# BT2 - Phân tích thiết kế hệ thống

Môn học: Kiểm thử phần mềm

Năm học: 2025-2026

Giảng viên: TS. Đỗ Như Tài

Thành viên:

Hồ Nguyên Minh 3122411123

Thông Trung Quyền 3122411175

Lục Trọng Khoa 3122411096

Mục lục:

[Mục lục: 1](#_Toc28650)

[Phân công công việc: 1](#_Toc13263)

[Bài tập minh họa 2](#_Toc13077)

[Bài 1. 2](#_Toc22037)

[Bài 2. 5](#_Toc17805)

[Bài 3. 7](#_Toc31976)

[Bài 4. 11](#_Toc22528)

[Bài 6. 18](#_Toc19434)

[Bài 7. 21](#_Toc1675)

[Bài tập ứng dụng 27](#_Toc25022)

[1) Hãy liệt kê các yêu cầu chức năng và phi chức năng của hệ thống CoolStore dựa trên mô tả kịch bản nghiệp vụ. 27](#_Toc1105)

[2) Vẽ qui trình nghiệp vụ cho website CoolStore 28](#_Toc16322)

[3) Xây dựng use case diagram cho website CoolStore. 29](#_Toc9437)

[4) Mô tả chi tiết một Use Case "Mua sản phẩm từ trang chi tiết sản phẩm". Trình bày theo mẫu chuẩn gồm: Tên use case, Tác nhân chính, Mục tiêu, Tiền điều kiện, Luồng chính, Luồng thay thế, Hậu điều kiện, Ghi chú (nếu có). 29](#_Toc27765)

[5) Sử dụng sơ đồ Activity Diagram để mô tả quy trình mua hàng từ khi người dùng nhấn “Mua hàng” cho đến khi hoàn tất thanh toán. 30](#_Toc14695)

[6) Thiết kế sơ đồ Class Diagram cho module Giỏ hàng. 31](#_Toc3603)

[7) Thiết kế giao diện màn hình cho chức năng “Quản lý giỏ hàng” (có thể thiết kế một hoặc nhiều màn hình) gồm các thông tin: 31](#_Toc10617)

[8) Đề xuất cách tổ chức kiến trúc phần mềm cho hệ thống CoolStore theo mô hình kiến trúc ba lớp (3-tier architecture). 32](#_Toc26427)

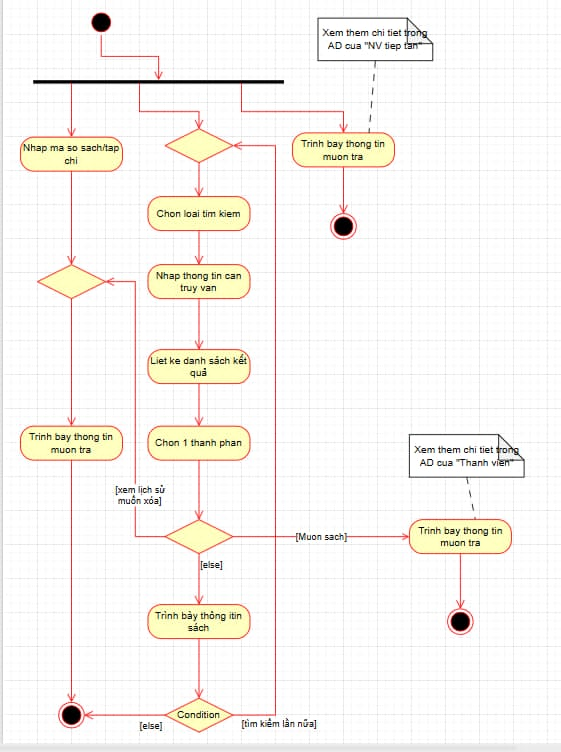
[9) Viết ít nhất 3 ca kiểm thử (test cases) cho chức năng “Thêm sản phẩm vào giỏ hàng” – bao gồm dữ liệu đầu vào, bước thực hiện và kết quả mong đợi. 33](#_Toc10631)

**Phân công công việc:**

|  |  |
| --- | --- |
| Thành viên | Công việc |
| Thông Trung Quyền | Làm phần bài tập ứng dụng |
| Lục Trọng Khoa | Vẽ hình, làm phần bài tập minh họa |
| Hồ Nguyên Minh | Tìm ý nghĩa các hình trong bài tập minh họa |

**Bài tập minh họa**

Bài 1.



**GIẢI THÍCH CHỨC NĂNG CÁC THÀNH PHẦN (CÁC LỚP)**

1. **Nhập mã số sách/tạp chí:**
   * **Chức năng:** Đây là hành động người dùng nhập một mã định danh duy nhất (ví dụ: ISBN, mã vạch) của quyển sách mà họ muốn tìm.
   * **Ý nghĩa:** Đây là luồng đi tắt, dành cho người dùng đã biết chính xác mình cần tìm gì.
2. **Chọn loại tìm kiếm:**
   * **Chức năng:** Người dùng lựa chọn phương thức tìm kiếm, ví dụ: tìm theo "Tên sách", "Tên tác giả", "Thể loại"...
   * **Ý nghĩa:** Bắt đầu luồng tìm kiếm nâng cao, cho phép người dùng lọc thông tin theo nhiều cách khác nhau.
3. **Nhập thông tin cần truy vấn:**
   * **Chức năng:** Dựa vào "loại tìm kiếm" đã chọn ở bước trên, người dùng sẽ nhập từ khóa tương ứng. Ví dụ, nếu chọn "Tên sách" thì họ sẽ nhập tên quyển sách vào đây.
   * **Ý nghĩa:** Cung cấp dữ liệu đầu vào để hệ thống thực hiện tìm kiếm.
4. **Liệt kê danh sách kết quả:**
   * **Chức năng:** Hệ thống xử lý thông tin truy vấn và hiển thị một danh sách các quyển sách phù hợp với từ khóa người dùng đã nhập.
   * **Ý nghĩa:** Trả về kết quả tìm kiếm cho người dùng.
5. **Chọn 1 thành phần:**
   * **Chức năng:** Từ danh sách kết quả, người dùng nhấp chuột hoặc chọn một quyển sách cụ thể mà họ quan tâm.
   * **Ý nghĩa:** Người dùng đã xác định được đối tượng mình muốn xem thông tin chi tiết.
6. **Trình bày thông tin sách:**
   * **Chức năng:** Hệ thống hiển thị các thông tin chi tiết về quyển sách đã chọn, ví dụ: tên sách, tác giả, nhà xuất bản, tóm tắt nội dung, số lượng còn lại trong thư viện.
   * **Ý nghĩa:** Cung cấp đầy đủ thông tin về một đầu sách.
7. **Trình bày thông tin mượn trả:**
   * **Chức năng:** Đây là một hành động chuyên biệt, có thể là hiển thị lịch sử mượn trả của sách đó, hoặc bắt đầu quy trình cho người dùng mượn/đặt trước quyển sách này.
   * **Ý nghĩa:** Xử lý các nghiệp vụ liên quan đến việc mượn và trả sách.
8. **Xem thêm chi tiết trong AD của "NV tiếp tân" / "Thành viên":**
   * Đây là **ô ghi chú (Note)**, không phải là một hành động.
   * **Ý nghĩa:** Nó chỉ ra rằng hành động "Trình bày thông tin mượn trả" là một quy trình phức tạp, và để hiểu rõ hơn các bước chi tiết của nó, người xem cần tham khảo một Sơ đồ hoạt động (Activity Diagram - AD) khác dành riêng cho vai trò "Nhân viên tiếp tân" hoặc "Thành viên".

**LUỒNG 1: TÌM KIẾM NHANH BẰNG MÃ SỐ**

1. **Bắt đầu:** Người dùng khởi động chức năng tìm kiếm.
2. **Rẽ nhánh:** Người dùng chọn đi theo con đường tìm kiếm bằng mã số.
3. **Hành động: Nhập mã số sách/tạp chí** 
   * Người dùng nhập trực tiếp mã định danh (ID, ISBN, mã vạch...) của cuốn sách vào ô tìm kiếm.
4. **Kiểm tra điều kiện (Hình thoi):** 
   * Hệ thống ngầm kiểm tra xem mã số này có hợp lệ và tồn tại trong cơ sở dữ liệu hay không.
   * Nếu mã số hợp lệ (đây là nhánh được vẽ trong sơ đồ), luồng sẽ đi tiếp. (Sơ đồ này đang thiếu nhánh xử lý khi mã không hợp lệ, ví dụ: báo lỗi).
5. **Hành động: Trình bày thông tin mượn trả** 
   * Hệ thống hiển thị ngay lập tức các thông tin liên quan đến việc mượn/trả của sách đó (ví dụ: sách đang được ai mượn, khi nào đến hạn trả, lịch sử mượn...).
   * *Ghi chú "Xem thêm chi tiết trong AD của 'NV tiếp tân'"* cho biết rằng quy trình chi tiết để xử lý việc mượn trả được mô tả ở một sơ đồ khác.
6. **Kết thúc:** Luồng xử lý cho kịch bản này hoàn tất.

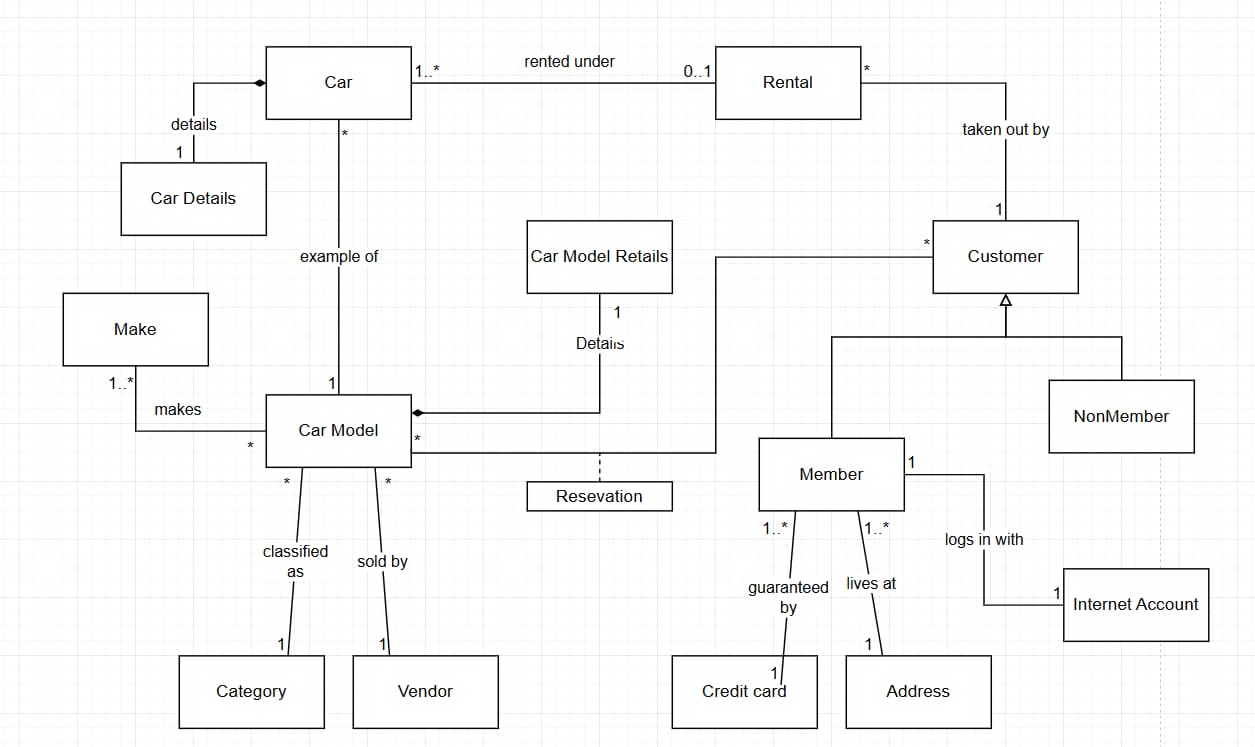
**Tóm tắt Luồng 1: Nhập mã -> Hệ thống tìm thấy -> Hiển thị thông tin -> Kết thúc. Rất nhanh và hiệu quả.**

