1. Tổng quan kiến trúc

Hệ thống được thiết kế theo kiến trúc **phân tầng** (layered architecture) gồm:

- **Controller Layer**: chịu trách nhiệm tiếp nhận request từ UI, trả kết quả/hiển thị lỗi (BaseController.showError).
- **Service Layer**: xử lý nghiệp vụ, gọi tầng dữ liệu hoặc external services.
- **Database / Entity Layer**: các thực thể lưu trữ (User, Student, Course, Feedback, StudyMaterial, ...).
- **External Services**: HCMUT_SSO (SSO), HCMUT_Library (học liệu), HCMUT_DataCore (đồng bộ/thu thập dữ liệu).

Nguyên tắc: Controller gọi Service; Service truy xuất Entity (DB) hoặc gọi External; Controller hiển thị kết quả hoặc gọi showError() khi có lỗi.

---

2. Giải thích ngắn về các lớp chính (role & phương thức quan trọng)

Controller Layer

- **BaseController**
    - +showError(message) — phương thức chung để hiển thị lỗi/feedback cho người dùng.
    - Tất cả controller kế thừa để tái sử dụng xử lý lỗi.
- **LoginController**
    - +login(username,password), +logout() — nhận credential từ UI, gọi AuthService.
- **ProfileController**
    - +viewProfile(), +updateProfile() — hiển thị/cập nhật hồ sơ; gọi ProfileService.
- **CourseController**

    - +registerCourse(courseId), +cancelCourse(courseId), +viewCourseList() — quản lý đăng ký khóa học; gọi CourseService.

- **FeedbackController**
    - +viewFeedback(sessionId), +makeFeedback(sessionId, rating, comment) — gửi/hiển thị feedback; gọi FeedbackService.

- **ProgressController**
    - +viewProgress(studentId) — hiển thị tiến độ; gọi ProgressService.

- **NotificationController**
    - +getNotifications(), +markAsRead(notificationId) — lấy và quản lý thông báo; gọi NotificationService.

- **ScheduleController**
    - +createSchedule(), +createBonusSession(), +viewSchedule(), +cancelSchedule() — quản lý lịch dạy; gọi ScheduleService.

- **AdminController**
    - +manageSystem(), +viewLogs() — chức năng quản trị; gọi AdminService.

- **StudyMaterialController** *(mới cho UC002)*
    - +viewStudyMaterials(courseId), +downloadMaterial(materialId) — lấy và trả học liệu; gọi StudyMaterialService.

Service Layer

- **AuthService**
    - +authenticate(username,password), +validateToken(token) — gọi HCMUT_SSO để xác thực/kiểm tra token.

- **ProfileService**
    - +getProfile(userId), +updateProfile(userId,data) — thao tác với User entity; có thể gọi HCMUT_DataCore để sync.

- **CourseService**
    - +getAvailableCourses(), +registerStudentToCourse(...), +cancelCourseRegistration(...) — quản lý Course & Enrollment.
- **FeedbackService**
    - +saveFeedback(studentId,courseId,rating,comment), +getFeedbackBySession(sessionId) — thao tác Feedback entity; có thể fetch từ HCMUT_DataCore khi cần tổng hợp.
- **ProgressService**
    - +getStudentProgress(studentId), +generateProgressChart(studentId) — lấy dữ liệu ProgressReport/DataCore, tạo biểu đồ.
- **NotificationService**
    - +sendNotification(userId,message), +getUserNotifications(userId) — thao tác Notification entity và gửi/đồng bộ.
- **ScheduleService**
    - +createSchedule(...), +createBonusSession(...), +cancelSchedule(...) — thao tác Schedule, BonusSession.
- **AdminService**
    - +monitorSystem(), +adjustSystemSettings() — truy vấn SystemLog, sync HCMUT_DataCore.
- **StudyMaterialService** *(mới cho UC002)*
    - +getMaterialsByCourse(courseId), +getMaterialDetail(materialId) — gọi HCMUT_Library, lưu/đọc StudyMaterial entity.

Database / Entity Layer (chính)

- **User (abstract)**: userId, name, email, role. Cha của Student, Tutor, Department, ITStaff.
- **Student, Tutor**: thuộc tính chuyên biệt (studentId, tutorId, expertise).
- **Course, Enrollment**: Course có nhiều Session; Enrollment biểu diễn quan hệ Student–Course.

- **Session, BonusSession**: Buổi học và buổi bổ sung.
- **Feedback**: feedbackId, rating, comment, date.
- **ProgressReport**: lưu trữ điểm/ mốc thời gian.
- **Notification, Schedule, StudyMaterial, SystemLog**: tương ứng.

