ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC BÁCH KHOA

Khoa Khoa học và Kỹ thuật Máy tính

BÀI TẬP LỚN CÔNG NGHỆ PHẦN MỀM

HỆ THỐNG HỖ TRỢ TUTOR TẠI TRƯỜNG ĐẠI HỌC BÁCH KHOA - ĐHQG TP.HCM Nhóm: Mon

GVHD: Thầy Mai Đức Trung

STT	Họ và tên	MSSV
1	Lê Triều Kha	2311388
2	Võ Tiến Nam	2312205
3	Trịnh Vũ Đức Hải	2310886
4	Trịnh Nguyên Phát	2312603
5	Phan Đình Khang	2311459
6	Dương Bá Khang	2311403
7	Trương Trung Kiên	2311745

Thành phố Hồ Chí Minh, tháng 9 năm 2025

BẢNG PHÂN VIỆC

MSSV	Họ tên	Công việc	HT
2311388	Lê Triều Kha	- Vẽ các diagram về Request và Message - Đảm nhiệm code cho 2 phần trên - Tham gia xây dựng UIUX	100%
2312205	Võ Tiến Nam	- Vẽ các diagram về Session - Đảm nhiệm code cho phần trên - Vẽ Deployment View	100%
2310886	Trịnh Vũ Đức Hải	- Vẽ các diagram về Admin (Staff, Affair) - Đảm nhiệm code cho 2 phần trên - Tham gia xây dựng UIUX	100%
2312603	Trịnh Nguyên Phát	- Vẽ các diagram về Authorization và Library - Đảm nhiệm code cho 2 phần trên - Vẽ Class Diagram	100%
2311459	Phan Đình Khang	- Vẽ các diagram về Student - Đảm nhiệm code cho phần trên - Tham gia xây dựng UIUX	100%
2311403	Dương Bá Khang	- Vẽ các diagram về Tutor - Đảm nhiệm code cho phần trên - Vẽ Development View	100%
2311745	Trương Trung Kiên	- Vẽ các diagram về Evaluation (đánh giá) - Đảm nhiệm code cho phần trên - Tham gia xây dựng UIUX	100%

Bảng 1: Bảng phân việc nhóm

MỤC LỤC

1 Specification **6**

2 Functional and non-functional requirements **8**
2.1 Functional requirements . . . 8
2.1.1 Các functional requirements chung . . . 8
2.1.2 Đối với Sinh viên (Student) . . . 8
2.1.3 Đối với Tutor . . . 9
2.1.4 Đối với Điều phối viên/Khoa (Coordinator/Dept) . . . 9
2.1.5 Đối với Hệ tích hợp bên ngoài . . . 10
2.2 Non-functional Requirements . . . 11

3 Usecase Diagram **13**
3.1 Use case diagram for the whole system . . . 13
3.2 Use case diagram detail and scenario . . . 14
3.2.1 User Login and Synchronization another system . . . 14
3.2.2 Academic Overview Generation and Data Distribution . . . 16
3.2.3 Progress Evaluation and Academic Data Collection . . . 18
3.2.4 Learning Material Access and Synchronization . . . 20
3.2.5 Admin manage profile user . . . 21
3.2.6 System Suggest Tutor for Student . . . 22
3.2.7 Tutor manage calender . . . 23
3.2.8 Student enroll session . . . 25
3.2.9 Student Request Summary . . . 26

4 Sequence and Activity Diagram **28**
4.1 User Login and Synchronization . . . 28
4.2 Academic Overview Generation and Data Distribution . . . 30
4.3 Progress Evaluation and Academic Data Collection . . . 31
4.4 Learning Material Access . . . 33
4.4.1 Access Learning Material . . . 34
4.4.2 Provide Course Data . . . 35
4.5 Admin Management . . . 36
4.6 System Suggest Tutor for Student . . . 38
4.7 Tutor's Calendar Management . . . 39

