

TÀI LIỆU ĐẶC TẢ YÊU CẦU (SRS) - PROJECT JAVA WEB

Tên dự án: Nền tảng Hỗ trợ Học thuật & Quản lý Thiết bị Thực hành (Smart Academic & Lab Support Platform)

Kiến trúc: Monolithic (Ứng dụng nguyên khối)

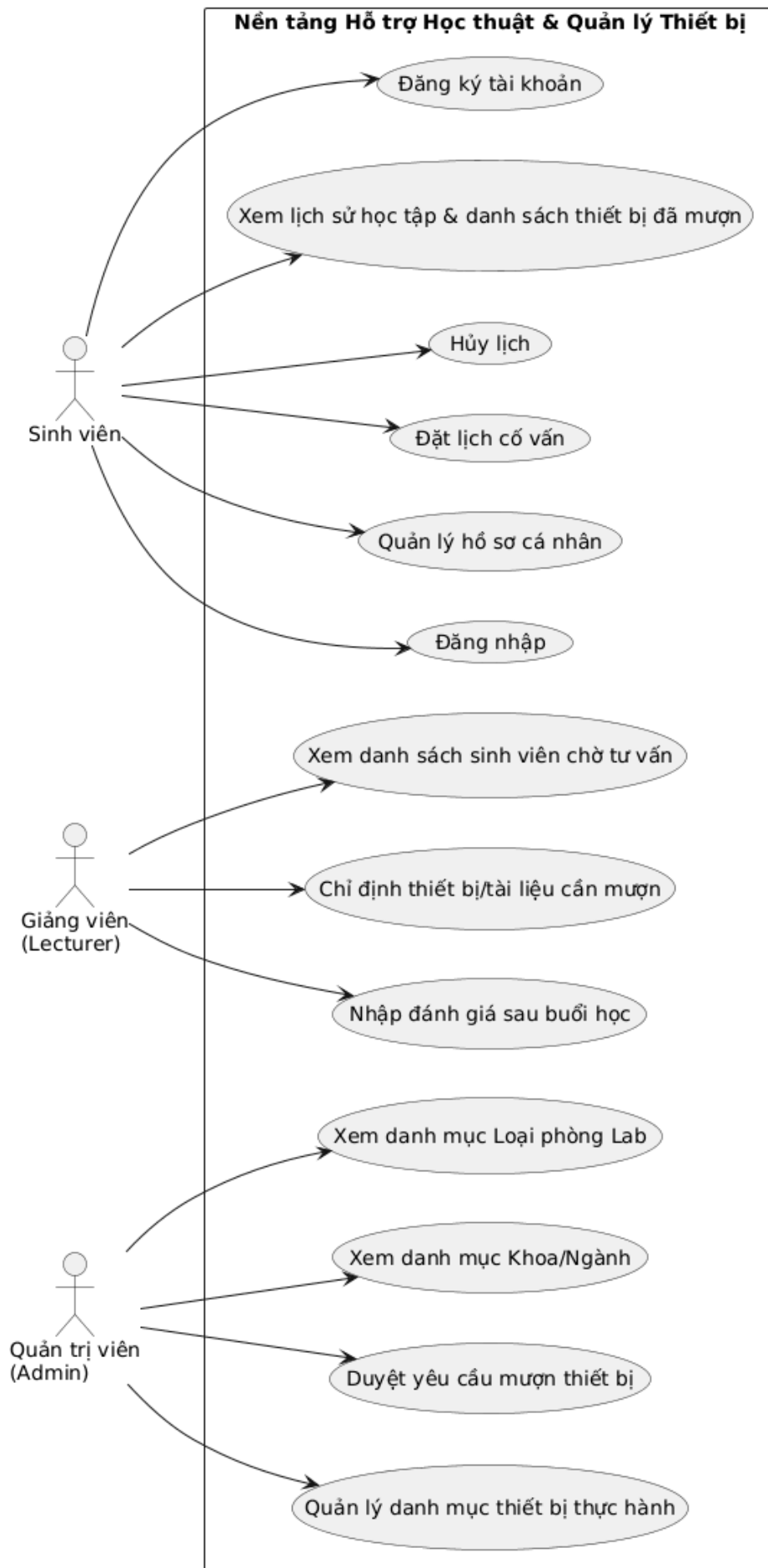
Tính chất: Phần Cốt lõi (Bắt buộc) + Các Module Mở rộng (Tự chọn)

1. MÔ TẢ TỔNG QUAN

Hệ thống là một ứng dụng Web hoàn chỉnh phục vụ quy trình đặt lịch cố vấn học tập (mentoring) và mượn trả thiết bị/tài liệu thực hành. Toàn bộ mã nguồn (giao diện, xử lý nghiệp vụ, kết nối cơ sở dữ liệu) được đóng gói và triển khai trên cùng một máy chủ (VD: Tomcat).