External Services

- **HCMUT_SSO** (<<external>>): authenticate, validateToken — dùng cho tất cả action cần login.
- **HCMUT_Library** (<<external>>): fetchLearningMaterials(courseId), getLibraryResource(id) — dùng cho UC002.
- **HCMUT_DataCore** (<<external>>): syncStudentData, fetchCourseFeedback, fetchStudentProgress — dùng để đồng bộ/thu thập dữ liệu thống kê (UC001, UC004, UC005,...).

---

3. Mapping chi tiết lớp → Use Case (UC001–UC012)

Mỗi UC mình nêu luồng chính (success path), luồng alternative/exception, và classes tham gia chính.

UC001 — View Feedback

Mục tiêu: Bộ môn / Phòng Đào tạo / Tutor xem dữ liệu đánh giá buổi học (danh sách/biểu đồ/export).

Luồng chính:

1. UI: người dùng nhấn “Xem dữ liệu đánh giá” → FeedbackController.viewFeedback(sessionId).
2. FeedbackController gọi AuthService.validateToken(token) (đảm bảo đã login).
3. FeedbackController gọi FeedbackService.getFeedbackBySession(sessionId).
4. FeedbackService truy vấn Feedback entity; nếu cần tổng hợp hoặc dữ liệu lịch sử hơn, gọi HCMUT_DataCore.fetchCourseFeedback(sessionId/courseId).

5. FeedbackService trả dữ liệu cho controller; controller render biểu đồ/danh sách; nếu user yêu cầu export thì controller gọi service để xuất (Excel/PDF).

Classes tham gia: FeedbackController, AuthService, FeedbackService, Feedback (DB), HCMUT_DataCore (nếu cần).

Exception handling:

- Nếu validateToken() fail → FeedbackController.showError("Vui lòng đăng nhập").
- Nếu HCMUT_DataCore lỗi → FeedbackService ném exception; FeedbackController catch → showError("Không thể tải dữ liệu đánh giá, thử lại sau").

Alternative: lọc/ tìm kiếm/ xuất file → thực hiện trên FeedbackService (thêm params filter/sort) và FeedbackController hiển thị tùy chọn export.

UC002 — Access Study Material

Mục tiêu: Sinh viên/Tutor xem/tải tài nguyên học tập từ HCMUT_Library.

Luồng chính:

1. UI → StudyMaterialController.viewStudyMaterials(courseId).
2. StudyMaterialController (nếu cần) gọi AuthService.validateToken(token) để kiểm quyền.
3. StudyMaterialController gọi StudyMaterialService.getMaterialsByCourse(courseId).
4. StudyMaterialService gọi HCMUT_Library.fetchLearningMaterials(courseId).
5. Response từ HCMUT_Library trả về danh sách tài liệu; StudyMaterialService có thể:
    - Lưu/đồng bộ bản sao vào StudyMaterial entity (cache), hoặc
    - Trực tiếp trả cho controller.
6. StudyMaterialController trả danh sách cho UI; khi user chọn download → downloadMaterial(materialId) gọi StudyMaterialService.getMaterialDetail() → fetch resource (via HCMUT_Library.getLibraryResource) hoặc trả fileUrl.

Classes tham gia: StudyMaterialController, StudyMaterialService, HCMUT_Library, StudyMaterial entity, AuthService (kiểm token).

Exception handling:

- Nếu HCMUT_Library unreachable → StudyMaterialService ném lỗi; StudyMaterialController catch → showError("Không thể kết nối thư viện HCMUT").
- Nếu không tìm thấy tài liệu → trả empty → controller hiển thị “Không có kết quả phù hợp”.

Notes: quyền truy cập (public/private) do StudyMaterialService kiểm tra dựa trên Enrollment/role.

UC003 — Manage Personal Profile

Mục tiêu: Sinh viên/Tutor quản lý hồ sơ cá nhân.

Luồng chính:

1. ProfileController.viewProfile() gọi AuthService.validateToken() để lấy userId.
2. ProfileController gọi ProfileService.getProfile(userId).
3. ProfileService đọc User (Student/Tutor) entity; nếu cần cập nhật dữ liệu chính thức, có thể gọi HCMUT_DataCore.syncStudentData(userId) hoặc cập nhật local DB.
4. Khi user edit → ProfileController.updateProfile(data) gọi ProfileService.updateProfile(userId,data) → lưu User entity.