4.7.1 Register Teaching Schedule . . . 39
4.7.2 Open Session . . . 41
4.7.3 Manage Session . . . 43
4.8 Student's Enrollment . . . 45
4.8.1 Student Book Session . . . 45
4.8.2 Student Delete Session . . . 47
4.8.3 Student Change session . . . 50
4.9 Student Request Summary . . . 52
4.9.1 Request Summary . . . 52

5 Mockup **54**
5.1 Student UI . . . 54
5.1.1 Overview . . . 54
5.1.2 Find Tutor . . . 55
5.1.3 Open Section . . . 56
5.1.4 My Session . . . 57
5.1.5 Library . . . 58
5.1.6 Profile . . . 58
5.2 Tutor UI . . . 59
5.2.1 Overview . . . 59
5.2.2 Session . . . 60
5.2.3 Availability . . . 61
5.2.4 Student Progress . . . 61
5.2.5 Library . . . 61
5.2.6 Profile . . . 62
5.2.7 Messages . . . 62
5.3 Academic Affair UI . . . 63
5.3.1 Overview . . . 63
5.3.2 Analysis . . . 63
5.3.3 Resource . . . 64
5.3.4 Report . . . 65
5.4 Student Affair UI . . . 65
5.4.1 Overview . . . 65
5.4.2 Training Credit . . . 66
5.4.3 **Scholarship** . . . 66
5.4.4 Participation . . . 67

6 Architecture design **68**
6.1 Deployment view . . . 68
6.1.1 Client PC . . . 68
6.1.2 Web Server . . . 68
6.1.3 Application Server . . . 69
6.1.4 AI Service Server . . . 70
6.1.5 Database Server . . . 70
6.1.6 Cache Server . . . 71
6.1.7 Storage Server . . . 71
6.1.8 External Systems . . . 71
6.1.9 Tương tác giữa các node . . . 71
6.2 Development view . . . 72
6.2.1 Component Diagram . . . 72

6.2.2 Phân hệ Student Web UI . . . 73
6.2.3 Phân hệ Tutor Web UI . . . 73
6.2.4 Phân hệ Staff Web UI . . . 73
6.2.5 Phân hệ Backend Server . . . 74
6.2.6 Phân hệ Database . . . 74
6.2.7 Hệ thống bên ngoài (External Systems) . . . 74
6.2.8 Tương tác giữa các phân hệ . . . 75

7 Class diagram **76**
7.1 User . . . 77
7.1.1 Student . . . 77
7.1.2 Tutor . . . 77
7.1.3 Admin . . . 78
7.1.4 Student affairs office . . . 78
7.2 Other entities . . . 78
7.2.1 Session . . . 78
7.2.2 StudentEvaluation (Đánh giá sinh viên) . . . 79
7.2.3 Message (Tin nhắn) . . . 80
7.2.4 RescheduleRequest (Yêu cầu dời lịch) . . . 80
7.2.5 Library (Thư viện) . . . 81
7.2.6 LibraryResource (Tài liệu Thư viện) . . . 82

8 Vận hành hệ thống (Working Demonstration) **83**
8.1 Quy trình Đăng nhập và Dashboard . . . 83
8.2 Luồng Tạo và Tham gia Lịch học . . . 85
8.2.1 Dành cho Gia sư (Tutor) . . . 85
8.2.2 Dành cho Sinh viên (Student) . . . 86
8.3 Luồng Điều khiển Lịch học . . . 88
8.3.1 Gia sư (Tutor) . . . 88
8.3.2 Sinh viên (Student) . . . 90
8.4 Luồng Đánh giá (Evaluation Flow) . . . 90
8.4.1 Đánh giá Sinh viên . . . 90
8.4.2 Đánh giá Giảng viên . . . 92
8.5 Luồng Thư viện (Library Flow) . . . 93
8.6 Luồng Tin nhắn (Messaging Flow) . . . 94

