****

**BÁO CÁO BÀI TẬP LAB 2**

**KIỂM THỬ PHẦN MỀM**

**GV hướng dẫn: TS. Đỗ Như Tài**

**Lớp: DCT122C3 – Mã học phần: 841408**

**Sinh viên: Trịnh Long Phát – 3122411150**

**Lê Hồng Phát – 3122411145**

**Trương Phú Kiệt – 3122411109**

**Trà Đức Toàn – 3122411217**

**Thành phố Hồ Chí Minh, tháng 9 năm 2025**

**BẢNG PHÂN CÔNG CỦA NHÓM**

|  |  |  |
| --- | --- | --- |
| **Tên các thành viên thực hiện** | **Nội dung công việc** | **Tiến độ**  **công việc** |
| **Trịnh Long Phát** | **Tổng hợp và phân chia công việc** | **100%** |
| **Bài tập ứng dụng: câu 1, 2, 3, 4** | **100%** |
|  | **100%** |
| **Lê Hồng Phát** | **Bài 7: Phân tích dữ liệu cho ABC Bank** | **100%** |
| **Bài tập ứng dụng: câu 5, 6** | **100%** |
| **Viết báo cáo word, tổng hợp drawio** | **100%** |
| **Trương Phú Kiệt** | **Bài 4: Vẽ sơ đồ hoạt động và tương tác sau và tóm tắt ý nghĩa sơ đồ**  **Bài 5: Vẽ lược đồ lớp cho bài toán Quản lý thư viện**  **Bài 6: Vẽ lược đồ** | **100%** |
| **Bài tập ứng dụng: câu 8, 9** | **100%** |
| **Trà Đức Toàn** | **Bài 1: Vẽ lại qui trình nghiệp vụ sau (Độc giả, NV Tiếp tân, Thành viên)**  **Bài 2: Vẽ mô hình khái niệm**  **Bài 3: Vẽ lại business use case sau** | **100%** |
| **Bài tập ứng dụng: câu 5, 6** | **100%** |

**MỤC LỤC**

Contents

Bài 1: Vẽ lại qui trình nghiệp vụ sau (Độc giả, NV Tiếp tân, Thành viên) 5

Bài 2: Vẽ mô hình khái niệm 6

Bài 3: Vẽ lại business use case 7

a. Các Actor xuất hiện trong sơ đồ gồm 7

b. Liệt kê và giải thích các trường hợp sử dụng use case trong sơ đồ 8

c. Phân tích các quan hệ trong sơ đồ 9

d. Viết kịch bản cho trường hợp cụ thể 10

Bài 4: Vẽ sơ đồ hoạt động và tương tác sau và tóm tắt ý nghĩa sơ đồ 12

Bài 5: Vẽ lược đồ lớp cho bài toán Quản lý thư viện 14

Bài 6: Vẽ lại lược đồ 15

Bài 7: Phân tích dữ liệu cho ABC Bank 16

1. Thiết kế Cơ sở dữ liệu ở mức Khái niệm, Luận lý và Vật lý 17

Mức khái niệm (Conceptual Model) 17

Mức luận lý (Logical Model) 18

Mức vật lý (Physical Model) 19

2. Query: Insert, Update, Delete, Select 20

BÀI TẬP ỨNG DỤNG 25

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. 26

2. Vẽ qui trình nghiệp vụ cho website CoolStore 28

3. Xây dựng use case diagram cho website CoolStore. 29

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

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. 33

6. Thiết kế sơ đồ Class Diagram cho module Giỏ hàng. 34

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: 34

8. Đề xuất cách tổ chức kiến trúc phần mềm cho hệ thống CoolStore theo mô hình kiến trúc ba lớp (3-tier architecture). 36