**LUỒNG 2: TÌM KIẾM NÂNG CAO THEO TIÊU CHÍ**

1. **Bắt đầu:** Người dùng khởi động chức năng tìm kiếm.
2. **Rẽ nhánh:** Người dùng chọn đi theo con đường tìm kiếm nâng cao.
3. **Hành động:** Chọn loại tìm kiếm
   * Người dùng chọn một tiêu chí để tìm, ví dụ: "Tên sách", "Tác giả", "Thể loại", "Nhà xuất bản".
4. **Hành động: Nhập thông tin cần truy vấn** 
   * Người dùng nhập từ khóa tương ứng với tiêu chí đã chọn. (Ví dụ: chọn "Tác giả" rồi nhập "Nguyễn Nhật Ánh").
5. **Hành động: Liệt kê danh sách kết quả** 
   * Hệ thống truy vấn cơ sở dữ liệu và hiển thị một danh sách các cuốn sách khớp với từ khóa tìm kiếm.
6. **Hành động: Chọn 1 thành phần** 
   * Từ danh sách kết quả, người dùng nhấp vào một cuốn sách cụ thể mà họ quan tâm.
7. **Kiểm tra quyết định của người dùng (Hình thoi lớn ở giữa):** 
   * Đây là điểm rẽ nhánh quan trọng nhất. Dựa vào hành động tiếp theo của người dùng trên cuốn sách đã chọn, hệ thống sẽ đi theo các hướng khác nhau:
   * Nhánh [Mượn sách] hoặc [xem lịch sử mượn xóa]: Nếu người dùng nhấn nút "Mượn sách" hoặc "Xem lịch sử", luồng sẽ chuyển đến hành động Trình bày thông tin mượn trả. Sau đó, quy trình kết thúc.
   * Nhánh [else] (Trường hợp còn lại): Nếu người dùng chỉ đơn giản là muốn xem thông tin chi tiết (ví dụ: nhấp vào tên sách), luồng sẽ đi tiếp đến hành động Trình bày thông tin sách.
8. **Hành động: Trình bày thông tin sách** 
   * Hệ thống hiển thị các thông tin mô tả chi tiết về sách như tên đầy đủ, tác giả, tóm tắt, số lượng...
9. **Kiểm tra quyết định cuối cùng (Hình thoi cuối cùng):** 
   * Sau khi xem xong thông tin, người dùng có 2 lựa chọn:
   * Nhánh [tìm kiếm lần nữa]: Nếu người dùng muốn thực hiện một tìm kiếm khác, luồng sẽ quay trở lại (vòng lặp) bước Chọn loại tìm kiếm.
   * Nhánh [else]: Nếu người dùng không muốn làm gì thêm (ví dụ: đóng cửa sổ thông tin), luồng sẽ đi đến điểm Kết thúc.

**Tóm tắt Luồng 2: Chọn tiêu chí -> Nhập từ khóa -> Xem danh sách -> Chọn 1 sách -> Xem chi tiết/Mượn sách -> (Tùy chọn) Tìm lại từ đầu hoặc Kết thúc.**

Bài 2.



**GIẢI THÍCH CHỨC NĂNG CÁC THÀNH PHẦN (CÁC LỚP)**

1. **Car (Xe ô tô):**
   * **Chức năng**: Đại diện cho một chiếc xe ô tô cụ thể, hữu hình trong bãi xe, có mã số định danh riêng (VD: xe Toyota Vios biển số 51K-123.45).
   * **Ý nghĩa:** Đây là đối tượng cốt lõi mà khách hàng sẽ thuê.
2. **Car Details (Chi tiết xe):**
   * **Chức năng:** Lưu trữ các thông tin chi tiết chỉ thuộc về một chiếc xe cụ thể như biển số, màu sơn, số km đã đi, tình trạng hiện tại.
   * **Ý nghĩa:** Tách biệt thông tin chi tiết của từng xe ra khỏi thông tin chung, giúp quản lý dễ dàng hơn. Mối quan hệ kim cương đen (Composition) cho thấy Car Details không thể tồn tại nếu khôngcó Car.
3. **Car Model (Mẫu xe):**
   * **Chức năng:** Đại diện cho một dòng xe/mẫu xe chung (VD: Toyota Vios 2022, Ford Ranger). Đây là một bản thiết kế, không phải một chiếc xe cụ thể.
   * **Ý nghĩa:** Cho phép quản lý nhiều chiếc xe có cùng đặc điểm chung. Khách hàng thường chọn thuê theo mẫu xe.
4. **Make (Hãng sản xuất):**
   * **Chức năng:** Lưu trữ thông tin về hãng sản xuất xe (VD: Toyota, Ford, Honda).
   * **Ý nghĩa:** Phân loại các mẫu xe theo nhà sản xuất.
5. **Category (Loại xe/Phân khúc):**
   * **Chức năng:** Phân loại các mẫu xe vào các nhóm như "Sedan", "SUV", "Xe bán tải".
   * **Ý nghĩa:** Giúp khách hàng tìm kiếm và lọc xe theo nhu cầu sử dụng.
6. **Vendor (Nhà cung cấp):**
   * **Chức năng:** Lưu thông tin về đơn vị đã bán xe cho công ty cho thuê.
   * **Ý nghĩa:** Phục vụ cho việc quản lý mua sắm, bảo hành tài sản.
7. **Customer (Khách hàng):**
   * **Chức năng:** Đại diện cho người thuê xe. Đây là một lớp trừu tượng (lớp cha).
   * **Ý nghĩa:** Quản lý thông tin chung của tất cả những người sử dụng dịch vụ.
8. **Member (Thành viên) & NonMember (Khách vãng lai):**
   * **Chức năng:** Là hai loại khách hàng cụ thể, kế thừa từ Customer. Member có các thông tin bổ sung như tài khoản, thẻ tín dụng, địa chỉ.
   * **Ý nghĩa:** Thể hiện hệ thống có chính sách ưu đãi hoặc quy trình khác nhau cho khách hàng thân thiết và khách vãng lai.
9. **Internet Account (Tài khoản trực tuyến):**
   * **Chức năng:** Lưu thông tin đăng nhập (tên người dùng, mật khẩu) cho Member.
   * **Ý nghĩa:** Cho phép thành viên đăng nhập vào hệ thống website/ứng dụng để quản lý đặt xe.
10. **Address (Địa chỉ) & Credit card (Thẻ tín dụng):**
    * **Chức năng:** Lưu trữ thông tin địa chỉ và thẻ tín dụng của Member.
    * **Ý nghĩa:** Phục vụ cho việc xác thực, thanh toán và đảm bảo cho các giao dịch thuê xe.
11. **Rental (Lượt thuê xe):**
    * **Chức năng:** Đại diện cho một giao dịch thuê xe, lưu các thông tin như ngày thuê, ngày trả, tổng chi phí.
    * **Ý nghĩa:** Ghi lại lịch sử và quản lý các giao dịch thuê xe đang diễn ra.
12. **Reservation (Đặt xe trước):**
    * **Chức năng:** Đại diện cho một yêu cầu đặt trước một Car Model cho một thời điểm trong tương lai.
    * **Ý nghĩa:** Cho phép khách hàng giữ chỗ cho loại xe họ muốn thuê.

**GIẢI THÍCH CÁC LUỒNG NGHIỆP VỤ**

**LUỒNG 1: QUẢN LÝ ĐỘI XE (NGHIỆP VỤ NỘI BỘ)**

1. Bắt đầu: Công ty mua một dòng xe mới.
2. Hành động: Tạo Make và Car Model
   * Hệ thống kiểm tra xem Make (Hãng sản xuất, VD: "Toyota") đã tồn tại chưa. Nếu chưa, tạo mới.
   * Hệ thống tạo một Car Model mới (VD: "Vios 2022").
   * Gán Car Model này vào một Category (VD: "Sedan") và một Vendor (nhà cung cấp).
3. Hành động: Nhập xe cụ thể vào bãi
   * Với mỗi chiếc xe vật lý được nhập về, hệ thống tạo một đối tượng Car mới.
   * Đối tượng Car này được liên kết ("example of") với Car Model đã tạo ở trên.
   * Đồng thời, một đối tượng Car Details tương ứng được tạo ra và gắn vào Car này để lưu biển số, màu sắc...
4. Kết thúc: Các xe mới đã sẵn sàng trong hệ thống để cho thuê.

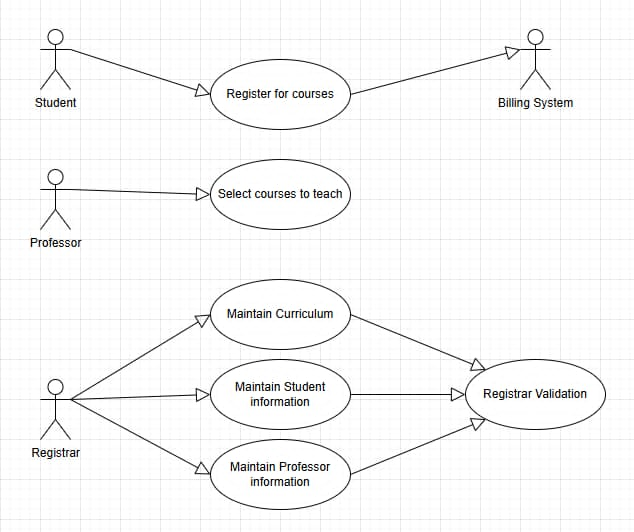
**Tóm tắt Luồng 1: Quản trị viên nhập thông tin từ Hãng sản xuất -> Mẫu xe -> Xe cụ thể.**

**LUỒNG 2: MỘT KHÁCH HÀNG THUÊ XE**

1. Bắt đầu: Một Customer (khách hàng) muốn thuê xe.
2. Hành động: Xác định loại khách hàng
   * Khách hàng có thể là Member (đã có tài khoản) hoặc NonMember (khách mới).
3. Hành động: Tìm và chọn xe
   * Khách hàng tìm kiếm các Car còn trống, có thể lọc theo Make, Car Model, Category.
4. Hành động: Tạo giao dịch thuê
   * Khi khách hàng chọn được xe, hệ thống tạo một đối tượng Rental (lượt thuê).
   * Rental này được liên kết ("taken out by") với đúng một Customer.
   * Rental này cũng được liên kết ("rented under") với một hoặc nhiều (1..\*) chiếc Car mà khách hàng đã chọn.
   * Quan trọng: Mỗi chiếc Car chỉ có thể được liên kết với 0 hoặc 1 Rental tại một thời điểm. Điều này đảm bảo một chiếc xe đang được thuê thì không thể cho người khác thuê nữa.
5. Kết thúc: Giao dịch hoàn tất, khách hàng nhận xe.

**Tóm tắt Luồng 2: Khách hàng chọn xe -> Hệ thống tạo giao dịch thuê -> Gán giao dịch đó cho khách hàng và chiếc xe đã chọn -> Xe chuyển sang trạng thái "đã được thuê".**

Bài 3.



### a) Xác định và mô tả các tác nhân (Actors) trong sơ đồ

**Student (Sinh viên)** — người đăng ký học phần; tương tác chính với chức năng Register for Courses.

**Professor (Giảng viên)** — chọn môn/khóa để dạy; tương tác với Select Courses to Teach.

**Registrar (Phòng đào tạo / Cán bộ đăng ký)** — quản lý dữ liệu (curriculum, student info, professor info) và thực hiện các xác thực/duyệt.