9 Generative AI Contribution **97**

CHƯƠNG 1

SPECIFICATION

Mục tiêu dự án

Trong môi trường giáo dục đại học hiện nay, việc hỗ trợ học tập và phát triển kỹ năng cho sinh viên là một nhu cầu ngày càng quan trọng. Tại Trường Đại học Bách Khoa – Đại học Quốc gia TP.HCM (HCMUT), chương trình Tutor/Mentor đã được triển khai nhằm giúp sinh viên được hướng dẫn, tư vấn và đồng hành trong quá trình học tập. Tuy nhiên, công tác quản lý, phân công và theo dõi các hoạt động tutor hiện nay còn phụ thuộc nhiều vào thủ công, thiếu tính đồng bộ và khó mở rộng. Do đó, việc phát triển một hệ thống hỗ trợ Tutor hiện đại, hiệu quả và có khả năng tích hợp cao là rất cần thiết để đáp ứng nhu cầu quản lý và hỗ trợ học tập trong toàn trường.

Hệ thống HCMUT_Tutor được xây dựng nhằm hỗ trợ toàn bộ quy trình vận hành của chương trình Tutor tại trường. Hệ thống cho phép quản lý thông tin của sinh viên và tutor (bao gồm hồ sơ cá nhân, lĩnh vực chuyên môn, nhu cầu hỗ trợ), đồng thời cung cấp cơ chế đăng ký, lựa chọn hoặc ghép cặp tự động giữa sinh viên và tutor. Các tutor có thể thiết lập lịch rảnh, mở các buổi tư vấn, tổ chức buổi học trực tiếp hoặc trực tuyến, cũng như quản lý và cập nhật thông tin buổi hướng dẫn. Hệ thống còn hỗ trợ đặt lịch, hủy lịch, đổi lịch, gửi thông báo tự động và nhắc lịc qua email hoặc giao diện hệ thống, giúp đảm bảo sự chủ động và thuận tiện cho cả hai bên.

Bên cạnh đó, HCMUT_Tutor tích hợp các công cụ phản hồi và đánh giá, giúp sinh viên có thể nhận xét chất lượng buổi học, còn tutor có thể ghi nhận tiến bộ của sinh viên. Các dữ liệu này được tổng hợp để các khoa, bộ môn và phòng ban liên quan có thể phân tích và theo dõi: khoa có thể nắm bắt tình hình học tập của sinh viên trong từng môn học, phòng Đào tạo khai

thác báo cáo để tối ưu phân bổ nguồn lực học thuật, trong khi phòng Công tác Sinh viên có thể sử dụng dữ liệu về mức độ tham gia và phản hồi của sinh viên để xét học bổng, cộng điểm rèn luyện hoặc ghi nhận thành tích hỗ trợ học tập.

Nhờ sự tích hợp này, hệ thống HCMUT_Tutor không chỉ nâng cao hiệu quả quản lý và vận hành chương trình Tutor mà còn tạo ra một nền tảng hỗ trợ học tập toàn diện, linh hoạt và bền vững, góp phần nâng cao chất lượng đào tạo và trải nghiệm học tập tại Đại học Bách Khoa.

CHƯƠNG 2

FUNCTIONAL AND NON-FUNCTIONAL REQUIREMENTS

2.1 Functional requirements

2.1.1 Các functional requirements chung

- Tài khoản: Người dùng phải đăng nhập thông qua dịch vụ HCMUT_SSO trước khi sử dụng hệ thống.
- Giao diện: Ứng dụng cung cấp giao diện web cho các thao tác: đặt lịch, tham gia cộng đồng, xem tài nguyên, quản trị chương trình, báo cáo, đồng bộ tích hợp.

2.1.2 Đối với Sinh viên (Student)