8.1. Tổng quan 36

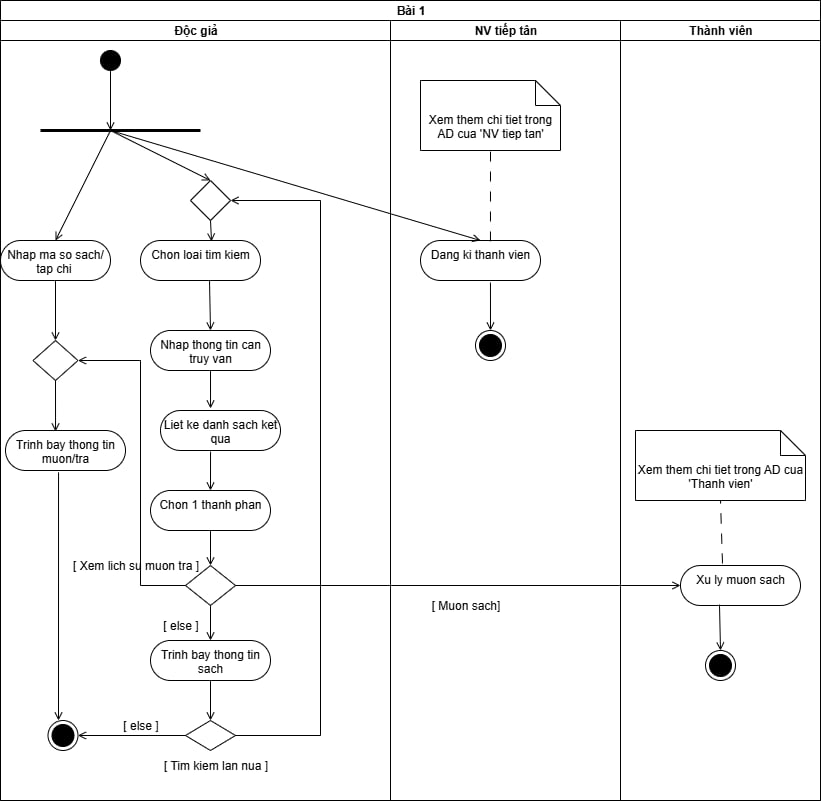
8.2. Các tầng trong kiến trúc 36

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. 37

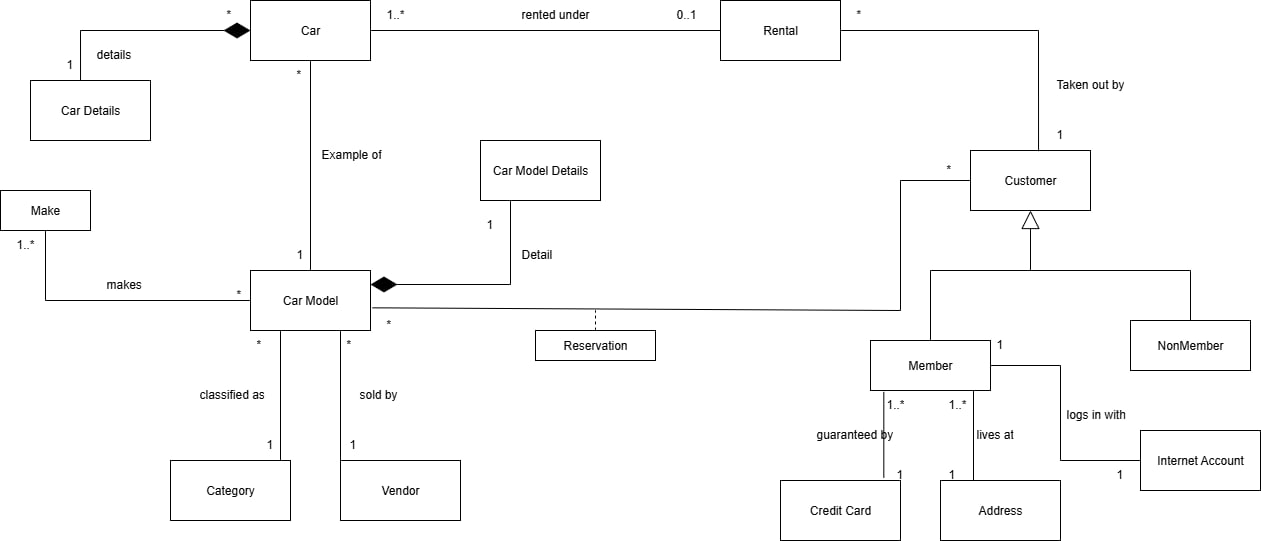
9.1. Nguyên tắc xây dựng ca kiểm thử 37

9.2. Danh sách ca kiểm thử 37

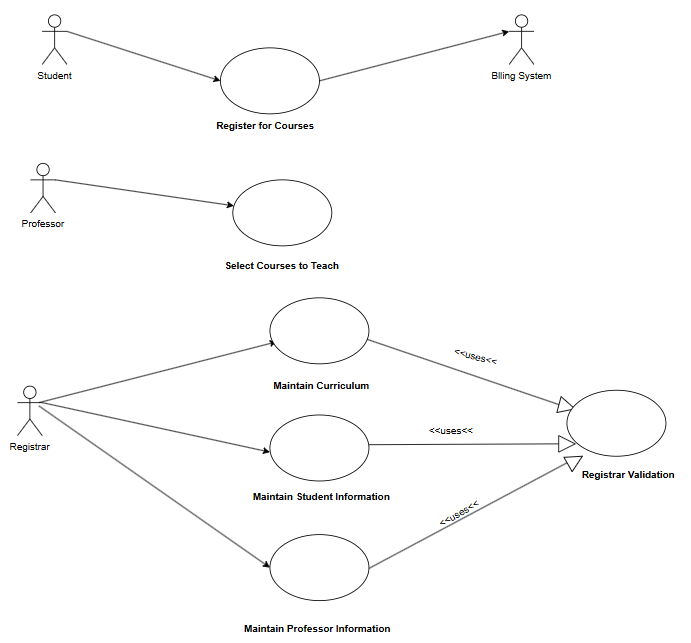
# **Bài 1: Vẽ lại qui trình nghiệp vụ sau (Độc giả, NV Tiếp tân, Thành viên)**

****

# **Bài 2: Vẽ mô hình khái niệm**

****

# **Bài 3: Vẽ lại business use case**

****

1. **Các Actor xuất hiện trong sơ đồ gồm**

**Student :** Học sinh

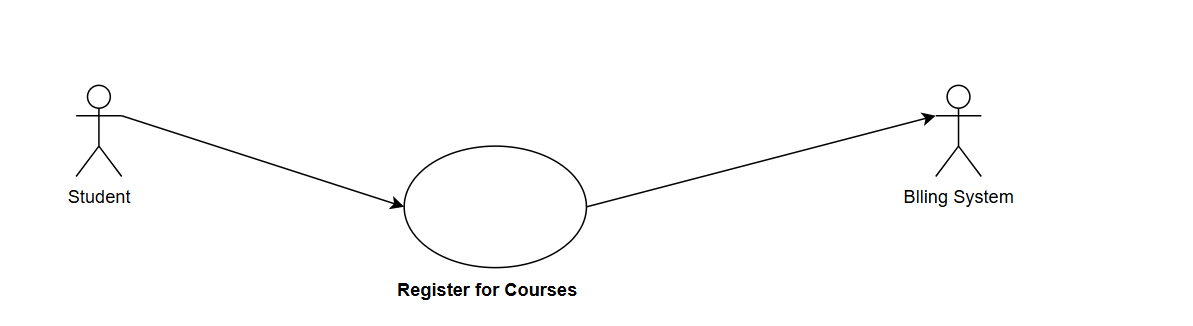
**Professor :** Giảng viên

**Registrar** : Cán bộ đào tạo (người chịu trách nhiệm về việc quản lí học phần )

**Billing System** : Hệ thống thanh toán ( tương tác từ bên ngoài nên nó là actor)

1. **Liệt kê và giải thích các trường hợp sử dụng use case trong sơ đồ**

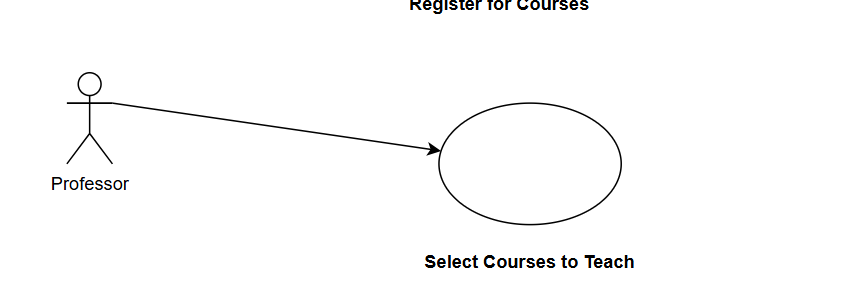
**Trường hợp 1 : Học sinh đăng ký khóa học ( Register for Courses )**



Use case : Đăng ký học phần

Học sinh thực hiện đăng ký học phần

**Trường hợp 2 : Giảng viên lựa chọn học phần ( Select Courses to Teach )**

****

Use case : Lựa chọn học phần

Giảng viên thực hiện lựa chọn học phần để dạy



**Trường hợp 3 : Cán bộ đào tạo cập nhật chương trình đào tạo ( Maintain Curriculum)**

Use case : Quản lí chương trình đào tạo

Cán bộ đào tạo tiến hành xem sửa xóa thêm đối với chương trình đào tạo

**Trường hợp 4 : Cán bộ đào tạo cập nhật thông tin sinh viên (Maintain Student Information)**

Use case : Quản lí sinh viên

Cán bộ đào tạo tiến thành xem sửa xóa thêm đối với thông tin sinh viên

**Trường hợp 5 : Cán bộ đào tạo cập nhật thông tin giảng viên (Maintain Professor Information)**

Use case : Quản lí giảng viên

Cán bộ đào tạo tiến hành xem sửa xóa thêm với thông tin giảng viên

**Trường hợp 6 : Xác nhận thông tin (Registrar Validation)**

Use case : Xác nhận thông tin   
Thực hiện kiểm tra, xác thực thông tin khi thêm/sửa dữ liệu.

1. **Phân tích các quan hệ trong sơ đồ**

**Register for Courses**

**Quan hệ**: Liên kết trực tiếp với **Student** (actor chính).

**Quan hệ**: Liên kết trực tiếp với **Billing System** để xử lý học phí khi sinh viên đăng ký.

**Ghi chú**: Đây là mối quan hệ **association** (giao tiếp trực tiếp).

**Maintain Curriculum**

**Quan hệ**: Liên kết với **Registrar** (cán bộ đào tạo) để cập nhật chương trình đào tạo.

**Quan hệ**: Có thể có liên kết với **Professor** (giảng viên) trong trường hợp giảng viên tham gia đóng góp/chỉnh sửa chương trình đào tạo.

**Ghi chú**: Đây là mối quan hệ trực tiếp giữa actor và use case.

**Maintain Student Information**

**Quan hệ**: Liên kết trực tiếp với **Registrar** (cán bộ đào tạo).

**Quan hệ**: «uses» (include) với **Registrar Validation** để xác thực thông tin sinh viên.

**Maintain Professor Information**

**Quan hệ**: Liên kết trực tiếp với **Registrar** (cán bộ đào tạo).

**Quan hệ**: «uses» (include) với **Registrar Validation** để xác thực thông tin giảng viên.

**Select Courses to Teach**

**Quan hệ**: Liên kết trực tiếp với **Professor** (giảng viên) để lựa chọn môn dạy.

**Registrar Validation**

**Quan hệ**: Được **«uses» (include)** trong 2 use case:

* + Maintain Student Information
  + Maintain Professor Information

**Ghi chú**: Đây là use case hỗ trợ, không phải actor trực tiếp gọi.

1. **Viết kịch bản cho trường hợp cụ thể**

Focus use case cho Register for Courses

**Actor** : Student

**Description:** Đăng ký thành công học phần cho kỳ học.

**Pre-condition:** Sinh viên đã đăng nhập vào hệ thống, có tài khoản hợp lệ.

**Luồng sự kiện chính**:

1. Sinh viên chọn chức năng “Register for Courses”.
2. Hệ thống hiển thị danh sách môn học mở trong kỳ.
3. Sinh viên chọn môn học mong muốn và xác nhận.
4. Hệ thống kiểm tra điều kiện tiên quyết (prerequisite, số tín chỉ còn lại, số lượng chỗ trống).
5. Nếu hợp lệ, hệ thống kết nối với **Billing System** để tính học phí.
6. Sau khi thanh toán thành công, hệ thống cập nhật lịch học của sinh viên.

**Post-condition**: Sinh viên được ghi danh vào môn học, thông tin cập nhật trong hệ thống và gửi thông báo xác nhận.

**Exception**:

* Nếu môn học đã đầy, hệ thống thông báo lỗi.
* Nếu thanh toán thất bại, hệ thống không ghi nhận đăng ký.

