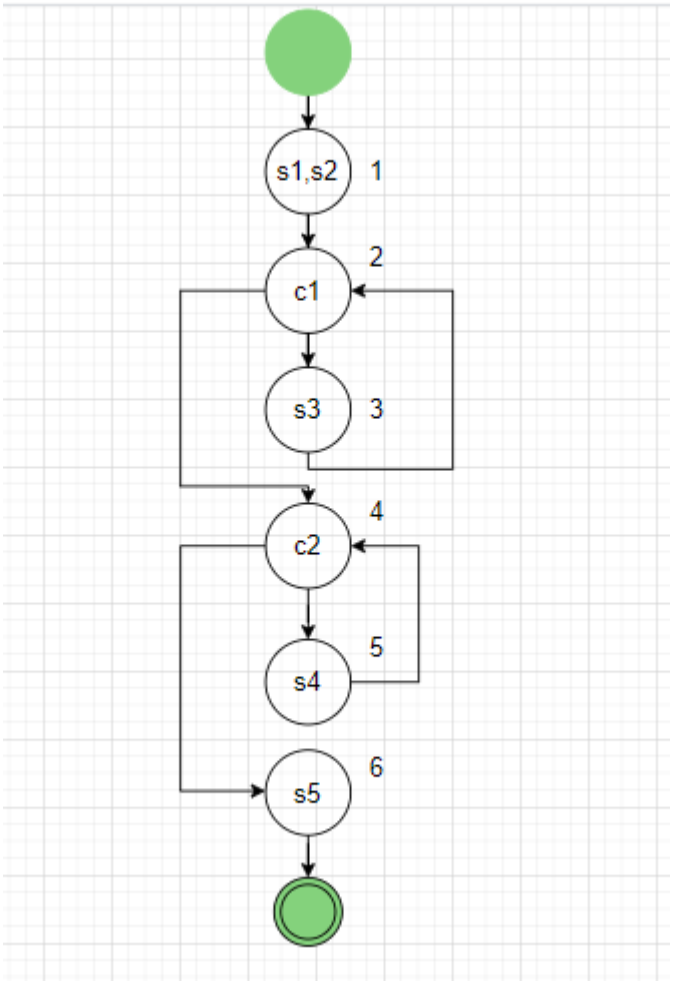
Lấy dữ liệu từ csdl và thiết lập vào danh sách hiển thị các chương trình khuyến mãi.

❖ Mã nguồn

```
    //set list lên table list
        public  void setListNV(DefaultTableModel model){
/*s1*/          this.resetListNV();
/*s2*/          this.loadListNV();

/*c1*/          while(model.getRowCount()>0){
/*s3*/              model.removeRow(row: 0);
                }

/*c2*/          for (NhanVienDTO nhanVienDTO : listNV) {
/*s4*/              model.addRow(new Object[]{
                      nhanVienDTO.getMa_nhan_vien(),
                      nhanVienDTO.getTen_nhan_vien(),
                      nhanVienDTO.getNgay_sinh(),
                      nhanVienDTO.getSdt(),
                      nhanVienDTO.getMail(),
                      nhanVienDTO.getTai_khoan()
                  });
              }
/*s5*/          model.fireTableDataChanged();
    }
```

Hình 4.2.1. Mã nguồn chức năng lấy dữ liệu từ csdl và thiết lập vào danh sách hiển thị các nhân viên.

❖ Đồ thị dòng điều khiển cơ bản



Hình 4.2.2. Đồ thị dòng diều khiển cơ bản chức năng lấy dữ liệu từ csdl và thiết lập vào danh sách hiển thị các nhân viên.

- ⇨ Độ phức tạp Cyclomatic M = N + 1, với N là số nút điều kiện rẽ nhánh nhị phân.
- ⇨ M = 2 + 1 = 3
- ⇨ Có 3 đường độc lập tuyến tính cơ bản là:
  - 1 – 2 – 4 – 6
  - 1 – 2 – 3 – 2 – 4 – 6
  - 1 – 2 – 4 – 5 – 4 – 6

❖ Test case

ID	Basis path	Data	Expected results

1	1-2-4-6	model.getRowCount() <= 0	Không cần xóa dữ liệu modelNV hiện có vì nó đang rỗng
2	1-2-3-2-4-6	model.getRowCount() > 0 listNV rỗng (listNV.size() = 0)	Reset dữ liệu trong modelNV hiện tại. List chứa dữ liệu được lấy về từ csdl rỗng nên không có dữ liệu để nạp vào model.
3	1-2-4-5-4-6	model.getRowCount() <= 0 listNV có dữ liệu (listNV.size() > 0)	Không cần xóa dữ liệu modelNV hiện có vì nó đang rỗng. Nạp dữ liệu từ listNV vào modelNV để hiển thị lên giao diện người dùng.

❖ Test report

Số lượng test case	3
Số test case Pass	3
Số test case Fail	0
Số test case Error	0
Số test case chưa thực hiện	0
Tỉ lệ thành công	100%

Thêm nhân viên, xóa nhân viên, sửa nhân viên đều tương tự nhau về đòng điều khiển cơ bản và đều có chung đường độc lập tuyến tính cơ bản.

❖ Mã nguồn lần lượt các chức năng thêm nhân viên, sửa nhân viên, xóa nhân viên.

```
        public void ThemNV(String []str){
/*s1*/          nhanVienDAL.addData(str);
/*s2*/          this.resetListNV();
/*s3*/          this.loadListNV();
        }
```

Hình 4.2.3. Mã nguồn chức năng thêm nhân viên.

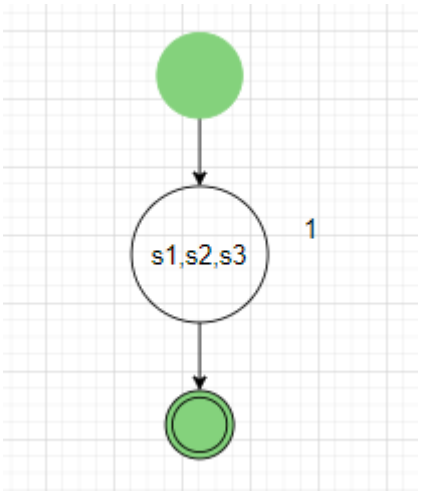
```
         public void suaNV(String []str){
/*s1*/           nhanVienDAL.updateData(str);
/*s2*/           this.resetListNV();
/*s3*/           this.loadListNV();
         }
```

Hình 4.2.4. Mã nguồn chức năng sửa nhân viên.

```
        public void delNV(String maNV){
/*s1*/          nhanVienDAL.delData(maNV);
/*s2*/          this.resetListNV();
/*s3*/          this.loadListNV();
        }
```

Hình 4.2.5. Mã nguồn chức năng xóa nhân viên.

❖ Đồ thị dòng điều khiển cơ bản



Hình 4.2.6. Đồ thị dòng điều khiển cơ bnar các chức năng thêm, xóa, sửa nhân viên.

⇨ Độ phức tạp Cyclomatic M = N + 1, với N là số nút điều kiện rẽ nhánh nhị phân.

⇨ M = 0 + 1 = 1 (với N = 0)

⇨ Đường độc lập tuyến tính cơ bản duy nhất là 1

❖ Test case chức năng thêm nhân viên

ID	Basis path	Data	Expected results
1	1	Mảng arr chứa dữ liệu là thông tin nhân viên người quản lý nhập vào, cụ thể: Arr[0]: “NV_01”; Arr[1]: “A”; Arr[2]: “10/11/2000”; Arr[3]: “012345”; Arr[4]: “123@gmail”;	Nhân viên mới được thêm vào csdl. Đồng thời reset và cập nhật lại listNV (danh sách chứa các NV được lấy xuống từ csdl).

		Arr[5]: “1”;	

❖ Test report

Số lượng test case	1
Số test case Pass	1
Số test case Fail	0
Số test case Error	0
Số test case chưa thực hiện	0
Tỉ lệ thành công	100%

❖ Test case chức năng sửa nhân viên

ID	Basis path	Data	Expected results
1	1	Mảng arr chứa dữ liệu là thông tin NV người quản lý nhập vào, cụ thể: Arr[0]: “NV_01”; Arr[1]: “B”; Arr[2]: “10/11/2001”; Arr[3]: “012345”; Arr[4]: “123@gmail”; Arr[5]: “1”;	Nhân viên được cập nhật lại ở csdl. Đồng thời reset và cập nhật lại listNV (danh sách chứa các NV được lấy xuống từ csdl).

❖ Test report

Số lượng test case	1

Số test case Pass	1
Số test case Fail	0
Số test case Error	0
Số test case chưa thực hiện	0
Tỉ lệ thành công	100%

❖ Test case chức năng xóa nhân viên

ID	Basis path	Data	Expected results
1	1	Ma_NV: NV_01 (Ma_NV là mã của nhân viên được người quản lý click vào).	Nhân viên được chọn bị xóa khỏi csdl. Đồng thời reset và cập nhật lại listNV (danh sách chứa các NV được lấy xuống từ csdl).

❖ Test report

Số lượng test case	1
Số test case Pass	1
Số test case Fail	0
Số test case Error	0
Số test case chưa thực hiện	0
Tỉ lệ thành công	100%

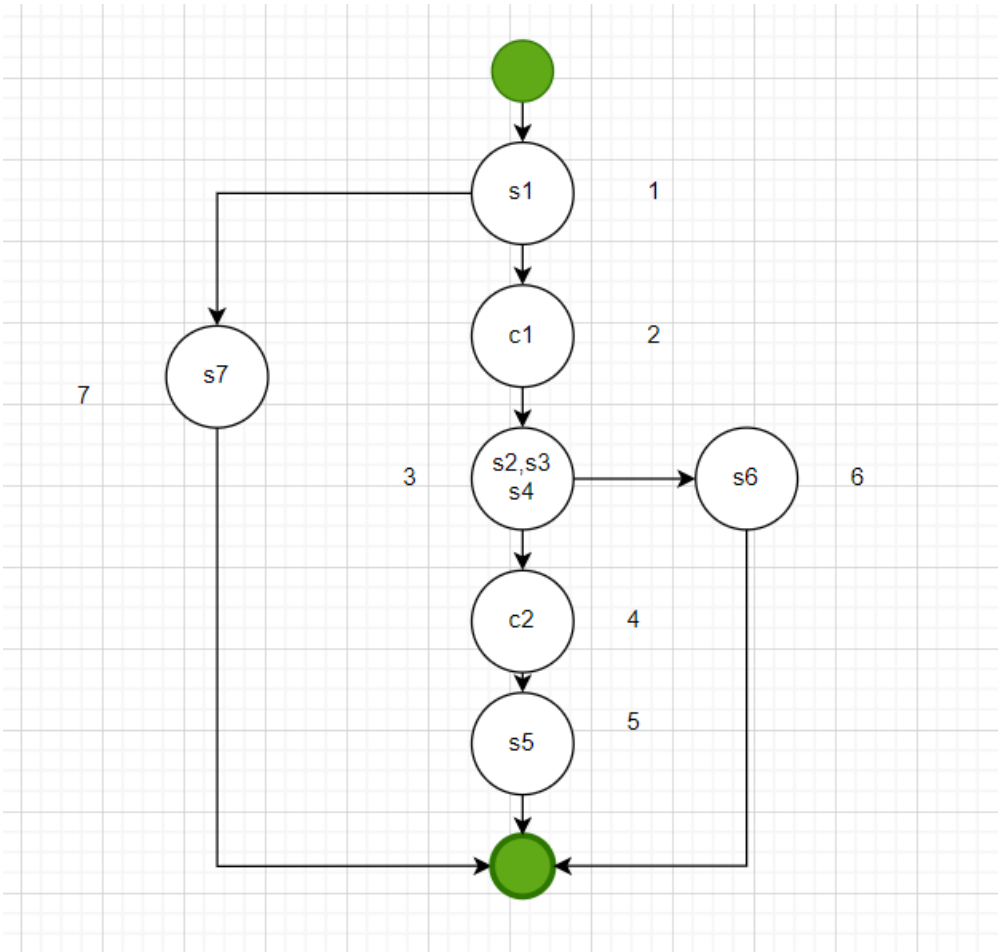
Tạo mới mã nhân viên

❖ Mã nguồn

```
    public String maMoi() {
/*s1*/          String ma = nhanVienDAL.getMaMax();

/*c1*/          if(ma!=null){
/*s2*/              int size = ma.length();


/*s3*/              int maHienTai = Integer.parseInt(ma.substring(beginIndex: 2,size));
/*s4*/              int maMoi = maHienTai + 1;
/*c2*/              if(maMoi < 10){
/*s5*/                  return "0" + maMoi;
                    }
/*s6*/              return maMoi+"";
                }
/*s7*/          return "01";
    }
```

Hình 4.2.7. Mã nguồn chức năng tạo mới mã nhân viên.

❖ Đồ thị dòng điều khiển cơ bản



Hình 4.2.7. Đồ thị dòng điều khiển cơ bản chức năng tạo mới mã nhân viên.

⇨ Độ phức tạp Cyclomatic M = N + 1, với N là số nút điều kiện rẽ nhánh nhị phân.

⇨ M = 2 + 1 = 3 (với N = 2)

⇨ Đường độc lập tuyến tính cơ bản:

- 1 – 7
- 1 – 2 – 3 – 6
- 1 – 2 – 3 – 4 – 5

❖ Test case

ID	Basis path	Data	Expected results
1	1 – 7	maHienTai: -1 (MaHienTai được lấy từ csdl)	Phương thức trả về một mã mới là 01
2	1 – 2 – 3 – 6	maHienTai: 11 (MaHienTai được lấy từ csdl)	Phương thức trả về mã mới là 11 (maHienTai + 1)
3	1 – 2 – 3 – 4 – 5	maMoi: 4	Phương thức trả về mã mới là 04

❖ Test report

Số lượng test case	3
Số test case Pass	3
Số test case Fail	0
Số test case Error	0
Số test case chưa thực hiện	0

Tỉ lệ thành công	100%

Các phương thức tìm kiếm nhân viên theo mã và hiển thị ra giao diện, tìm kiếm và hiển thị theo tên, tìm kiếm và hiển thị theo ngày sinh, tìm kiếm và hiển thị theo số điện thoại, tìm kiếm và hiển thị theo mail, tìm kiếm và hiển thị theo tài khoản đều tương tự nhau về đồ thị dòng điều kiển và đều có chung số đường độc lập cơ bản. Chỉ khác về test case.

❖ Mã nguồn

```
    public void timkiemNVTheoMa(DefaultTableModel model, String ma_nv) {
/*s1*/          this.resetListNV();
/*s2*/          this.loadListNV();

/*c1*/          while(model.getRowCount() > 0){
/*s3*/              model.removeRow(row: 0);
        }
/*c2*/          for (NhanVienDTO nhanVienDTO : listNV) {
/*c3*/              if(nhanVienDTO.getMa_nhan_vien().contains(ma_nv)){
/*s4*/                  model.addRow(new Object[]{
                        nhanVienDTO.getMa_nhan_vien(),
                        nhanVienDTO.getTen_nhan_vien(),
                        nhanVienDTO.getNgay_sinh(),
                        nhanVienDTO.getSdt(),
                        nhanVienDTO.getMail(),
                        nhanVienDTO.getTai_khoan(),
                    });
                }
            }
/*s5*/          model.fireTableDataChanged();
    }
```

Hình 4.2.8. Mã nguồn chức năng tìm kiếm nhân viên theo mã nhân viên

```
    public void timkiemNVTheoTen(DefaultTableModel model, String ten_nv) {
/*s1*/          this.resetListNV();
/*s2*/          this.loadListNV();

/*c1*/          while(model.getRowCount() > 0){
/*s3*/              model.removeRow(row: 0);
        }
/*c2*/          for (NhanVienDTO nhanVienDTO : listNV) {
/*c3*/              if(nhanVienDTO.getTen_nhan_vien().contains(ten_nv)){
/*s4*/                  model.addRow(new Object[]{
                        nhanVienDTO.getMa_nhan_vien(),
                        nhanVienDTO.getTen_nhan_vien(),
                        nhanVienDTO.getNgay_sinh(),
                        nhanVienDTO.getSdt(),
                        nhanVienDTO.getMail(),
                        nhanVienDTO.getTai_khoan(),
                    });
            }
        }
/*s5*/          model.fireTableDataChanged();
    }
```

Hình 4.2.9. Mã nguồn chức năng tìm kiếm nhân viên theo tên nhân viên

```
    public void timkiemNVTheoSDT(DefaultTableModel model, String sdt) {
/*s1*/          this.resetListNV();
/*s2*/          this.loadListNV();

/*c1*/          while(model.getRowCount() > 0){
/*s3*/              model.removeRow(row: 0);
        }
/*c2*/          for (NhanVienDTO nhanVienDTO : listNV) {
/*c3*/              if(nhanVienDTO.getSdt().contains(sdt)){
/*s4*/                  model.addRow(new Object[]{
                        nhanVienDTO.getMa_nhan_vien(),
                        nhanVienDTO.getTen_nhan_vien(),
                        nhanVienDTO.getNgay_sinh(),
                        nhanVienDTO.getSdt(),
                        nhanVienDTO.getMail(),
                        nhanVienDTO.getTai_khoan(),
                    });
            }
        }
/*s5*/          model.fireTableDataChanged();
    }
```