- Đăng ký/Tham gia chương trình: Sinh viên có thể xem danh mục chương trình (học thuật/phi học thuật) và Enroll in Program.
- Tìm & xem hồ sơ Tutor: Tra cứu danh sách Tutor, View Tutor Profile, xem kỹ năng/chuyên môn/đánh giá.
- Gợi ý ghép cặp thông minh (AI Match): Hệ thống Get Tutor Suggestion (AI/ML match) dựa trên hồ sơ, mục tiêu học tập, lịch rảnh và chính sách ghép cặp.
- Đặt/Quản lý lịch hẹn: Sinh viên có thể Book Session, Reschedule Session, Cancel Session; hệ thống Manage Booking, Notify Participants, Generate Meeting Link (Online) và Sync to Calendar (Cal).
- Hỏi đáp & trợ lý học tập AI: Sinh viên Ask Learning Question và dùng Start AI Study Coach (Q&A); hệ thống Cite Sources cho câu trả lời AI.
- Kế hoạch cá nhân hóa: Hệ thống Generate Personalized Plan và Recommend Resources; sinh viên xem/gán nhiệm vụ, mốc tiếp theo (Update Next Steps).
- Tài nguyên học tập: Access Learning Resources, Retrieve from Library (từ HCMUT_LIBRARY) và Attach Learning Resource vào phiên/trao đổi.

- Cộng đồng học tập: Tham gia Create/Join Community Space, Post & Comment, Share Resource; tuân thủ điều phối & kiểm duyệt.
- Tham dự phiên: Attend Session (online/offline). Sau phiên, có thể xem Session Summary/Minutes, điểm danh, tài nguyên liên quan.
- Theo dõi tiến độ: Xem Track Student Progress theo mục tiêu/chương trình/khóa học.

2.1.3 Đối với Tutor

- Cập nhật hồ sơ và năng lực: Cập nhật hồ sơ, lĩnh vực, tài nguyên giảng dạy; hồ sơ được hiển thị trong View Tutor Profile.
- Công bố lịch rảnh: Publish Availability Slots; hệ thống hiển thị cho sinh viên/điều phối để đặt lịch.
- Duyệt/Quản lý phiên dạy: Conduct Session, Record Attendance, ghi Session Summary/Minutes, đính kèm tài nguyên và Share Resource.
- Theo dõi học viên: Xem Track Student Progress, đề xuất Update Next Steps sau mỗi phiên.
- Cộng đồng: Tham gia/khởi tạo chủ đề trong Community Space, đăng bài, bình luận, chia sẻ tài nguyên theo quy định.

2.1.4 Đối với Điều phối viên/Khoa (Coordinator/Dept)

- Quản trị chương trình: Create/Edit Program (Academic/Non-academic); thiết lập mục tiêu, tiêu chí hoàn thành, quy tắc báo cáo.
- Chính sách ghép cặp & lịch: Configure Matching Rules (tiêu chí AI match) và Manage Scheduling Policies (khung giờ, giới hạn phiên, khoảng đệm...).
- Quản lý Tutor–Sinh viên: Assign/Reassign Tutors to Students, xử lý Conflict Resolution khi phát sinh khiếu nại/va chạm lịch.
- Cộng đồng & sự kiện: Create Community Event (có thể Link to Session/Booking); Report/Moderate Content (warn/mute/remove) trong các Community Space.
- Đồng bộ dữ liệu: Sync Master Data với hệ thống của trường ở 2 chế độ: Scheduled Sync và Manual Sync.
- Báo cáo – phân tích: View Program Reports với View Dashboard & KPIs (hiệu suất chương trình, mức độ tham gia, tỉ lệ hoàn thành), Export Data (CSV/PDF), Anonymize/Aggregate Data khi cần chia sẻ bên ngoài (phục vụ OAA/OSA).
- Thông báo & tích hợp: Cấu hình kênh thông báo (email/in-app) cho đặt lịch, thay đổi, hủy; cấu hình Sync to Calendar (Cal) và nhà cung cấp Generate Meeting Link (Online).

2.1.5 Đối với Hệ tích hợp bên ngoài

- HCMUT_SSO: Xác thực người dùng & phân quyền truy cập theo vai trò (Student/Tutor/Coordinator).
- HCMUT_LIBRARY: Cung cấp API để Retrieve from Library (metadata, link truy cập, điều kiện bản quyền) cho tài nguyên được gợi ý/trích dẫn.
- Office of Academic Affairs (OAA) / Office of Student Affairs (OSA): Nhận báo cáo tổng hợp/ẩn danh từ View Program Reports; nhận cảnh báo khi có vụ việc Conflict Resolution vượt quyền xử lý của khoa.
- Calendar/Meeting Providers: Kết nối để Sync to Calendar (Cal) và Generate Meeting Link (Online) (Teams/Zoom/Meet) khi đặt/đổi/hủy phiên.