1. **Đề xuất cải tiến hoặc bổ sung**

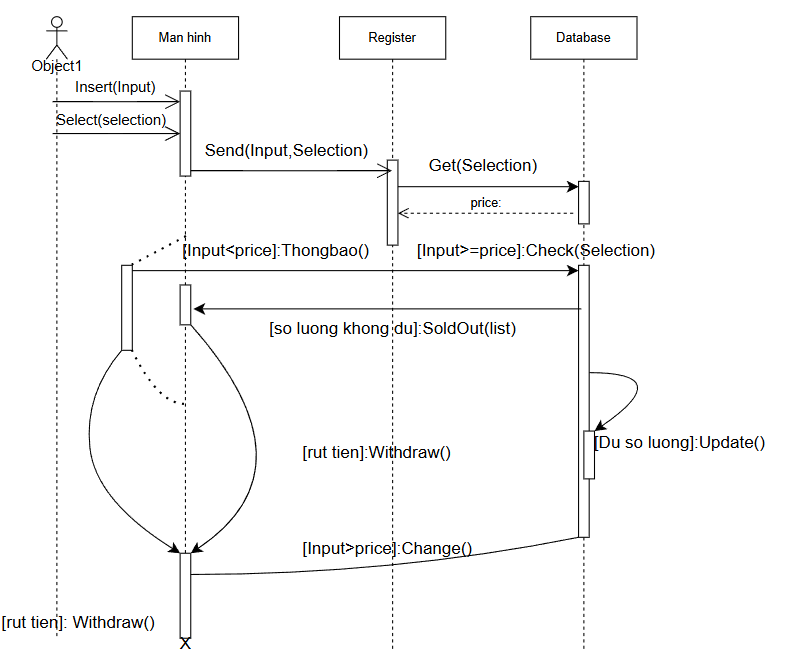
Thêm use case View Course Schedule: Sinh viên có thể xem thời khóa biểu sau khi đăng ký.

Thêm use case Generate Reports: Quản trị viên có thể xuất báo cáo về số lượng sinh viên đăng ký, số môn học mở.

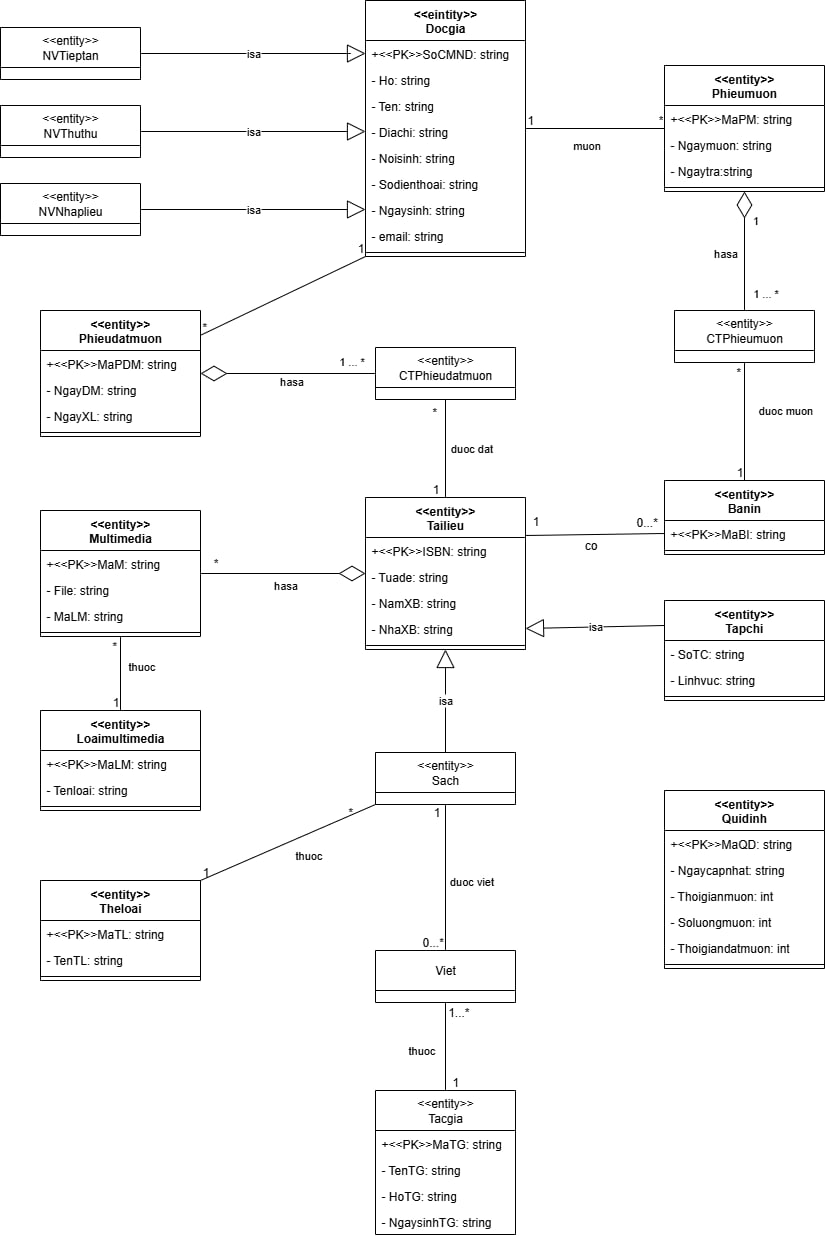
Thêm use case Grade Submission: Giảng viên nhập điểm cho sinh viên.

Thêm chức năng Notifications/Alerts: Gửi thông báo khi có thay đổi trong chương trình học hoặc học phí.

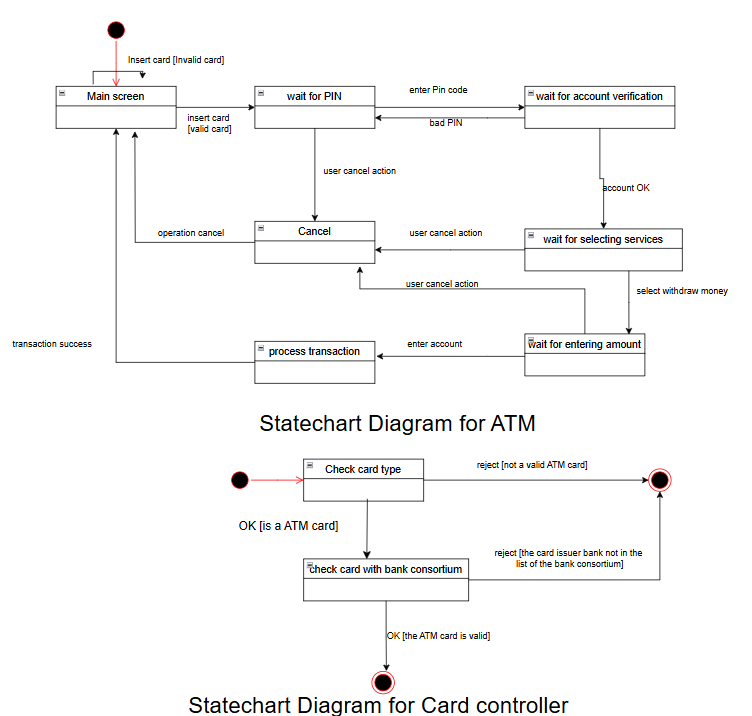
# **Bài 4****: Vẽ sơ đồ hoạt động và tương tác sau và tóm tắt ý nghĩa sơ đồ**

****

# **Bài 5: Vẽ lược đồ lớp cho bài toán Quản lý thư viện**

****

# **Bài 6: Vẽ lại lược đồ**

****

# **Bài 7: Phân tích dữ liệu cho ABC Bank**

ABC Bank has many branches. Each branch has a unique name, an address (including No., Street, District, City, and Region), phone numbers, fax numbers, email, and total number of employees working there. Each branch has many employees and an employee must work at a branch. There is always one employee assigned to manage that branch. The employee can only manage the office to which he or she is assigned. For each employee, we need to store a unique code, first name, last name, date of birth, phone numbers, email, and home address (including No., Street, District, and City). The bank issues three different types of bank accounts for their customers. For Savings Accounts, the bank must keep track of the account’s balance, interest rate, and the date the account was opened. Checking Accounts pay no interest, so the bank keeps track of just the balance and date opened. The third type of account, Loans, requires tracking the date the loan was taken, the balance due, and the interest rate of the loan. Each customer may have any number of bank accounts, and all accounts have a unique account number. The bank save the following information of each customer: a unique code, first name, last name, home address, office address, phone numbers, email. Each customer is attended by an employee and an employee can serve many customers.