Hình 4.2.10. Mã nguồn chức năng tìm kiếm nhân viên theo số điện thoại nhân viên

```
        public void timkiemNVTheoNS(DefaultTableModel model, Date ngayPd, Date ngayKt) {
/*s1*/        this.resetListNV();
/*s2*/         this.loadListNV();

/*c1*/         while(model.getRowCount() > 0){
/*s3*/             model.removeRow(row: 0);
            }

/*c2*/         for (NhanVienDTO nhanVienDTO : listNV) {
/*c3*/             if(nhanVienDTO.getNgay_sinh().before(ngayKt) && nhanVienDTO.getNgay_sinh().after(ngayPd)){
/*s4*/                model.addRow(new Object[]{
                        nhanVienDTO.getMa_nhan_vien(),
                        nhanVienDTO.getTai_khoan(),
                        nhanVienDTO.getNgay_sinh(),
                        nhanVienDTO.getSdt(),
                        nhanVienDTO.getMail(),
                        nhanVienDTO.getTai_khoan(),
                    });
                }
            }
/*s5*/         model.fireTableDataChanged();
        }
```

Hình 4.2.11. Mã nguồn chức năng tìm kiếm nhân viên theo ngày sinh nhân viên

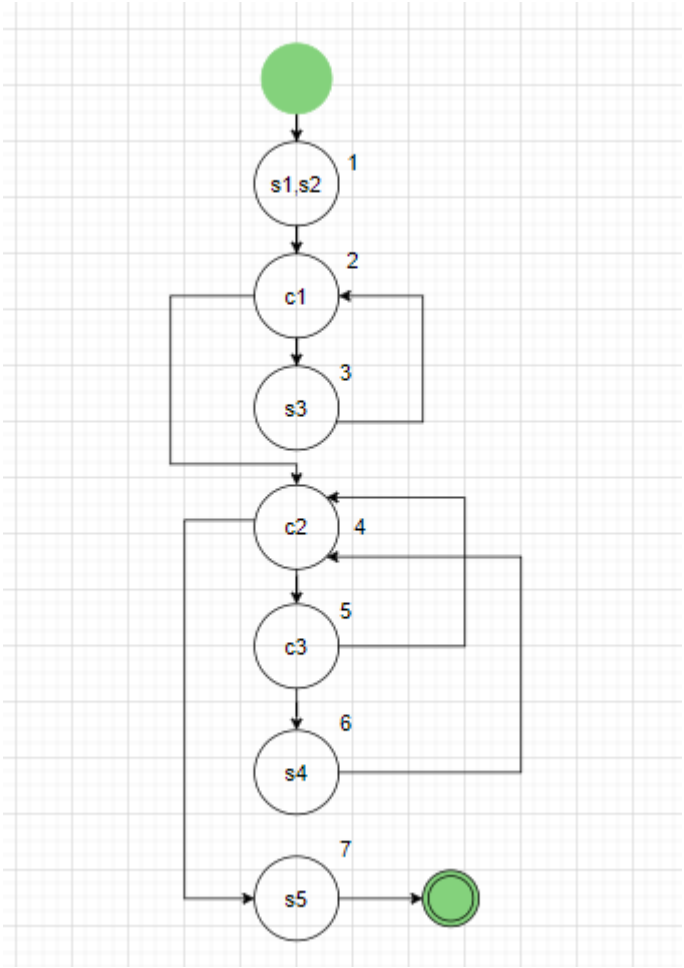
```
    public void timkiemNVTheoMail(DefaultTableModel model, String mail) {
/*s1*/        this.resetListNV();
/*s2*/        this.loadListNV();

/*c1*/        while(model.getRowCount() > 0){
/*s3*/            model.removeRow(row: 0);
            }
/*c2*/        for (NhanVienDTO nhanVienDTO : listNV) {
/*c3*/            if(nhanVienDTO.getMail().contains(mail)){
/*s4*/                model.addRow(new Object[]{
                        nhanVienDTO.getMa_nhan_vien(),
                        nhanVienDTO.getTen_nhan_vien(),
                        nhanVienDTO.getNgay_sinh(),
                        nhanVienDTO.getSdt(),
                        nhanVienDTO.getMail(),
                        nhanVienDTO.getTai_khoan(),
                    });
                }
            }
/*s5*/        model.fireTableDataChanged();
    }
```

Hình 4.2.12. Mã nguồn chức năng tìm kiếm nhân viên theo email nhân viên

```
    public void timkiemNVTheoTK(DefaultTableModel model, String tk) {
/*s1*/          this.resetListNV();
/*s2*/          this.loadListNV();

/*c1*/          while(model.getRowCount() > 0){
/*s3*/              model.removeRow(row: 0);
        }
/*c2*/          for (NhanVienDTO nhanVienDTO : listNV) {
/*c3*/              if(String.valueOf(nhanVienDTO.getTai_khoan()).contains(tk)){
/*s4*/                  model.addRow(new Object[]{
                        nhanVienDTO.getMa_nhan_vien(),
                        nhanVienDTO.getTai_khoan(),
                        nhanVienDTO.getNgay_sinh(),
                        nhanVienDTO.getSdt(),
                        nhanVienDTO.getMail(),
                        nhanVienDTO.getTai_khoan(),
                    });
                }
            }
/*s5*/          model.fireTableDataChanged();
    }
```

Hình 4.2.12. Mã nguồn chức năng tìm kiếm nhân viên theo tài khoản nhân viên

❖ Đồ thị dòng điều khiển cơ bản



Hình 4.2.13. Đồ thị dòng điều khiển cơ bản các chức năng kể trên.

⇨ Độ phức tạp Cyclomatic M = N + 1, với N là số nút điều kiện rẽ nhánh nhị phân.

⇨ M = 3 + 1 = 4(với N = 3)

⇨ Đường độc lập tuyến tính cơ bản:

- 1 – 2 – 4 – 7
- 1 – 2 – 3 – 2 – 4 – 7
- 1 – 2 – 4 – 5 – 6 – 4 – 7
- 1 – 2 – 4 – 5 – 4 – 7

❖ Test case

ID	Basis path	Data	Expected results

1	1-2-4-7	defaultTableModel: modelNV (model của nhân viên) search: “NV_01” (search theo mã); “A” (search theo tên); “10/11/2000” (seach theo ngày sinh), “012345” (seach theo số điện thoại), “123@gmail” (seach theo mail), “1” (seach theo tài khoản) modelNV.getRowCount() <= 0	Cập nhật lại giao diện nhân viên, (Do listNV rỗng nên giao diện NV vẫn không cập nhật)
2	1-2-3-2-4-7	defaultTableModel: modelNV (model của nhân viên) search: “NV_01” (search theo mã); “A” (search theo tên); “10/11/2000” (seach theo ngày sinh), “012345” (seach theo số điện thoại), “123@gmail” (seach theo mail), “1” (seach theo tài khoản) modelNV.getRowCount() > 0	Reset lại modelNV trước đó và cập nhật lại giao diện NV, (Do listNV rỗng nên giao diện NV vẫn không cập nhật)
3	1-2-4-5-6-4-7	defaultTableModel: modelNV	Reset lại modelNV trước đó.

		(model của nhân viên) search: “NV_01” (search theo mã); “A” (search theo tên); “10/11/2000” (seach theo ngày sinh), “012345” (seach theo số điện thoại), “123@gmail” (seach theo mail), “1” (seach theo tài khoản) modelNV.getRowCount() > 0	NV được tìm thấy và nạp vào modelNV. Cập nhật lại giao diện NV
4	1-2-4-5-4-7	defaultTableModel: modelNV (model của nhân viên) search: “NV_01” (search theo mã); “A” (search theo tên); “10/11/2000” (seach theo ngày sinh), “012345” (seach theo số điện thoại), “123@gmail” (seach theo mail), “1” (seach theo tài khoản) modelNV.getRowCount() > 0	Reset lại modelNV trước đó. NV không tìm thấy. Cập nhật lại giao diện NV

❖ Test report

Số lượng test case	4
Số test case Pass	3

Số test case Fail	1
Số test case Error	0
Số test case chưa thực hiện	0
Tỉ lệ thành công	75%

5. Quản lý tài khoản

Đặc tả: Để đăng nhập vào hệ thống cần có tài khoản, thông tin tài khoản. Tài khoản được cấp bởi quản lý.

5.1. Kiểm thử hộp đen

- Thành phần: 3 Textfield là tìm kiếm, tên tài khoản và mật khẩu; 1 Label là mã tài khoản; 5 Button là tạo mã mới, thêm, sửa, xóa, load; 2 ComboBox là tìm kiếm theo tiêu chí nào và chọn loại tài khoản.

Tên trường	Bắt buộc	Yêu cầu	Ghi chú
Label Mã tài khoản	Có	Không được bỏ trống, Click vào button tạo mã mới để hệ thống tự tạo mới mã TK	Dữ liệu trong trường này dùng để tạo mới, sửa, hoặc xóa tài khoản TK.
TextField Tên tài khoản	Có	Không được bỏ trống	Dữ liệu trong trường này dùng để tạo mới, sửa, hoặc xóa tài khoản TK.
TextField Mật khẩu	Có	Không được bỏ trống	Dữ liệu trong trường này dùng để tạo mới, sửa,

			hoặc xóa tài khoản TK.
ComboBox Loại tài khoản	Có	Không được bỏ trống, click chọn loại tài khoản	Dữ liệu trong trường này dùng để tạo mới, sửa, hoặc xóa TK.
ComboBox Loại tìm kiếm	Không	Chỉ Click chọn tiêu chí tìm kiếm khi cần tìm kiếm NV, sau khi nhập thông tin cần tìm kiếm nhấn button tìm kiếm	Tìm kiếm theo mã TK, tên TK…

❖ Test case

ID	Test Step	Test data	Expected Result	A Result
1	-Nhập mã tài khoản -Nhập tên tài khoản -Nhập mật khẩu -Chọn loại tài khoản -Click vào nút thêm	Mã tài khoản: “TK_01” Tên tài khoản: “TkA” Mật khẩu: “123” Loại tài khoản: “1”	Thêm thành công	Pass
2		Mã tài khoản: “TK_01” Tên tài khoản: “” Mật khẩu: “” Loại tài khoản: “1” (Tên và mật khẩu không hợp lệ)	Thêm không thành công	Pass

3	-Nhập mã tài khoản -Nhấn nút xóa.	Mã tài khoản: TK_03	Xóa thành công thông tin tài khoản	Pass
4		Mã tài khoản: TK_A00 (Mã tài khoản không tồn tại).	Xóa thành công thông tin tài khoản	Fail
5	-Thực hiện tìm kiếm thông tin tài khoản muốn chỉnh sửa -Nhập thông tin muốn chỉnh sửa -Nhấn nút sửa.	Mã tài khoản: “TK_01” Tên tài khoản: “TkA”(cũ) Mật khẩu: “123” Loại tài khoản: “1” ↓ Mã tài khoản: “TK_01” Tên tài khoản: “TkB” Mật khẩu: “123” Loại tài khoản: “1”	Sửa thành công thông tin tài khoản	Pass
6		Mã tài khoản: “TK_01” ↓ Mã tài khoản : “”(mã không hợp lệ)	Sửa thành công thông tin tài khoản	Fail

- ❖ Các trường hợp kiểm thử (bảng quyết định):

  BangQuyetDinh_QL_TaiKhoan_WhiteBoxTesting.xlsx

- ❖ Test report

Số lượng test case	6
Số test case Pass	4
Số test case Fail	2
Số test case Error	0
Số test case chưa thực hiện	0
Tỉ lệ thành công	66.67%

5.2. Kiểm thử hộp trắng

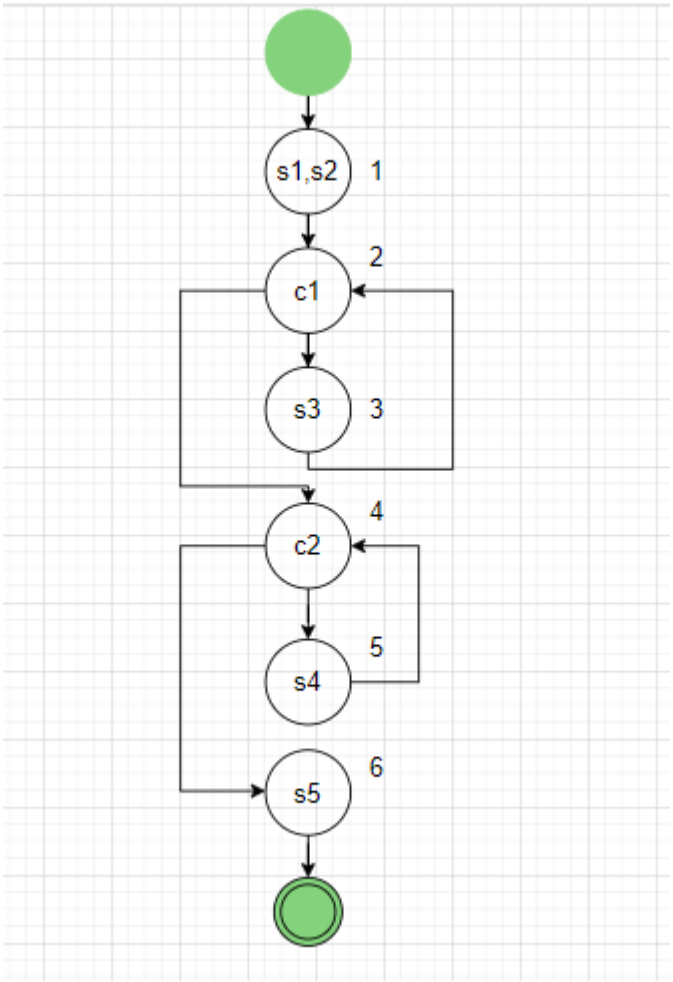
Lấy dữ liệu từ csdl và thiết lập vào danh sách hiển thị các tài khoản

❖ Mã nguồn

```
    //thiết lập danh sách tài khoản của DefaultTableModel
    public void setListTk(DefaultTableModel model){
/*s1*/          this.resetListTk();
/*s2*/          this.loadListTk();

/*c1*/          while(model.getRowCount()>0){
/*s3*/              model.removeRow(row: 0);
        }

/*c2*/          for (TaiKhoanDTO taiKhoanDTO : listTK) {
/*s4*/              model.addRow(new Object[]{
                  taiKhoanDTO.getMa_tai_khoan(),
                  taiKhoanDTO.getTen_tai_khoan(),
                  taiKhoanDTO.getMat_khau(),
                  new PhanQuyenBLL().getQuyenByMaQuyen(taiKhoanDTO.getPhan_quyen()),
                  (taiKhoanDTO.getTrang_thai()!= 0 ? "Đã sở hữu" : "Chưa sở hữu")
              });
          }
/*s5*/          model.fireTableDataChanged();
    }
```

Hình 5.2.1. Mã nguồn chức năng kể lấy dữ liệu từ csdl và thiết lập vào danh sách hiển thị các chương trình khuyến mãi.

❖ Đồ thị dòng điều khiển cơ bản



Hình 5.2.2. Đồ thị dòng điều khiển cơ bản chức năng kể lấy dữ liệu từ csdl và thiết lập vào danh sách hiển thị các chương trình khuyến mãi.

⇨ Độ phức tạp Cyclomatic M = N + 1, với N là số nút điều kiện rẽ nhánh nhị phân.

⇨ M = 2 + 1 = 3

⇨ Có 3 đường độc lập tuyến tính cơ bản là:

- 1 – 2 – 4 – 6
- 1 – 2 – 3 – 2 – 4 – 6
- 1 – 2 – 4 – 5 – 4 – 6

❖ Test case

ID	Basis path	Data	Expected results
1	1-2-4-6	model.getRowCount() <= 0	Không cần xóa dữ liệu modelTK hiện có vì nó đang rỗng
2	1-2-3-2-4-6	model.getRowCount() > 0 listTK rỗng (listTK.size() = 0)	Reset dữ liệu trong modelTK hiện tại. List chứa dữ liệu được lấy về từ csdl rỗng nên không có dữ liệu để nạp vào model.
3	1-2-4-5-4-6	model.getRowCount() <= 0 listTK có dữ liệu (listTK.size() > 0)	Không cần xóa dữ liệu modelTK hiện có vì nó đang rỗng. Nạp dữ liệu từ listTK vào modelTK để hiển thị lên giao diện người dùng.

❖ Test report

Số lượng test case	3
Số test case Pass	2
Số test case Fail	1
Số test case Error	0
Số test case chưa thực hiện	0
Tỉ lệ thành công	66.67%

Thêm tài khoản, xóa tài khoản, sửa tài khoản đều tương tự nhau về đòng điều khiển cơ bản và đều có chung đường độc lập tuyến tính cơ bản.

❖ Mã nguồn lần lượt các chức năng thêm tài khoản, sửa tài khoản, xóa tài khoản.

```
//    Thêm tài khoản
    public void ThemTK(String []arr){
/*s1*/        taiKhoanDAL.addData(arr);
/*s2*/         this.resetListTk();
/*s3*/         this.loadListTk();
    }
```

Hình 5.2.3. Mã nguồn các chức năng thêm tài khoản

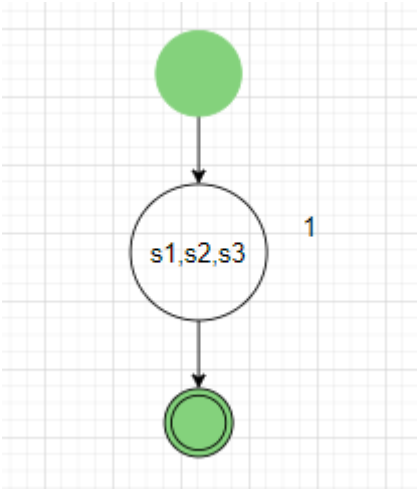
```
//    sửa tk
    public void SuaTk(String []arr){
/*s1*/         taiKhoanDAL.updateData(arr);
/*s2*/         this.resetListTk();
/*s3*/         this.loadListTk();
    }
```

Hình 5.2.3. Mã nguồn các chức năng sửa tài khoản

```
//    Xóa tài khoản
    public void XoaTK(int ma_tk){
/*s1*/         taiKhoanDAL.delData(ma_tk);
/*s2*/         this.resetListTk();
/*s3*/         this.loadListTk();
    }
```

Hình 5.2.4. Mã nguồn các chức năng xóa tài khoản

❖ Đồ thị dòng điều khiển cơ bản



Hình 5.2.5. Đồ thị dòng điều khiển cơ bản các chức năng trên.