Classes tham gia: ProfileController, AuthService, ProfileService, User/Student/Tutor entity, HCMUT_DataCore (nếu đồng bộ).

Exception handling:

- Lỗi get/save dữ liệu → controller bắt lỗi và showError("Lỗi khi lưu thông tin, thử lại").

UC004 — View Progress

Mục tiêu: Tutor/Phòng Đào tạo/Bộ Môn xem tiến bộ học tập (điểm, mốc).

Luồng chính:

1. ProgressController.viewProgress(studentId) → AuthService.validateToken() (đảm bảo role được phép xem).

2. ProgressController gọi ProgressService.getStudentProgress(studentId).

3. ProgressService lấy ProgressReport từ DB và/hoặc gọi HCMUT_DataCore.fetchStudentProgress(studentId) để lấy dữ liệu đầy đủ.

4. ProgressService.generateProgressChart() tạo dữ liệu biểu đồ; trả controller hiển thị dạng cột.

Classes tham gia: ProgressController, AuthService, ProgressService, ProgressReport, HCMUT_DataCore.

Exception handling:

- Nếu không có dữ liệu → ProgressService trả empty; ProgressController.showError("Không có dữ liệu").

- Lỗi external → showError + ghi log.

UC005 — Make Feedback

Mục tiêu: Sinh viên gửi đánh giá sau buổi học.

Luồng chính:

1. Sau buổi, UI hiển thị form → user submit → FeedbackController.submitFeedback(courseId, rating, comment).

2. FeedbackController kiểm token → gọi FeedbackService.saveFeedback(studentId, courseId, rating, comment).

3. FeedbackService lưu Feedback entity; gọi NotificationService.sendNotification(tutorId, "Bạn có phản hồi mới").

4. Feedback được dùng sau đó cho UC001 (báo cáo).

Classes tham gia: FeedbackController, AuthService, FeedbackService, Feedback entity, NotificationService.

Exception handling:

- Lỗi lưu DB → controller showError("Lưu phản hồi thất bại").
- (Optional) Nếu cần tổng hợp lịch sử từ HCMUT_DataCore, gọi external.

UC006 — Cancel Register Course

Mục tiêu: Sinh viên hủy buổi đã đăng ký; system thông báo tutor.

Luồng chính:

1. UI → CourseController.cancelCourse(courseId) (có thể truyền sessionId để xác thực).
2. CourseController gọi CourseService.cancelCourseRegistration(studentId, courseId).
3. CourseService cập nhật Enrollment (trạng thái canceled) trong DB.
4. Gọi NotificationService.sendNotification(tutorId, "Sinh viên X đã hủy buổi").

Classes tham gia: CourseController, CourseService, Enrollment entity, NotificationService.

Exception handling / Alternative:

- Nếu hủy trong 30 phút có thể hoàn tác → UI/Controller xử lý undo (call CourseService.reinstateEnrollment).
- Nếu vi phạm rule (hủy trước 24h => cảnh cáo) → CourseService kiểm rule, có thể update record cấm/ghi log.

UC007 — Log in

Mục tiêu: Người dùng đăng nhập via HCMUT_SSO.

Luồng chính:

1. LoginController.login(username,password) nhận request.

2. LoginController gọi AuthService.authenticate(username,password).
3. AuthService gọi external HCMUT_SSO.authenticate(...).
4. Nếu success, AuthService nhận token, lưu/issue session/token cho client; LoginController trả success.
5. Nếu fail → LoginController.showError("Tên đăng nhập hoặc mật khẩu không chính xác").

Classes tham gia: LoginController, AuthService, HCMUT_SSO, User (để map account).

Exception handling:

- Nếu HCMUT_SSO unreachable → AuthService ném lỗi → controller showError("Không thể đăng nhập, thử lại sau").

UC008 — Receive notifications and reminders

Mục tiêu: Người dùng xem thông báo.

Luồng chính:

1. UI → NotificationController.getNotifications() → AuthService.validateToken.
2. NotificationController gọi NotificationService.getNotifications(userId).
3. NotificationService query Notification entity (filter by user), trả controller hiển thị danh sách.
4. Người dùng click vào 1 thông báo → NotificationController.viewNotificationDetail(id) → NotificationService.getNotificationDetail(id).

Classes tham gia: NotificationController, NotificationService, Notification entity.

Exception handling:

- Nếu DB lỗi hoặc trình render lỗi → showError("Có lỗi, hãy thử lại").