BRANCH (Name, AddressNo, Street, District, City, Region, Email, MngCode)

BRANCHPHONE (BraName, Phone)

BRANCHFAX (BraName, Fax)

EMPLOYEE (Code, FName, LName, AddressNo, Street, District, City, DOB, Email, BraName)

EMPLOYEEPHONE (EmpCode, Phone)

CUSTOMER (Code, FName, LName, OfficeAddress, HomeAddress, Email, EmpCode)

CUSTOMERPHONE (CusCode, Phone)

ACCOUNT (Number, CusCode)

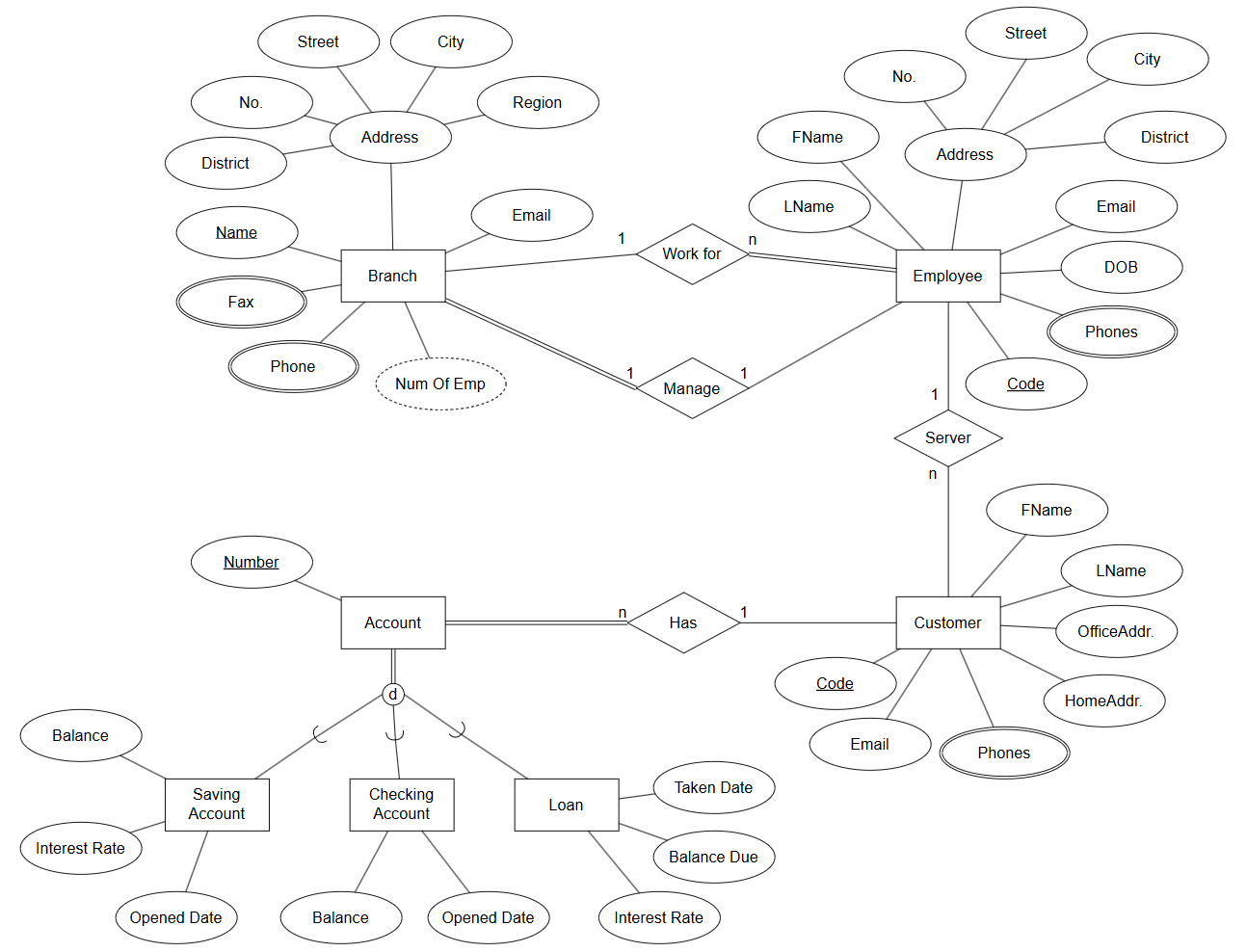
SAVINGACCOUNT (AccNumber, Balance, InterestRate, OpenedDate)

CHECKINGACCOUNT (AccNumber, Balance, OpenedDate)

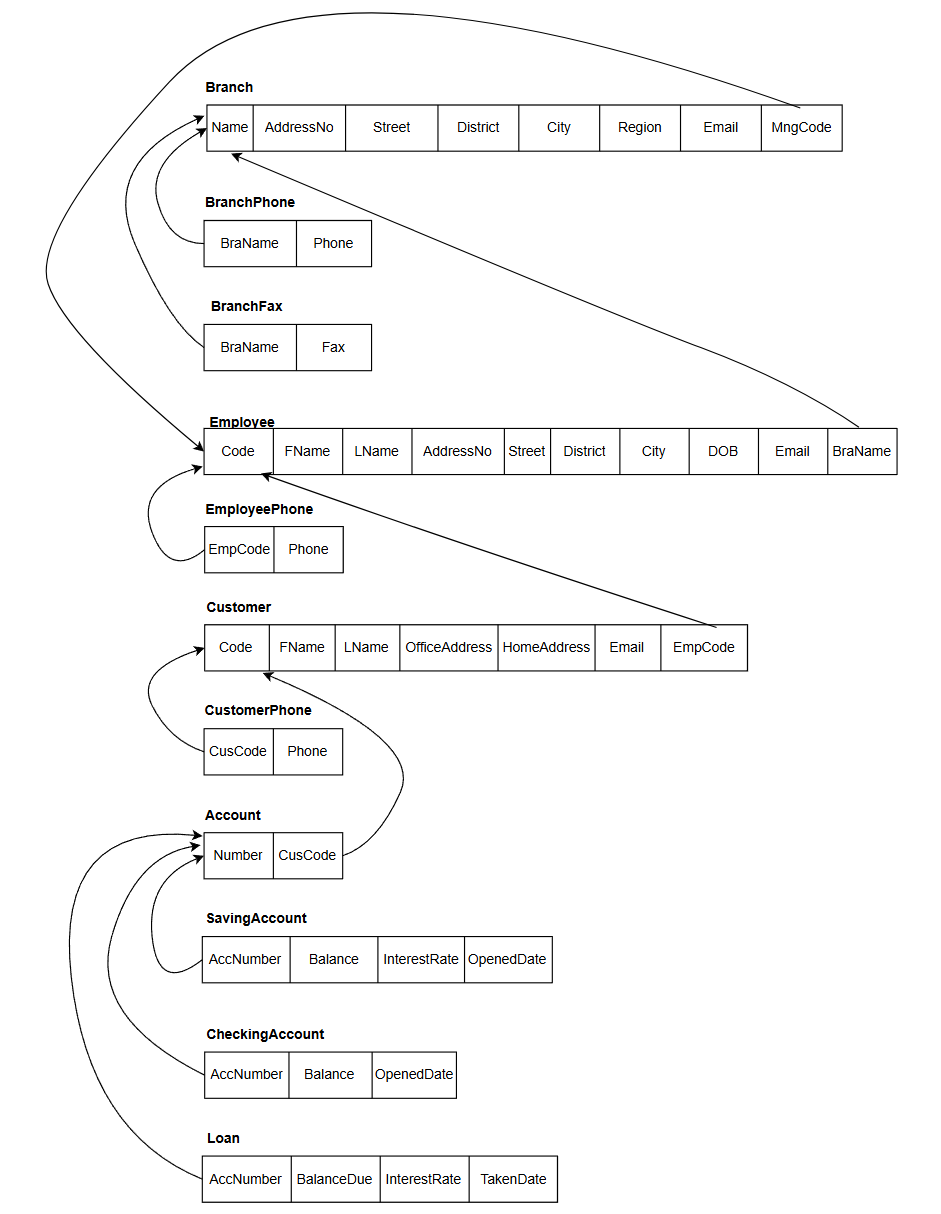
LOAN (AccNumber, BalanceDue, InterestRate, TakenDate)

## **Thiết kế Cơ sở dữ liệu ở mức Khái niệm, Luận lý và Vật lý**

### **Mức khái niệm (Conceptual Model)**

****

### **Mức luận lý (Logical Model)**



### **Mức vật lý (Physical Model)**

1. **Query: Insert, Update, Delete, Select**
   1. **You are required to insert valid and meaningful data into the database. Each table has at least 4 rows.**

-- BRANCH

INSERT INTO BRANCH (Name, AddressNo, Street, District, City, Region, Email, MngCode) VALUES