Các Tác nhân (Actors)

- Sinh viên (Student):** Đăng ký tài khoản, đăng nhập, quản lý hồ sơ cá nhân, đặt lịch cố vấn, hủy lịch, xem lịch sử học tập và danh sách thiết bị đã mượn.
- Giảng viên/Mentor (Lecturer):** Xem danh sách sinh viên chờ tư vấn, nhập đánh giá sau buổi học, chỉ định thiết bị/tài liệu cần mượn cho sinh viên.
- Quản trị viên (Admin):** Quản lý danh mục thiết bị thực hành, duyệt yêu cầu mượn thiết bị, xem danh mục Khoa/Ngành, xem danh mục Loại phòng Lab.



2. PHẦN CỐT LÕI (CORE REQUIREMENTS) - BẮT BUỘC

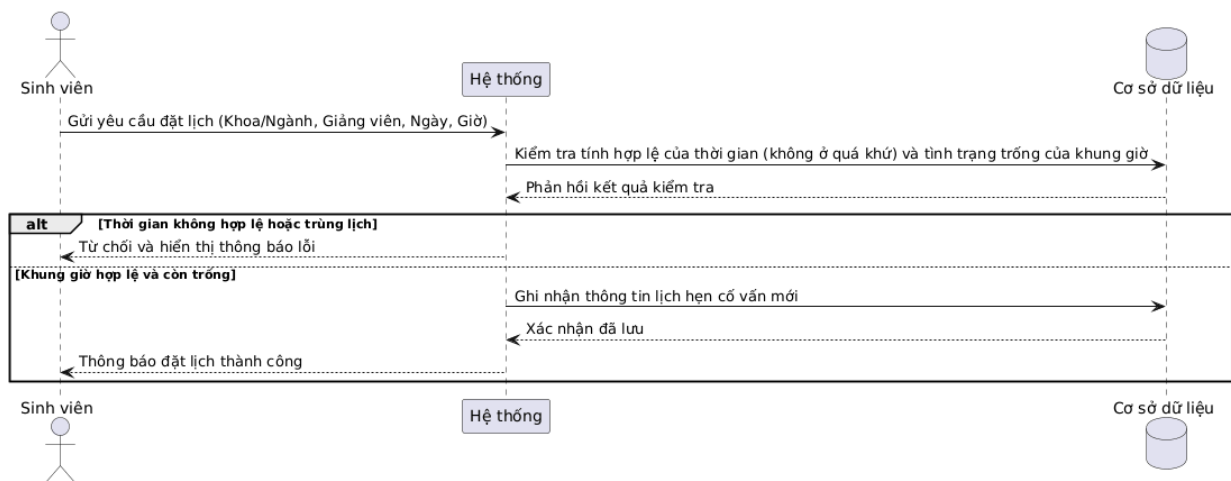
Phần này yêu cầu sinh viên phải xây dựng hoàn chỉnh luồng nghiệp vụ cơ bản. Dữ liệu được lưu trữ trong Relational Database (MySQL, PostgreSQL...).

2.1. Quản lý Tài khoản & Phân quyền cơ bản

- **CORE-01:** Đăng ký, đăng nhập. Mật khẩu lưu trong database bắt buộc phải được băm (hash) bảo mật.
- **CORE-02:** Kiểm soát truy cập. Sinh viên không được vào trang quản lý của Giảng viên; Giảng viên không được vào trang cấu hình của Admin.
- **CORE-03:** Quản lý Hồ sơ cá nhân (Profile) cho từng loại người dùng.