UC009 — Schedule appointments (Tutor đăng ký khung thời gian rảnh)

Luồng chính:

1. Tutor → ScheduleController.createSchedule() (gửi payload).
2. ScheduleController gọi ScheduleService.createSchedule(tutorId, timeSlot).
3. ScheduleService kiểm tra trùng lịch (logic), lưu Schedule entity.
4. Gửi thông báo thành công cho tutor via NotificationService.

Classes tham gia: ScheduleController, ScheduleService, Schedule entity, NotificationService.

Exception handling / Alternative:

- Nếu trùng lịch → ScheduleService ném businessException → controller showError("Trùng lịch dạy").
- Nếu lưu DB lỗi → showError + ghi SystemLog.

UC010 — Select a tutor (Student đăng ký khóa học)

Luồng chính:

1. CourseController.registerCourse(courseId) (student chọn khung/học phần).
2. CourseController call CourseService.registerStudentToCourse(studentId, courseId).
3. CourseService kiểm các rule (quota, trùng lịch), tạo Enrollment.
4. Gọi NotificationService.sendNotification(tutorId, "Sinh viên A đăng ký").

Classes tham gia: CourseController, CourseService, Enrollment, NotificationService.

Exception handling:

- Nếu course không tồn tại → controller showError("Khóa học không tồn tại").
- Nếu lưu lỗi → showError("Có lỗi xảy ra, thử lại").

Alternative: hệ thống gợi ý tutor tự động → logic nằm ở CourseService (recommendation).

UC011 — Bonus Schedule

Mục tiêu: Tutor tạo buổi dạy bổ sung.

Luồng chính:

1. Tutor → ScheduleController.createBonusSession() (gửi nội dung form).
2. ScheduleService.createBonusSession(tutorId, info) kiểm dữ liệu.
3. Nếu need admin approval → set status = Pending (entity BonusSession).
4. Lưu BonusSession entity; gửi notify đến student list via NotificationService.
5. Nếu student accept → update RSVP (cơ sở dữ liệu).

Classes tham gia: ScheduleController, ScheduleService, BonusSession, NotificationService, AdminService (nếu approval).

Exception handling:

- Nếu notification service down → ScheduleService vẫn lưu; controller showError("Buổi đã lưu, nhưng thông báo chưa gửi — sẽ thử lại").

UC012 — Manage and maintain the system

Mục tiêu: IT staff quản trị hệ thống (đăng nhập admin, xem log, điều chỉnh).

Luồng chính:

1. AdminController.manageSystem() (sau AuthService.validateToken với role admin).
2. AdminController gọi AdminService.monitorSystem() để đọc SystemLog hoặc AdminService.adjustSettings().
3. AdminService có thể gọi HCMUT_DataCore để sync, hoặc cập nhật cấu hình.

Classes tham gia: AdminController, AuthService, AdminService, SystemLog, HCMUT_DataCore.

Exception handling: showError cho admin nếu thao tác thất bại.

---

4. Cách xử lý Exception Flow và showError()

- **Nguyên tắc**: Service layer trả lỗi (throw exception hoặc trả error code). Controller chịu trách nhiệm bắt lỗi và phản hồi UI bằng BaseController.showError(message).
- **Ghi log**: khi exception xảy ra, Controller/Service gọi AdminService hoặc SystemLog để ghi chi tiết lỗi; có thể trigger NotificationService gửi cảnh báo tới admin.
- **Ví dụ**: UC002 — khi HCMUT_Library unreachable:
    - StudyMaterialService throws ExternalServiceException.
    - StudyMaterialController catch → showError("Không thể kết nối đến thư viện") + gọi AdminService để log.

---

5. Một số ghi chú thiết kế / best-practices

- **Xác thực / phân quyền**: mọi controller action quan trọng phải validate token qua AuthService.validateToken() để đảm bảo pre-condition “đã login”.
- **Adapter cho external**: trong hiện thực nên tạo adapter/wrapper (ví dụ HCMUTLibraryClient) để dễ mock/tests; in UML ta biểu diễn là dependency đến HCMUT_Library.
- **Transaction & Consistency**: các thao tác multi-step (ví dụ tạo buổi + notify) nên xử lý transaction hoặc compesation nếu notify fail.
- **Caching**: StudyMaterialService có thể cache kết quả từ HCMUT_Library (lưu StudyMaterial) để giảm load và cho phép hiển thị khi library tạm thời unreachable.
- **Centralized Error Handling**: nếu muốn thống nhất, thêm ErrorHandler service để controller forward lỗi, log, và decide notification.