('HCMC', '12', 'Le Loi', '1', 'Ho Chi Minh', 'South', 'hcmc@bank.com', NULL),

('Hanoi', '88', 'Tran Hung', 'Hoan Kiem', 'Hanoi', 'North','hanoi@bank.com', NULL),

('Danang','25', 'Vo Nguyen', 'Son Tra', 'Danang','Central','danang@bank.com', NULL),

('CanTho','40', 'Nguyen Trai','Ninh Kieu', 'CanTho','South','cantho@bank.com', NULL);

-- BRANCHPHONE

INSERT INTO BRANCHPHONE (BraName, Phone) VALUES

('HCMC','028-1111111'),('Hanoi','024-2222222'),('Danang','0236-3333333'),('CanTho','0292-4444444');

-- BRANCHFAX

INSERT INTO BRANCHFAX (BraName, Fax) VALUES

('HCMC','028-1111112'),('Hanoi','024-2222223'),('Danang','0236-3333334'),('CanTho','0292-4444445');

-- EMPLOYEE

INSERT INTO EMPLOYEE (Code, FName, LName, AddressNo, Street, District, City, DOB, Email, BraName) VALUES

('N1080','John','Smith', '10','Pasteur','1','Ho Chi Minh','2010-03-15','john.smith@bank.com','HCMC'),

('N1081','Mary','Brown', '22','Ba Trieu','Hoan Kiem','Hanoi','1989-06-20','mary.brown@bank.com','Hanoi'),

('N1082','Peter','Nguyen','35','Pham Van Dong','Son Tra','Danang','2010-03-05','peter.nguyen@bank.com','Danang'),

('N1083','Alice','Tran', '47','3-2','Ninh Kieu','CanTho','1992-11-02','alice.tran@bank.com','CanTho');

-- Gán Manager cho từng chi nhánh (sau khi có EMPLOYEE)

UPDATE BRANCH SET MngCode = 'N1081' WHERE Name='Hanoi';

UPDATE BRANCH SET MngCode = 'N1080' WHERE Name='HCMC';

UPDATE BRANCH SET MngCode = 'N1082' WHERE Name='Danang';

UPDATE BRANCH SET MngCode = 'N1083' WHERE Name='CanTho';

-- EMPLOYEEPONE

INSERT INTO EMPLOYEEPHONE (EmpCode, Phone) VALUES

('N1080','0901-000001'),('N1081','0902-000002'),('N1082','0903-000003'),('N1083','0904-000004');

-- CUSTOMER

INSERT INTO CUSTOMER (Code, FName, LName, OfficeAddress, HomeAddress, Email, EmpCode) VALUES

('C001','Peter','Johnson','01 Office St','01 Home St','peter.j@cus.com','N1080'),

('C002','Linda','Pham', '02 Office St','02 Home St','linda.p@cus.com','N1081'),

('C003','David','Lee', '03 Office St','03 Home St','david.l@cus.com','N1082'),

('C004','Sara','Nguyen', '04 Office St','04 Home St','sara.n@cus.com','N1083');

-- CUSTOMERPHONE

INSERT INTO CUSTOMERPHONE (CusCode, Phone) VALUES

('C001','0911-111111'),('C002','0912-222222'),('C003','0913-333333'),('C004','0914-444444');

-- ACCOUNT (tạo 12 tài khoản để đủ cho Saving/Checking/Loan)

INSERT INTO ACCOUNT (Number, CusCode) VALUES

('A1001','C001'),('A1002','C002'),('A1003','C001'),('A1004','C003'),

('A2001','C001'),('A2002','C002'),('A2003','C003'),('A2004','C004'),

('A3001','C001'),('A3002','C002'),('A3003','C003'),('A3004','C004');

-- SAVINGACCOUNT (≥ 4)

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

('A1001', 5000, 0.030, '2020-01-10'),

('A1003', 1500, 0.028, '2021-06-05'),

('A2001', 7000, 0.031, '2022-03-12'),

('A2004', 2200, 0.029, '2019-09-01');

-- CHECKINGACCOUNT (≥ 4)

INSERT INTO CHECKINGACCOUNT (AccNumber, Balance, OpenedDate) VALUES

('A2002', 1200, '2019-02-02'),

('A2003', 800, '2020-07-07'),

('A1002', 2500, '2021-08-08'),

('A3001', 600, '2023-03-03');

-- LOAN (≥ 4)

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

('A3001', 10000, 0.095, '2022-10-01'),

('A3002', 8000, 0.090, '2021-05-15'),

('A3003', 15000, 0.100, '2023-01-20'),

('A3004', 5000, 0.085, '2020-12-11');

* 1. **Change last name of an employee whose code is “N1080” to “Brown”.**

UPDATE EMPLOYEE

SET LName = 'Brown'

WHERE Code = 'N1080';

* 1. **Delete the employee having code “N1080”. Explain what will happen to the customers who are severed by this employee.**

DELETE FROM EMPLOYEE WHERE Code = 'N1080';

→ Nếu CUSTOMER.EmpCode có khóa ngoại RESTRICT (mặc định), lệnh sẽ báo lỗi do khách hàng C001 đang gán EmpCode='N1080'.

* 1. **Retrieve all account numbers of a customer whose name is Peter Johnson.**

SELECT a.Number

FROM CUSTOMER c

JOIN ACCOUNT a ON a.CusCode = c.Code

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

* 1. **Find all employees who have their birthdays in March 2010.**

SELECT \*

FROM EMPLOYEE

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

* 1. **Retrieve the total balance of all saving accounts owned by a customer named “Peter Johnson” (Supposing that he has many saving accounts)**

SELECT SUM(s.Balance) AS TotalBalance

FROM SAVINGACCOUNT s

JOIN ACCOUNT a ON a.Number = s.AccNumber

JOIN CUSTOMER c ON c.Code = a.CusCode

WHERE c.FName = ‘Peter’

AND c.LName = ‘Johnson’;

# **BÀI TẬP ỨNG DỤNG**

Tham khảo: https://vietnam-devs.github.io/coolstore-microservices

Website CoolStore có kịch bản kinh doanh cơ bản cho **Danh mục sản phẩm (Product Catalog), Giỏ hàng (Shopping Cart), Quy trình thanh toán (Payment Process), Tồn kho (Inventory), Đánh giá (Rating)** và **Kiểm soát truy cập (Access Control)**.

Với **Danh mục sản phẩm**, người mua (Buyer) có thể duyệt danh sách sản phẩm với các chức năng lọc và sắp xếp theo tên sản phẩm và giá. Người mua có thể 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)** trong hệ thống có thể **quản lý sản phẩm** và có quyền **gán sản phẩm vào kho hiện có**.

Với **Giỏ hàng**, người mua có thể **mua bất kỳ sản phẩm nào** trên danh sách sản phẩm thông qua nút "Mua hàng" trên mỗi sản phẩm. Ngoài ra, người mua cũng có thể mua sản phẩm tại **trang chi tiết sản phẩm**. Sau khi mua sản phẩm, người mua sẽ thấy các sản phẩm này trong giỏ hàng và **bảng tổng hợp** với thông tin cơ bản như tổng chi phí giỏ hàng, tổng chi phí sau khuyến mãi, chi phí vận chuyển, khuyến mãi tiết kiệm, tổng giá trị đơn hàng. Bất cứ khi nào người mua **thêm sản phẩm** hoặc **xóa sản phẩm** khỏi giỏ hàng, thì **bảng tổng hợp và giỏ hàng sẽ được cập nhật**. Sau đó, người mua có thể thực hiện **quy trình thanh toán** bằng cách nhấn nút Thanh toán trên trang giỏ hàng. **Quản trị viên có thể xem toàn bộ giỏ hàng của bất kỳ người dùng nào** để có thể **kích hoạt hoặc vô hiệu hóa** bất kỳ giỏ hàng không hợp lệ nào trên website CoolStore.

Với **Quy trình thanh toán**, sau khi người mua nhấn nút thanh toán, hệ thống sẽ bắt đầu **xác thực thông tin sản phẩm**, **xử lý thanh toán**, và sau đó **gửi email cho người mua** để người mua biết chuyện gì đang xảy ra.

Với **Tồn kho**, **Quản trị viên** có thể **quản lý kho hàng**.

Với **Đánh giá**, người mua có thể **đánh giá bất kỳ sản phẩm nào** mà cô ấy thấy tốt (thang điểm 1 -> 5 sao).

Với **Kiểm soát truy cập**, người mua hoặc quản trị viên có thể **đăng nhập/đăng xuất hệ thống**. Nếu là người mua, người đó sẽ được đưa đến **trang danh mục sản phẩm**, còn nếu là quản trị viên, sẽ được đưa đến **trang quản trị**.