**Billing System (Hệ thống thanh toán / Kế toán)** — hệ thống bên ngoài xử lý thanh toán/hoá đơn, liên kết với chức năng đăng ký học phần.

b) Liệt kê và giải thích các Use Cases có trên sơ đồ

**Register for Courses** — Sinh viên đăng ký các học phần/khoá học cho học kỳ.

**Select Courses to Teach** — Giảng viên chọn các khóa/môn họ sẽ giảng dạy.

**Maintain Curriculum** — Cán bộ quản lý chương trình/hệ thống môn học (thêm/sửa/xóa chương trình, môn).

**Maintain Student Information** — Cập nhật dữ liệu sinh viên (thông tin cá nhân, trạng thái học tập, học phí, …).

**Maintain Professor Information** — Cập nhật thông tin giảng viên (chi tiết liên hệ, bộ môn, quyền dạy).

**Registrar Validation** (Xác thực của phòng đăng ký) — chức năng kiểm tra/duyệt (ví dụ kiểm tra điều kiện, trạng thái học phí, giới hạn tín chỉ).

c) Phân tích các mối quan hệ giữa các use case (semantic)

Trong sơ đồ có biểu diễn mối quan hệ kiểu «uses»/«includes» giữa một vài use case với **Registrar Validation**. Giải thích ý nghĩa:

**Association (liên kết)**: actor ↔ use case (ví dụ Student — Register for Courses).

**<<include>> / <<uses>> (tái sử dụng hành vi chung)**: khi một use case luôn cần thực hiện một hành vi con chung, thì use case chính **include** hành vi con đó. Ở sơ đồ:

Maintain Student Information **uses/includes** Registrar Validation — tức việc thay đổi thông tin sinh viên phải qua bước xác thực/duyệt của phòng đăng ký.

Tương tự Maintain Curriculum và Maintain Professor Information đều **uses/includes** Registrar Validation

**<<extend>> (mở rộng điều kiện)**: không có use case dạng extend

Tóm lại: Registrar Validation là một use case tái sử dụng (include) cho nhiều use case quản trị nhằm đảm bảo mọi thay đổi đều phải được xác thực.

d) Viết kịch bản chi tiết cho một use case (**Register for Courses**)

|  |  |  |
| --- | --- | --- |
| Tên use case | Register for Courses | |
| Tác nhân chính | Student | |
| Bên liên quan | Billing System (thanh toán), Registrar (xác thực/duyệt), Hệ thống quản lý học phần. | |
| Mục tiêu | Sinh viên đăng ký thành công các học phần phù hợp cho học kỳ. | |
| Tiền điều kiện | Sinh viên đã đăng nhập hợp lệ vào hệ thống; Sinh viên đủ điều kiện để đăng ký (đang học, không bị cấm đăng ký). | |
| Luồng chính | Actor action | System Respone |
| 1. Sinh viên mở màn hình đăng ký học phần. |  |
|  | 2. Hệ thống hiển thị danh sách các học phần có sẵn (với mã, tên, số tín chỉ, giáo viên, thời khóa biểu, số chỗ còn lại, điều kiện tiền quyết). |
| 3. Sinh viên chọn một hoặc nhiều học phần muốn đăng ký. |  |
|  | 4. Hệ thống **tự động kiểm tra** xung đột thời khóa biểu, điều kiện tiền quyết, số tín chỉ tối đa, và yêu cầu học phí (gọi các use case/con-tác vụ: CheckPrerequisites, CheckScheduleConflict, CheckCreditLimit). **A1, A2** |
|  | 5. Hệ thống gửi yêu cầu xác thực đến **Registrar Validation** (include). |
|  | 6. Registrar validation phê duyệt (hoặc tự động phê duyệt nếu không cần can thiệp). **A3** |
|  | 7. Hệ thống chuyển sang **Billing System** để xử lý thanh toán/ghi nhận phí nếu học phần cần đóng phí. **A4** |
|  | 8. hệ thống ghi nhận đăng ký, cập nhật số chỗ còn lại, gửi thông báo xác nhận cho sinh viên (email/portal). **A5** |
|  | 9. Sinh viên đã được đăng ký học phần thành công. |
| Luồng thay thế | A1. Nếu có xung đột thời gian → hệ thống cảnh báo, đề xuất các học phần thay thế hoặc yêu cầu sinh viên bỏ chọn học phần xung đột.  A2. Nếu điều kiện tiền quyết không đạt → hệ thống từ chối đăng ký và hiển thị lý do; sinh viên có thể yêu cầu xét duyệt (đi vào luồng khác).  A3. Nếu Registrar từ chối → hiển thị lý do; có thể hướng dẫn làm khiếu nại/điều chỉnh.  A4. Nếu thanh toán thất bại → đơn đăng ký được giữ ở trạng thái chờ thanh toán; sinh viên có thể thử lại hoặc chọn hình thức khác.  A5. Nếu môn đã đầy trong lúc đăng ký → hệ thống có thể **extend** luồng bằng chức năng AddToWaitlist hoặc thông báo từ chối. | |
| Hậu điều kiện: | Nếu thành công: bản ghi đăng ký học phần tồn tại, sinh viên được tính tín chỉ, hóa đơn/ghi nhận thanh toán được tạo; Nếu thất bại: trạng thái tương ứng (bị từ chối, chờ thanh toán, trong danh sách chờ). | |

**Yêu cầu đặc biệt:**

Kiểm tra thời khóa biểu phải nhanh/real-time.

Bảo mật/ấn định quyền (chỉ sinh viên được đăng ký cho chính mình).

Giao tiếp an toàn với Billing System.

e) Đề xuất cải tiến hoặc bổ sung use cases cần thiết

**Use cases nên thêm:**

**Login / Authenticate** — xác thực người dùng (student/professor/registrar).

**View Academic Schedule / Timetable** — sinh viên/giảng viên xem thời khóa biểu cá nhân.

**Drop Course** — huỷ đăng ký một học phần.

**Check Prerequisites** — kiểm tra tự động điều kiện tiền quyết (nên **include** trong Register for Courses).

**Process Payment** — tách riêng chức năng thanh toán (Billing System là actor, Process Payment được include trong Register for Courses khi cần).

**Add to Waitlist** — nếu lớp đầy, thêm vào danh sách chờ (**extend** của Register for Courses).

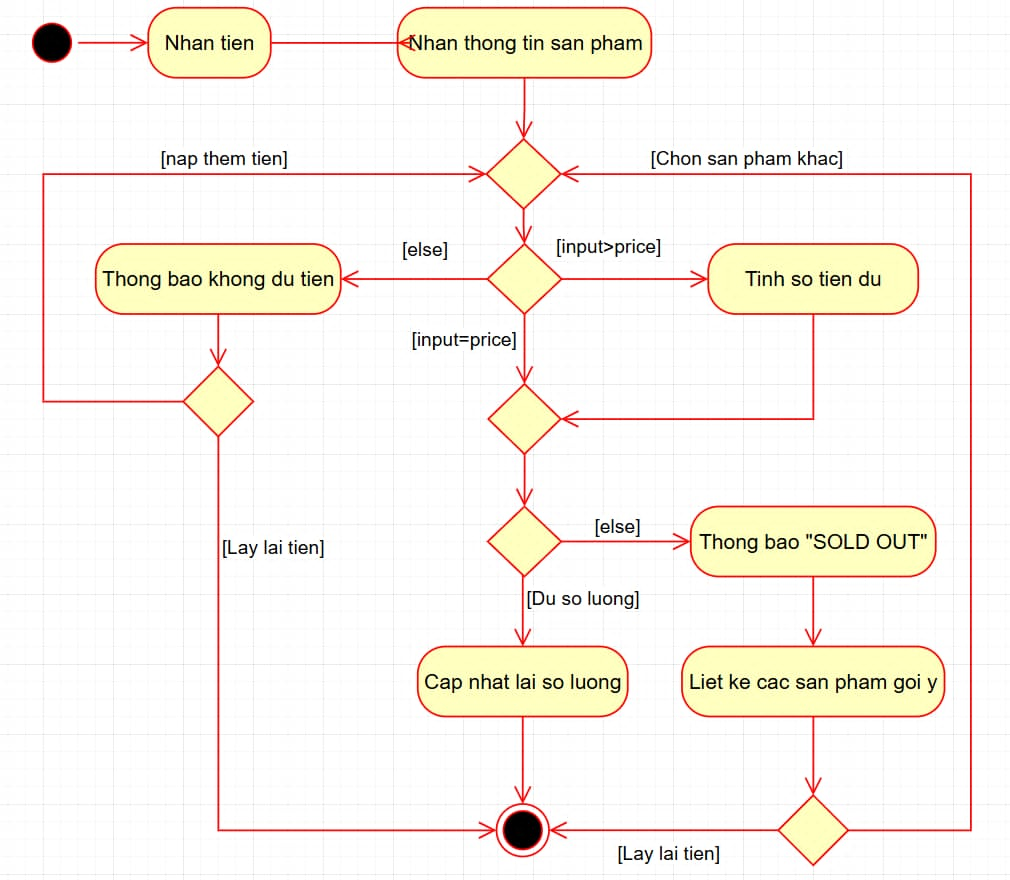
**Request Override / Special Approval** — khi sinh viên cần phép vượt tín chỉ hoặc đăng ký khi chưa đủ điều kiện (luồng mở rộng, liên hệ Registrar).

**Generate Transcript / Academic Report** — báo cáo học tập, điểm.

**Manage Rooms/Timeslot** — cho phòng đào tạo phân bố phòng/giờ (liên quan tới Maintain Curriculum).

**Notify / Announce** — gửi thông báo (thời hạn đăng ký, thay đổi lớp,…) — có thể là service chung.

Bài 4.



**GIẢI THÍCH CHỨC NĂNG CÁC THÀNH PHẦN**

1. **Nhập mã số sách/tạp chí (Interpreted as: Nhập mã sản phẩm):**
   * **Chức năng:** Người dùng nhập mã của món hàng họ muốn mua (ví dụ: A1, B5...). Hành động này được lặp lại 2 lần một cách thừa thãi trong sơ đồ, nhưng về bản chất chúng là một.
   * **Ý nghĩa:** Khởi đầu quy trình mua một sản phẩm cụ thể.
2. **Trình bày thông tin mượn trả (Interpreted as: Hiển thị thông báo / Thực hiện hành động):**
   * **(Khi input < price):** 
     + **Chức năng:** Hiển thị thông báo "Số tiền không đủ, vui lòng nạp thêm" hoặc hiển thị số tiền còn thiếu.
     + **Ý nghĩa:** Yêu cầu người dùng nạp thêm tiền để hoàn tất giao dịch.
   * **(Khi input > price):**
     + **Chức năng:** Hệ thống tính toán số tiền thừa cần trả lại cho khách hàng.
     + **Ý nghĩa:** Chuẩn bị cho hành động trả lại tiền thừa sau khi giao hàng.
   * **(Khi không đủ số lượng):**
     + **Chức năng:** Hiển thị thông báo "Sản phẩm đã hết hàng".
     + **Ý nghĩa:** Thông báo cho người dùng rằng giao dịch không thể thực hiện và gợi ý hành động tiếp theo.
   * **(Khi đủ số lượng):**
     + **Chức năng:** Máy đẩy sản phẩm ra khay cho người dùng.
     + **Ý nghĩa:** Hoàn thành phần chính của giao dịch là giao hàng cho khách.
   * **(Sau khi trả hàng và có tiền thừa):** 
     + **Chức năng:** Máy nhả tiền thừa ra khay trả tiền.
     + **Ý nghĩa:** Hoàn trả lại phần tiền chênh lệch cho khách hàng.

**GIẢI THÍCH CÁC LUỒNG XỬ LÝ**

**LUỒNG 1: GIAO DỊCH THÀNH CÔNG (TRẢ TIỀN ĐỦ HOẶC DƯ)**

1. Bắt đầu: Người dùng muốn mua hàng.
2. Hành động: Nhập mã sản phẩm
   * Người dùng bấm mã sản phẩm mong muốn.
3. Hành động: Nạp tiền (Đây là hành động ngầm trước khi vào điểm rẽ nhánh đầu tiên)
   * Người dùng đưa tiền vào máy.
4. Kiểm tra điều kiện tiền (Condition đầu tiên):
   * Hệ thống so sánh số tiền người dùng nạp (input) với giá sản phẩm (price).
   * Nếu [input > price]: Luồng đi sang phải. Hệ thống ghi nhận sẽ phải trả lại tiền thừa.
   * Nếu [input = price]: Luồng đi thẳng xuống.
5. Kiểm tra điều kiện số lượng (Condition thứ hai):
   * Hệ thống kiểm tra xem sản phẩm được chọn còn hàng hay không.
   * Nếu [Đủ số lượng]: Luồng đi thẳng xuống.
6. Hành động: Trình bày thông tin mượn trả (Giao sản phẩm)
   * Máy đẩy sản phẩm ra cho khách hàng.
7. Kiểm tra và trả tiền thừa (Nếu có):
   * Nếu ban đầu input > price, luồng sẽ đi qua hành động tính tiền thừa và cuối cùng là Trình bày thông tin mượn trả (Trả lại tiền thừa).
8. Kết thúc: Giao dịch hoàn tất.