⇨ Độ phức tạp Cyclomatic M = N + 1, với N là số nút điều kiện rẽ nhánh nhị phân.

⇨ M = 0 + 1 = 1 (với N = 0)

⇨ Đường độc lập tuyến tính cơ bản duy nhất là 1

❖ Test case chức năng thêm tài khoản

ID	Basis path	Data	Expected results
1	1	Mảng arr chứa dữ liệu là thông tin tài khoản người quản lý nhập vào, cụ thể: Arr[0]: “TK_01”; Arr[1]: “TkA”; Arr[2]: “1”; Arr[3]: “1”;	Tài khoản mới được thêm vào csdl. Đồng thời reset và cập nhật lại listTK (danh sách chứa các TK được lấy xuống từ csdl).

❖ Test report

Số lượng test case	1
Số test case Pass	1

Số test case Fail	0
Số test case Error	0
Số test case chưa thực hiện	0
Tỉ lệ thành công	100%

❖ Test case chức năng sửa tài khoản

ID	Basis path	Data	Expected results
1	1	Mảng arr chứa dữ liệu là thông tin TK người quản lý nhập vào, cụ thể: Arr[0]: “TK_01”; Arr[1]: “TkA”; Arr[2]: “1”; Arr[3]: “1”;	Tài khoản được cập nhật lại ở csdl. Đồng thời reset và cập nhật lại listTK (danh sách chứa các TK được lấy xuống từ csdl).

❖ Test report

Số lượng test case	1
Số test case Pass	1
Số test case Fail	0
Số test case Error	0
Số test case chưa thực hiện	0
Tỉ lệ thành công	100%

❖ Test case xóa tài khoản

ID	Basis path	Data	Expected results

1	1	Ma_TK: TK_01 (Ma_TK là mã của nhân viên được người quản lý click vào).	Tài khoản được chọn bị xóa khỏi csdl. Đồng thời reset và cập nhật lại listTK (danh sách chứa các TK được lấy xuống từ csdl).

❖ Test report

Số lượng test case	1
Số test case Pass	1
Số test case Fail	0
Số test case Error	0
Số test case chưa thực hiện	0
Tỉ lệ thành công	100%

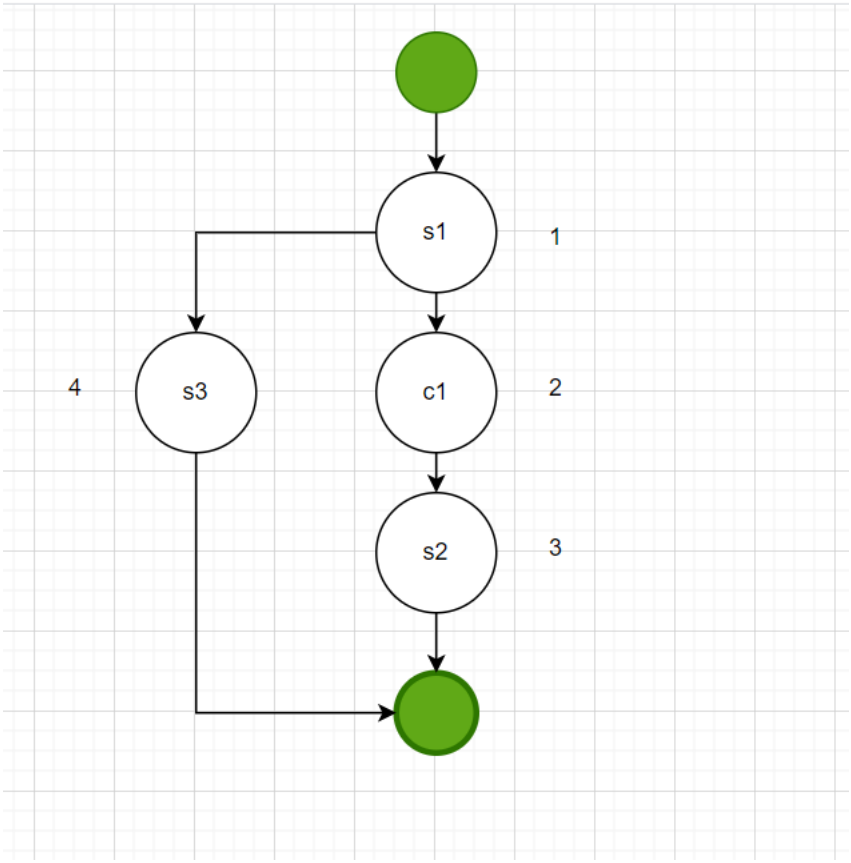
Tạo mới mã nhân viên

❖ Mã nguồn

```
    public int maMoi(){
/*s1*/          int ma = taiKhoanDAL.getMaMax();
/*c1*/          if(ma!=-1){
/*s2*/              return ma+1;
        }
/*s3*/          return 1;

    }
```

Hình 5.2.6. Mã nguồn chức năng tạo mã tài khoản mới.

❖ Đồ thị dòng điều khiển cơ bản



Hình 5.2.7. Đồ thị dòng điều khiển cơ bản chức năng tạo mã tài khoản mới.

⇨ Độ phức tạp Cyclomatic M = N + 1, với N là số nút điều kiện rẽ nhánh nhị phân.

⇨ M = 1 + 1 = 2 (với N = 1)

⇨ Đường độc lập tuyến tính cơ bản:

- 1 – 4
- 1 – 2 – 3

❖ Test case

ID	Basis path	Data	Expected results
1	1 – 4	maHienTai: -1 (MaHienTai được lấy từ csdl)	Phương thức trả về một mã mới là 01
2	1 – 2 – 3 –	maHienTai: 03 (MaHienTai được lấy từ csdl)	Phương thức trả về mã mới là 04

			(maHienTai + 1)

❖ Test report

Số lượng test case	2
Số test case Pass	2
Số test case Fail	0
Số test case Error	0
Số test case chưa thực hiện	0
Tỉ lệ thành công	100%

Các phương thức tìm kiếm tài khoản theo mã và hiển thị ra giao diện, tìm kiếm và hiển thị theo tên tương tự nhau về đồ thị dòng điều kiển và đều có chung số đường độc lập cơ bản, chỉ khác về test case và nó có mã nguồn và test case tương tự như giống như các chức năng tìm kiếm của phần quản lý nhân viên đã đề cập ở mục 4.

6. Cập nhật thông tin tài khoản

Đặc tả: Nhân viên bán hàng sau khi được cấp tài khoản sẽ đăng nhập với tài khoản mặc định đó vào hệ thống rồi tiến hành sửa lại thông tin tài khoản của mình cho phù hợp với thông tin nhân viên ban đầu đã khai báo.

6.1. Kiểm thử hộp đen

❖ Thành phần: 4 Textfields là tên tài khoản, mật khẩu, mật khẩu mới và nhập lại mật khẩu mới, 2 button là cập nhật và thoát.

Tên trường	Bắt buộc	Yêu cầu	Ghi chú

TextField Tên tài khoản	Có	Không được bỏ trống	Dữ liệu trong trường này dùng để sửa tài khoản.
TextField Mật khẩu	Có	Không được bỏ trống	Dữ liệu trong trường này dùng để sửa tài khoản.
TextField Mật khẩu mới	Có	Không được bỏ trống	Dữ liệu trong trường này dùng để sửa tài khoản.
TextField Nhập lại mật khẩu mới	Có	Không được bỏ trống	Dữ liệu trong trường này dùng để sửa tài khoản.

- ❖ Các trường hợp kiểm thử (bảng quyết định): BangQuyetDInh_SuaThongTinTK_WhiteBoxTesting.xlsx
- ❖ Test case

ID	Test Step	Test data	Expected Result	A Result
1	-Nhập tên tài khoản -Nhập mật khẩu -Nhập mật khẩu mới -Nhập lại mật khẩu mới -Click vào nút cập nhật	Tên tài khoản: “TkA” Mật khẩu: “123” Mật khẩu mới: “1234@” Nhập lại mật khẩu mới: “1234@”	Cập nhật thành công	Pass
2		Tên tài khoản: “TkA” Mật khẩu: “123” Mật khẩu mới: “12345@” Nhập lại mật khẩu mới: “1234@”	Thêm thành công	Fail

		(nhập lại mật khẩu mới không đúng)		

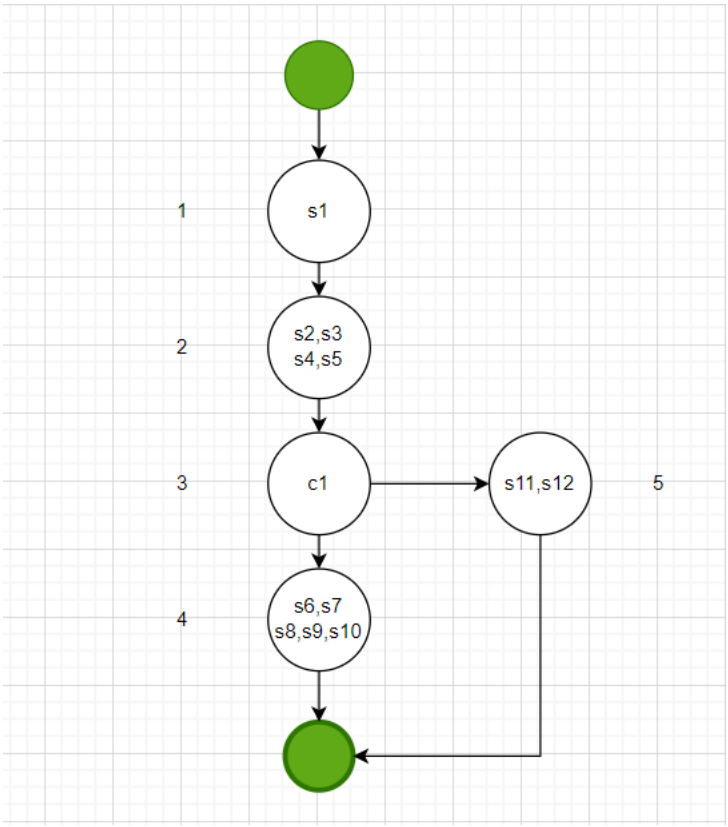
6.2. Kiểm thử hộp trắng

❖ Mã nguồn

```
    private void btnUpdateActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event_btnUpdateActionPerformed

/*s1*/            TaiKhoanDTO taiKhoanDTO = new TaiKhoanBLL().getTkByMaTK(ma_tk);
/*s2*/            String []str = {
/*s3*/                String.valueOf(ma_tk),
/*s4*/                txtUsername.getText(),
/*s5*/                txtNewPass.getText()
            };
/*c1*/            if(txtNewPass.getText().equals(txtRePass.getText())){
/*s6*/                new TaiKhoanDAL().updateUsernameAndPass(str);
/*s7*/                JOptionPane.showMessageDialog(this,message: "Sửa thành công",title: "Thông báo",JOptionPane.INFO
/*s8*/                txtNewPass.setText(t: "");
/*s9*/                txtRePass.setText(t: "");
/*s10*/                loadInfoTK();
            }
/*s11*/            else{
/*s12*/                JOptionPane.showMessageDialog(this,message: "Mật khẩu không khớp", title: "Lỗi",JOptionPane.ERR
            }


    }//GEN-LAST:event_btnUpdateActionPerformed
```

Hình 6.2.1 Mã nguồn chức năng sửa thông tin tài khoản.

❖ Đồ thị dòng điều khiển cơ bản



Hình 6.2.2 Sơ đồ dòng điều khiển cơ bản chức năng sửa thông tin tài khoản.

⇨ Độ phức tạp Cyclomatic M = N + 1, với N là số nút điều kiện rẽ nhánh nhị phân.

⇨ M = 1 + 1 = 2(với N = 1)

⇨ Đường độc lập tuyến tính cơ bản:

- 1 – 2 – 3 – 4
- 1 – 2 – 3 – 5

❖ Test case

ID	Basis path	Data	Expected results
1	1-2-3-4	Tài khoản: admin Mật khẩu: admin (Tài khoản và mật khẩu đúng)	Cập nhật lại tài khoản thành công

2	1-2-3-5	Tài khoản: admin Mật khẩu: 1 (Tài khoản và mật khẩu sai)	Dialog hiện lên thông báo người dùng nhập sai tài khoản hoặc mật khẩu.

❖ Test report

Số lượng test case	2
Số test case Pass	1
Số test case Fail	1
Số test case Error	0
Số test case chưa thực hiện	0
Tỉ lệ thành công	50%

7. Quản lý nhà cung cấp

Đặc tả : Người quản lý sẽ quản lý các nhà cung cấp sản phẩm cho cửa hàng bao gồm việc tạo mới, sửa, xóa và xem chi tiết các nhà cung cấp ứng với kinh doanh loại sản phẩm nào.

7.1. Kiểm thử hộp đen

Tên trường	Bắt buộc	Yêu cầu	Ghi chú
Label Mã Nhà Cung Cấp	Có	Không được bỏ trống, Click vào button tạo mã	Dữ liệu trong trường này dùng để tạo mới,

		mới để hệ thống tự tạo mới mã nhà cung cấp	sửa, hoặc xóa nhà cung cấp.
TextFields Tên Nhà Cung Cấp	Có	Không được bỏ trống	Dữ liệu trong trường này dùng để tạo mới, sửa, hoặc xóa nhà cung cấp.
TextFields Địa chỉ	Có	Không được bỏ trống	Dữ liệu trong trường này dùng để tạo mới, sửa, hoặc xóa nhà cung cấp.
ComboBox Loại tìm kiếm	Không	Chỉ Click chọn tiêu chí tìm kiếm khi cần tìm kiếm danh sách nhà cung cáp, sau khi nhập thông tin cần tìm kiếm nhấn button tìm kiếm	Tìm kiếm theo mã nhà cung cấp, tên nhà cung cấp, địa chỉ

- Thành phần: 3 Text fields là "Mã nhà cung cấp", "Tên nhà cung cấp" và "Địa chỉ"; 6 Button là tạo mới, thêm, sửa, xoá, load, chi tiết; 1 ComboBox là tìm kiếm theo tiêu chí nào.

- Các trường hợp kiểm thử (bảng quyết định)

BangQuyetDinh_QL_NCC_WhiteBoxTesting.xlsx

- Test case

ID	Test Step	Test data	Expected Result	A Result

1	-Nhập mã nhà cung cấp -Nhập tên nhà cung cấp -Nhập địa chỉ nhà cung cấp -Click vào nút thêm	Mã nhà cung cấp : “NCC_15” Tên nhà cung cấp : “SEA” Địa chỉ : “Phú Thọ”	Thêm thành công	Pass
2		Mã nhà cung cấp : “NCC_09” Tên nhà cung cấp : “” Địa chỉ : “” (Tên và địa chỉ không hợp lệ)	Thêm không thành công	Fail
3	-Nhập mã nhà cung cấp -Nhấn nút xóa.	Mã thành viên: NCC_01	Xóa thành công thông tin nhà cung cấp	Pass
4		Mã nhà cung cấp : NCC_999 (Mã nhà cung cấp không tồn tại).	Xóa không thành công thông tin nhà cung cấp	Fail
5	-Thực hiện tìm kiếm thông tin nhà cung cấp muốn chỉnh sửa -Nhập thông tin muốn chỉnh sửa -Nhấn nút sửa.	MãNCC : 01 TênNCC:“ABCD”(cũ) Địa chỉ : “Tân Phú” ↓ MãNCC : 02 TênNCC:”XYZ” Địa chỉ : “Bình Thạnh”	Sửa thành công thông tin nhà cung cấp	Pass

6		MãNCC : 01 ↓ MãNCC : “”(mã không hợp lệ)	Sửa không thành công thông tin nhà cung cấp	Fail

7.2. Kiểm thử hộp trắng

Lấy dữ liệu từ csdl và thiết lập vào danh sách hiển thị danh sách nhà cung cấp

❖ Mã nguồn

Hình 7.2.1 Mã nguồn chức năng lấy dữ liệu từ csdl và thiết lập vào danh sách hiển thị danh sách nhà cung cấp.