Một số **nhiệm vụ thiết lập ban đầu** cần được thực hiện khi website CoolStore khởi động như: tạo **người dùng quản trị**, **hai người dùng mua hàng**, và **dữ liệu mẫu** cho **sản phẩm, kho, đánh giá** cho một vài sản phẩm.

**Câu hỏi**

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 (Product Catalog):

* + Người mua (Buyer) có thể duyệt danh sách sản phẩm với các chức năng lọc và sắp xếp theo tên sản phẩm và giá.
  + Xem chi tiết sản phẩm, bao gồm tên, mô tả, số lượng tồn kho, thông tin kho hàng, trạng thái "sản phẩm nổi bật", và đánh giá sản phẩm.
  + Quản trị viên (SysAdmin) có thể quản lý sản phẩm và gán sản phẩm vào kho.

Giỏ hàng (Shopping Cart):

* + Người mua có thể mua sản phẩm từ danh sách hoặc trang chi tiết sản phẩm qua nút "Mua hàng".
  + Hiển thị giỏ hàng và bảng tổng hợp gồm tổng chi phí, chi phí sau khuyến mãi, phí vận chuyển, và tổng giá trị đơn hàng.
  + Giỏ hàng và bảng tổng hợp cập nhật liên tục khi người mua thêm hoặc xóa sản phẩm.
  + Cho phép người mua thực hiện quy trình thanh toán (checkout).
  + Quản trị viên có thể xem giỏ hàng của bất kỳ người dùng nào và kích hoạt/vô hiệu hóa giỏ hàng không hợp lệ.

Quy trình thanh toán (Payment Process):

* + Xác thực thông tin sản phẩm.
  + Xử lý thanh toán.
  + Gửi email thông báo tình trạng tới người mua.

Tồn kho (Inventory):

* + Quản trị viên quản lý kho hàng.

Đánh giá (Rating):

* + Người mua có thể đánh giá sản phẩm với thang điểm 1 đến 5 sao.

Kiểm soát truy cập (Access Control):

* + Người mua và quản trị viên có thể đăng nhập/đăng xuất.
  + Người mua được điều hướng về trang danh mục sản phẩm sau khi đăng nhập.
  + Quản trị viên được điều hướng về trang quản trị.

Các nhiệm vụ khởi tạo:

* + Tạo người dùng quản trị viên, hai người mua hàng, và dữ liệu mẫu cho sản phẩm, kho và đánh giá.

Yêu cầu phi chức năng:

Hệ thống phải hỗ trợ đa dịch vụ microservices với khả năng mở rộng và bảo trì dễ dàng.

Đảm bảo bảo mật với kiểm soát truy cập theo vai trò.

Hệ thống phải cung cấp trải nghiệm người dùng mượt mà, cập nhật giỏ hàng và tổng hợp tức thì.

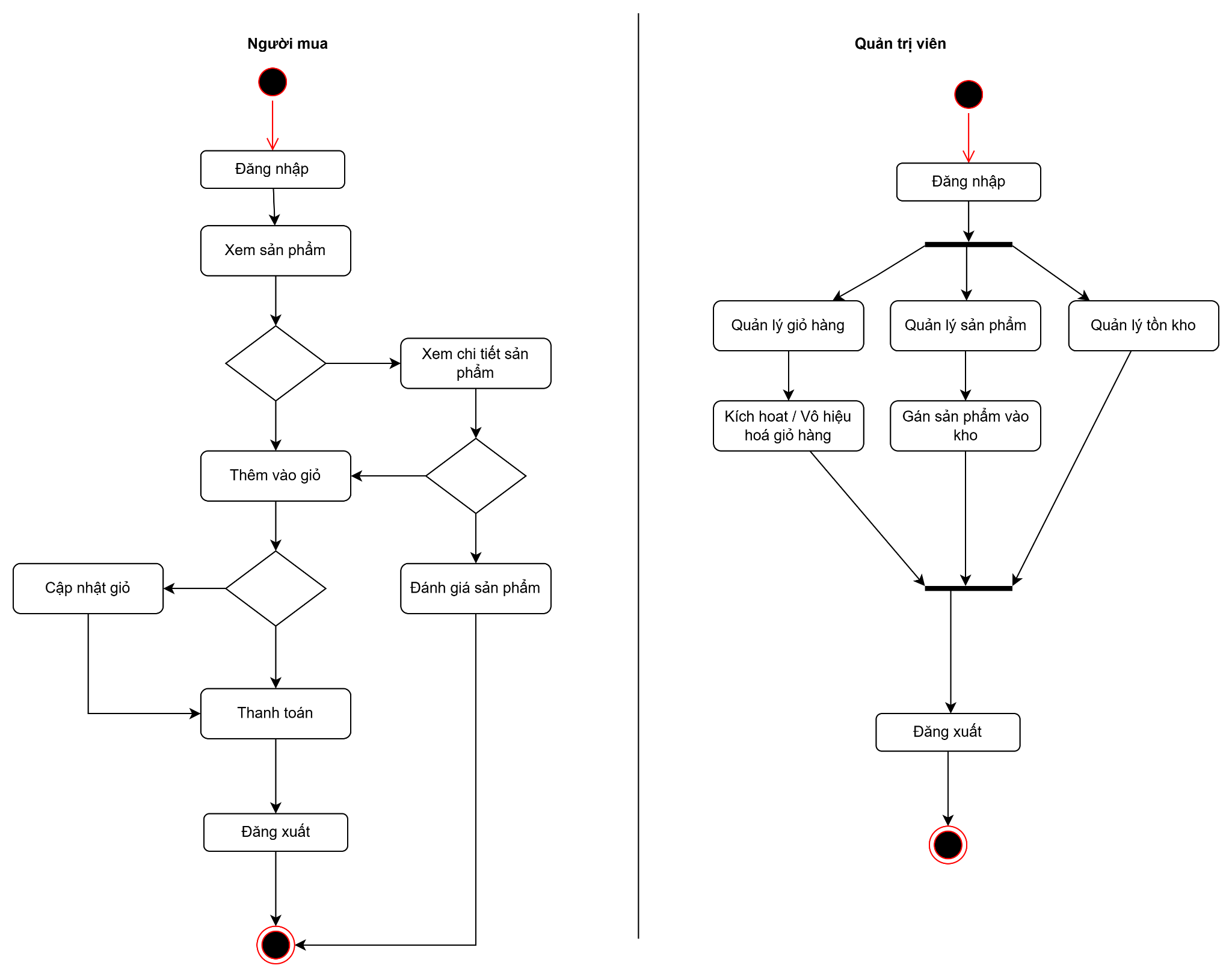
Giao tiếp giữa các dịch vụ phải ổn định và tin cậy.

Hỗ trợ thông báo qua email.