**Tóm tắt Luồng 1: Chọn hàng -> Nạp tiền đủ/dư -> Máy kiểm tra còn hàng -> Giao hàng -> (Nếu có) Trả tiền thừa -> Kết thúc.**

**LUỒNG 2: SỐ TIỀN NẠP KHÔNG ĐỦ**

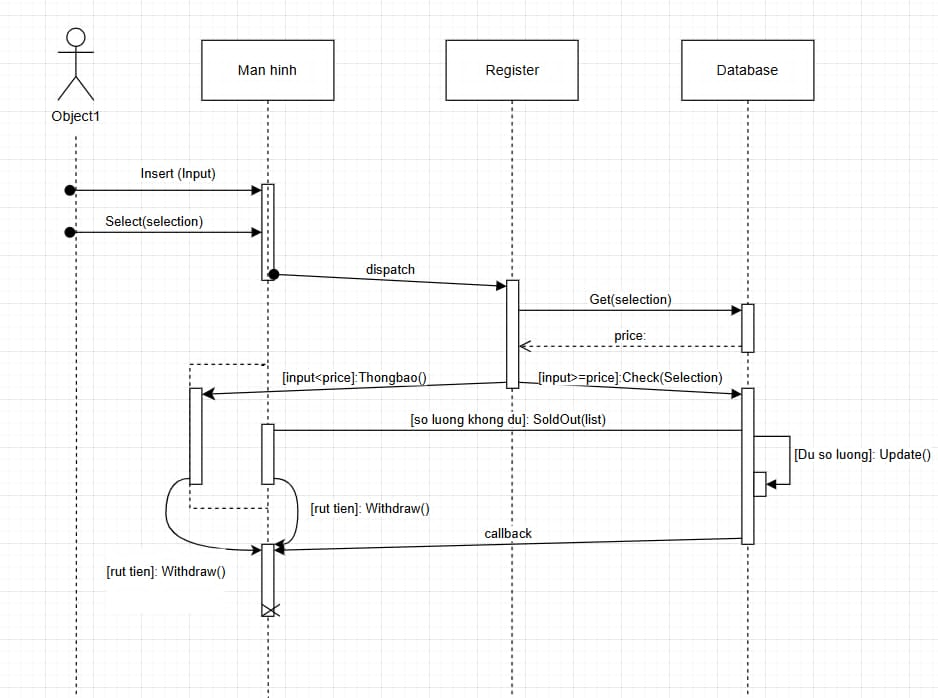
1. Bắt đầu -> Nhập mã sản phẩm -> Nạp tiền.
2. Kiểm tra điều kiện tiền (Condition đầu tiên):
   * Hệ thống phát hiện [else] (tức là input < price). Luồng đi sang trái.
3. Hành động: Trình bày thông tin mượn trả (Thông báo thiếu tiền)
   * Máy hiển thị thông báo yêu cầu nạp thêm tiền.
4. Kiểm tra quyết định của người dùng (Condition bên trái):
   * Nếu [nạp thêm tiền]: Luồng sẽ quay trở lại điểm rẽ nhánh đầu tiên để kiểm tra lại tổng số tiền.
   * Nếu [Lấy lại tiền]: Người dùng quyết định hủy giao dịch. Luồng đi thẳng đến điểm Kết thúc (máy sẽ trả lại toàn bộ số tiền đã nạp).

**Tóm tắt Luồng 2: Chọn hàng -> Nạp thiếu tiền -> Máy thông báo -> Người dùng có thể nạp thêm (lặp lại) hoặc hủy giao dịch (kết thúc).**

**LUỒNG 3: SẢN PHẨM HẾT HÀNG**

1. Bắt đầu -> Nhập mã sản phẩm -> Nạp tiền đủ.
2. Kiểm tra điều kiện tiền: Giả sử input >= price.
3. Kiểm tra điều kiện số lượng (Condition thứ hai):
   * Hệ thống phát hiện [else] (tức là không đủ số lượng). Luồng đi sang phải.
4. Hành động: Trình bày thông tin mượn trả (Thông báo hết hàng)
   * Máy hiển thị thông báo sản phẩm đã hết.
5. Kiểm tra quyết định của người dùng (Condition cuối cùng bên phải):
   * Nếu [Chọn sản phẩm khác]: Luồng quay trở lại bước Nhập mã sản phẩm để người dùng mua món hàng khác với số tiền đã nạp.
   * Nếu [Lấy lại tiền]: Người dùng hủy giao dịch. Luồng đi đến điểm Kết thúc và máy trả lại tiền.

**Tóm tắt Luồng 3: Chọn hàng -> Nạp tiền đủ -> Máy báo hết hàng -> Người dùng có thể chọn món khác (lặp lại) hoặc lấy lại tiền (kết thúc).**



**GIẢI THÍCH CHỨC NĂNG CÁC THÀNH PHẦN (CÁC ĐỐI TƯỢNG)**

1. **Object1 (Người dùng):**
   * **Chức năng:** Đại diện cho người dùng (khách hàng) đang tương tác với máy. Đây là người khởi tạo mọi hành động.
   * **Ý nghĩa:** Tác nhân bên ngoài hệ thống.
2. **Man hinh (Màn hình / Giao diện người dùng - UI):**
   * **Chức năng:** Là lớp đối tượng chịu trách nhiệm nhận các thao tác trực tiếp từ người dùng (như nạp tiền, chọn sản phẩm) và hiển thị thông tin phản hồi (như thông báo lỗi, kết quả giao dịch).
   * **Ý nghĩa:** Cầu nối giữa người dùng và lõi xử lý của hệ thống.
3. **Register (Bộ xử lý / Bộ điều khiển):**
   * **Chức năng:** Là "bộ não" của hệ thống. Nó nhận yêu cầu từ Man hinh, điều phối các hoạt động, thực hiện logic nghiệp vụ (so sánh tiền, kiểm tra hàng) bằng cách giao tiếp với Database.
   * **Ý nghĩa:** Tầng xử lý logic nghiệp vụ (Business Logic Layer).
4. **Database (Cơ sở dữ liệu):**
   * **Chức năng:** Chịu trách nhiệm lưu trữ và truy xuất dữ liệu về sản phẩm, bao gồm giá (price) và số lượng tồn kho.
   * **Ý nghĩa:** Tầng truy cập dữ liệu (Data Access Layer).

**GIẢI THÍCH CÁC LUỒNG XỬ LÝ**

**LUỒNG 1: GIAO DỊCH THÀNH CÔNG**

1. Bắt đầu: Object1 (Người dùng) thực hiện 2 hành động lên Man hinh:
   * Insert(Input): Nạp tiền vào máy.
   * Select(selection): Chọn một sản phẩm.
2. Hành động: Man hinh gửi yêu cầu xử lý
   * Man hinh đóng gói thông tin (tiền nạp và sản phẩm chọn) và gửi một thông điệp dispatch đến Register.
3. Hành động: Register lấy thông tin giá
   * Register gửi yêu cầu Get(selection) đến Database để lấy giá của sản phẩm đã chọn.
   * Database trả về price.
4. Hành động: Register kiểm tra số lượng
   * Register so sánh và thấy điều kiện [input >= price] là đúng. Nó tiếp tục gửi yêu cầu Check(Selection) đến Database để kiểm tra tồn kho.
5. Hành động: Database xác nhận và cập nhật
   * Database kiểm tra và thấy điều kiện [Đủ số lượng] là đúng.
   * Nó tự thực hiện hành động Update() để trừ đi 1 sản phẩm khỏi kho.
6. Hành động: Phản hồi thành công
   * Database gửi một thông điệp callback trực tiếp đến Man hinh (để ra lệnh cho máy nhả sản phẩm và trả tiền thừa nếu có).
7. Kết thúc: Người dùng nhận được sản phẩm.

**Tóm tắt Luồng 1: Người dùng nạp tiền & chọn hàng -> Giao diện gửi yêu cầu -> Bộ xử lý lấy giá, kiểm tra hàng tồn -> Cơ sở dữ liệu xác nhận, trừ kho -> Máy trả hàng.**

**LUỒNG 2: NẠP KHÔNG ĐỦ TIỀN**

1. Bắt đầu -> Bước 1, 2, 3 diễn ra như Luồng 1.
2. Kiểm tra điều kiện:
   * Sau khi nhận được price từ Database, Register so sánh và thấy điều kiện [input < price] là đúng.
3. Hành động: Thông báo lỗi
   * Register gửi thông điệp Thongbao() đến Man hinh.
   * Man hinh hiển thị thông báo "Không đủ tiền".
4. Hành động: Hủy giao dịch (Tùy chọn)
   * Người dùng có thể chọn [rút tiền]. Họ gửi yêu cầu Withdraw() đến Man hinh.
   * Man hinh thực hiện hành động trả lại tiền và luồng xử lý kết thúc (đánh dấu bằng dấu 'X').

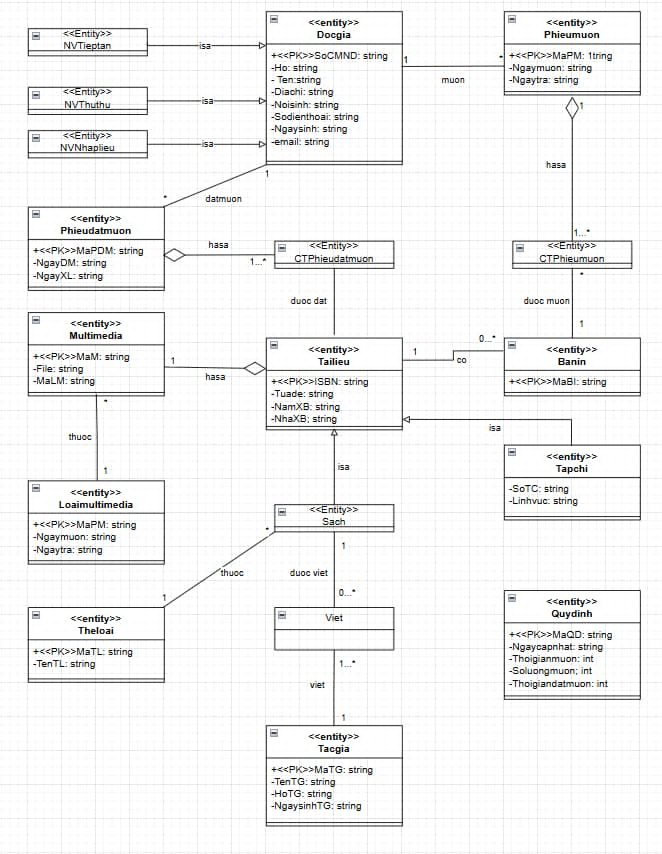
**Tóm tắt Luồng 2: Các bước đầu như trên -> Bộ xử lý phát hiện thiếu tiền -> Gửi thông báo cho giao diện -> Người dùng có thể rút tiền và hủy giao dịch.**

**LUỒNG 3: SẢN PHẨM HẾT HÀNG**

1. Bắt đầu -> Bước 1, 2, 3, 4 diễn ra như Luồng 1 (người dùng đã nạp đủ tiền).
2. Kiểm tra điều kiện:
   * Sau khi Register gửi yêu cầu Check(Selection) đến Database, Database kiểm tra và thấy điều kiện [số lượng không đủ] là đúng.
3. Hành động: Thông báo hết hàng
   * Database gửi thông điệp SoldOut(list) ngược lại cho Register, thông báo sản phẩm đã hết (có thể kèm theo danh sách các sản phẩm gợi ý).
4. Hành động: Hủy giao dịch
   * (Luồng này bị thiếu một bước là Register phải thông báo cho Man hinh). Giả định là người dùng đã được thông báo và họ chọn [rút tiền].
   * Người dùng yêu cầu Withdraw(), Man hinh xử lý trả tiền và kết thúc giao dịch.

**Tóm tắt Luồng 3: Người dùng nạp đủ tiền -> Bộ xử lý kiểm tra kho -> Cơ sở dữ liệu báo hết hàng -> Giao dịch bị hủy và người dùng nhận lại tiền.**

Bài 5.