❖ Đồ thị dòng điều khiển cơ bản

```
    //set list lên table list
    public  void setListNCC(De
/*s1*/          this.resetListNC
/*s2*/          this.loadListNCC

/*c1*/          while(model.getR
/*s3*/              model.remove
        }

/*c2*/          for (NhaCungCapD
/*s4*/              model.addRow
                  nhaCungCapDTO.
                  nhaCungCapDTO.
                  nhaCungCapDTO.
              });
        }
/*s5*/          model.fireTableD
    }
```

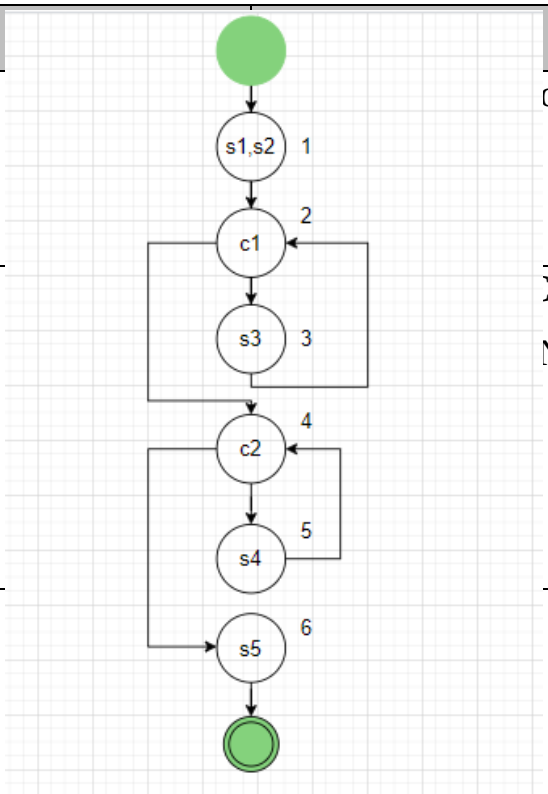
⇨ Độ phức tạp Cyclomatic M = N + 1, với N là số nút điều kiện rẽ nhánh nhị phân.

⇨ M = 2 + 1 = 3

⇨ Có 3 đường độc lập tuyến tính cơ bản là:

- 1 – 2 – 4 – 6
- 1 – 2 – 3 – 2 – 4 – 6
- 1 – 2 – 4 – 5 – 4 – 6

Hình 7.2.2 Đồ thị dòng điều khiển c
danh sách hiể



ID			Expected results
1		ount() <= 0	Không cần xóa dữ liệu modelNCC hiện có vì nó đang rỗng
2		Count() > 0 NCC.size() =	Reset dữ liệu trong modelNCC hiện tại. List chứa dữ liệu được lấy về từ csdl rỗng nên không có dữ liệu để nạp vào model.

3	1-2-4-5-4-6	model.getRowCount() <= 0 listNCC có dữ liệu (listNCC.size() > 0)	Không cần xóa dữ liệu modelNCC hiện có vì nó đang rỗng. Nạp dữ liệu từ listKM vào modelNCC để hiển thị lên giao diện người dùng.

❖ Test report

Số lượng test case	3
Số test case Pass	3
Số test case Fail	0
Số test case Error	0
Số test case chưa thực hiện	0
Tỉ lệ thành công	100%

Thêm, sửa, xoá danh sách nhà cung cấp đều tương tự nhau về đòng điều khiển cơ bản và đều có chung đường độc lập tuyến tính cơ bản.

❖ Mã nguồn

```
    public void ThemNCC(String []str){
/*s1*/          nhaCungCapDAL.addData( arr: str);
/*s2*/          this.resetListNCC();
/*s3*/          this.loadListNCC();
    }
    public void suaNCC(String []str){
/*s1*/          nhaCungCapDAL.updateData( arr: str);
/*s2*/          this.resetListNCC();
/*s3*/          this.loadListNCC();
    }
    public void delNCC(String maNV){
/*s1*/          nhaCungCapDAL.delData( ma_ncc: maNV);
/*s2*/          this.resetListNCC();
/*s3*/          this.loadListNCC();
    }
```

Hình 7.2.3 Mã nguồn chức năng thêm, sửa, xóa các nhà cung cấp

❖ Đồ thị dòng điều khiển cơ bản

Hình 7.2.4 Đồ thị dòng điều

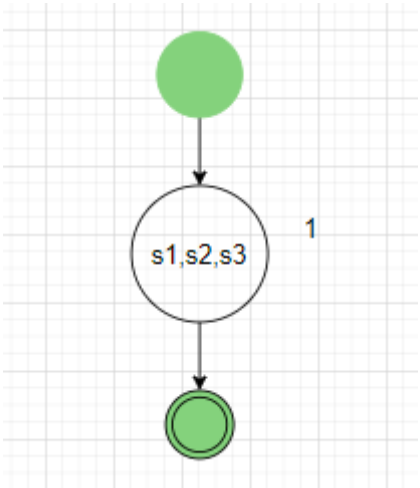
⇨ Độ phức tạp Cyclomatic M = N + 1, với N là số nút điều kiện rẽ nhánh nhị phân.

⇨ M = 0 + 1 = 1 (với N = 0)

⇨ Đường độc lập tuyến tính cơ bản duy nhất là 1

❖ Test case thêm nhà c

ID	Basis path	Da



1	1	Mảng arr chứa dữ liệu là thông tin nhà cung cấp người quản lý nhập vào, cụ thể: Arr[0]: “NCC_VN”; Arr[1]: “VietGap”; Arr[2]: “Hà Nội”;	Nhà cung cấp mới được thêm vào csdl. Đồng thời reset và cập nhật lại listNCC (danh sách chứa các nhà cung cấp được lấy xuống từ csdl).

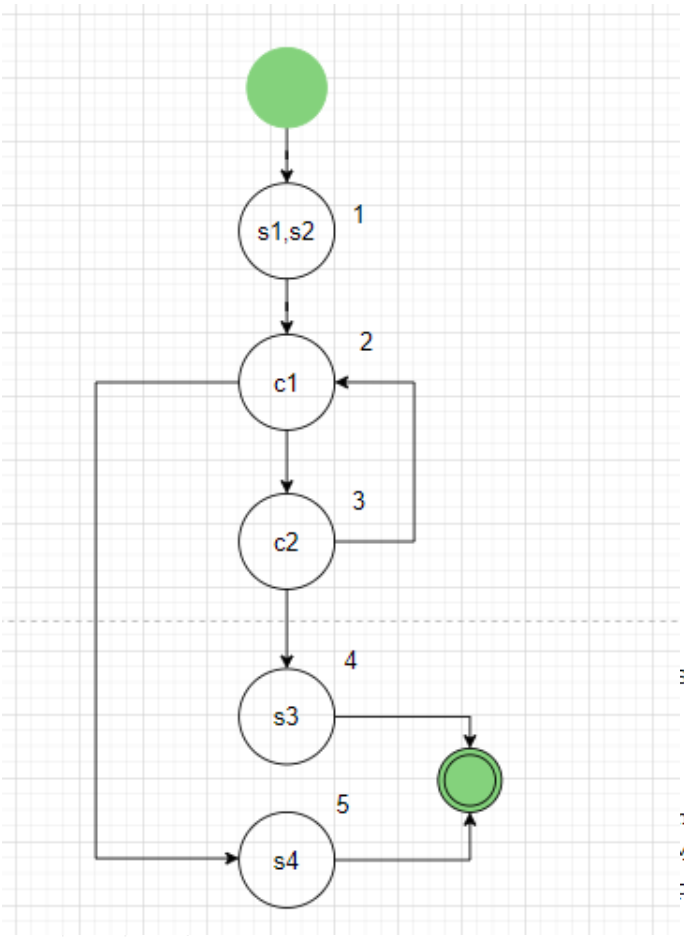
❖ Test case sửa nhà cung cấp

ID	Basis path	Data	Expected results
1	1	Mảng arr chứa dữ liệu là thông tin ncc mà người quản lý nhập vào, cụ thể: Arr[0]: “NCC_VN”; Arr[1]: “CTTNHH Thực phẩm sạch”; Arr[2]: “Phú Nhuận”;	Nhà cung cấp được cập nhật lại ở csdl. Đồng thời reset và cập nhật lại listNCC (danh sách chứa các NCC được lấy xuống từ csdl).

❖ Test case xóa nhà cung cấp

ID	Basis path	Data	Expected results
1	1	Ma_NCC:”NCC_VN” (Ma_NCC là mã của ncc được người quản lý click vào).	NCC được chọn bị xóa khỏi csdl. Đồng thời reset và cập nhật lại listNCC (danh sách chứa các NCC được lấy xuống từ csdl).

Tìm kiếm nhà cung cấp theo mã




```
String ma_ncc) {

igCapDTO : listNCC) {
Ma_ncc().equals( anObject:ma_ncc)){
oDTO;

        }
/*s4*/          return null;
    }
```

❖ Mã nguồn

Hình 7.2.5 Mã nguồn chức năng tìm kiếm nhà cung cấp

❖ Đồ thị dòng điều khiển cở bản

Hình 7.2.6. Đồ thị dòng điều khiển c

⇨ Độ phức tạp Cyclomatic M = N + 1, với N là số nút điều kiện rẽ nhánh nhị phân.

⇨ M = 2 + 1 = 3 (với N = 2)

⇨ Đường độc lập tuyến tính cơ bản:

- 1 – 2 – 5
- 1 – 2 – 3 – 4
- 1 – 2 – 3 – 2 – 3 – 5

❖ Test case

ID	Basis path	Data

1	1-2-5	ma_ncc : 15 (Nhập từ bàn phím của người quản lý)	Trả về null (do listNCC rỗng, chưa có dữ liệu trong csdl tức chưa có ctk ncc m nào)
2	1-2-3-4	ma_ncc: 01 (Nhập từ bàn phím của người quản lý)	Trả về ncc có mã ncc là 01
3	1-2-3-2-3-5	ma_ncc: 15 (Nhập từ bàn phím của người quản lý)	Trả về null do không tìm thấy ncc nào có mã ncc là 15 trong listNCC

❖ Test report

Số lượng test case	3
Số test case Pass	1
Số test case Fail	2
Số test case Error	0
Số test case chưa thực hiện	0
Tỉ lệ thành công	33.33%

Các phương thức tìm kiếm chương trình khuyến mãi theo mã và hiển thị ra giao diện, tìm kiếm và hiển thị theo trạng thái, tìm kiếm và hiển thị theo tên, tìm kiểm và hiển thị theo ngày và khoảng ngày đều tương tự nhau về đồ thị dòng điều kiển và đều có chung số đường độc lập cơ bản, chỉ khác về test case và nó tương tự như những phương thức tìm kiếm của chức năng quản lý tài khoản, nhân viên đã làm ở những mục trước.

8. Quản lý loại sản phẩm

Đặc tả : Người quản lý sẽ quản lý các loại sản phẩm của cửa hàng bao gồm việc tạo mới, sửa, xóa loại sản phẩm đó.

8.1. Kiểm thử hộp đen

- Thành phần : 2 Text fields là "Mã loại thức ăn", "Tên loại thức ăn", 6 Button là tạo mới, thêm, sửa, xoá, load; 1 ComboBox là tìm kiếm theo tiêu chí nào.

Tên trường	Bắt buộc	Yêu cầu	Ghi chú
Label Mã Loại Thức Ăn	Có	Không được bỏ trống, Click vào button tạo mã mới để hệ thống tự tạo mới mã loại thức ăn	Dữ liệu trong trường này dùng để tạo mới, sửa, hoặc xóa loại sản phẩm
TextFields Tên Loại Thức Ăn	Có	Không được bỏ trống	Dữ liệu trong trường này dùng để tạo mới, sửa, hoặc xóa loại sản phẩm
ComboBox Loại tìm kiếm	Không	Chỉ Click chọn tiêu chí tìm kiếm khi cần tìm kiếm danh sách nhà cung cáp, sau khi nhập thông tin cần tìm kiếm nhấn button tìm kiếm	Tìm kiếm theo mã loại thức ăn, tên loại thức ăn.

- Các trường hợp kiểm thử (bảng quyết định)

BangQuyetDinh_QL_LoaiSanPham_WhiteBoxTesting.xlsx

❖ Test case

ID	Test Step	Test data	Expected Result	A Result
1	-Nhập mã loại thức ăn -Nhập tên loại thức ăn -Click vào nút thêm	Mã loại thức ăn : “01” Tên loại thức ăn : “Tráng miệng”	Thêm thành công	Pass
2		Mã loại thức ăn : “02” Tên loại thức ăn : “” (Tên không hợp lệ)	Thêm không thành công	Fail
3	-Nhập mã loại thức ăn -Nhấn nút xóa.	Mã loại thức ăn : “01”	Xóa thành công thông tin loại thức ăn	Pass
4		Mã loại thức ăn : “999” (Mã loại thức ăn không tồn tại).	Xóa không thành công thông tin loại thức ăn	Fail

5	-Thực hiện tìm kiếm thông tin nhà cung cấp muốn chỉnh sửa -Nhập thông tin muốn chỉnh sửa -Nhấn nút sửa.	Mã loại thức ăn : “01” Tên loại thức ăn : “Ăn nhanh” ↓ Mã loại thức ăn : “02” Tên loại thức ăn : “Rau củ”	Sửa thành công thông tin loại thức ăn	Pass
6		Mã loại thức ăn : 01 Tên loại thức ăn :“Tráng miệng” ↓ Mã loại thức ăn : “” (Mã không hợp lệ)	Sửa không thành công thông tin loại thức ăn	Fail

❖ Test report

Số lượng test case	6
Số test case Pass	3
Số test case Fail	3
Số test case Error	0
Số test case chưa thực hiện	0
Tỉ lệ thành công	50%

8.2. Kiểm thử hộp trắng

Lấy dữ liệu từ csdl và thiết lập vào danh sách hiển thị loại sản phẩm

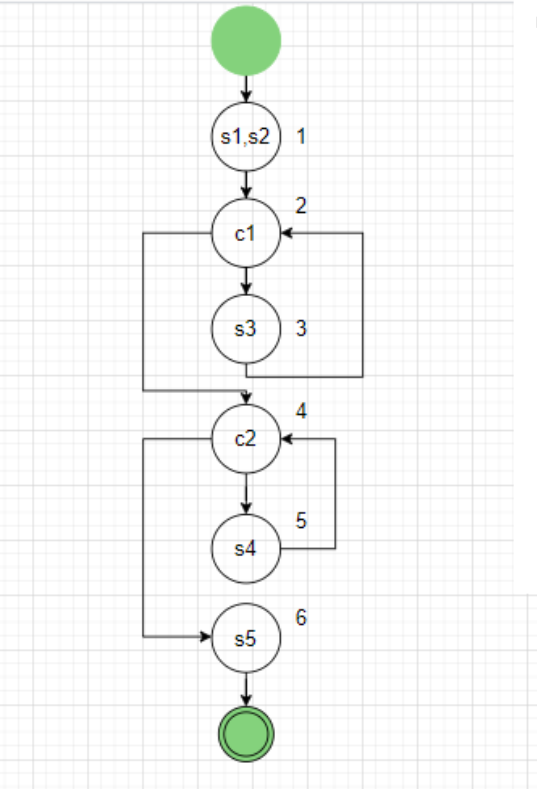
❖ Mã nguồn

```
    public void setListLM(DefaultTableModel model){
/*s1*/          this.resetListLM();
/*s2*/           this.loadListLM();

/*c1*/           while(model.getRowCount() > 0){
/*s3*/               model.removeRow( row: 0);
        }
/*c2*/           for (LoaiMonDTO loaiMonDTO : listLM) {
/*s4*/               model.addRow(new Object[]{
                    loaiMonDTO.getMa_loai(),
                    loaiMonDTO.getTen_loai()
                });
        }
/*s5*/           model.fireTableDataChanged();
    }
```

Hình 8.2.1. Mã nguồn chức năng lấy dữ liệu từ csdl và thiết lập vào danh sách hiển thị loại sản phẩm.

❖ Đồ thị dòng điều khiển

Hình 8.2.1. Đồ thị dòng điều khiển sách h



⇨ Độ phức tạp Cyclomatic M = N + 1, với N là số nút điều kiện rẽ nhánh nhị phân.

⇨ M = 2 + 1 = 3

⇨ Có 3 đường độc lập tuyến tính cơ bản là:

- 1 – 2 – 4 – 6
- 1 – 2 – 3 – 2 – 4 – 6
- 1 – 2 – 4 – 5 – 4 – 6

❖ Test case

ID	Basis path	Data	Expected results
1	1-2-4-6	listLoaiSP.getRowCount() <= 0	Không cần xóa dữ liệu modelLoaiSP hiện có vì nó đang rỗng
2	1-2-3-2-4-6	listLoaiSP.getRowCount() > 0 listLoaiSP rỗng (listLoaiSP.size() = 0)	Reset dữ liệu trong modelLoaiSP hiện tại. List chứa dữ liệu được lấy về từ csdl rỗng nên không có dữ liệu để nạp vào model.

3	1-2-4-5-4-6	listLoaiSP.getRowCount() <= 0 listLoaiSP có dữ liệu (listLoaiSP.size() > 0)	Không cần xóa dữ liệu modelLoaiSP hiện có vì nó đang rỗng. Nạp dữ liệu từ listLoaiSP vào modelLoaiSP để hiển thị lên giao diện người dùng.

❖ Test report

Số lượng test case	3
Số test case Pass	3
Số test case Fail	0
Số test case Error	0
Số test case chưa thực hiện	0
Tỉ lệ thành công	50%

Các chức năng như thêm, xóa, sửa loại sản phẩm nhìn chung đều có mã nguồn và đồ thị dòng điều khiển, đường độc lập tuyến tính cơ bản tương tự với các chức năng quản lý khác như tài khoản, nhân viên, nhà cung cấp,…

Tìm kiếm loại sản phẩm theo mã, theo tên đều tương tự với những yêu cầu kiểm thử của những chức năng quản lý khác.

9. Quản lý sản phẩm

Đặc tả : Người quản lý sẽ quản lý các loại sản phẩm của cửa hàng bao gồm việc tạo mới, sửa, xóa sản phẩm. Lưu ý chỉ được sửa các thông tin khác ngoại trừ thông tin liên quan đến số lượng sản phẩm, nó chỉ thay đổi khi có đơn hàng mua (giảm) và khi có đơn hàng nhập (tăng).