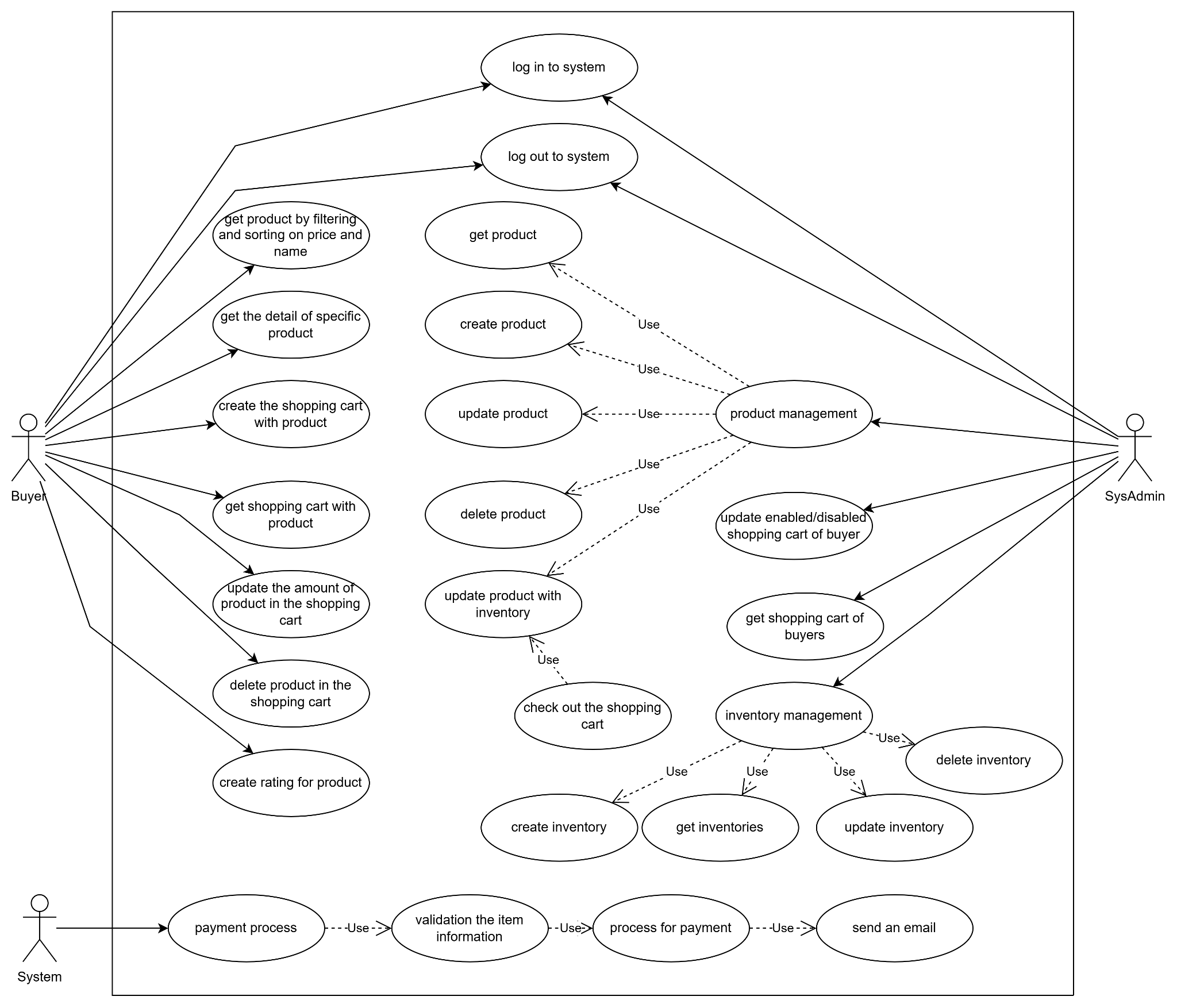
Hệ thống hỗ trợ giao diện dễ sử dụng cho cả người mua và quản trị viên.

Có khả năng khởi tạo dữ liệu mẫu ban đầu để chạy hệ thống thuận tiện.

1. **Vẽ qui trình nghiệp vụ cho website CoolStore**



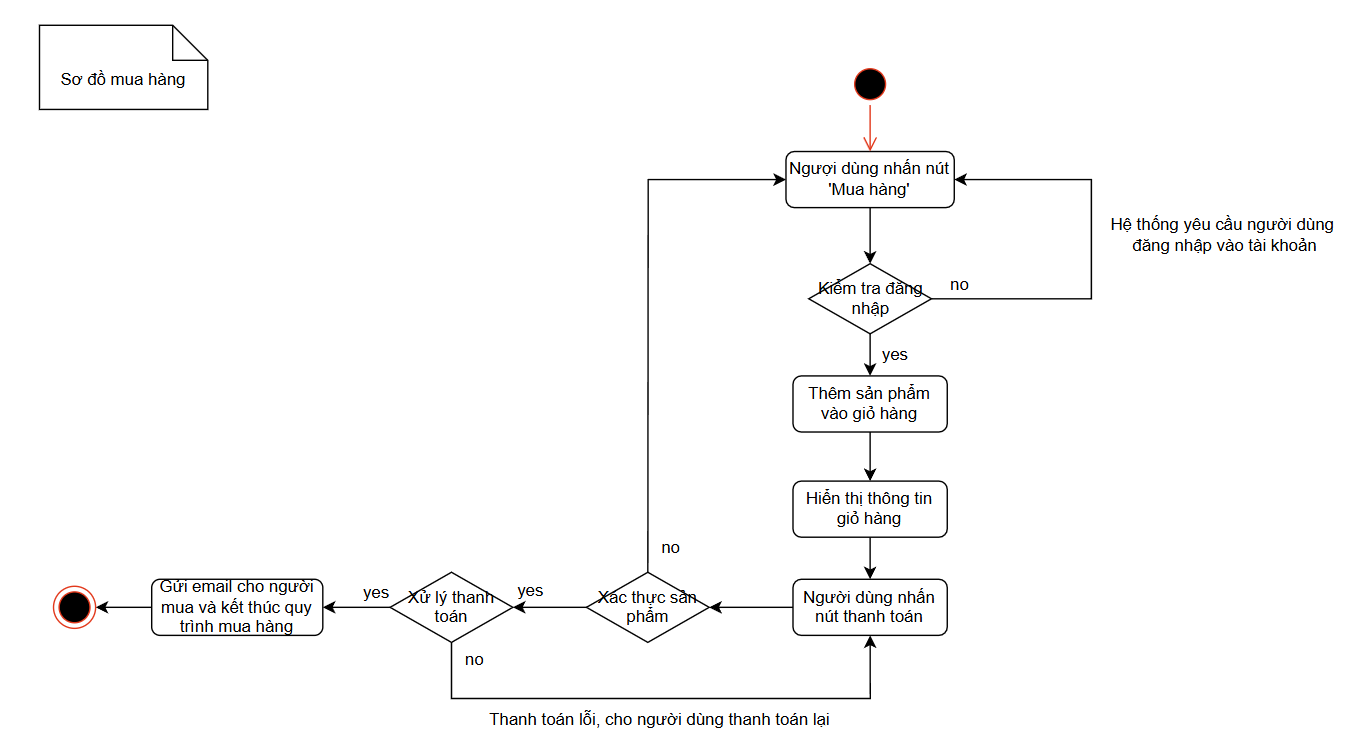
1. **Xây dựng use case diagram cho website CoolStore.**



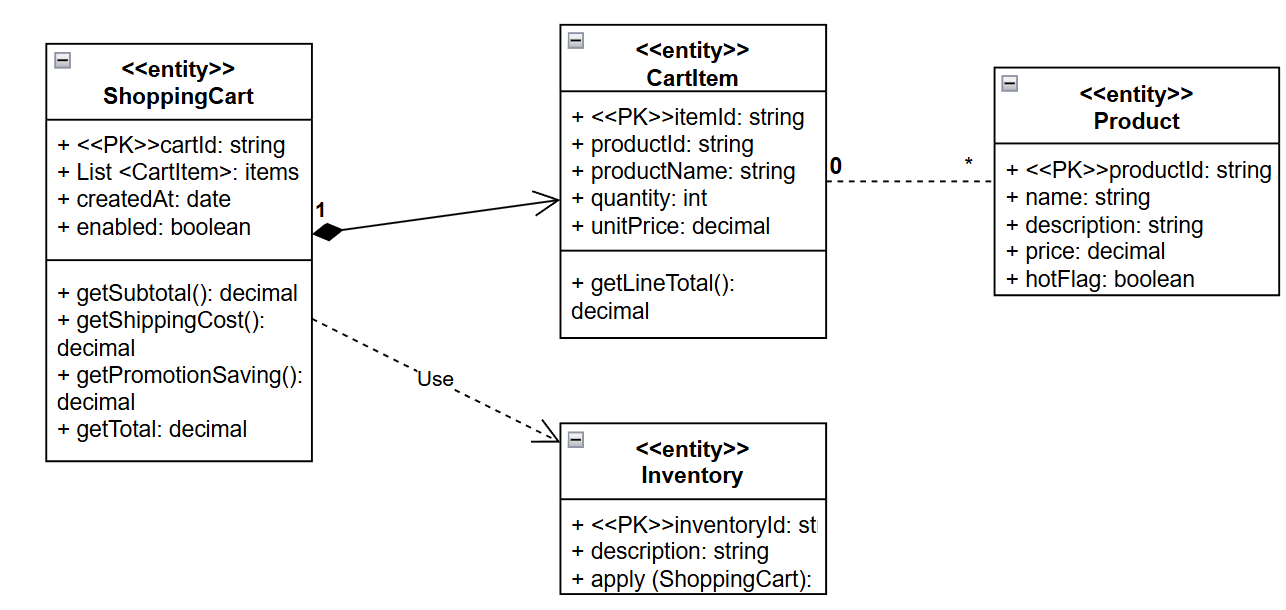
1. **Mô tả chi tiết một Use Case "Mua sản phẩm từ trang chi tiết sản phẩm". Trình bày theo mẫu chuẩn gồm: Tên use case, Tác nhân chính, Mục tiêu, Tiền điều kiện, Luồng chính, Luồng thay thế, Hậu điều kiện, Ghi chú (nếu có).**