**GIẢI THÍCH CHỨC NĂNG CÁC THÀNH PHẦN (CÁC THỰC THỂ)**

**Nhóm Thực thể Người dùng:**

1. **Docgia (Độc giả):**
   * **Chức năng:** Đại diện cho một người đọc được cấp thẻ thư viện, lưu trữ các thông tin cá nhân cơ bản như họ tên, địa chỉ, số CMND.
   * **Ý nghĩa:** Là đối tượng chính sử dụng các dịch vụ mượn, trả, đặt trước tài liệu.
2. **NVTieptan, NVThuthu, NVNhaplieu (Nhân viên):**
   * **Chức năng:** Đây là các loại nhân viên khác nhau trong thư viện (Tiếp tân, Thủ thư, Nhập liệu). Mối quan hệ "is-a" (kế thừa) cho thấy tất cả các nhân viên này cũng là một Docgia.
   * **Ý nghĩa:** Phân chia vai trò và quyền hạn của nhân viên trong hệ thống. Việc kế thừa từ Docgia có thể ngụ ý rằng nhân viên cũng có quyền mượn sách như một độc giả thông thường.
3. **Tacgia (Tác giả):**
   * **Chức năng:** Lưu trữ thông tin về tác giả của các cuốn sách.
   * **Ý nghĩa:** Cho phép tìm kiếm sách theo tên tác giả.

**Nhóm Thực thể Tài liệu:**

1. **Tailieu (Tài liệu):**
   * **Chức năng:** Là một lớp cha, đại diện cho khái niệm chung về một đầu tài liệu trong thư viện (sách, tạp chí...). Lưu các thông tin chung như mã ISBN, tựa đề, năm/nhà xuất bản.
   * **Ý nghĩa:** Giúp quản lý các loại tài liệu khác nhau một cách thống nhất.
2. **Sach (Sách), Tapchi (Tạp chí), Multimedia:**
   * **Chức năng:** Là các lớp con kế thừa từ Tailieu, đại diện cho các loại tài liệu cụ thể. Sach có thêm liên kết tới Theloai, Tapchi có Số tạp chí, Multimedia có thông tin file.
   * **Ý nghĩa:** Chuyên biệt hóa thông tin cho từng loại tài liệu.
3. **Banin (Bản in):**
   * **Chức năng:** Đại diện cho một bản sao vật lý, cụ thể của một Tailieu. Ví dụ, Tailieu là "Harry Potter tập 1", nhưng thư viện có 5 cuốn thì sẽ có 5 Banin khác nhau, mỗi bản có một mã riêng (MaBI).
   * **Ý nghĩa:** Đây là thực thể cốt lõi để quản lý việc mượn/trả. Hệ thống theo dõi trạng thái của từng bản in riêng lẻ (đang được mượn hay có sẵn).
4. **Theloai, Loaimultimedia (Thể loại):**
   * **Chức năng:** Phân loại cho Sach và Multimedia.
   * **Ý nghĩa:** Giúp tổ chức và tìm kiếm tài liệu theo danh mục.

**Nhóm Thực thể Giao dịch và Quy định:**

1. **Phieumuon (Phiếu mượn):**
   * **Chức năng:** Ghi lại một giao dịch mượn tài liệu, liên kết một Docgia với ngày mượn và ngày trả dự kiến.
   * **Ý nghĩa:** Quản lý toàn bộ thông tin của một lần mượn.
2. **CTPhieumuon (Chi tiết Phiếu mượn):**
   * **Chức năng:** Là một dòng chi tiết trong Phieumuon, liên kết phiếu mượn đó với một Banin cụ thể. Một Phieumuon có thể có nhiều CTPhieumuon (mượn nhiều cuốn cùng lúc).
   * **Ý nghĩa:** Ghi lại chính xác bản sao nào đã được mượn trong một giao dịch.
3. **Phieudatmuon (Phiếu đặt mượn):**
   * **Chức năng:** Ghi lại yêu cầu đặt trước một Tailieu của một Docgia.
   * **Ý nghĩa:** Quản lý danh sách chờ cho các tài liệu đang được người khác mượn.
4. **CTPhieudatmuon (Chi tiết Phiếu đặt mượn):**
   * **Chức năng:** Là một dòng chi tiết, liên kết Phieudatmuon với một Tailieu cụ thể mà độc giả muốn đặt.
   * **Ý nghĩa:** Ghi lại chính xác đầu tài liệu nào đã được đặt trước.
5. **QuyDinh (Quy định):**
   * **Chức năng:** Lưu trữ các tham số, quy định của thư viện như thời gian mượn tối đa, số lượng sách được mượn tối đa...
   * **Ý nghĩa:** Giúp hệ thống hoạt động một cách linh hoạt, dễ dàng thay đổi quy định mà không cần sửa code.

**GIẢI THÍCH CÁC LUỒNG NGHIỆP VỤ**

**LUỒNG 1: MỘT ĐỘC GIẢ MƯỢN TÀI LIỆU**

1. Bắt đầu: Một Docgia mang các Banin (bản in cụ thể) đến quầy của NVThuthu (Thủ thư).
2. Hành động: Tạo Phiếu mượn
   * Thủ thư tạo một Phieumuon mới trong hệ thống.
   * Phiếu này được liên kết với Docgia đang thực hiện giao dịch và ghi nhận Ngaymuon.
3. Hành động: Thêm chi tiết mượn
   * Với mỗi Banin mà độc giả mượn, thủ thư tạo một bản ghi CTPhieumuon.
   * Mỗi bản ghi CTPhieumuon này được liên kết với Phieumuon vừa tạo và Banin tương ứng.
4. Hành động: Cập nhật trạng thái
   * Hệ thống sẽ cập nhật trạng thái của từng Banin thành "đã cho mượn".
5. Kết thúc: Giao dịch hoàn tất, độc giả nhận tài liệu.

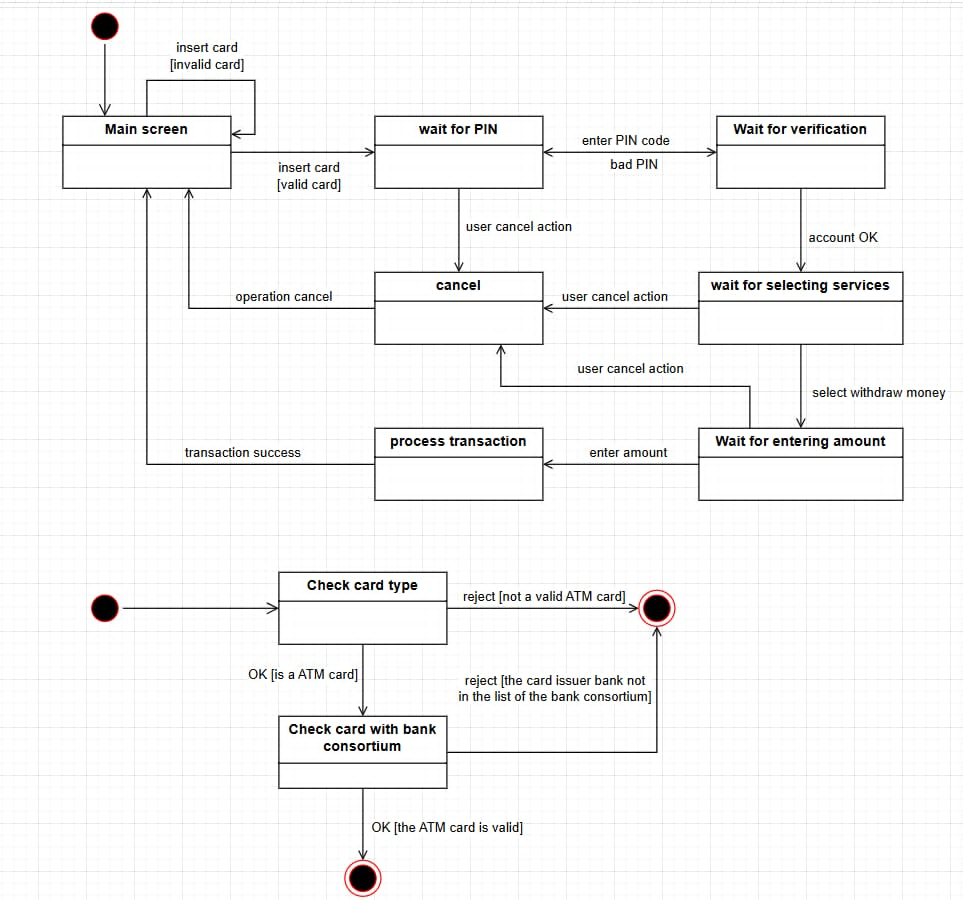
**Tóm tắt Luồng 1: Độc giả chọn bản in -> Thủ thư tạo 1 Phiếu mượn -> Với mỗi bản in, tạo 1 Chi tiết phiếu mượn -> Cập nhật trạng thái bản in -> Hoàn tất.**

**LUỒNG 2: MỘT ĐỘC GIẢ ĐẶT TRƯỚC TÀI LIỆU**

1. Bắt đầu: Một Docgia muốn đặt trước một Tailieu (một đầu sách) hiện đang hết bản in có sẵn.
2. Hành động: Tạo Phiếu đặt mượn
   * NVTieptan (Nhân viên tiếp tân) tạo một Phieudatmuon mới, liên kết với Docgia.
3. Hành động: Thêm chi tiết đặt mượn
   * Nhân viên tạo một bản ghi CTPhieudatmuon, liên kết Phieudatmuon với Tailieu mà độc giả muốn đặt.
4. Hành động: Xử lý khi có sách
   * Khi một Banin của Tailieu đó được trả về, hệ thống sẽ kiểm tra danh sách Phieudatmuon và thông báo cho Docgia đầu tiên trong danh sách chờ.
5. Kết thúc: Yêu cầu đặt trước được ghi nhận vào hệ thống.

**Tóm tắt Luồng 2: Độc giả yêu cầu đặt trước 1 đầu tài liệu -> Nhân viên tạo 1 Phiếu đặt mượn -> Tạo 1 Chi tiết phiếu đặt mượn tương ứng -> Hệ thống ghi nhận yêu cầu.**

Bài 6.



**SƠ ĐỒ 1: LUỒNG TƯƠNG TÁC NGƯỜI DÙNG VỚI ATM**

**GIẢI THÍCH CHỨC NĂNG CÁC THÀNH PHẦN (CÁC TRẠNG THÁI)**

1. **Main screen (Màn hình chính):**
   * **Chức năng:** Đây là trạng thái khởi đầu và trạng thái chờ của máy ATM. Tại đây, máy hiển thị màn hình chào và sẵn sàng nhận thẻ từ người dùng.
   * **Ý nghĩa:** Trạng thái nghỉ, sẵn sàng phục vụ.
2. **wait for PIN (Chờ nhập mã PIN):**
   * **Chức năng:** Sau khi máy chấp nhận thẻ hợp lệ, nó chuyển sang trạng thái này để yêu cầu người dùng nhập mã PIN.
   * **Ý nghĩa:** Trạng thái xác thực bước đầu.
3. **Wait for verification (Chờ xác thực):**
   * **Chức năng:** Sau khi người dùng nhập PIN, máy chuyển sang trạng thái này để gửi thông tin thẻ và mã PIN đến hệ thống ngân hàng để kiểm tra.
   * **Ý nghĩa:** Trạng thái giao tiếp với máy chủ ngân hàng để xác thực thông tin đăng nhập.
4. **wait for selecting services (Chờ chọn dịch vụ):**
   * **Chức năng:** Mã PIN đã được xác thực thành công. Máy hiển thị một menu các dịch vụ cho người dùng lựa chọn (ví dụ: Rút tiền, Xem số dư, Chuyển khoản...).
   * **Ý nghĩa:** Trạng thái cung cấp các chức năng chính của ATM.
5. **Wait for entering amount (Chờ nhập số tiền):**
   * **Chức năng:** Người dùng đã chọn dịch vụ "rút tiền". Máy chuyển sang trạng thái này để yêu cầu người dùng nhập số tiền muốn rút.
   * **Ý nghĩa:** Trạng thái thu thập thông tin cho giao dịch.
6. **process transaction (Xử lý giao dịch):**
   * **Chức năng:** Máy ATM đã có đủ thông tin (dịch vụ, số tiền). Nó đang thực hiện giao dịch: kiểm tra số dư, gửi yêu cầu trừ tiền đến ngân hàng, chuẩn bị nhả tiền.
   * **Ý nghĩa:** Trạng thái thực thi nghiệp vụ cốt lõi.
7. **cancel (Hủy giao dịch):**
   * **Chức năng:** Là trạng thái trung gian khi người dùng nhấn nút "Cancel". Trạng thái này xử lý việc dừng hoạt động hiện tại và chuẩn bị trả lại thẻ.
   * **Ý nghĩa:** Trạng thái xử lý yêu cầu hủy bỏ từ người dùng.