2.2. Luồng Hỗ trợ Học thuật (Nghiệp vụ chính)

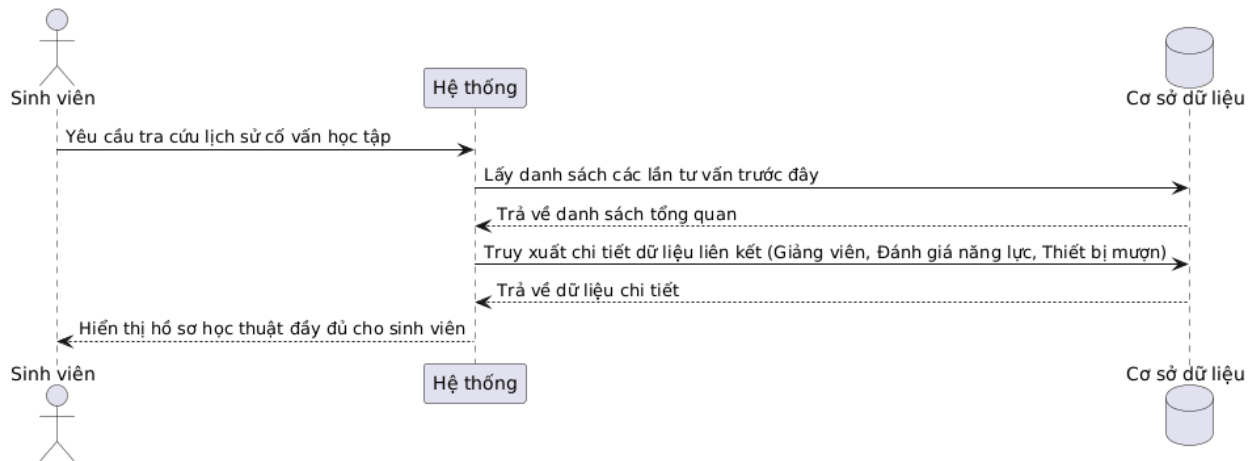
- **CORE-04:** Admin thực hiện quản lý (Thêm/Xem/Sửa/Xóa) cho duy nhất danh mục Thiết bị/Tài liệu. Các danh mục "Khoa/Ngành", "Loại phòng Lab" sẽ được hardcode (seed data) trực tiếp trong **Database** để làm dữ liệu nền tảng cho hệ thống.
- **CORE-05 - Đặt lịch Cố vấn & Chống xung đột tài nguyên:** Sinh viên chọn Khoa/Ngành -> Giảng viên -> Ngày, Giờ và thực hiện Đặt lịch.
 - **Yêu cầu nghiệp vụ (Core Logic):** Hệ thống bắt buộc phải kiểm tra và chặn 2 trường hợp: (1) Cùng 1 giảng viên không thể bị đặt 2 lần trong cùng 1 khung giờ (Tránh duplicate), và (2) Không cho phép đặt lịch ngược về ngày/giờ trong quá khứ.



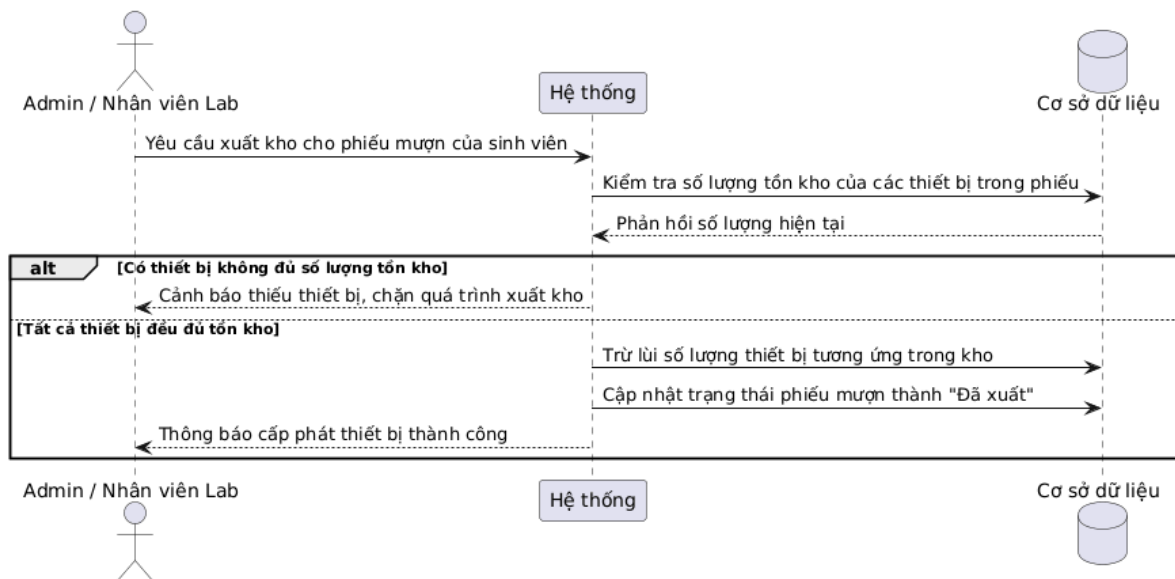
- **CORE-06 - Đánh giá & Tính toán vận dữ liệu (Transaction):** Giảng viên tiếp nhận ca tư vấn đang ở trạng thái "Chờ xác nhận". Giảng viên nhập đánh giá năng lực, chọn thiết bị/tài liệu từ danh mục cần cấp cho sinh viên, lưu kết quả.
 - **Yêu cầu nghiệp vụ (Core Logic):** Việc lưu hồ sơ tư vấn phải tác động đến nhiều bảng cùng lúc. Sinh viên phải áp dụng Transaction: Trạng thái lịch cập nhật thành "Đã hoàn thành", bảng đánh giá được lưu, chi tiết phiếu mượn thiết bị được tạo. Nếu một trong các thao tác này thất bại, toàn bộ quá trình phải bị Rollback để hệ thống không sinh ra rác dữ liệu.