9.1. Kiểm thử hộp đen

- Thành phần: 5 Text fields : “Mã món” , “Tên món”, “Giá”, “Danh mục món”, “Số lượng”, khoảng giá từ bao nhiêu đến bao nhiêu, số lượng từ bao nhiêu đến bao nhiêu; 5 Button là tạo mới, thêm, sửa, xoá, load; 1 ComboBox là tìm kiếm theo tiêu chí nào.

Tên trường	Bắt buộc	Yêu cầu	Ghi chú
Label Mã Món	Có	Không được bỏ trống, Click vào button tạo mã mới để hệ thống tự tạo mới mã món ăn	Dữ liệu trong trường này dùng để tạo mới, sửa, hoặc xóa món ăn
TextFields Tên Món	Có	Không được bỏ trống	Dữ liệu trong trường này dùng để tạo mới, sửa, hoặc xóa món ăn.
TextFields Giá	Có	Không được bỏ trống	Dữ liệu trong trường này dùng để tạo mới, sửa, hoặc xóa món ăn.
TextFields Danh Mục Món	Có	Không được bỏ trống	Dữ liệu trong trường này dùng để tạo mới, sửa, hoặc xóa món ăn.
TextFields Số Lượng	Có	Không được bỏ trống	Dữ liệu trong trường này dùng để tạo mới, sửa, hoặc xóa món ăn.
ComboBox Tìm Kiếm	Không	Chỉ Click chọn tiêu chí tìm kiếm khi cần tìm kiếm danh sách món ăn, sau khi nhập thông tin cần tìm kiếm nhấn button tìm kiếm	Tìm kiếm theo mã món ăn, tên món ăn, giá , số lượng, danh mục món
TextField giá (from)	Không	Nhập giá khởi điểm	Tìm kiếm món ăn theo giá
TextField giá (to)	Không	Nhập giá kết thúc	Tìm kiếm món ăn theo giá
TextField số lượng(from)	Không	Nhập số lượng bắt đầu	Tìm kiếm món ăn theo số lượng

TextField số lượng(to)	Không	Nhập số lượng kết thúc	Tìm kiếm món ăn theo số lượng

❖ Các trường hợp kiểm thử (bảng quyết định)

BangQuyetDinh_QL_SanPham_WhiteBoxTesting.xlsx

❖ Test case

ID	Test Step	Test data	Expected Result	A Result
1	-Nhập mã món ăn -Nhập tên món ăn -Nhập giá từ - đến -Nhập danh mục món -Nhập số lượng từ - đến -Click vào nút thêm	Mã món ăn : “15” Tên món ăn : “Snack” Giá : “5.000 – 8.000 VNĐ” Danh mục món : “Ăn vặt” Số lượng : “20 - 30”	Thêm thành công	Pass
2		Mã món ăn : “15” Tên món ăn : “Snack” Giá : “5.000 VNĐ” Danh mục món : “” Số lượng : “” (Danh mục món và số lượng không hợp lệ)	Thêm không thành công	Fail
3	-Nhập mã món ăn -Nhấn nút xóa.	Mã món ăn : 10	Xóa thành công thông tin món ăn	Pass

4		Mã món ăn : 999 (Mã món ăn không tồn tại).	Xóa không thành công thông tin món ăn	Fail

5	-Thực hiện tìm kiếm thông tin nhà cung cấp muốn chỉnh sửa -Nhập thông tin muốn chỉnh sửa -Nhấn nút sửa.	Mã món ăn : “15” Tên món ăn : “Dưa hấu” Giá : “20.000 – 30.000 VNĐ” Danh mục món : “Rau củ” Số lượng : “20 - 30” ↓ Mã món ăn : “30” Tên món ăn : “Dưa hấu” Giá : “35.000 – 40.000 VNĐ” Danh mục món : “Tráng miệng” Số lượng : “15 - 18”	Sửa thành công thông tin món ăn	Pass
6		Mã món ăn : “15” Tên món ăn : “Đu đủ” Giá : “30.000 – 35.000 VNĐ” Danh mục món : “Rau củ” Số lượng : “20 - 23” ↓	Sửa không thành công thông tin món ăn	Fail

		Mã món ăn : “”(không hợp lệ) Tên món ăn : “Đu đủ” Giá : “25.000 – 30.000 VNĐ” Danh mục món : “Tráng miệng” Số lượng : “10 - 15”		

❖ Test report

Số lượng test case	6
Số test case Pass	3
Số test case Fail	3
Số test case Error	0
Số test case chưa thực hiện	0
Tỉ lệ thành công	50%

9.2. Kiểm thử hộp trắng

Các chức năng như lấy giá trị từ csdl và hiển thị hay thêm, xóa, sửa đều tương tự như các chức năng quản lý khác đã kiểm thử ở những phần trên.

Tìm kiếm sản phẩm theo khoảng giá, theo khoảng số lượng đều tương tự nhau và chỉ khác nhau về test case

❖ Mã nguồn

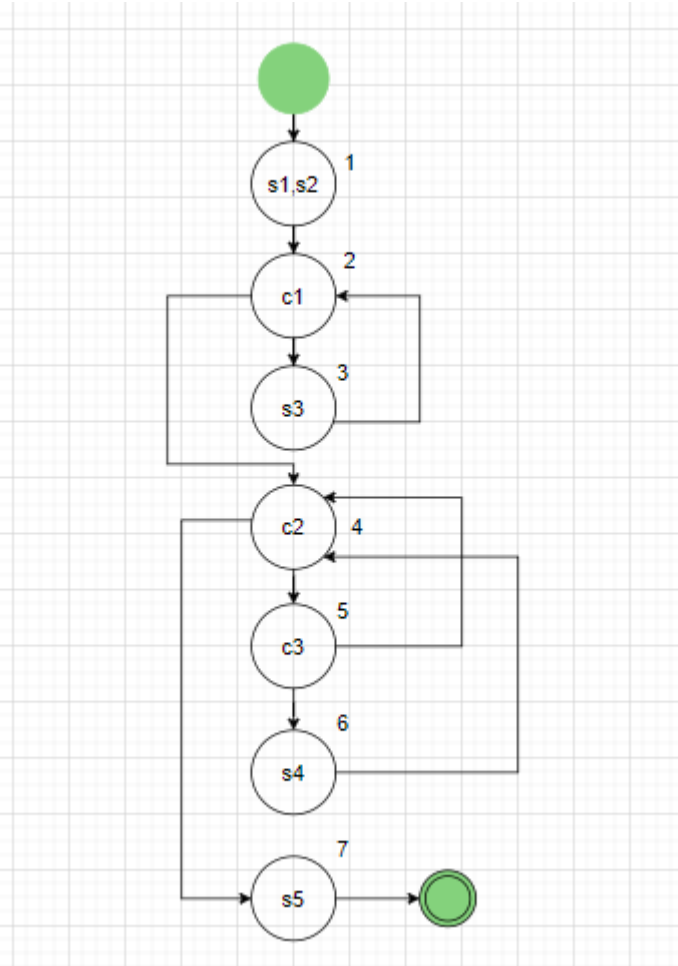
```
    public void timkiemMonAnTheoKhoangGia(DefaultTableModel model, int giaFrom, int giaTo) {
/*s1*/          this.resetListMA();
/*s2*/          this.loadListMA();

/*c1*/          while(model.getRowCount()>0){
/*s3*/              model.removeRow( row: 0);
        }
/*c2*/          for (MonAnDTO monAnDTO : listMA) {
/*c3*/              if(monAnDTO.getGia() > giaFrom && monAnDTO.getGia() < giaTo){
/*s4*/                  model.addRow(new Object[]{
                        monAnDTO.getMa_mon(),
                        monAnDTO.getTen_mon(),
                        monAnDTO.getSo_luong(),
                        monAnDTO.getGia(),
                        monAnDTO.getGia_giam(),
                        monAnDTO.getGia()-monAnDTO.getGia_giam(),
                        new LoaiMonAnBLL().getTenLoaiByMaLoai( ma_loai: monAnDTO.getMa_loai())
                    });
                }
            }
/*s5*/          model.fireTableDataChanged();
    }
```

Hình 9.2.1. Mã nguồn chức năng tìm kiếm sản phẩm theo khoảng giá.

```
    public  void timkiemMonAnTheoKhoangSL(DefaultTableModel model, int soLuongFrom, int soLuongTo){
/*s1*/          this.resetListMA();
/*s2*/          this.loadListMA();

/*c1*/          while(model.getRowCount()>0){
/*s3*/              model.removeRow( row: 0);
        }
/*c2*/          for (MonAnDTO monAnDTO : listMA) {
/*c3*/              if(monAnDTO.getSo_luong() > soLuongFrom && monAnDTO.getSo_luong() < soLuongTo){
/*s4*/                  model.addRow(new Object[]{
                        monAnDTO.getMa_mon(),
                        monAnDTO.getTen_mon(),
                        monAnDTO.getSo_luong(),
                        monAnDTO.getGia(),
                        monAnDTO.getGia_giam(),
                        monAnDTO.getGia()-monAnDTO.getGia_giam(),
                        new LoaiMonAnBLL().getTenLoaiByMaLoai( ma_loai: monAnDTO.getMa_loai())
                    });
                }
            }
/*s5*/          model.fireTableDataChanged();
    }
```

Hình 9.2.2. Mã nguồn chức năng tìm kiếm sản phẩm theo khoảng số lượng

❖ Đồ thị dòng điều khiển cơ bản



- ⇨ Độ phức tạp Cyclomatic M = N + 1, với N là số nút điều kiện rẽ nhánh nhị phân.
- ⇨ M = 3 + 1 = 4(với N = 3)
- ⇨ Đường độc lập tuyến tính cơ bản:
  - 1 – 2 – 4 – 7
  - 1 – 2 – 3 – 2 – 4 – 7
  - 1 – 2 – 4 – 5 – 6 – 4 – 7
  - 1 – 2 – 4 – 5 – 4 – 7

Hình 9.2.3. Đồ thị dòng điều khiển cơ bản các chức năng tìm kiếm sản phẩm theo khoảng số lượng, giá tiền

❖ Test case

ID	Basis path	Data	Expected results
1	1-2-4-7	defaultTableModel: modelSP (model của danh sách món ăn)	Cập nhật lại giao diện món ăn,

		search: “01” (search theo mã); “BingChilling” (search theo tên); “15.000 VNĐ”(search theo giá);”Tráng miệng”(search theo danh mục món);”14”(search theo số lượng);”4.000 – 20.000 VNĐ”(search theo khoảng giá); “3 - 20”(search theo khoảng số lượng) modelNCC.getRowCount() <= 0	(Do list rỗng nên giao diện loại món ăn vẫn không cập nhật)
2	1-2-3-2-4-7	defaultTableModel: modelSP (model của danh sách món ăn) search: “01” (search theo mã); “BingChilling” (search theo tên); “15.000 VNĐ”(search theo giá);”Tráng miệng”(search theo danh mục món);”14”(search theo số lượng);”4.000 – 20.000 VNĐ”(search theo khoảng giá); “3 - 20”(search theo khoảng số lượng)	Cập nhật lại giao diện loại món ăn, (Do list rỗng nên giao diện loại món ăn vẫn không cập nhật)

		modelNCC.getRowCount() <= 0	
3	1-2-4-5-6-4-7	defaultTableModel: modelSP (model của danh sách món ăn) search: “01” (search theo mã); “BingChilling” (search theo tên); “15.000 VNĐ”(search theo giá);”Tráng miệng”(search theo danh mục món);”14”(search theo số lượng);”4.000 – 20.000 VNĐ”(search theo khoảng giá); “3 - 20”(search theo khoảng số lượng) modelNCC.getRowCount() <= 0	Reset lại modelProductCategory trước đó. Món ăn được tìm thấy và nạp vào modelProductCategory. Cập nhật lại giao diện loại món ăn
4	1-2-4-5-4-7	defaultTableModel: modelSP (model của danh sách món ăn) search: “06” (search theo mã); “Bắp rang” (search theo tên); “15.000 VNĐ”(search theo giá);”Tráng miệng”(search	Reset lại modelProductCategory trước đó. Loại món ăn không tìm thấy. Cập nhật lại giao diện loại món ăn

		theo danh mục món);”13”(search theo số lượng);”4.000 – 20.000 VNĐ”(search theo khoảng giá); “3 - 10”(search theo khoảng số lượng) modelNCC.getRowCount() <= 0	

❖ Test report

Số lượng test case	4
Số test case Pass	3
Số test case Fail	1
Số test case Error	0
Số test case chưa thực hiện	0
Tỉ lệ thành công	75%

10. Quản lý thành viên

Đặc tả: Cửa hàng thực hiện chức năng quản lý danh sách thành viên thông qua các phương thức như đọc, sửa, xóa, tìm kiếm nhằm giúp cửa hàng nắm bắt được số lượng, thông tin của các thành viên một cách chính xác để phục vụ cho các nội dụng khác của cửa hàng.

10.1. Kiểm thử hộp đen

❖ Thành phần: 10 TextField là search, lọc thông tin theo ngày sinh chặn trên, chặn dưới; lọc thông tin theo điểm chặn trên, chặn dưới, Tên thành viên, số điện thoại, mail, điểm, mật khẩu;1 Label mã thành viên;1

ComboBox tiêu chí tìm kiếm; 1 DatePicker ngày sinh; 3 Button là sửa, xóa, load.

Tên trường	Bắt buộc	Yêu cầu	Ghi chú
Mã thành viên (Label)	Có	Không được bỏ trống, mã thành viên sẽ tự động được tạo khi đăng ký một tài khoản thành viên	Dữ liệu ở đây dùng để sửa, xóa
Tên thành viên (Textfield)	Có	Không được bỏ trống	Dữ liệu ở đây dùng để sửa, xóa
Số điện thoại (Textfield)	Có	Không được bỏ trống, số điện thoại phải có 10 số	Dữ liệu ở đây dùng để sửa, xóa
Mail (Textfield)	Có	Không được bỏ trống, đúng cú pháp	Dữ liệu ở đây dùng để sửa, xóa
Điểm (Textfield)	Có	Không được bỏ trống	Dữ liệu ở đây dùng để sửa, xóa
Mật khẩu (Textfield)	Có	Không được bỏ trống	Dữ liệu ở đây dùng để sửa, xóa
Search (Textfield)	Không	Nhập thông tin cần tìm sau khi đã chọn tiêu chí tìm kiếm	Nhập từ khóa để tìm kiếm

Chặn trên cho lọc thông tin theo ngày sinh (Textfield)	Không	Nhập ngày sinh chặn trên	Chọn ngày sinh kết thúc để lọc thông tin
Chặn dưới cho lọc thông tin theo ngày sinh (Textfield)	Không	Nhập ngày sinh chặn dưới	Chọn ngày bắt đầu để lọc thông tin
Chặn trên cho lọc thông tin theo điểm (Textfield)	Không	Nhập điểm chặn trên	Chọn ngày kết thúc để lọc thông tin
Chặn dưới cho lọc thông tin theo điểm (Textfield)	Không	Nhập điểm chặn dưới	Chọn ngày bắt đầu để lọc thông tin
Tiêu chí tìm kiếm (ComboBox)	Không	Click chọn tiêu chí tìm kiếm thông tin thành viên	Tìm kiếm theo mã thành viên, tên thành viên,…
Ngày sinh (DatePicker)	Có	Không được bỏ trống, click chọn ngày sinh hợp lệ	Dữ liệu ở đây dùng để sửa, xóa

- ❖ Các trường hợp kiểm thử (bảng quyết định)

  BangQuyetDinh_QL_ThanhVien_WhiteBoxTesting.xlsx

- ❖ Test case

ID	Test Step	Test data	Expected Result	A Result
1	-Nhập mã thành viên. -Nhấn nút xóa.	Mã thành viên: TV_01	Xóa thành công thành viên	Pass
2		Mã thành viên: TV_999 (Mã thành viên không tồn tại).	Xóa thành công thành viên	Fail
3	-Thực hiện tìm kiếm thành viên muốn chỉnh sửa -Nhập thông tin muốn chỉnh sửa -Nhấn nút sửa.	TênTV:“ABCD”(cũ) → “NVA”(mới)	Sửa thành công thông tin thành viên	Pass
4		TênTV:“ABCD”(cũ) → “”(mới)(tên không hợp lệ)	Sửa thành công thông tin thành viên	Fail

❖ Test report

Số lượng test case	4
Số test case Pass	2
Số test case Fail	2
Số test case Error	0
Số test case chưa thực hiện	0
Tỉ lệ thành công	50%

10.2. Kiểm thử hộp trắng

Các nghiệp vụ như thêm, xóa, sửa thành viên đều tương tự như những nghiệp vụ đã kiểm thử của những chức năng quản lý khác như tài khoản, nhân viên, tài khoản. Đều có chung dòng điều khiển cơ bản nên ở đây ta sẽ suy ra luôn test case

❖ Test case thêm thành viên

ID	Basis path	Data	Expected results
1	1	Mảng arr chứa dữ liệu là thông tin thành viên người quản lý nhập vào, cụ thể: Arr[0]: “TV_01”; Arr[1]: “NVA”; Arr[2]: “20/12/2002”; Arr[3]:“0934772284”; Arr[4]:nva@gmail.com Arr[5]:0 Arr[6]:123	Thành viên mới được thêm vào csdl. Đồng thời reset và cập nhật lại listTV (danh sách chứa thông tin các thành viên được lấy xuống từ csdl).