**GIẢI THÍCH CÁC LUỒNG XỬ LÝ (KỊCH BẢN)**

**LUỒNG 1: RÚT TIỀN THÀNH CÔNG**

1. Bắt đầu: Máy đang ở trạng thái Main screen.
2. Sự kiện insert card [valid card]: Người dùng đưa thẻ hợp lệ vào, máy chuyển sang trạng thái wait for PIN.
3. Sự kiện enter PIN code: Người dùng nhập mã PIN, máy chuyển sang Wait for verification.
4. Sự kiện account OK: Ngân hàng xác nhận PIN đúng, máy chuyển sang wait for selecting services.
5. Sự kiện select withdraw money: Người dùng chọn chức năng rút tiền, máy chuyển sang Wait for entering amount.
6. Sự kiện enter amount: Người dùng nhập số tiền, máy chuyển sang process transaction.
7. Sự kiện transaction success: Giao dịch thành công (máy nhả tiền và biên lai), máy quay trở lại trạng thái Main screen và trả thẻ.

**Tóm tắt Luồng 1: Đây là kịch bản "happy path", mô tả một giao dịch rút tiền hoàn chỉnh và thành công từ đầu đến cuối.**

**LUỒNG 2: NHẬP SAI MÃ PIN**

1. Các bước diễn ra như Luồng 1 cho đến khi máy ở trạng thái Wait for verification.
2. Sự kiện bad PIN: Ngân hàng báo mã PIN sai. Máy sẽ chuyển ngược lại trạng thái wait for PIN để cho phép người dùng nhập lại.

**Tóm tắt Luồng 2: Mô tả trường hợp người dùng nhập sai thông tin xác thực và hệ thống cho phép thử lại.**

**LUỒNG 3: NGƯỜI DÙNG HỦY GIAO DỊCH**

1. Bắt đầu: Giả sử máy đang ở trạng thái wait for selecting services.
2. Sự kiện user cancel action: Người dùng nhấn nút hủy. Máy chuyển sang trạng thái cancel.
3. Sự kiện operation cancel: Trạng thái cancel xử lý xong (ví dụ: chuẩn bị trả thẻ), máy chuyển về Main screen.

**Tóm tắt Luồng 3: Mô tả cách hệ thống xử lý khi người dùng chủ động muốn kết thúc giao dịch giữa chừng.**

**SƠ ĐỒ 2: QUY TRÌNH KIỂM TRA TÍNH HỢP LỆ CỦA THẺ**

**GIẢI THÍCH CÁC LUỒNG XỬ LÝ (KỊCH BẢN)**

**LUỒNG 1: THẺ HỢP LỆ HOÀN TOÀN (KỊCH BẢN THÀNH CÔNG)**

1. Bắt đầu: Người dùng đưa thẻ vào máy ATM.
2. Hành động: Check card type
   * Hệ thống kiểm tra và xác nhận đây là một thẻ ATM đúng chuẩn. Điều kiện [OK is a ATM card] được thỏa mãn, luồng xử lý đi tiếp.
3. Hành động: Check card with bank consortium
   * Hệ thống tiếp tục kiểm tra và xác nhận ngân hàng phát hành thẻ nằm trong mạng lưới được hỗ trợ. Điều kiện [OK the ATM card is valid] được thỏa mãn.
4. Kết thúc: Quy trình kiểm tra thẻ kết thúc thành công. Máy ATM sẽ chuyển sang trạng thái tiếp theo (ví dụ: yêu cầu nhập mã PIN).

**Tóm tắt Luồng 1: Thẻ được xác nhận vừa là thẻ ATM, vừa thuộc mạng lưới hỗ trợ -> Thẻ hợp lệ để giao dịch.**

**LUỒNG 2: THẺ KHÔNG PHẢI LÀ THẺ ATM (KỊCH BẢN THẤT BẠI)**

1. Bắt đầu: Người dùng đưa một vật thể (ví dụ: thẻ thư viện) vào máy.
2. Hành động: Check card type
   * Hệ thống kiểm tra và phát hiện đây không phải là một thẻ ATM đúng chuẩn. Điều kiện [reject not a valid ATM card] được thỏa mãn.
3. Kết thúc: Quy trình ngay lập tức bị từ chối và đi đến trạng thái kết thúc (thất bại). Máy sẽ trả lại thẻ và hiển thị thông báo lỗi.

**Tóm tắt Luồng 2: Vật thể đưa vào không đúng định dạng thẻ ATM -> Bị từ chối ngay lập tức.**

**LUỒNG 3: THẺ ATM KHÔNG ĐƯỢC HỖ TRỢ (KỊCH BẢN THẤT BẠI)**

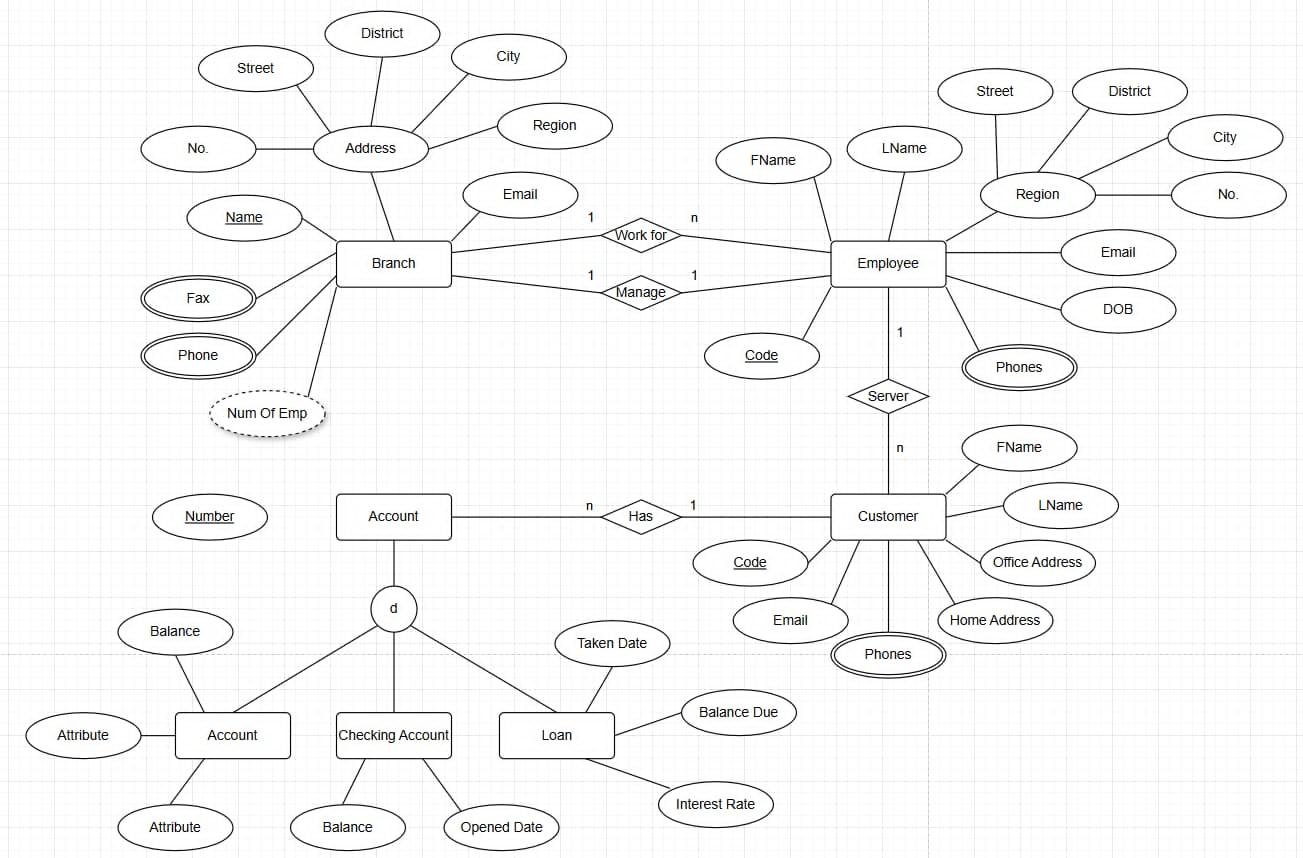
1. Bắt đầu: Người dùng đưa một thẻ ATM hợp lệ vào máy.
2. Hành động: Check card type
   * Hệ thống xác nhận đây là thẻ ATM ([OK is a ATM card]). Luồng đi tiếp.
3. Hành động: Check card with bank consortium
   * Hệ thống kiểm tra và phát hiện ngân hàng phát hành thẻ này không thuộc mạng lưới được máy ATM hỗ trợ. Điều kiện [reject the card issuer bank not in the list of the bank consortium] được thỏa mãn.
4. Kết thúc: Quy trình bị từ chối và đi đến trạng thái kết thúc (thất bại). Máy sẽ trả lại thẻ và hiển thị thông báo "Thẻ không được hỗ trợ".

**Tóm tắt Luồng 3: Thẻ là thẻ ATM thật, nhưng không tương thích với máy ATM -> Bị từ chối.**

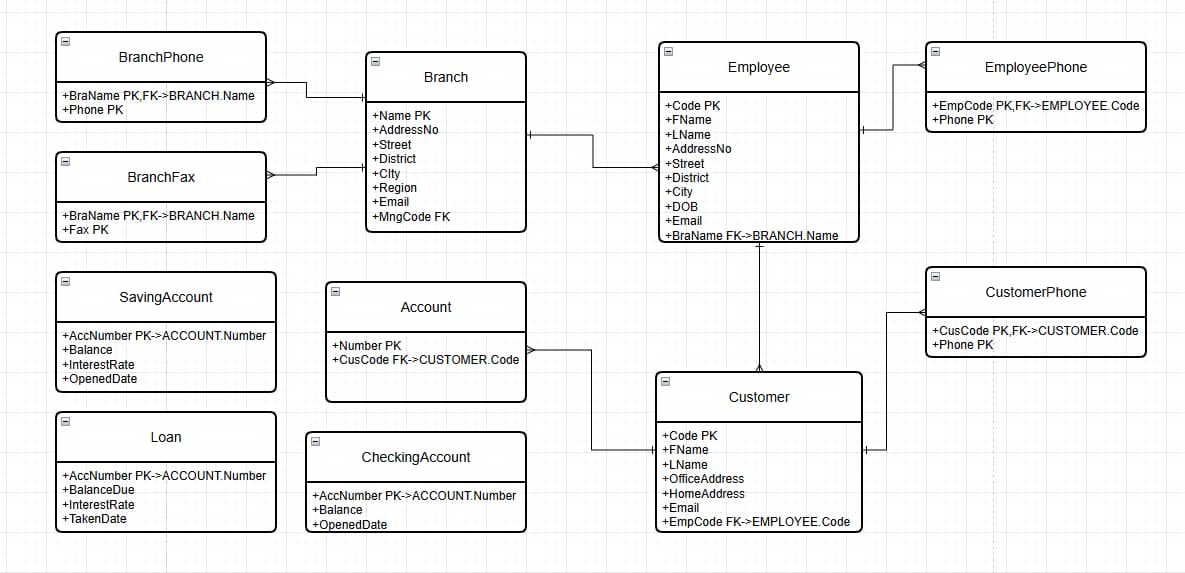
Bài 7.