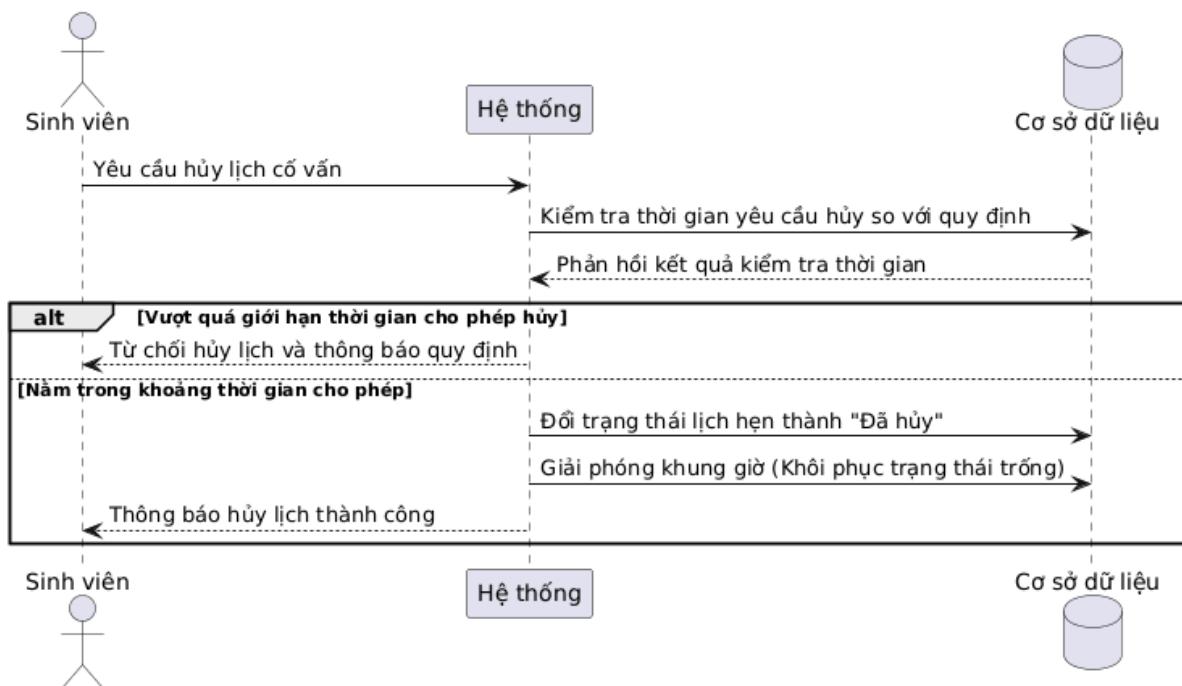
- **CORE-07 - Tra cứu Hồ sơ Học thuật Liên kết:** Sinh viên xem lại lịch sử các lần tư vấn trước đây.
 - **Yêu cầu nghiệp vụ (Core Logic):** Không dừng lại ở việc xem một danh sách đơn thuần, hệ thống phải truy xuất dữ liệu liên kết phức tạp giữa các bảng (JOIN) để trả về một "Hồ sơ học tập" hoàn chỉnh bao gồm: Tên giảng viên, Đánh giá năng lực và Chi tiết danh sách từng loại thiết bị đã mượn.