2.2 Non-functional Requirements

Non-functional requirements (NFR) là các ràng buộc, tiêu chuẩn áp dụng cho hệ thống, ảnh hưởng đến trải nghiệm người dùng và chất lượng sản phẩm. Việc tuân thủ các yêu cầu này đảm bảo hệ thống vận hành ổn định, hiệu quả, bảo mật và dễ bảo trì.

Trong chương trình Tutor/Mentor dành cho trường Đại học Bách Khoa - ĐHQG TP.HCM (HCMUT), ta chú tâm vào các yêu cầu chủ chốt bao gồm độ bảo mật (Security), hiệu năng (Performance), khả năng sử dụng (Usability), tính sẵn sàng (Availibility), độ tin cậy (Reliability), khả năng bảo trì và mở rộng (Maintainance and Extensibility), khả năng tích hợp (Interoperability) và hỗ trợ đa nền tảng (Portability)

ID	Requirement	Description
NFR-1	Security	Khả năng bảo vệ dữ liệu, ngăn chặn truy cập trái phép
NFR-2	Performance	Tốc độ xử lý, thời gian phản hồi của hệ thống
NFR-3	Usability	Khả năng dễ sử dụng của hệ thống
NFR-4	Availibility	Các yếu tố về thời gian sử dụng và yêu cầu sử dụng của hệ thống
NFR-5	Reliability	Khả năng vận hành ổn định
NFR-6	Maintainance and Extensibility	Các cơ chế phát hiện và xử lý lỗi, mở rộng phạm vi các chức năng hệ thống
NFR-7	Interoperability	Tích hợp với hệ thống nội bộ và bên ngoài
NFR-8	Portability	Khả năng chạy trên đa nền tảng, đa hệ điều hành

Bảng 2.1: Non-functional Requirements cho hệ thống HCMUT Tutor

1. **Security**
   - Phải tích hợp với dịch vụ xác thực tập trung HCMUT_SSO để quản lý đăng nhập thống nhất.
   - Có cơ chế phân quyền phù hợp với vai trò người dùng theo thông tin từ hệ thống tập trung của trường như RBAC.
   - Có hệ thống log hoạt động để theo dõi thao tác người dùng.
2. **Performance**
   - Thời gian phản hồi dưới 2s cho các thao tác cơ bản gồm đăng nhập, tra cứu lịch, đặt lịch.
   - Hỗ trợ tối thiểu 10000 người dùng đồng thời bao gồm sinh viên, tutor, giảng viên, phòng ban, ...
   - Xử lý đồng bộ dữ liệu từ HCMUT_DATACORE và HCMUT_SSO trong thời gian thực.
3. **Usability**
   - Có giao diện web thân thiện, người dùng có thể sử dụng thành thạo sau 30 phút sử dụng.