1. Thiết kế Cơ sở dữ liệu ở mức Khái niệm, Luận lý và Vật lý

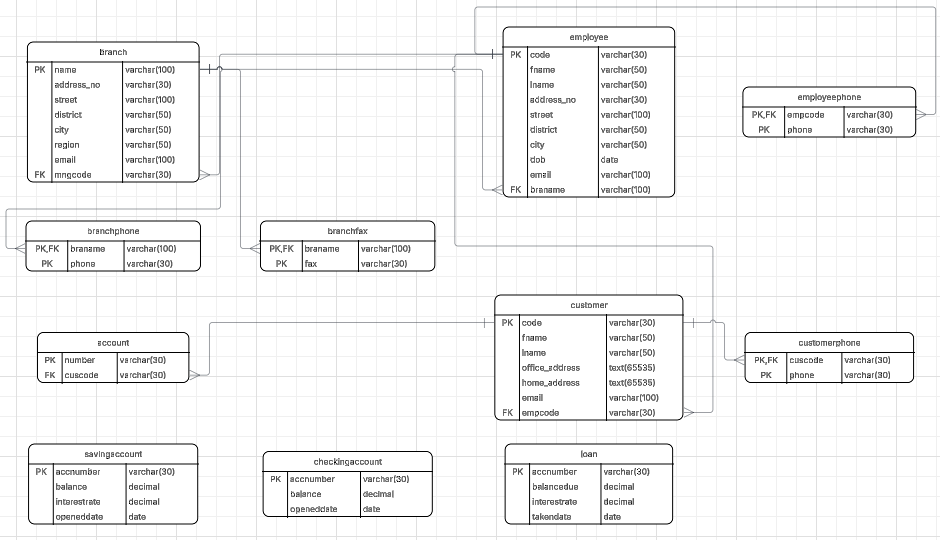
- Mức khái niệm:



- Mức luận lý:



- Mức vật lý:



1. Query: Insert, Update, Delete, Select

*a.*

-- BRANCH

INSERT INTO Branch(BranchID, Name, AddressNo, Street, District, City, Region, Email, NumEmployees, ManagerEmpCode)

VALUES

('B001','ABC Central','10','Hai Ba Trung','Dist1','HCMC','South','central@abc.com', 3, NULL),

('B002','ABC North','22','Nguyen Trai','Dist2','Hanoi','North','north@abc.com', 2, NULL),

('B003','ABC West','5','Le Loi','Dist3','Da Nang','Central','west@abc.com', 2, NULL),

('B004','ABC East','12','Tran Hung Dao','Dist4','Can Tho','South','east@abc.com', 1, NULL);

-- BRANCH\_PHONE

INSERT INTO BranchPhone(BranchPhoneID, BranchID, PhoneNumber) VALUES

(1,'B001','(028)1234567'),

(2,'B001','(028)7654321'),

(3,'B002','(024)1112222'),

(4,'B003','(0236)3334444');

-- BRANCH\_FAX

INSERT INTO BranchFax(BranchFaxID, BranchID, FaxNumber) VALUES

(1,'B001','(028)1234000'),

(2,'B002','(024)1112000'),

(3,'B003','(0236)3330000'),

(4,'B004','(0292)4440000');

-- EMPLOYEE - EmpCode là mã string

INSERT INTO Employee(EmpCode, FName, LName, DOB, Email, AddressNo, Street, District, City, Region, BranchID)

VALUES

('N1001','Alice','Smith','1985-07-10','alice.smith@abc.com','15','Le Loi','Dist3','Da Nang','Central','B003'),

('N1002','Bob','Lee','1978-11-05','bob.lee@abc.com','20','Nguyen Trai','Dist2','Hanoi','North','B002'),

('N1080','John','Doe','1982-03-15','john.doe@abc.com','3','Hai Ba Trung','Dist1','HCMC','South','B001'),

('N1004','Karen','Tran','1990-03-22','karen.tran@abc.com','8','Tran Hung Dao','Dist4','Can Tho','South','B004');

-- EMPLOYEE\_PHONE

INSERT INTO EmployeePhone(EmpPhoneID, EmpCode, PhoneNumber) VALUES

(1,'N1001','0908123456'),

(2,'N1002','0912345678'),

(3,'N1080','0987654321'),

(4,'N1004','0977123456');

-- CUSTOMER

INSERT INTO Customer(CusCode, FName, LName, HomeAddress, OfficeAddress, Email, EmpCode)

VALUES

('C001','Mai','Nguyen','HNo1 StA','Office A','mai.nguyen@x.com','N1001'),

('C002','Peter','Johnson','HNo2 StB','Office B','p.johnson@example.com','N1080'),

('C003','Tran','Vu','HNo3 StC','Office C','tran.vu@x.com','N1002'),

('C004','Linh','Pham','HNo4 StD','Office D','linh.pham@x.com','N1004');

-- CUSTOMER\_PHONE

INSERT INTO CustomerPhone(CusPhoneID, CusCode, PhoneNumber) VALUES

(1,'C001','0911000111'),

(2,'C002','0912000222'),

(3,'C003','0913000333'),

(4,'C004','0914000444');

-- ACCOUNT, AccountType có thể là 'SAVING','CHECKING','LOAN'

INSERT INTO Account(AccNumber, CusCode, AccountType, Balance, OpenedDate) VALUES

('A2001','C002','SAVING', 5000.00, '2019-06-01'), -- Peter Johnson saving

('A2002','C002','SAVING', 12000.50, '2020-01-15'), -- Peter Johnson another saving

('A3001','C001','CHECKING', 2500.00, '2018-09-10'),

('L4001','C003','LOAN', -15000.00, '2021-03-05');

-- SAVING\_ACCOUNT (matching AccNumber)

INSERT INTO SavingAccount(AccNumber, InterestRate, OpenedDate, Balance) VALUES

('A2001', 0.0150, '2019-06-01', 5000.00),

('A2002', 0.0200, '2020-01-15', 12000.50),

('A2003', 0.0180, '2017-04-20', 300.00), -- if extra saving account exists

('A2004', 0.0120, '2016-12-12', 1000.00);

-- CHECKING\_ACCOUNT

INSERT INTO CheckingAccount(AccNumber, OpenedDate, Balance) VALUES

('A3001','2018-09-10',2500.00),

('A3002','2019-05-05', 400.00),

('A3003','2017-07-07', 900.00),

('A3004','2020-11-11', 150.00);

-- LOAN

INSERT INTO Loan(AccNumber, BalanceDue, InterestRate, TakenDate) VALUES

('L4001',15000.00, 0.0750,'2021-03-05'),

('L4002',8000.00, 0.0900,'2019-08-10'),

('L4003',20000.00,0.0600,'2018-02-20'),

('L4004',5000.00, 0.1000,'2022-01-01');

*b.*

UPDATE Employee

SET LName = 'Brown'

WHERE EmpCode = 'N1080';

*c.*

DELETE FROM Employee

WHERE EmpCode = 'N1080';

*d.*

SELECT a.AccNumber

FROM Account a

JOIN Customer c ON a.CusCode = c.CusCode

WHERE c.FName = 'Peter' AND c.LName = 'Johnson';

*e.*

SELECT EmpCode, FName, LName, DOB

FROM Employee

WHERE MONTH(DOB) = 3 AND YEAR(DOB) = 2010;

*f.*

SELECT c.FName, c.LName, SUM(s.Balance) AS TotalSavingBalance

FROM Customer c

JOIN Account a ON c.CusCode = a.CusCode

JOIN SavingAccount s ON s.AccNumber = a.AccNumber

WHERE c.FName = 'Peter' AND c.LName = 'Johnson'

AND a.AccountType = 'SAVING'

GROUP BY c.CustCode, c.FName, c.LName;

Bài tập ứng dụng

1) Hãy liệt kê các yêu cầu chức năng và phi chức năng của hệ thống CoolStore dựa trên mô tả kịch bản nghiệp vụ.

**Yêu cầu chức năng**

**Danh mục sản phẩm:**

**Người mua (Buyer):** Duyệt danh sách sản phẩm với các chức năng lọc và sắp xếp theo tên sản phẩm và giá. Xem chi tiết sản phẩm trên trang danh sách sản phẩm bằng cách nhấp vào sản phẩm đó. Trên trang chi tiết, người mua có thể xem tên, mô tả, số lượng tồn kho, thông tin kho như địa chỉ kho, trạng thái "sản phẩm nổi bật" (nếu có) và đánh giá

**Quản trị viên (SysAdmin):** Quản lý sản phẩm; gán sản phẩm vào kho hàng.

**Giỏ hàng:**

**Người mua:** Thêm sản phẩm từ danh sách sản phẩm hoặc trang chi tiết sản phẩm.

Xem sản phẩm trên giỏ hàng và bảng tổng hợp (tổng chi phí giỏ hàng, tổng chi phí sau khuyến mãi, chi phí vận chuyển, khuyến mãi

tiết kiệm, tổng giá trị đơn hàng).

Giỏ hàng và bảng sẽ được cập nhật khi thêm hay xóa sản phẩm.

Nhấn nút Thanh toán trên trang giỏ hàng để thực hiện quy trình thanh toán.

**Quản trị viên**: Xem giỏ hàng bất kỳ người dùng và kích hoạt hoặc vô hiệu hóa giỏ hàng không hợp lệ của người dùng.

**Thanh toán:**

Hệ thống sẽ xác thực thông tin, xử lý thanh toán gửi email xác nhận cho người mua.

**Tồn kho:** Quản trị viên có thểquản lý kho hàng

**Đánh giá:**

Người mua có thể đánh giá bất kỳ sản phẩm nào (thang điểm 1 -> 5 sao)

**Kiểm soát truy cập:**

Người dùng (Người mua, Quản trị viên) có thể đăng nhập và đăng xuất.

Hệ thống điều hướng người dùng đến trang phù hợp sau khi đăng nhập (Người mua tới trang sản phẩm, Quản trị viên tới trang quản trị).

**Yêu cầu phi chức năng (Non-functional Requirements)**

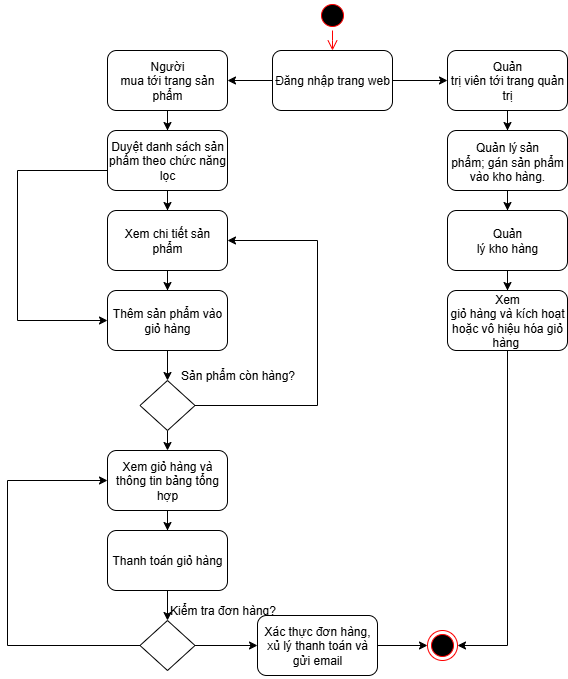
**Tính khả dụng (Usability):** Giao diện phải thân thiện, dễ sử dụng cho cả Người mua và Quản trị viên. Quy trình mua sắm phải trực quan và đơn giản.

**Hiệu năng (Performance):** Thời gian tải trang, đặc biệt là trang danh mục sản phẩm có chức năng lọc và sắp xếp, phải nhanh và chính xác. Giỏ hàng phải được cập nhật ngay lập tức.