- **CORE-08 - Cấp phát Thiết bị & Quản lý Tồn kho:** Sau khi Giảng viên chỉ định thiết bị thành công (CORE-06), phiếu mượn tự động rơi vào trạng thái "Chờ cấp phát". Admin (hoặc nhân viên phòng Lab) kiểm tra và bấm "Xác nhận xuất kho".
 - **Yêu cầu nghiệp vụ:** Khi ấn xác nhận, hệ thống phải chạy logic kiểm tra Số lượng tồn kho của từng loại thiết bị trong phiếu. Nếu Đủ, tiến hành trừ lùi số lượng trong kho và cập nhật phiếu thành "Đã xuất". Nếu Không đủ tồn kho đối với bất kỳ thiết bị nào, chặn toàn bộ quá trình và thông báo lỗi.



- CORE-09 - Hủy lịch chủ động & Giải phóng Slot:** Sinh viên được phép chủ động "Hủy lịch" trước thời điểm tư vấn một khoảng thời gian nhất định (VD: trước 24 giờ).
 - Yêu cầu nghiệp vụ:** Khi người dùng hủy thành công, trạng thái lịch cập nhật thành "Đã hủy". Quan trọng nhất, hệ thống phải "giải phóng" (Unlock) khung giờ đó ngay lập tức. Khung giờ này sẽ tự động hiển thị trở lại là "Còn trống" ở luồng CORE-05 để các sinh viên khác có thể đặt vào.



3. GỢI Ý THIẾT KẾ CƠ SỞ DỮ LIỆU (DATABASE SCHEMA GUIDELINES)

Để triển khai thành công các luồng nghiệp vụ trên, đặc biệt là việc liên kết dữ liệu (JOIN) và quản lý giao dịch (Transaction), sinh viên cần thiết kế một hệ cơ sở dữ liệu chuẩn hóa (ít nhất đạt chuẩn 3NF). Dưới đây là gợi ý các thực thể (Entities) cốt lõi hệ thống cần có:

- **Bảng users (Tài khoản):** Quản lý thông tin xác thực và phân quyền.
- **Bảng user_profiles (Hồ sơ người dùng):** Lưu thông tin cá nhân chi tiết. Có thể gộp chung vào bảng **users** hoặc tách riêng để tối ưu.
- **Bảng departments (Khoa/Ngành):** Dữ liệu nền để phân loại giảng viên.
- **Bảng lecturers (Thông tin Giảng viên):** Mở rộng thông tin chuyên môn của user có role Giảng viên.
- **Bảng equipments (Danh mục Thiết bị):** Quản lý thiết bị/tài liệu và tồn kho (phục vụ CORE-04, CORE-08).
- **Bảng mentoring_sessions (Lịch hẹn tư vấn):** Bảng trung tâm giải quyết bài toán chống xung đột thời gian (CORE-05, CORE-09).
- **Bảng academic_evaluations (Hồ sơ Đánh giá):** Lưu trữ kết quả sau khi tư vấn (CORE-06).
- **Bảng borrowing_records (Phiếu mượn):** Quản lý trạng thái cấp phát thiết bị tổng thể của một lần tư vấn.
- **Bảng borrowing_details (Chi tiết phiếu mượn):** Bảng trung gian xử lý quan hệ nhiều-nhiều (N-N) giữa **borrowing_records** và **equipments**.

(Lưu ý: Sinh viên cần tự xác định đúng các Khóa chính (Primary Key), Khóa ngoại (Foreign Key) và vận dụng kiến thức Hibernate/JDBC mapping để ánh xạ các bảng này thành Entity trong code Java)

4. KHÔNG GIAN MỞ RỘNG (OPEN EXTENSIONS) - DÀNH CHO NHÓM XUẤT SẮC

Sinh viên tự chọn 1 hoặc nhiều hướng dưới đây để mở rộng ứng dụng Java Web của mình, chứng minh khả năng áp dụng các kỹ thuật lập trình nâng cao.