|  |  |  |
| --- | --- | --- |
|  | | |
| **Use Case Number:** | UC | |
| **Use Case Name:** | Mua sản phẩm từ trang chi tiết sản phẩm | |
| **Actor (s):** | Buyer | |
| **Maturity:** | Focused | |
| **Summary:** | Nguời mua có thể mua sản phẩm trên trang chi tiết của sản phẩm đó, sau đó chuyển sang thanh toán. | |
| **Basic Course of Events:**  (luồng nghiệp vụ chính) | **Actor Action** | **System Response** |
| 1. Thực hiện đăng nhập |  |
|  | 2. Hệ thống hiển thị trang chủ với các chức năng chính |
| 3. Use case bắt đầu khi Buyer chọn vào 1 sản phẩm cụ thể. |  |
|  | 4. Hệ thống hiển thị thông tin chi tiết của sản phẩm đó, gồm: tên, mô tả, số lượng tồn kho, địa chỉ kho, trạng thái (nếu có) và đánh giá. |
| 5. Buyer chọn số lượng sản phẩm và bấm nút “Mua hàng” trên sản phẩm.  **A1** |  |
|  | 6. Hệ thống xử lý đơn hàng và cập nhật sản phẩm vào giỏ hàng và bảng tổng hợp. E1  **{Kiểm tra đơn hàng hợp lệ}** |
| 7. Use case kết thúc khi Buyer nhấn vào nút “Thanh toán” trên trang Gio hàng. |  |
| **Alternative Paths:**  (luồng phụ) | **A1**. | |
| **Actor Action** | **System Response** |
| 1. Buyer thêm sản phẩm mới vào giỏ hoặc xoá sản phẩm trong giỏ. |  |
|  | 2. Hệ thống cập nhật lại giỏ hàng và bảng tổng hợp. |
|  | Quay lại bước 6 của luồng nghiệp vụ chính |
| **Exception Paths:** | E1. {Kiểm tra đơn hàng hợp lệ} nếu có lỗi (số lượng không hợp lệ, lớn hơn tồn kho), hệ thống sẽ chỉ ra lỗi và thông báo lên màn hình.  Quay trở về luồng nghiệp vụ chính bước 5. | |
| **Extension Points:** | Kiểm tra đơn hàng hợp lệ  Kiểm tra đơn hàng thông qua việc số lượng đơn hàng phải phù hợp, không âm, là số nguyên và phải nhỏ hơn số lượng của hàng tồn kho | |
| **Triggers:** | None | |
| **Assumptions:** | None | |
| **Preconditions:** | None. | |
| **Post Conditions:** | Giỏ hàng và bảng tổng hợp hiển thị tổng chi phí của toàn bộ sản phẩm. | |

1. **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.**



1. **Thiết kế sơ đồ Class Diagram cho module Giỏ hàng.**



1. **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:**

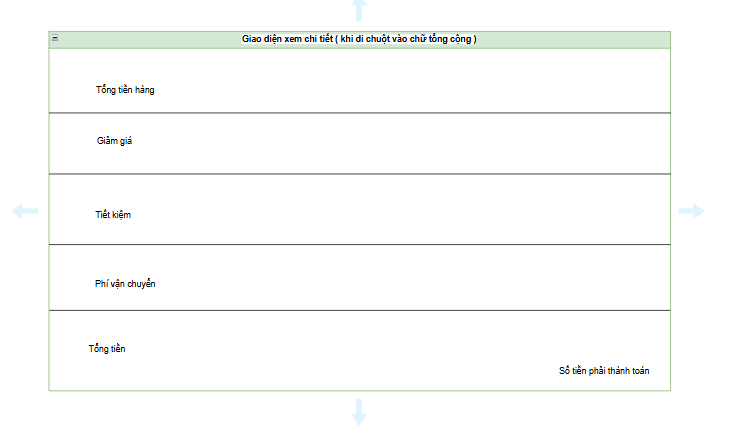
Danh sách sản phẩm trong giỏ hàng: tên sản phẩm, hình ảnh, số lượng, giá từng món, tổng phụ.

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).

****

****

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

Mô hình kiến trúc ba lớp (3-tier architecture) phân chia hệ thống thành ba tầng chính: **Presentation layer**, **Application/Business Logic layer**, và **Data layer**. Cách tổ chức này giúp hệ thống dễ bảo trì, mở rộng, tái sử dụng và tăng tính bảo mật.

* 1. **Các tầng trong kiến trúc**

**Tầng Presentation (Giao diện người dùng)**

Cung cấp giao diện web cho Người mua (Buyer) và Quản trị viên (SysAdmin).  
Chức năng chính: duyệt sản phẩm, xem chi tiết, quản lý giỏ hàng, đánh giá, thanh toán, đăng nhập/đăng xuất.  
Admin Portal: quản lý sản phẩm, kho, và giỏ hàng không hợp lệ.

**Tầng Application/Business Logic (Xử lý nghiệp vụ)**

Đóng vai trò trung gian giữa giao diện và cơ sở dữ liệu.

Các module/dịch vụ chính:

1. **Catalog Service**: quản lý sản phẩm, danh mục, lọc, sắp xếp.
2. **Cart Service**: thêm/xoá/cập nhật sản phẩm trong giỏ hàng, tính tổng chi phí và khuyến mãi.
3. **Checkout/Order Service**: xác thực thông tin, xử lý đơn hàng, thanh toán.
4. **Inventory Service**: quản lý kho hàng và tồn kho.
5. **Rating Service**: quản lý đánh giá sản phẩm.
6. **Identity & Access Control**: quản lý đăng nhập, phân quyền Buyer/Admin.
7. **Notification Service**: gửi email xác nhận thanh toán, thông báo hệ thống.

**Tầng Data (Dữ liệu)**

**Cơ sở dữ liệu quan hệ/phi quan hệ**: quản lý dữ liệu sản phẩm, kho, giỏ hàng, đơn hàng, người dùng và đánh giá.

**Cache** (ví dụ Redis): lưu giỏ hàng tạm thời, token đăng nhập.

**Search Engine** (ví dụ Elasticsearch): hỗ trợ tìm kiếm/lọc sản phẩm.

**File Storage**: lưu trữ hình ảnh sản phẩm và hóa đơn.

1. **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.** 
   1. **Nguyên tắc xây dựng ca kiểm thử**

Mỗi ca kiểm thử bao gồm: **Mục tiêu, Dữ liệu đầu vào, Các bước thực hiện, Kết quả mong đợi**.

Các ca kiểm thử đảm bảo bao phủ cả trường hợp hợp lệ (happy path) và bất hợp lệ (negative cases).

* 1. **Danh sách ca kiểm thử**

Test Case 1: Thêm sản phẩm hợp lệ

**Mục tiêu**: Kiểm tra chức năng thêm sản phẩm còn hàng vào giỏ.

**Dữ liệu đầu vào**: userId=U1, productId=P001, qty=2 (tồn kho 10).  
**Bước thực hiện**:

* Người mua mở chi tiết sản phẩm P001.
* Nhấn nút “Mua hàng” với số lượng 2.

**Kết quả mong đợi**:

* Giỏ hàng hiển thị sản phẩm P001 với số lượng 2.
* Tổng tiền được cập nhật chính xác (2 × giá sản phẩm).
* Hệ thống trả về trạng thái **200 OK**.

Test Case 2: Thêm sản phẩm vượt quá số lượng tồn

**Mục tiêu**: Kiểm tra hệ thống xử lý khi số lượng yêu cầu > số lượng tồn kho.

**Dữ liệu đầu vào**: userId=U1, productId=P003, qty=10 (tồn kho 5).

**Bước thực hiện**: Người mua nhấn nút “Mua hàng” với số lượng 10.  
**Kết quả mong đợi**:

* Hệ thống trả về thông báo lỗi: *“Số lượng vượt quá tồn kho khả dụng (5)”*.
* Giỏ hàng không thay đổi.
* HTTP status: **409 Conflict** hoặc **422 Unprocessable Entity**.

Test Case 3: Thêm sản phẩm đã hết hàng

**Mục tiêu**: Kiểm tra khi người mua chọn sản phẩm không còn tồn.

**Dữ liệu đầu vào**: userId=U1, productId=P002, qty=1 (tồn kho 0).

**Bước thực hiện**: Người mua nhấn nút “Mua hàng”

**Kết quả mong đợi**:

* + Hệ thống trả về thông báo: *“Sản phẩm tạm hết hàng”*.
  + Giỏ hàng không thay đổi.
  + HTTP status: **409/422**.