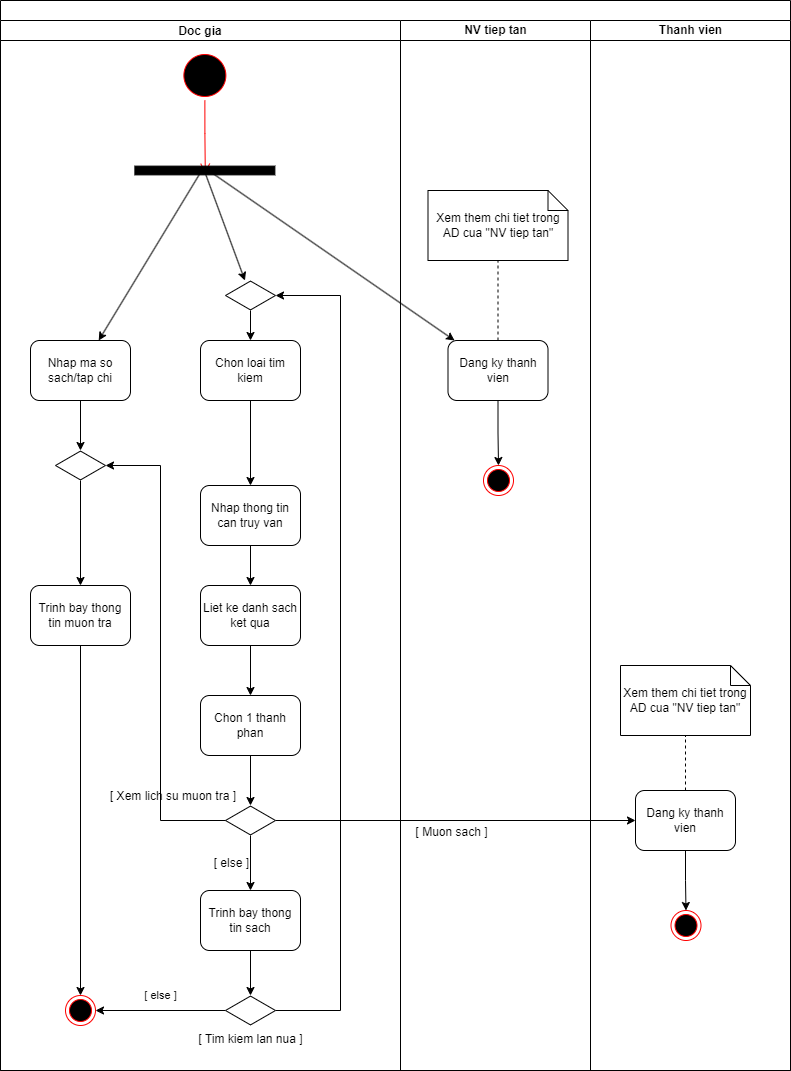
**Bài tập 1: Vẽ lại qui trình nghiệp vụ sau (độc giả, NV tiếp tân, Thành viên)**

****

**1. Tổng quát quy trình**

* Quy trình được chia thành 2 nhánh: tra cứu thông tin và mượn sách/đăng ký thành viên.
* Nhân viên tiếp tân đóng vai trò quan trọng trong việc xác thực thông tin và quản lý thành viên.
* Hệ thống hỗ trợ độc giả:
  + Tra cứu sách (không cần đăng nhập).
  + Mượn sách & xem lịch sử (chỉ dành cho thành viên).

**2. Các tác nhân tham gia (actor)**

* **Độc giả**: người sử dụng hệ thống để tìm kiếm sách/tạp chí, đăng ký thành viên, mượn sách, xem lịch sử mượn trả.
* **Nhân viên tiếp tân**: quản lý đăng ký thành viên, hỗ trợ độc giả trong việc mượn sách, lưu trữ thông tin.
* **Thành viên**: độc giả đã đăng ký và có quyền mượn sách.

**3. Quy trình nghiệp vụ chính**

**a) Tìm kiếm tài liệu**

1. **Nhập mã số sách/tạp chí** hoặc **chọn loại tìm kiếm**.
2. **Nhập thông tin cần truy vấn** (tên sách, tác giả, chuyên ngành...).
3. Hệ thống **liệt kê danh sách kết quả**.
4. Độc giả **chọn một tài liệu** trong danh sách.
5. Hệ thống **trình bày thông tin chi tiết sách**.
   * Có thể **xem thêm chi tiết trong AD của nhân viên tiếp tân** (AD = Activity Diagram của NV tiếp tân).
6. Người dùng có thể:
   * **Xem lịch sử mượn trả** (nếu đã là thành viên).
   * **Thực hiện tìm kiếm lần nữa** nếu chưa tìm được sách mong muốn.

**b) Đăng ký thành viên**

* Nếu độc giả chưa là thành viên, có thể **đăng ký thành viên** thông qua nhân viên tiếp tân.
* Sau khi đăng ký, họ trở thành **thành viên chính thức** và được quyền mượn sách.

**c) Mượn sách**

* Khi đã là thành viên, độc giả có thể chọn chức năng **mượn sách**.
* Hệ thống/nhân viên tiếp tân **trình bày thông tin mượn trả** (ngày mượn, ngày trả, tình trạng).

**Bài tập ứng dụng:**

**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 kiến trúc 3 lớp**

a) Presentation Layer (UI / Client Layer):

* Công nghệ**:** React/Angular
* Thành phần chính**:**
  + Buyer UI: Product Catalog, Product Detail, Cart Page, Checkout Page, Login/Register, Rating.
  + SysAdmin UI: Admin Dashboard (quản lý sản phẩm, kho, giỏ hàng).
  + Shared: Authentication (login/logout), Navigation.

b) Business Logic Layer:

* Công nghệ**:** Node.js (Express/NestJS)
* Cấu trúc gợi ý:
  + ProductController (quản lý danh mục, chi tiết, filter/sort).
  + CartController(thêm/xóa/cập nhật giỏ hàng, tính toán tổng tiền).
  + OrderController (checkout, payment, gửi email).
  + InventoryController (quản lý kho, tồn kho).
  + RatingController (quản lý đánh giá).
  + UserController (auth, access control).

c) Data Access Layer

* Công nghệ:
  + ORM: Sequelize (Node.js)
  + Database: PostgreSQL
* Thành phần chính (Repository):
  + ProductRepository
  + CartRepository
  + OrderRepository
  + InventoryRepository
  + UserRepository
  + RatingRepository

**2. Database Design sơ bộ**

* **User** (UserID, Username, PasswordHash, Role).
* **Product** (ProductID, Name, Description, Price, HotFlag).
* **Inventory** (InventoryID, Stock, Address, Location, ProductID).
* **Cart** (CartID, UserID, Status).
* **CartItem** (CartItemID, CartID, ProductID, Quantity, Price).
* **Order** (OrderID, CartID, PaymentStatus, TotalAmount).
* **Rating** (RatingID, UserID, ProductID, Stars, Comment, Date).

**3. Luồng xử lý ví dụ (Checkout Flow)**

* Buyer click **Checkout** trên React UI.
* React gọi API POST /orders/checkout
* OrderController -> checkout: kiểm trả giỏ hàng (CartController), xác nhận tồn kho (validate) (InventoryController), xử lý thanh toán, lưu order vào Postgres.
* CartController cập nhật trạng thái giỏ hàng (Closed).
* Trả kết quả về React UI -> hiển thị thông báo thành công.