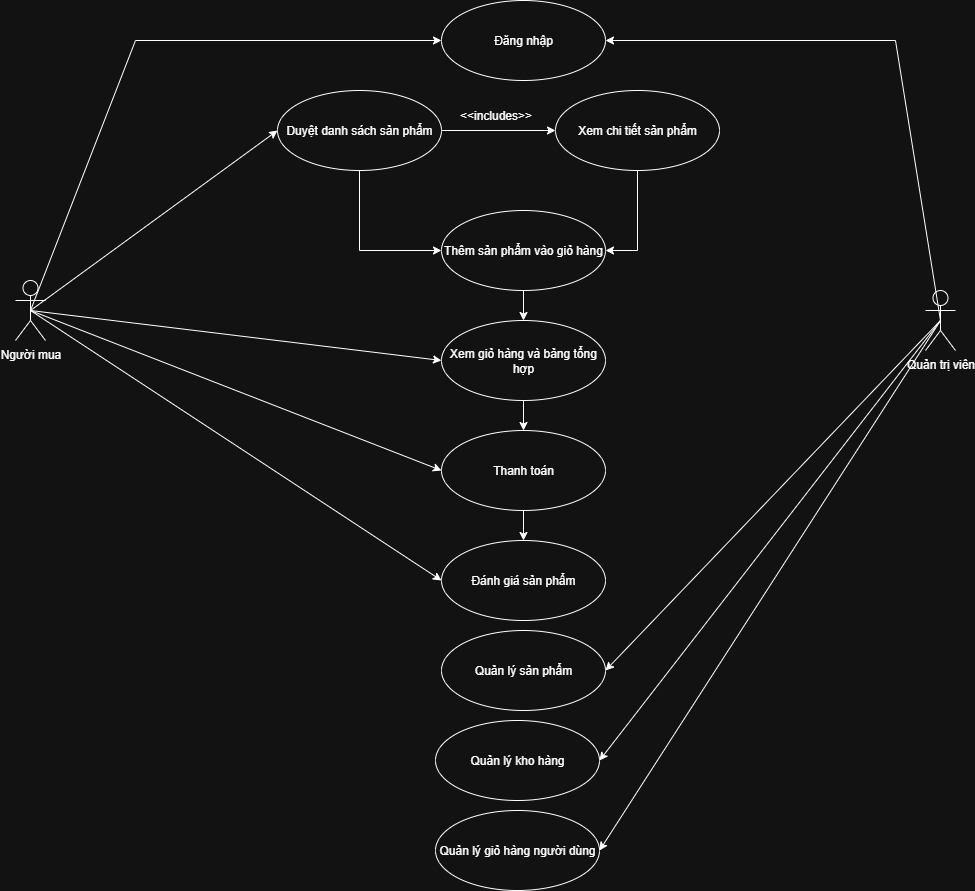
**Độ tin cậy (Reliability):** Hệ thống phải hoạt động ổn định, đặc biệt là quy trình thanh toán phải chính xác và không xảy ra lỗi. Dữ liệu tồn kho phải được cập nhật đồng bộ.

**Bảo mật (Security):** Hệ thống phải bảo mật thông tin tài khoản người dùng và thông tin thanh toán. Phải có sự phân quyền rõ ràng giữa Người mua và Quản trị viên.

2) Vẽ qui trình nghiệp vụ cho website CoolStore



3) Xây dựng use case diagram cho website CoolStore.

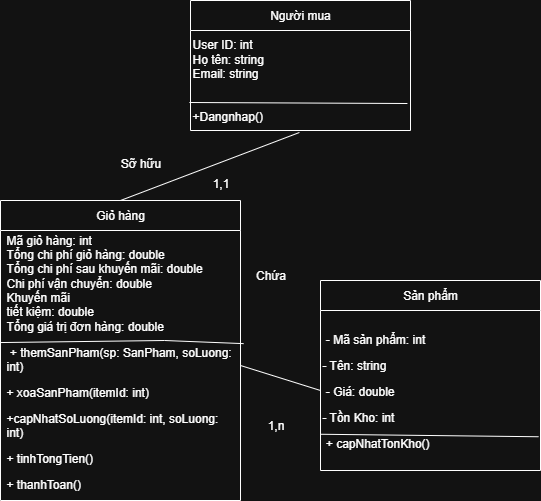


4) Mô tả chi tiết một Use Case "Mua sản phẩm từ trang chi tiết sản phẩm". Trình bày theo mẫu chuẩn gồm: Tên use case, Tác nhân chính, Mục tiêu, Tiền điều kiện, Luồng chính, Luồng thay thế, Hậu điều kiện, Ghi chú (nếu có).

|  |  |  |
| --- | --- | --- |
| Tên use case | Mua sản phẩm từ trang chi tiết sản phẩm | |
| Tác nhân chính | Người mua | |
| Mục tiêu | Cho phép người mua thêm một sản phẩm vào giỏ hàng từ trang hiển thị thông tin chi tiết của sản phẩm | |
| Tiền điều kiện | Sản phẩm còn hàng; Người mua đang ở trang duyệt sản phẩm | |
| Luồng chính | Actor action | System Respone |
| 1. Người mua mở trang chi tiết sản phẩm |  |
|  | 2. Hệ thống hiển thị thông tin sản phẩm (tên, mô tả, tồn kho, giá, đánh giá, trạng thái nổi bật). |
| 3. Người mua nhấn nút **“Mua hàng”**. |  |
|  | 4. Hệ thống kiểm tra tồn kho. **A1** |
|  | 5. Hệ thống thêm sản phẩm vào giỏ hàng. |
| Luồng thay thế | **A1** Nếu sản phẩm hết hàng, hệ thống hiển thị thông báo "Sản phẩm đã hết hàng" và không cho thêm vào giỏ. | |
| Hậu điều kiện: | Sản phẩm được thêm vào giỏ hàng thành công.  Giỏ hàng được cập nhật. | |

5) Sử dụng sơ đồ Activity Diagram để mô tả quy trình mua hàng từ khi người dùng nhấn “Mua hàng” cho đến khi hoàn tất thanh toán.

6) Thiết kế sơ đồ Class Diagram cho module Giỏ hàng.



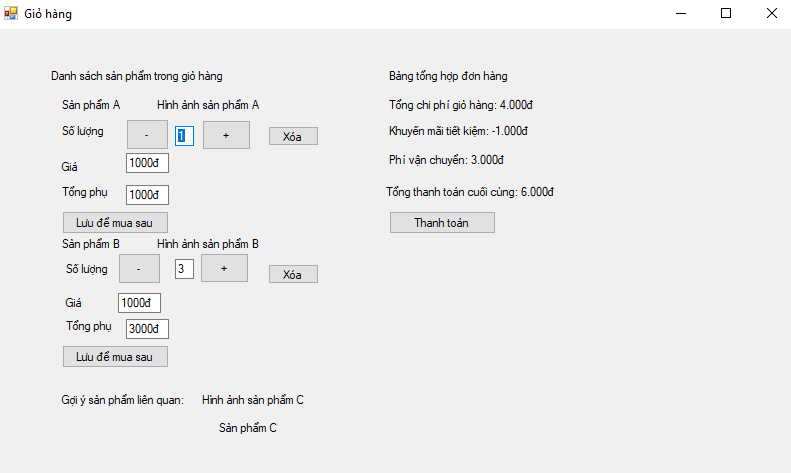
7) Thiết kế giao diện màn hình cho chức năng “Quản lý giỏ hàng” (có thể thiết kế một hoặc nhiều màn hình) gồm các thông tin:

• Danh sách sản phẩm trong giỏ hàng: tên sản phẩm, hình ảnh, số lượng, giá từng món, tổng phụ.Kiểm thử phần mềm Bài tập 2: Phân tích thiết kế hệ thống

• Chức năng thao tác nhanh: Tăng/giảm số lượng sản phẩm; Xóa sản phẩm khỏi giỏ; Lưu sản phẩm để mua sau.

• Bảng tổng hợp đơn hang: Tổng chi phí giỏ hang; Khuyến mãi tiết kiệm được; Phí vận chuyển; Tổng thanh toán cuối cùng

• Hành động chính: Nút “Thanh toán”; Gợi ý sản phẩm liên quan để thêm vào giỏ.



8) Đề xuất cách tổ chức kiến trúc phần mềm cho hệ thống CoolStore theo mô hình kiến trúc ba lớp (3-tier architecture).

**Lớp View**

**Trách nhiệm chính**:

* Hiển thị giao diện người dùng (UI).
* Thu thập dữ liệu đầu vào từ người dùng (như click chuột, nhập liệu vào form).
* Gửi yêu cầu đến lớp Controller để xử lý.

**Các thành phần trong CoolStore**:

* **Trang danh mục sản phẩm**: Hiển thị danh sách sản phẩm cho người mua duyệt, lọc và sắp xếp.
* **Trang chi tiết sản phẩm**: Hiển thị đầy đủ thông tin về một sản phẩm cụ thể.
* **Giao diện giỏ hàng**: Liệt kê các sản phẩm đã được thêm vào và hiển thị bảng tổng hợp đơn hàng.
* **Các trang quản trị**: Giao diện cho phép Quản trị viên (SysAdmin) quản lý sản phẩm, kho hàng và giỏ hàng của người dùng.
* **Form đăng nhập/đăng xuất**.

**Lớp Controller**

**Trách nhiệm chính**:

* Nhận yêu cầu từ lớp View.
* Thực thi các quy tắc nghiệp vụ (ví dụ: kiểm tra tồn kho, tính toán khuyến mãi).
* Tương tác với lớp Database để đọc hoặc ghi dữ liệu.
* Trả kết quả xử lý về lại cho lớp View để hiển thị.

**Các thành phần trong CoolStore**:

* **Xử lý logic giỏ hàng**: Logic thêm sản phẩm, xóa sản phẩm, và tự động cập nhật lại tổng tiền khi giỏ hàng thay đổi.
* **Xử lý thanh toán**: Xác thực thông tin, xử lý giao dịch và gửi email xác nhận cho người mua.
* **Xử lý kiểm soát truy cập**: Xác thực thông tin đăng nhập và phân quyền cho người dùng (Người mua hay Quản trị viên).
* **Xử lý logic quản lý**: Các quy trình thêm/sửa/xóa sản phẩm, gán sản phẩm vào kho do Quản trị viên thực hiện.

**Lớp Database**

**Trách nhiệm chính**:

* Lưu trữ dữ liệu (ví dụ: thông tin sản phẩm, tài khoản người dùng).
* Cung cấp các phương thức để lớp Controller có thể truy vấn, thêm, sửa, xóa dữ liệu (CRUD).
* Đảm bảo tính toàn vẹn và bảo mật của dữ liệu.

**Các thành phần trong CoolStore**:

* **Cơ sở dữ liệu sản phẩm**: Lưu trữ thông tin chi tiết về sản phẩm như tên, mô tả, số lượng tồn kho, đánh giá.
* **Cơ sở dữ liệu người dùng**: Lưu thông tin tài khoản của Người mua và Quản trị viên.
* **Cơ sở dữ liệu giỏ hàng và đơn hàng**: Lưu trạng thái giỏ hàng của từng người dùng và lịch sử các đơn hàng đã thanh toán.
* **Cơ sở dữ liệu kho hàng**: Lưu thông tin về các kho và sản phẩm trong đó.

9) Viết ít nhất 3 ca kiểm thử (test cases) cho chức năng “Thêm sản phẩm vào giỏ hàng” – bao gồm dữ liệu đầu vào, bước thực hiện và kết quả mong đợi.

|  |  |  |  |
| --- | --- | --- | --- |
| Mã TC | Dữ liệu đầu vào | Các bước thực hiện | Kết quả mong đợi |
| TC\_ADD\_01 | Người dùng đã đăng nhập; Sản phẩm "Laptop ABC" (tồn kho: 10); Giỏ hàng trống. | Truy cập trang sản phẩm "Laptop ABC" và nhấn "Thêm vào giỏ hàng". | Hiển thị thông báo thành công; Giỏ hàng cập nhật có 1 sản phẩm "Laptop ABC"; Tổng tiền được tính đúng. |
| TC\_ADD\_02 | Người dùng đã đăng nhập; Sản phẩm "Chuột XYZ" (tồn kho: 0). | Truy cập trang sản phẩm "Chuột XYZ" và nhấn "Thêm vào giỏ hàng". | Hiển thị thông báo lỗi "Sản phẩm đã hết hàng"; Giỏ hàng không thay đổi. |
| TC\_ADD\_03 | Người dùng chưa đăng nhập; Sản phẩm "Laptop ABC" (tồn kho: 10); Giỏ hàng đã có 1 "Laptop ABC". | Truy cập lại trang sản phẩm "Laptop ABC" và nhấn "Thêm vào giỏ hàng". | Hiển thị thông báo lỗi “Vui lòng đăng nhập”; Tự động di chuyển đến trang đăng nhập; Giỏ hàng không thay đổi |