❖ Test report

Số lượng test case	1
Số test case Pass	1
Số test case Fail	0
Số test case Error	0
Số test case chưa thực hiện	0
Tỉ lệ thành công	100%

❖ Test case sửa thông tin thành viên

ID	Basis path	Data	Expected results

1	1	Mảng arr chứa dữ liệu là thông tin thành viên người quản lý nhập vào, cụ thể: Arr[0]: “TV_01”; Arr[1]: “NVA”; Arr[2]: “29/12/2002”; Arr[3]:“0934774284”; Arr[4]:nva_11@gmail.com Arr[5]:0 Arr[6]:123	Thành viên được cập nhật lại ở csdl. Đồng thời reset và cập nhật lại listTV (danh sách chứa thông tin các thành viên được lấy xuống từ csdl).

❖ Test report

Số lượng test case	1
Số test case Pass	1
Số test case Fail	0
Số test case Error	0
Số test case chưa thực hiện	0
Tỉ lệ thành công	100%

❖ Test case xóa thành viên

ID	Basis path	Data	Expected results
1	1	Chuỗi chứa dữ liệu là mã thành viên người quản lý nhập vào, cụ thể: Ma_TV: TV_11	Thành viên được chọn bị xóa khỏi csdl. Đồng thời reset và cập nhật lại listTV (danh sách chứa thông

			tin các thành viên được lấy xuống từ csdl).

❖ Test report

Số lượng test case	1
Số test case Pass	1
Số test case Fail	0
Số test case Error	0
Số test case chưa thực hiện	0
Tỉ lệ thành công	100%

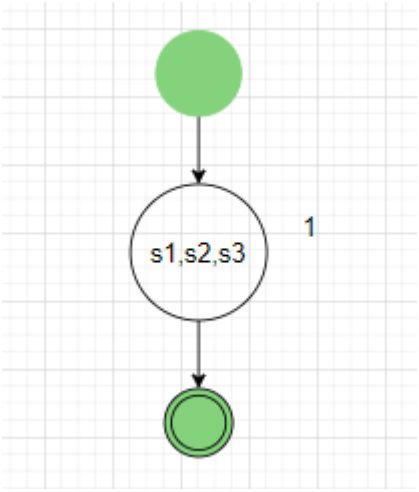
Cập nhật điểm tích lũy

❖ Mã nguồn

```
    public void capNhatDiemTichLuy(String []str){
/*s1*/        thanhVienDAL.updatePoint(str);
/*s2*/         this.resetListTV();
/*s3*/         this.loadListTV();
    }
```

Hình 10.2.1. Mã nguồn cập nhật điểm tích lũy

❖ Đồ thị dòng điều khiển cơ bản



Hình 10.2.2. Đồ thị dòng điều khiển cơ bản cập nhật điểm tích lũy

⇨ Độ phức tập Cyclomatic M=N+1, với N là số nút điều kiện rẽ nhánh nhị phân.

⇨ M = 0 + 1 = 1 (Với N=0).

⇨ Đường độc lập tuyến tính cơ bản duy nhất là 1.

❖ Test case

ID	Basis path	Data	Expected results
1	1	Chuỗi chứa dữ liệu là điểm tích lũy của thành viên người quản lý nhập vào, cụ thể: Diem: 100	Reset và cập nhật lại điểm tích lũy của thành viên trong listTV (danh sách chứa thông tin các thành viên được lấy xuống từ csdl).

❖ Test report

Số lượng test case	1
Số test case Pass	1
Số test case Fail	0
Số test case Error	0
Số test case chưa thực hiện	0

Tỉ lệ thành công	100%

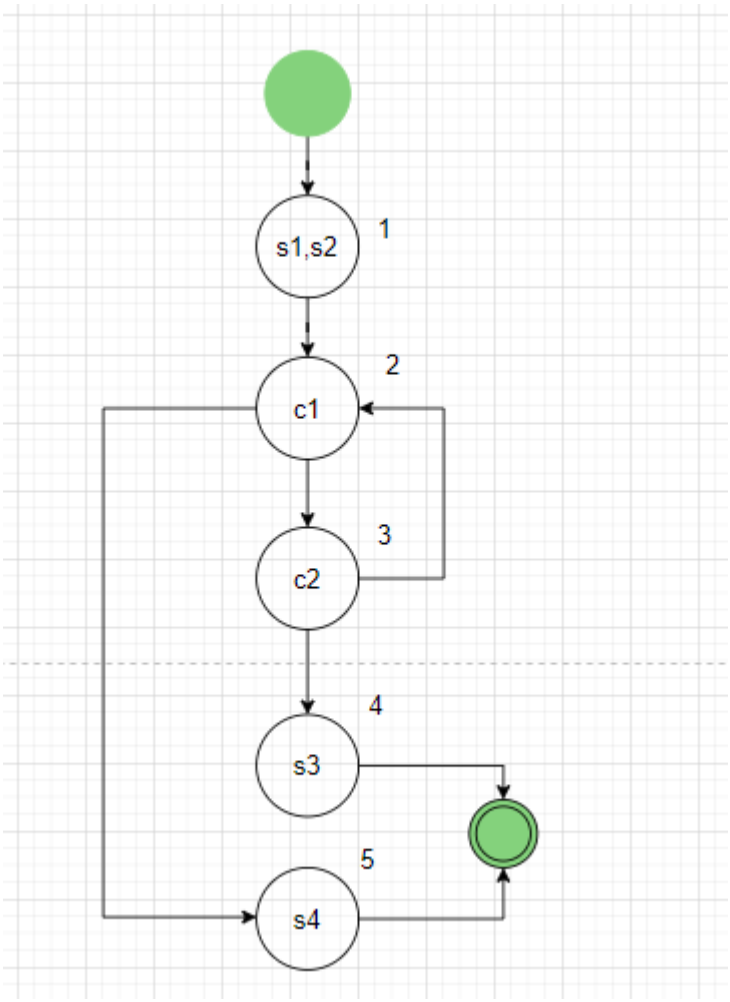
Tìm kiếm thành viên dựa vào mã thành viên

❖ Mã nguồn

```
    public ThanhVienDTO getTvByMaTV(String ma_thanh_vien){
/*s1*/          this.resetListTV();
/*s2*/          this.loadListTV();

/*c1*/          for (ThanhVienDTO thanhVienDTO : listTV) {
/*c2*/              if(thanhVienDTO.getMa_thanh_vien().equals( anObject: ma_thanh_vien)){
/*s3*/                  return thanhVienDTO;
                  }
              }
/*s4*/          return null;
    }
```

Hình 10.2.3. Mã nguồn tìm kiếm thành viên dựa trên mã thành viên

❖ Đồ thị dòng điều khiển cơ bản



Hình 10.2.4. Đồ thị dòng điều khiển cơ bản chức năng tìm kiếm thành viên dựa trên mã thành viên

⇨ Độ phức tạp Cyclomatic M = N + 1, với N là số nút điều kiện rẽ nhánh nhị phân.

⇨ M=2+1=3 (với N=2).

⇨ Đường độc lập tuyến tính cơ bản:

- 1 – 2 – 5
- 1 – 2 – 3 – 4
- 1 – 2 – 3 – 2 – 3 – 5

❖ Test case

ID	Basis path	Data	Expected results
1	1-2-5	Ma_TV: TV_15 (Nhập từ bàn phím của người quản lý)	Trả về null (do listTV rỗng, chưa có dữ liệu trong csdl tức chưa có thành viên nào)
2	1-2-3-4	Ma_TV: TV_01 (Nhập từ bàn phím của người quản lý)	Trả về thành viên có mã TV là TV_01
3	1-2-3-2-3-5	Ma_TV: TV_15 (Nhập từ bàn phím của người quản lý)	Trả về null do không tìm thấy thành viên nào có mã TV là TV_15 trong listTV

❖ Test report

Số lượng test case	3
Số test case Pass	3
Số test case Fail	0
Số test case Error	0
Số test case chưa thực hiện	0
Tỉ lệ thành công	100%

11.Quản lý phiếu giảm giá

Đặc tả: Người quản lý sẽ quản lý các phiếu giảm giá của cửa hàng bao gồm việc tạo mới, thêm, sửa, xóa, load.

11.1. Kiểm thử hộp đen

❖ Thành phần:7 Textfield (Tên phiếu, giá trị giảm, search, tìm kiếm theo khoảng ngày, tìm kiếm theo khoảng tiền giảm);1 Label (Mã giảm giá); 2 DatePicker (ngày bắt đầu, ngày kết thúc);1 ComboBox (tiêu chí tìm kiếm); 5 Button (Tạo mới, Thêm, Sửa, Xóa, Load).

Tên trường	Bắt buộc	Yêu cầu	Ghi chú
Mã giảm giá (Label)	Có	Không được bỏ trống, hệ thống thiết lập	Dữ liệu ở đây dùng để sửa, xóa, tạo mới, thêm
Tên phiếu (Textfield)	Có	Không được bỏ trống	Dữ liệu ở đây dùng để sửa, xóa, tạo mới, thêm
Giá trị giảm (Textfield)	Có	Không được bỏ trống	Dữ liệu ở đây dùng để sửa, xóa, tạo mới, thêm
Search (Textfield)	Không	Sau khi chọn tiêu chí tìm kiếm, nhập từ khóa vào đề thực hiện tìm kiếm	Nhập từ khóa như mã phiếu, tên phiếu…
Chặn dưới tìm kiếm theo ngày (Textfield)	Không	Nhập chặn dưới cho tìm kiếm theo ngày	Nhập ngày bắt đầu cho tìm kiếm
Chặn trên tìm kiếm theo ngày (Textfield)	Không	Nhập chặn trên cho tìm kiếm theo ngày	Nhập ngày kết thúc cho tìm kiếm
Chặn dưới tìm kiếm theo giá trị giảm (Textfield)	Không	Nhập chặn dưới cho tìm kiếm theo giá trị giảm	Nhập ngày bắt đầu cho tìm kiếm

Chặn trên tìm kiếm theo giá trị giảm (Textfield)	Không	Nhập chặn trên cho tìm kiếm theo giá trị giảm	Nhập ngày kết thúc cho tìm kiếm
Ngày bắt đầu (DatePicker)	Có	Không được bỏ trống	Dữ liệu ở đây dùng để sửa, xóa, tạo mới, thêm
Ngày kết thúc (DatePicker)	Có	Không được bỏ trống	Dữ liệu ở đây dùng để sửa, xóa, tạo mới, thêm
Tiêu chí tìm kiếm (ComboBox)	Không	Nhấp chọn tiêu chí tìm kiếm	Chọn các tiêu chí tìm kiếm: Mã phiếu, tên phiếu, giá trị giảm…

❖ Các trường hợp kiểm thử (bảng quyết định)

BangQuyetDinh_QL_PhieuGiamGia_WhiteBoxTesting.xlsx

❖ Test case

ID	Test Step	Test data	Expected Result	A Result
1	-Nhấn tạo mới một mã phiếu. -Nhập tên phiếu giảm giá, giá trị giảm, ngày bắt đầu giảm và ngày kết thúc. -Nhấn nút thêm.	Mã phiếu:06 Tên phiếu:”ABCD” Giảm: 20% Ngày bắt đầu:20/10/2022 Ngày kết thúc: 2/4/2023	Tạo mới thành công một phiếu giảm giá	Pass
2		Mã phiếu:06 Tên phiếu: Giảm: 20% Ngày bắt đầu:20/10/2022	Tạo mới thành công một phiếu giảm giá	Fail

		Ngày kết thúc: 2/4/2023		
3	-Nhập mã phiếu. -Nhập tên phiếu giảm giá, giá trị giảm, ngày bắt đầu giảm và ngày kết thúc. -Nhấn nút sửa.	Mã phiếu:06 Tên phiếu:"ABCD" Giảm: 20% Ngày bắt đầu:20/10/2022 Ngày kết thúc: 2/4/2023	Sửa thành công một phiếu giảm giá	Pass
3		Mã phiếu:100 (Mã phiếu chưa tồn tại) Tên phiếu:"ABCD" Giảm: 20% Ngày bắt đầu:20/10/2022 Ngày kết thúc: 2/4/2023	Tạo mới thành công một phiếu giảm giá	Fail
4	-Nhập mã phiếu. -Bấm nút xóa.	Mã phiếu: 06	Xóa thành công một phiếu giảm giá	Pass
5		Mã phiếu:100 (Mã phiếu chưa tồn tại)	Xóa thành công một phiếu giảm giá	Fail

❖ Test report

Số lượng test case	6
Số test case Pass	3
Số test case Fail	3
Số test case Error	0
Số test case chưa thực hiện	0
Tỉ lệ thành công	50%

11.2. Kiểm thử hộp trắng

Các chức nằng thêm, sửa, xóa phiếu giảm giá và cập nhật tình trạng phiếu giảm giá đều tương tự nhau về đòng điều khiển cơ bản và đều có chung đường độc lập tuyến tính cơ bản.

❖ Mã nguồn lần lượt các chức năng thêm, sửa, xóa phiếu giảm giá và cập nhật tình trạng phiếu giảm giá.

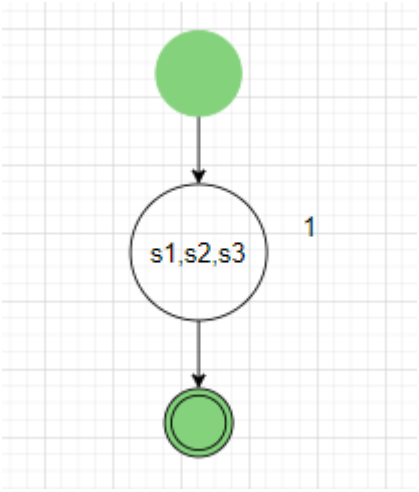
```
    public void ThemPGG(String[] str) {
/*s1*/          phieuGiamGiaDAL.addData( arr: str);
/*s2*/          this.resetListPGG();
/*s3*/          this.loadListPGG();
    }

    public void suaPGG(String[] str) {
/*s1*/          phieuGiamGiaDAL.updateData( arr: str);
/*s2*/          this.resetListPGG();
/*s3*/          this.loadListPGG();
    }

    public void delPGG(String maPGG) {
/*s1*/          phieuGiamGiaDAL.delData( ma_phieu_giam_gia: maPGG);
/*s2*/          this.resetListPGG();
/*s3*/          this.loadListPGG();
    }
```

Hình 11.2.1. Mã nguồn các chức năng thêm, xóa, sửa các phiếu giảm giá

```
    public void updateTrangThai(String ma_phieu_giam) {
/*s1*/          phieuGiamGiaDAL.updateTrangThaiPhieu( ma_pgg: ma_phieu_giam);
/*s2*/          this.resetListPGG();
/*s3*/          this.loadListPGG();
    }
```

Hình 11.2.2. Mã nguồn các chức năng cập nhật trạng thái phiếu giảm giá

❖ Đồ thị dòng điều khiển cơ bản



Hình 11.2.3. Đồ thị dòng điều khiển cơ bản các chức năng kể trên

⇨ Độ phức tạp Cyclomatic M = N + 1, với N là số nút điều kiện rẽ nhánh nhị phân.

⇨ M = 0 + 1 = 1 (với N = 0)

⇨ Đường độc lập tuyến tính cơ bản duy nhất là 1

❖ Test case thêm phiếu giảm giá

ID	Basis path	Data	Expected results
1	1	Maphieu: “01” Tên phiếu: “ABC” Giảm: “20%” Ngày bắt đầu:”11/10/2022” Ngày kết thúc:”11/12/2022”	Phiếu giảm giá mới được thêm vào csdl. Đồng thời reset và cập nhật lại listPGG

❖ Test report

Số lượng test case	1
Số test case Pass	1
Số test case Fail	0

Số test case Error	0
Số test case chưa thực hiện	0
Tỉ lệ thành công	100%

❖ Test case sửa phiếu giảm giá

ID	Basis path	Data	Expected results
1	1	Maphieu: “01” Tên phiếu: “ABC” Giảm: “30%” Ngày bắt đầu:”11/10/2022” Ngày kết thúc:”1/2/2023”	Phiếu giảm giá được cập nhật lại ở csdl. Đồng thời reset và cập nhật lại listPGG.

❖ Test report

Số lượng test case	1
Số test case Pass	1
Số test case Fail	0
Số test case Error	0
Số test case chưa thực hiện	0
Tỉ lệ thành công	100%

❖ Test case xóa phiếu giảm giá

ID	Basis path	Data	Expected results
1	1	Maphieu: “01” (Mã phiếu giảm giá được người quản lý chọn).	Phiếu giảm giá được chọn bị xóa khỏi csdl. Đồng thời reset và cập nhật lại listPGG.

❖ Test report

Số lượng test case	1
Số test case Pass	1
Số test case Fail	0
Số test case Error	0
Số test case chưa thực hiện	0
Tỉ lệ thành công	100%

❖ Test case cập nhật phiếu giảm giá

ID	Basis path	Data	Expected results
1	1	Maphieu: “01” (Mã phiếu giảm giá được người quản lý chọn).	Cập nhật lại trạng thái phiếu giảm giá rồi lưu vào csdl. Reset và cập nhật lại listPGG.

❖ Test report

Số lượng test case	1
Số test case Pass	1
Số test case Fail	0
Số test case Error	0
Số test case chưa thực hiện	0
Tỉ lệ thành công	100%

12.Quản lý chương trình khuyến mãi

Đặc tả: Người quản lý sẽ quản lý các chương trình khuyến mãi của cửa hàng bao gồm việc tạo mới, sửa, xóa, áp dụng và hủy bỏ chương trình khuyến mãi.