Hướng 1: Tích hợp Cổng thanh toán (Payment Integration)

- **Mô tả:** Chuyển đổi ứng dụng thành nền tảng có thu phí đối với các khóa huấn luyện đặc biệt hoặc phí đặt cọc mượn thiết bị đắt tiền.
- **Tính năng gợi ý:**
 - Tích hợp với một cổng thanh toán mô phỏng (Sandbox) như VNPay, Momo hoặc PayPal.
 - Khi sinh viên tạo lịch/đặt cọc, trạng thái là **CHỜ_THANH_TOÁN**. Chỉ khi cổng thanh toán trả về kết quả thành công, hệ thống mới chuyển sang **ĐÃ_XÁC_NHẬN**.
 - **Yêu cầu kỹ thuật:** Xử lý tốt các dữ liệu được gửi từ Controller, Java Backend ra dịch vụ bên ngoài và xử lý Callback/Webhook an toàn.

Hướng 2: Bảo mật Nâng cao & Quản lý Session (Advanced Security)

- **Mô tả:** Nâng cấp cơ chế bảo mật của ứng dụng Web
- **Tính năng gợi ý:**
 - Có thể sử dụng cơ chế **Interceptors** để làm và phân quyền.
 - Áp dụng Filter/Interceptor trong Spring MVC thuần hoặc **Spring Security** để quản lý luồng bảo mật tập trung.
 - Có thể sử dụng bằng **JWT (JSON Web Token)** hoặc thêm tính năng Đăng nhập bằng Google/Facebook (OAuth2).
 - Phân quyền chi tiết đến từng nút bấm (View) và từng đường dẫn (URI Endpoint).

Hướng 3: Xử lý Bất đồng bộ & Tự động hóa (Async & Background Jobs)

- **Mô tả:** Cải thiện trải nghiệm người dùng bằng cách không bắt họ chờ đợi các tác vụ tốn thời gian.
- **Tính năng gợi ý:**
 - **Gửi Email tự động:** Sau khi đặt lịch thành công, hệ thống gửi email xác nhận. Giao dịch gửi email phải chạy ngầm (Async/Thread riêng) để không làm chậm thao tác trên giao diện của người dùng.
 - **Background Task (Cron Job):** Viết một tiến trình chạy ngầm mỗi ngày vào lúc 00:00 để tự động quét và đánh dấu phạt các tài khoản mượn thiết bị quá hạn mà chưa trả lại.

Hướng 4: Tối ưu hóa Truy vấn & Báo cáo thống kê (Database Optimization)

- **Mô tả:** Đảm bảo ứng dụng chạy nhanh khi dữ liệu phình to.
- **Tính năng gợi ý:**
 - Xây dựng trang Dashboard cho Admin: Thống kê số lượng thiết bị đang được mượn, Top 5 giảng viên có lượt tư vấn nhiều nhất (vẽ Biểu đồ).
 - **Yêu cầu kỹ thuật:** Không dùng vòng lặp **for** trong Java để tính toán tổng. Sinh viên phải viết các câu lệnh truy vấn SQL nâng cao (JOIN, GROUP BY, HAVING) hoặc sử dụng các Query chuyên sâu của Hibernate/JPA/JDBC để lấy dữ liệu thống kê trực tiếp từ Database.

5. QUY ĐỊNH KỸ THUẬT & TIÊU CHÍ CHẤM ĐIỂM

1. **Kiến trúc mã nguồn (Code Architecture):** Tổ chức code chuẩn theo mô hình 3 lớp: **Controller** (Xử lý request/response) - **Service** (Chứa logic nghiệp vụ) - **Repository/DAO** (Tương tác Database). Tuyệt đối không viết logic kết nối Database trực tiếp trong Controller.
 - Sử dụng pattern **DTO (Data Transfer Object)** để truyền dữ liệu giữa Client và Server, không phơi bày trực tiếp Entity của Database ra ngoài.
2. **Quản lý Giao dịch (Transaction Management):** Xử lý đúng các nghiệp vụ lưu nhiều bảng cùng lúc (Ví dụ: Vừa lưu Lịch hẹn, vừa lưu Chi tiết Phiếu mượn). Nếu một bước lỗi, toàn bộ phải được **Rollback** để tránh rác dữ liệu.
3. **Tiêu chí đánh giá:**
 - Hoàn thành luồng CORE mượn mà, code sạch sẽ: Đạt điểm Khá.
 - Hoàn thành CORE + Tích hợp thành công 1 Hướng Mở rộng: Đạt điểm Giỏi.
 - Giao diện JSP/Thymeleaf thân thiện, xử lý bất lỗi (Validation) đầy đủ ở cả Client và Server, cấu trúc code tuân thủ chặt chẽ OOP: Đạt điểm Xuất sắc.