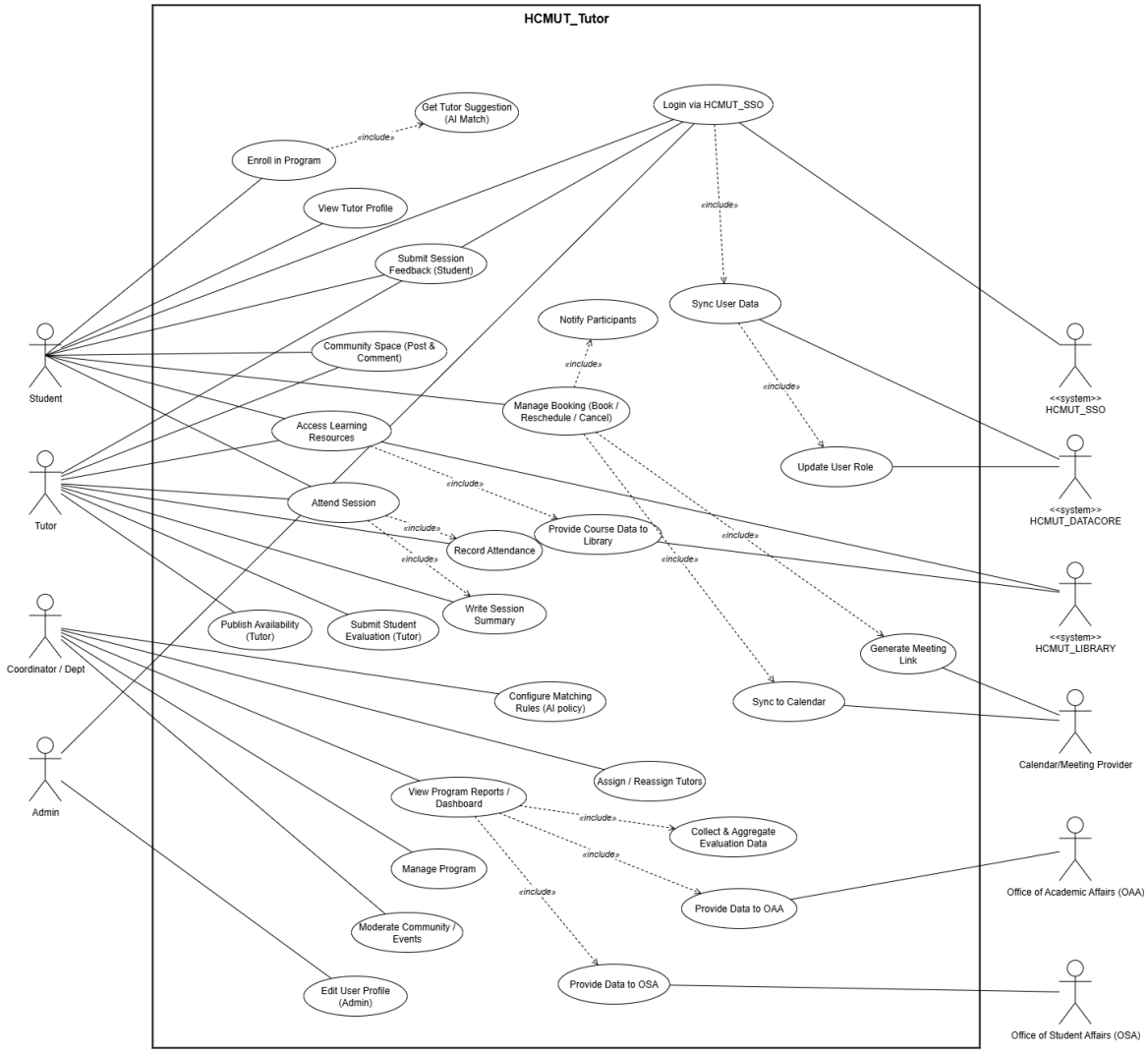
- Ngôn ngữ: Mặc định Tiếng Việt, có thể mở rộng sang các ngôn ngữ khác như Tiếng Anh, Tiếng Nhật cho các sinh viên hệ tài năng.

4. **Availability**
   - Hệ thống yêu cầu vận hành 24/7 trừ khi bảo trì hoặc nâng cấp.
   - Bảo trì hoặc nâng cấp tối đa 1 lần mỗi 3 tháng.
5. **Reliability**
   - Hỗ trợ cơ chế backup và restore dữ liệu ít nhất 1 lần mỗi ngày.
   - Có cơ chế xử lý lỗi, tránh mất dữ liệu khi vận hành.
6. **Maintainance and Extensibility**
   - Code tuân thủ clean code, có tài liệu rõ ràng.
   - Có hệ thống Log Monitoring để phát hiện lỗi, kiểm soát hiệu suất và bảo mật của hệ thống.
   - Cho phép mở rộng khi người dùng tăng hoặc mở rộng các chức năng nâng cao như AI hoặc cộng đồng trực tuyến mà không giảm quá 10% hiệu năng.
7. **Interoperability**
   - Tích hợp với hệ thống nội bộ bao gồm HCMUT_SSO, HCMUT_DATACORE và HCMUT_LIBRARY.
   - Có thể tích hợp các các hệ thống bên ngoài như Email và Google Calendar
8. **Portability**
   - Có khả năng đáp ứng trên nhiều nền tảng, đặc biệt là mobile.
   - Tương thích với hạ tầng Cloud.