12.1. Kiểm thử hộp đen

❖ Thành phần: bao gồm: 6 Textfield là Tên CTKM, ngày bắt đầu, ngày kết thúc, tìm kiếm, khuyến mãi từ ngày (date from), khuyến mãi đến ngày (date to) ; 2 Label là Mã CTKM và Trạng thái khuyến mãi (để biết CTKM đó có được áp dụng hay không); 8 Button là tạo mã mới, thêm, sửa, xóa, áp dụng, ngưng, load; 1 ComboBox là tìm kiếm theo tiêu chí nào.

Tên trường	Bắt buộc	Yêu cầu	Ghi chú
Label Mã CTKM	Có	Không được bỏ trống, Click vào button tạo mã mới để hệ thống tự tạo mới mã KM	Dữ liệu trong trường này dùng để tạo mới, sửa, hoặc xóa CTKM.
TextField Mật khẩu	Có	Không được bỏ trống	Dữ liệu trong trường này dùng để tạo mới, sửa, hoặc xóa CTKM.
TextField Ngày bắt đầu	Có	Không được bỏ trống	Dữ liệu trong trường này dùng để tạo mới, sửa, hoặc xóa CTKM.
TextField Ngày kết thúc	Có	Không được bỏ trống	Dữ liệu trong trường này dùng để tạo mới, sửa, hoặc xóa CTKM.
Label trạng thái	Không	Nhấp vào button áp dụng hoặc button ngưng để	Sau khi CTKM được áp dụng sẽ có trạng thái “áp

		bật / tắt trạng thái của chương trình (mặc định một CTKM có trạng thái "không áp dụng").	dụng", ngược lại sẽ có trạng thái "không áp dụng".
ComboBox Loại tìm kiếm	Không	Chỉ Click chọn tiêu chí tìm kiếm khi cần tìm kiếm CKTM, sau khi nhập thông tin cần tìm kiếm nhấn button tìm kiếm	Tìm kiếm theo mã KM, tên KM, trạng thái…
TextField khuyến mãi từ ngày (date from)	Không	Nhập ngày bắt đầu khuyến mãi	Tìm kiếm CTKM theo ngày khuyến mãi
TextField khuyến mãi từ ngày (date to)	Không	Nhập ngày kết thúc khuyến mãi	Tìm kiếm CTKM theo ngày khuyến mãi, sau khi nhập ngày tìm kiếm, hệ thống từ động lọc ra danh sách khuyến mãi hợp lệ

❖ Các trường hợp kiểm thử (bảng quyết định)

BangQuyetDinh_QL_CTKM_WhiteBoxTesting.xlsx

❖ Test case

ID	Test step	Test data	Expected Result	A Result
1	**Trường hợp chưa nhấn tạo mã mới hoặc chưa nhấn vào ctkm cần cập nhật** - Nhập các thông tin của ctkm - Nhấn button thêm, sửa, xóa **Trường hợp tạo mã ctkm mới hoặc nhấn vào ctkm cần cập nhật trước khi thêm, xóa, sửa** - Nhấn chọn button tạo mã mới hoặc nhấn chọn ctkm cần cập nhật - Nhấn các button thêm, xóa, sửa.	Mã ctkm: null (rỗng) Tên ctkm: KM_T11 Ngày bắt đầu:10/11/2022 Ngày kết thúc: 30/11/2022 Click button thêm, sửa	Thông báo lỗi	Pass
2		Mã ctkm: KM_01 Tên ctkm: KM_T11 Ngày bắt đầu:10/11/2022 Ngày kết thúc: 30/11 Click button thêm, sửa	Thông báo thành công	Fail (sai định dạng ngày)
3	**Lọc ctkm theo các chỉ tiêu tìm kiếm hoặc lọc cktm theo các khoảng cận trên và dưới.**	Mã ctkm: KM_01 Tên ctkm: KM_T11 Ngày bắt đầu:10/11/2022 Ngày kết thúc: 30/11/2022	Thông báo thành công	Pass

	- Click vào combobox chọn tiêu chí tìm kiếm	Click button thêm, sửa hoặc xóa		
4	- Nhập đúng định dạng dữ liệu cần tìm - Tương tự nhập dữ liệu cần tìm theo thứ tự cận dưới trước rồi đến cận trên.	Click vào combobox chọn tìm kiếm theo mã ctkm Search: 01	Ctkm có mã là 01 được tìm thấy	Fail (sai định dạng mã ctkm)
5		Nhập ngày bắt đầu Ngày bắt đầu: 11/11/2022 Nhập ngày kết thúc Ngày kết thúc: 30/11/2022	Các ctkm có ctkm nằm trong khoảng tg nhập sẽ được lọc ra	Pass

❖ Test report

Số lượng test case	5
Số test case Pass	3
Số test case Fail	2
Số test case Error	0
Số test case chưa thực hiện	0
Tỉ lệ thành công	60%

12.2. Kiểm thử hộp trắng

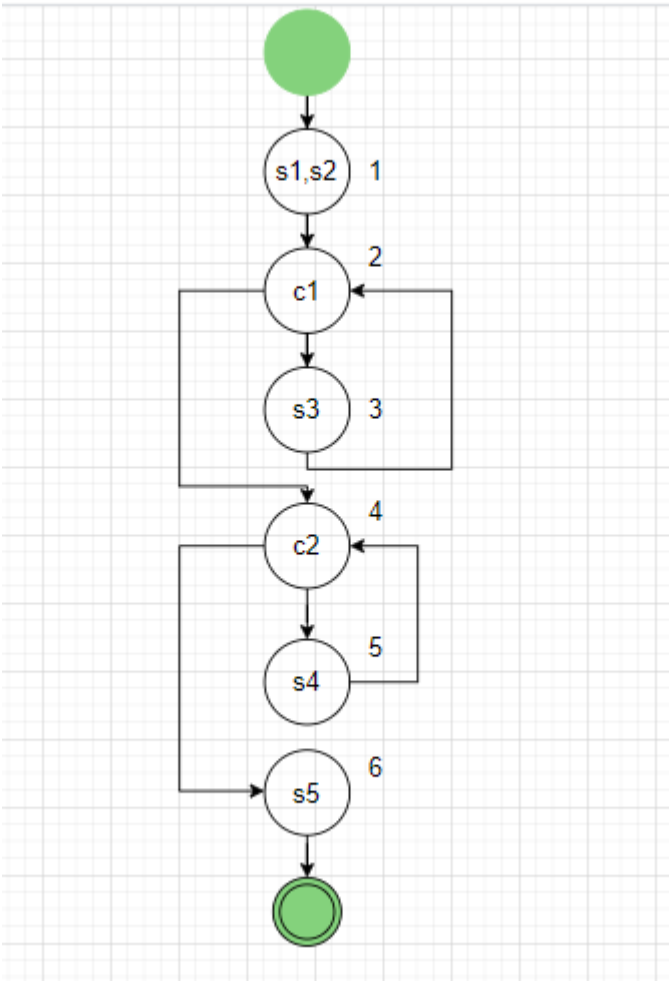
Lấy dữ liệu từ csdl và thiết lập vào danh sách hiển thị các chương trình khuyến mãi

❖ Mã nguồn

```
    public void setListTableKM(DefaultTableModel model){
/*s1*/  this.resetListKM();
/*s2*/  this.loadListKM();

/*c1*/  while(model.getRowCount() > 0){
/*s3*/      model.removeRow(0);
        }
/*c2*/  for (ChuongTrinhKMDTO chuongTrinhKMDTO : listKM) {
/*s4*/      model.addRow(new Object[]{
              chuongTrinhKMDTO.getMa_ctkm(),
              chuongTrinhKMDTO.getTen_ctkm(),
              chuongTrinhKMDTO.getNgay_bat_dau(),
              chuongTrinhKMDTO.getNgay_ket_thuc(),
              chuongTrinhKMDTO.getTrang_thai()
          });
        }
/*s5*/  model.fireTableDataChanged();
    }
```

Hình 12.2.1. Mã nguồn chức năng Lấy dữ liệu từ csdl và thiết lập vào danh sách hiển thị các chương trình khuyến mãi

❖ Đồ thị dòng điều khiển cơ bản



Hình 12.2.2. Đồ thị dòng điều khiển cơ bản chức năng Lấy dữ liệu từ csdl và thiết lập vào danh sách hiển thị các chương trình khuyến mãi

⇨ Độ phức tạp Cyclomatic M = N + 1, với N là số nút điều kiện rẽ nhánh nhị phân.

⇨ M = 2 + 1 = 3

⇨ Có 3 đường độc lập tuyến tính cơ bản là:

- 1 – 2 – 4 – 6
- 1 – 2 – 3 – 2 – 4 – 6
- 1 – 2 – 4 – 5 – 4 – 6

❖ Test case

ID	Basis path	Data	Expected results

1	1-2-4-6	model.getRowCount() <= 0	Không cần xóa dữ liệu modelCTKM hiện có vì nó đang rỗng
2	1-2-3-2-4-6	model.getRowCount() - 0 listKM rỗng (listKM.size() = 0)	Reset dữ liệu trong modelCTKM hiện tại. List chứa dữ liệu được lấy về từ csdl rỗng nên không có dữ liệu để nạp vào model.
3	1-2-4-5-4-6	model.getRowCount() <= 0 listKM có dữ liệu (listKM.size() - 0)	Không cần xóa dữ liệu modelCTKM hiện có vì nó đang rỗng. Nạp dữ liệu từ listKM vào modelCTKM để hiển thị lên giao diện người dùng.

❖ Test report

Số lượng test case	3
Số test case Pass	3
Số test case Fail	0
Số test case Error	0
Số test case chưa thực hiện	0
Tỉ lệ thành công	100%

Các đổi trạng thái chương trình khuyến mãi (bật/tắt trạng thái CTKM), thêm sản phẩm, xóa sản phẩm, sửa sản phẩm đều tương tự nhau về đòng điều khiển cơ bản và đều có chung đường độc lập tuyến tính cơ bản và nó đều tương tự với những chức năng quản lý từng xét nên ta sẽ suy ra các test case.

❖ Test case đổi trạng thái CTKM

ID	Basis path	Data	Expected results
1	1	Ma_ctkm: KM_01 Tt: true (Ma_ctkm là mã của ctkm được người dùng click vào, trạng thái là true nếu ctkm đó chưa áp dụng và ngược lại là false nếu nó áp dụng rồi)	Trạng thái của ctkm đó được cập nhật trên csdl. Đồng thời reset và cập nhật lại listKM (danh sách chứa các ctkm được lấy xuống từ csdl).

❖ Test report

Số lượng test case	1
Số test case Pass	1
Số test case Fail	0
Số test case Error	0
Số test case chưa thực hiện	0
Tỉ lệ thành công	100%

❖ Test case thêm CTKM

ID	Basis path	Data	Expected results
1	1	Mảng arr chứa dữ liệu là thông tin ctkm người quản lý nhập vào, cụ thể: Arr[0]: “KM_01”;	Ctkm mới được thêm vào csdl. Đồng thời reset và cập nhật lại listKM (danh sách chứa các

		Arr[1]: “CTKM_T11”; Arr[2]: “10/11/2022”; Arr[3]: “30/11/2022”;	ctkm được lấy xuống từ csdl).

❖ Test report

Số lượng test case	1
Số test case Pass	1
Số test case Fail	0
Số test case Error	0
Số test case chưa thực hiện	0
Tỉ lệ thành công	100%

❖ Test case sửa CTKM

ID	Basis path	Data	Expected results
1	1	Mảng arr chứa dữ liệu là thông tin ctkm người quản lý nhập vào, cụ thể: Arr[0]: “KM_01”; Arr[1]: “11_THANG_11”; Arr[2]: “10/11/2022”; Arr[3]: “12/11/2022”;	Ctkm được cập nhật lại ở csdl. Đồng thời reset và cập nhật lại listKM (danh sách chứa các ctkm được lấy xuống từ csdl).

❖ Test report

Số lượng test case	1
Số test case Pass	1
Số test case Fail	0

Số test case Error	0
Số test case chưa thực hiện	0
Tỉ lệ thành công	100%

❖ Test case xóa CTKM

ID	Basis path	Data	Expected results
1	1	Ma_ctkm: KM_01 (Ma_ctkm là mã của ctkm được người quản lý click vào).	Ctkm được chọn bị xóa khỏi csdl. Đồng thời reset và cập nhật lại listKM (danh sách chứa các ctkm được lấy xuống từ csdl).

❖ Test report

Số lượng test case	1
Số test case Pass	1
Số test case Fail	0
Số test case Error	0
Số test case chưa thực hiện	0
Tỉ lệ thành công	100%

13.Chức năng thống kê

Đặc tả: Chức năng thống kê doanh thu hàng tháng, hàng năm, tìm ra hóa dơn có doanh thu cao nhất hoặc nhỏ nhất,lọc hóa đơn theo từng tiêu chí khác nhau như mã đơn, ngày xuất, mã nhân viên xuất đơn,….

13.1. Kiểm thử hộp đen

❖ Thành phần: bao gồm 1 textField tìm kiếm để lọc hóa đơn và đơn nhập theo mã đơn, ngày xuất, mã nhân viên phụ trách xuất, tên nhân viên xuất đơn, mã thành viên hoặc tên thành viên (nếu có). 2 textField dùng để lọc các đơn theo khoảng, cụ thể là khoảng giá trị đơn, khoảng ngày (từ ngày … đến ngày …). 4 button lần lượt là Re-load, xem chi tiết, xuất file Excel và xác nhận filter, 1 combobox giúp chọn tiêu chí tìm kiếm (theo mã, theo ngày,…) và 1 combobox giúp chọn tiêu chí tìm kiếm theo khoảng giá trị (từ ngày bào nhiêu đến ngày bao nhiêu, giá trị đơn từ bao nhiều đến bao nhiêu), 3 label lần lượt là label giúp thống kê tự động doanh thu (hoặc chi phí) theo ngày, theo tháng, theo năm và 2 label giúp tự động tìm ra giá trị lơn nhất và nhỏ nhất của hóa đơn (hoặc đơn nhập), 1 table giúp hiện thị danh sách hóa đơn và đơn phiếu nhập.

Tên trường	Bắt buộc	Yêu cầu	Ghi chú
TextField tìm kiếm	Không	Sau khi nhập dữ liệu cần tìm thì click vào button xác nhận.	Dữ liệu trong trường này dùng để lọc danh sách hóa đơn hoặc danh sách đơn nhập theo các tiêu chí như mã, ngày xuất, mã nhân viên xuất,… Việc lọc danh sách hóa đơn hoặc phiếu nhập sẽ tự động khi người quản lý nhập

			đúng định dạng giá trị cần lọc.
TextField tìm kiếm từ	Không	Nhập chính xác giá trị tối thiểu cần tìm	Danh sách hóa đơn hoặc danh sách phiếu nhập được lọc từ giá trị của trường này.
TextField tìm kiếm đến	Có	Nhập chính xác giá trị tối đa cần tìm	Danh sách hóa đơn hoặc danh sách phiếu nhập được lọc đến giá trị của trường này. Việc lọc được tự động khi người quản lý nhập xong giá trị tại trương này mà không cần click xác nhận từ bất kỳ button nào.
TextField từ ngày	Không	Nhập chính xác ngày cần tìm	Danh sách các hóa đơn hoặc đơn nhập được lọc từ ngày mà quản lý nhập
TextField đến ngày	Không	Nhập chính xác ngày cần tìm	Danh sách các hóa đơn hoặc đơn nhập được lọc đến ngày mà quản lý nhập. Hệ thống tự động

			lọc ra danh sách hóa đơn phù hợp khi người quản lý nhập đúng định dạng
ComboBox Loại tìm kiếm	Không	Chỉ Click chọn tiêu chí tìm kiếm khi cần tìm kiếm	Tìm kiếm theo mã, theo ngày, theo mã nhân viên xuất đơn.
ComboBox Loại tìm kiếm theo khoảng giá trị	Không	Chỉ Click chọn tiêu chí tìm kiếm theo khoảng giá trị khi cần tìm kiếm	Tìm kiếm theo mã, theo tổng tiền (từ bao nhiêu đến bao nhiêu), theo điểm (từ bao nhiêu đến bao nhiêu).

❖ Các trường hợp kiểm thử (bảng quyết định): BangQuyetDinh_ThongKe_WhiteBoxTesting.xlsx

13.2. Kiểm thử hộp trắng

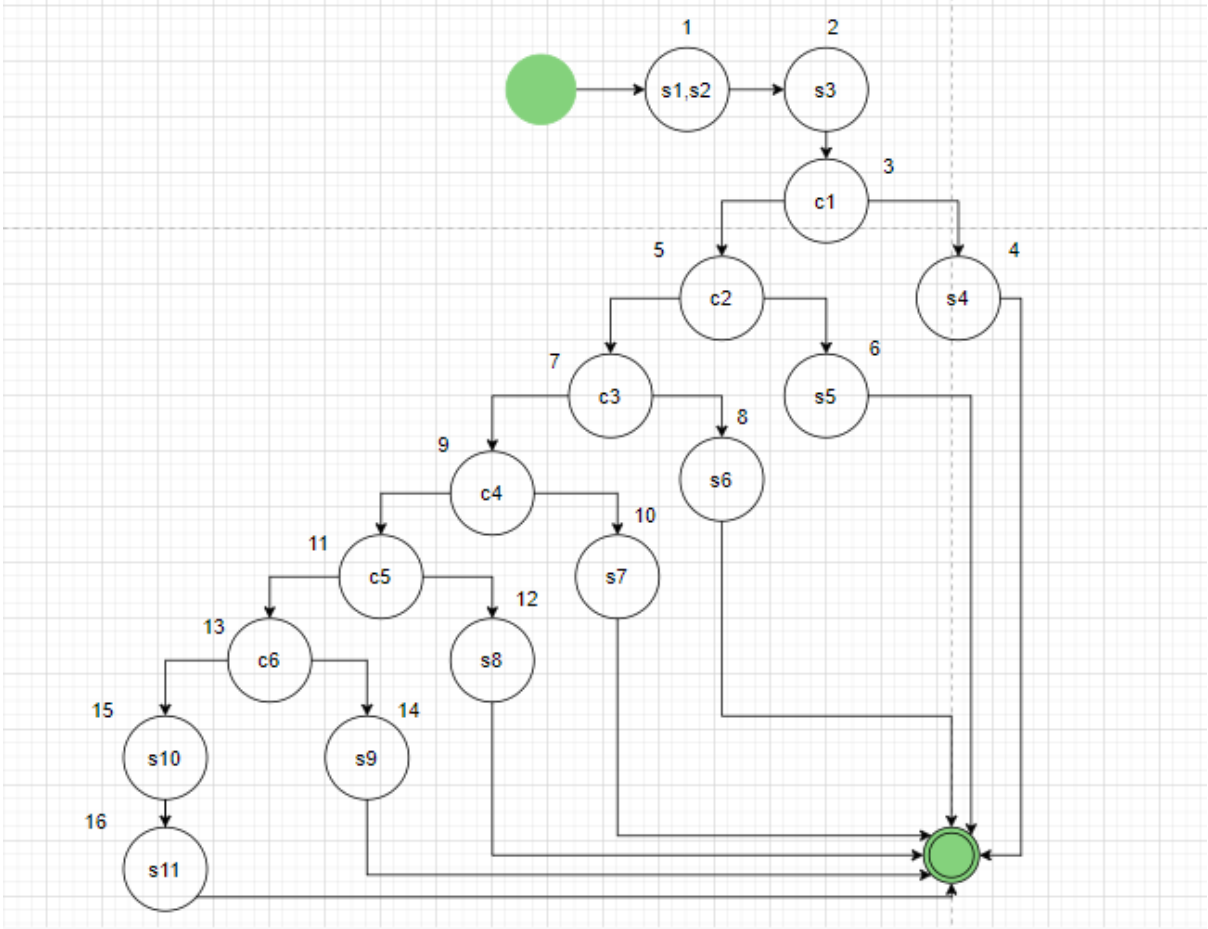
Tìm kiếm theo tiêu chí

❖ Mã nguồn

```
    private void txtSearchHDKeyReleased(java.awt.event.KeyEvent evt) {
/*s1*/  String search = txtSearchHD.getText();
/*s2*/  String typeSearch = String.valueOf(cbxSearchHD.getSelectedItem());
/*s3*/  switch (typeSearch) {
/*c1*/      case "Mã hóa đơn":
/*s4*/          hoaDonBLL.timKiemHoaDonTheoMaHD(modelHD,search);
                break;
/*c2*/      case "Ngày xuất":
/*s5*/          hoaDonBLL.timKiemHoaDonTheoNgay(modelHD,search);
                break;
/*c3*/      case "Mã nhân viên":
/*s6*/          hoaDonBLL.timKiemHoaDonTheoMaNV(modelHD,search);
                break;
/*c4*/      case "Tên nhân viên":
/*s7*/          hoaDonBLL.timKiemHoaDonTheoTenNV(modelHD,search);
                break;
/*c5*/      case "Mã thành viên":
/*s8*/          hoaDonBLL.timKiemHoaDonTheoMaTV(modelHD,search);
                break;
/*c6*/      case "Tên thành viên":
/*s9*/          hoaDonBLL.timKiemHoaDonTheoTenTV(modelHD,search);
                break;
/*c7*/      case "Mã voucher":
/*s10*/         hoaDonBLL.timKiemHoaDonTheoMaPGG(modelHD,search);
                break;
/*c8*/      default:
/*s11*/         return;
        }
    }
```

Hình 13.2.1. Mã nguồn chức năng tìm kiếm theo tiêu chí

❖ Đồ thị dòng điều khiển cơ bản



Hình 13.2.2. Đồ thị dòng điều khiển cơ bản của chức năng tìm kiếm theo tiêu chí

⇨ Độ phức tạp Cyclomatic M = N + 1, với N là số nút điều kiện rẽ nhánh nhị phân.

⇨ M = 6 + 1 = 7(với N = 6)

⇨ Đường độc lập tuyến tính cơ bản:

- 1 – 2 – 3 – 4
- 1 – 2 – 3 – 5 – 6
- 1 – 2 – 4 – 5 – 7 – 8
- 1 – 2 – 4 – 5 – 7 – 9 – 10
- 1 – 2 – 4 – 5 – 7 – 9 – 11 – 12
- 1 – 2 – 4 – 5 – 7 – 9 – 11 – 13 – 14
- 1 – 2 – 4 – 5 – 7 – 9 – 11 – 13 – 15 – 16

❖ Test case

ID	Basis path	Data	Expected results
1	1-2-3-4	typeSearch: "Mã hóa đơn"	Hệ thống lọc danh sách hóa đơn theo mã hóa đơn
2	1-2-3-5-6	typeSearch: "Ngày xuất"	Hệ thống lọc danh sách hóa đơn theo ngày xuất đơn
3	1-2-3-5-7-8	typeSearch: "Mã nhân viên"	Hệ thống lọc danh sách hóa đơn theo mã nhân viên phụ trách xuất đơn
4	1-2-3-5-7-9-10	typeSearch: "Tên nhân viên"	Hệ thống lọc danh sách hóa đơn theo tên nhân viên phụ trách xuất đơn
5	1-2-3-5-7-9-11-12	typeSearch: "Mã thành viên"	Hệ thống lọc danh sách hóa đơn theo mã thành viên (nếu có)
6	1-2-3-5-7-9-11-13-14	typeSearch: "Tên thành viên"	Hệ thống lọc danh sách hóa đơn theo tên thành viên (nếu có)
7	1-2-3-5-7-9-11-13-15-16	typeSearch: "Mã voucher"	Hệ thống lọc danh sách hóa đơn theo mã voucher (nếu có)

❖ Test report

Số lượng test case	7
Số test case Pass	7
Số test case Fail	0
Số test case Error	0
Số test case chưa thực hiện	0

Tỉ lệ thành công	100%

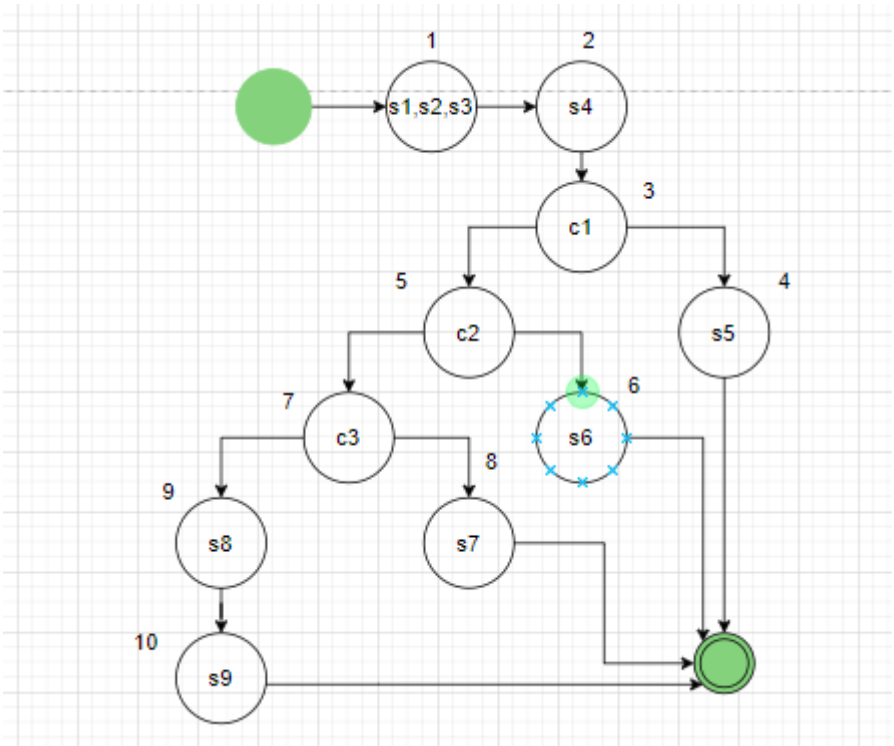
Tìm kiếm theo khoảng giá trị

❖ Mã nguồn

```
    private void txtSearchHDGiaDenKeyReleased(java.awt.event.KeyEvent evt) {
/*s1*/  int searchHDTu = Integer.parseInt(txtSearchHDGiaTu.getText());
/*s2*/  int searchHDDen = Integer.parseInt(txtSearchHDGiaDen.getText());
/*s3*/  String searchType = String.valueOf(cbxSearchHDGia.getSelectedItem());
/*s4*/  switch (searchType) {
/*c1*/      case "Tổng tiền":
/*s5*/          hoaDonBLL.thongKeHoaDonTheoTongTien(modelHD,searchHDTu,searchHDDen);
                break;
/*c2*/      case "Tiền giảm":
/*s6*/          hoaDonBLL.thongKeHoaDonTheoTienGiam(modelHD,searchHDTu,searchHDDen);
                break;
/*c3*/      case "Phải thanh toán":
/*s7*/          hoaDonBLL.thongKeHoaDonTheoTienConLai(modelHD,searchHDTu,searchHDDen);
                break;
/*s8*/      case "Điểm":
/*s9*/          hoaDonBLL.thongKeHoaDonTheoDiem(modelHD,searchHDTu,searchHDDen);
                break;
        }
    }
```

Hình 13.2.3. Mã nguồn chức năng tìm kiếm khoảng giá trị

❖ Đồ thị dòng điều khiển cơ bản



Hình 13.2.4. Đồ thị dòng điều khiển cơ bản chức năng tìm kiếm theo khoảng giá trị

⇨ Độ phức tạp Cyclomatic M = N + 1, với N là số nút điều kiện rẽ nhánh nhị phân.

⇨ M = 3 + 1 = 4 (với N = 3)

⇨ Đường độc lập tuyến tính cơ bản:

- 1-2-3-4
- 1-2-3-5-6
- 1-2-3-5-7-8
- 1-2-3-5-7-9-10

❖ Test case

ID	Basis path	Data	Expected results
1	1-2-3-4	typeSearch: "Tổng tiền" searchHDTu: 1000 searchHDDen: 10000	Hệ thống lọc danh sách hóa đơn theo khoảng tổng tiền
2	1-2-3-5-6	typeSearch: "Tiền giảm" searchHDTu: 5000 searchHDDen: 10000	Hệ thống lọc danh sách hóa đơn theo khoảng tiền giảm
3	1-2-3-5-7-8	typeSearch: " Phải thanh toán" searchHDTu: 10000 searchHDDen: 15000	Hệ thống lọc danh sách hóa đơn theo khoảng tiền phải thanh toán
4	1-2-3-5-7-9-10	typeSearch: "Điểm" searchHDTu: 500 searchHDDen: 2000	Hệ thống lọc danh sách hóa đơn theo khoảng điểm đã sử dụng

❖ Test report

Số lượng test case	4
Số test case Pass	4
Số test case Fail	0
Số test case Error	0
Số test case chưa thực hiện	0
Tỉ lệ thành công	100%

Trả về hóa đơn có doanh thu lớn nhất và nhỏ nhất

❖ Mã nguồn

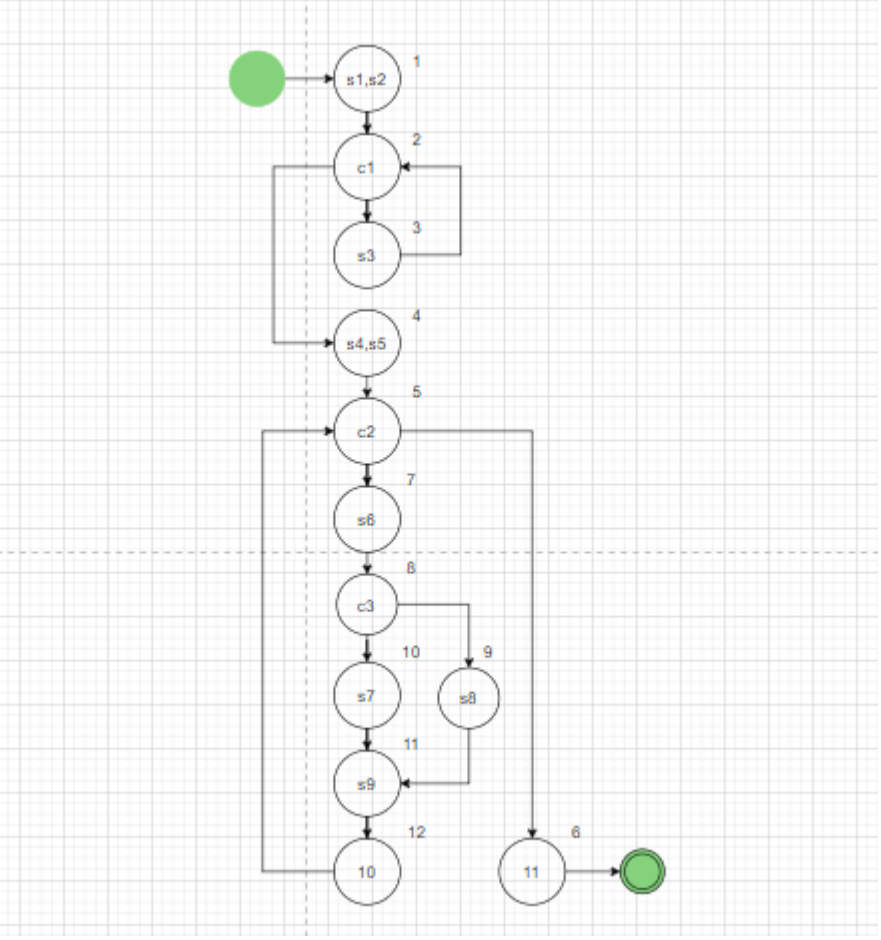
```
   public int getHoaDonCoGiaMax(){
/*s1*/  this.resetListHD();
/*s2*/  this.loadListHD();

/*s3*/  int max = listHD.get(0).getTien_con_lai();
/*c1*/  for(int i = 1 ; i < listHD.size() ; i++){
/*c2*/      if(listHD.get(i).getTien_con_lai() > max){
/*s4*/          max = listHD.get(i).getTien_con_lai();
          }
      }
/*s5*/  return max;
```

Hình 13.2.5. Mã nguồn chức năng tìm hóa đơn có doanh thu lớn nhất

```
    public int getHoaDonCoGiaMin(){
/*s1*/  this.resetListHD();
/*s2*/  this.loadListHD();

/*s3*/  int min = listHD.get(0).getTien_con_lai();
/*c1*/  for(int i = 1 ; i < listHD.size() ; i++){
/*c2*/      if(listHD.get(i).getTien_con_lai() < min){
/*s4*/          min = listHD.get(i).getTien_con_lai();
          }
      }
/*s5*/  return min;
```

Hình 13.2.5. Mã nguồn chức năng tìm hóa đơn có doanh thu nhỏ nhất

❖ Đồ thị dòng điều khiển cơ bản



Hình 13.2.6. Đồ thị dòng điều khiển cơ bản của chức năng tìm hóa đơn có doanh thu lớn nhất.

⇨ Độ phức tạp Cyclomatic M = N + 1, với N là số nút điều kiện rẽ nhánh nhị phân.

⇨ M = 2 + 1 = 3 (với N = 3)

⇨ Đường độc lập tuyến tính cơ bản:

- 1-2-5
- 1-2-3-2-5
- 1-2-3-4-2-5

❖ Test case

ID	Basis path	Data	Expected results

1	1-2-5	ListHD: -1 (rỗng) (listHD.size()<=0)	Không tìm thấy hóa đơn có giá trị lớn nhất (tương tự với nhỏ nhất)
2	1-2-3-2-5	ListHD: != -1 (không rỗng)	tìm thấy hóa đơn có giá trị lớn nhất là giá trị đầu tiên của listHD (tương tự với nhỏ nhất)
3	1-2-3-4-2-5	ListHD: != -1 (không rỗng)	tìm thấy hóa đơn có giá trị lớn nhất (tương tự với nhỏ nhất)

❖ Test report

Số lượng test case	3
Số test case Pass	3
Số test case Fail	0
Số test case Error	0
Số test case chưa thực hiện	0
Tỉ lệ thành công	100%

KẾT LUẬN

Trong quá trình kiểm thử và báo cáo kết quả dưới sự hướng dẫn của cô Vũ Thị Hạnh. Nhóm đã ít nhiều đạt được những mục tiêu mà ban đầu đã đặt ra. Nhưng đi kèm với đó cũng có những thiếu sót và hạn chế mà nhóm vẫn chưa khắc phục được, một phần do chưa có kinh nghiệm nhiều cũng như hạn chế về mặc thời gian khiến nhóm không thể kiểm tra đầy đủ được toàn bộ hệ thống.

Sau đây là những kết quả đã đạt được:

- Giới thiệu được về hệ thống cũng như cách hệ thống vận hành
- Sử dụng chiến lượt kiểm thử chức năng để kiểm thử phần mềm qua 2 phương pháp là kiểm thử hộp trắng và kiểm thử hộp đen.
- Phân tích thiết kế ra được các test case ứng với từng chức năng kiểm thử
- Triển khai việc kiểm thử và báo cáo kết quả đạt được.

Hạn chế:

- Chưa áp dụng thành công automatic testing để kiểm thử tự động hệ thống
- Hầu như tất cả các hoạt động test đều được làm thủ công bằng tay.
- Bỏ sót một vài chức năng.