CHƯƠNG 3

USECASE DIAGRAM

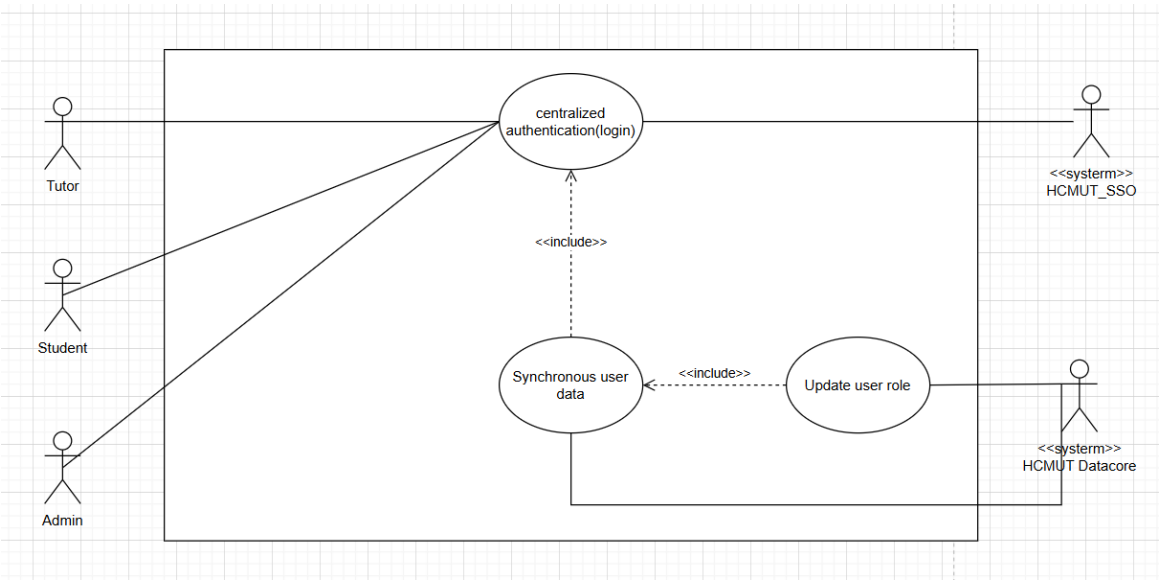
3.1 Use case diagram for the whole system



Hình 3.1: Main use case diagram

3.2 Use case diagram detail and scenario

3.2.1 User Login and Synchronization another system



Hình 3.2: Use case diagram: Login, Data Synchronization, and Role Update

Bảng sau mô tả chi tiết ba use case chính trong sơ đồ trên. Các chức năng này đảm bảo hệ thống Tutor Support tự động đồng bộ thông tin cá nhân và vai trò người dùng dựa trên dữ liệu từ HCMUT Datacore.

Use Case: Centralized authentication	
Actors	Tutor, Student, Admin, HCMUT SSO
Type	Semi-interactive (trigger for background processes)
Description	Người dùng đăng nhập bằng tài khoản HCMUT_SSO. Sau khi xác thực thành công, hệ thống tự động kích hoạt quá trình đồng bộ dữ liệu cá nhân và vai trò từ HCMUT Datacore.
Trigger	Người dùng thực hiện đăng nhập.
Main Flow	1. Người dùng đăng nhập qua HCMUT_SSO. 2. Hệ thống xác thực thông tin đăng nhập. 3. Hệ thống tự động kích hoạt use case *Synchronous user data.*
Post-condition	Người dùng được xác thực và dữ liệu cá nhân bắt đầu được đồng bộ hoá.

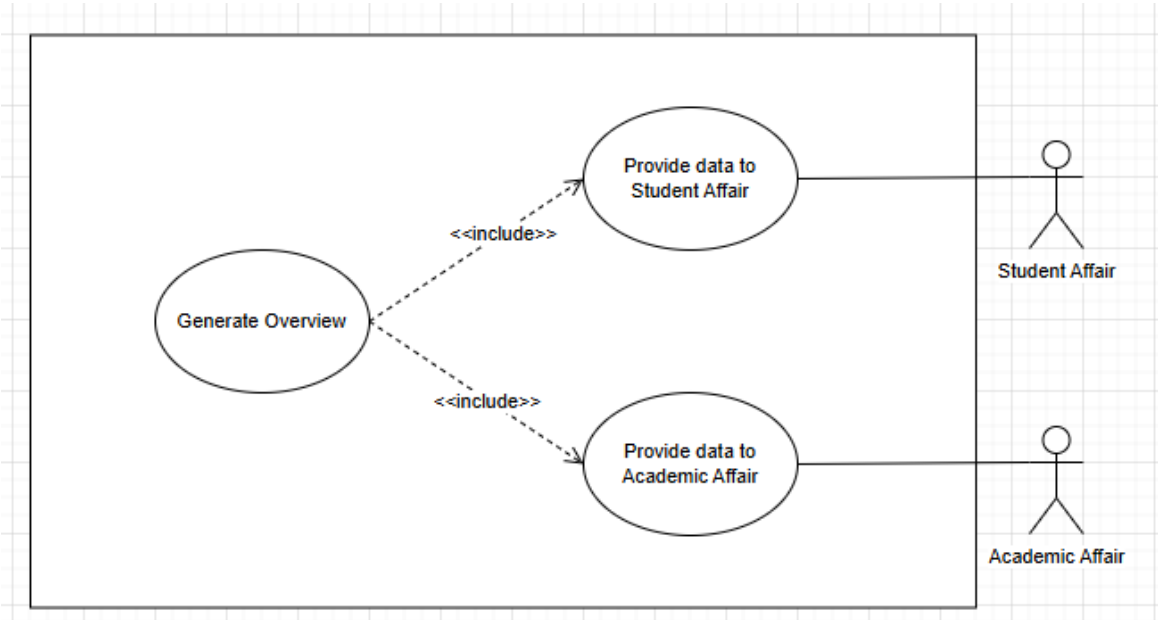
Use Case: Synchronous user data	
Actors	HCMUT Datacore
Type	Non-interactive

Description	Hệ thống tự động gửi yêu cầu đến HCMUT Datacore để lấy thông tin cá nhân, khoa, chuyên ngành và trạng thái học tập của người dùng. Quá trình này đảm bảo dữ liệu người dùng trong hệ thống Tutor Support luôn chính xác và nhất quán.
Trigger	Khi người dùng đăng nhập hoặc đến chu kỳ đồng bộ định kỳ.
Pre-condition	Kết nối với HCMUT Datacore ổn định.
Main Flow	1. Hệ thống gửi yêu cầu đồng bộ dữ liệu đến Datacore. 2. Datacore phản hồi dữ liệu cá nhân và vai trò. 3. Hệ thống cập nhật dữ liệu nội bộ. 4. Hệ thống tự động kích hoạt use case *Update user role*.
Post-condition	Thông tin người dùng trong hệ thống được đồng bộ chính xác với Datacore.

Use Case: Update user role	
Actors	HCMUT Datacore
Type	Non-interactive
Description	Sau khi dữ liệu người dùng được đồng bộ, hệ thống tự động xác định và cập nhật vai trò (Tutor hoặc Student) dựa trên thông tin từ HCMUT Datacore.
Trigger	Khi use case *Synchronous user data* hoàn tất.
Main Flow	1. Hệ thống nhận dữ liệu vai trò từ Datacore. 2. Xác định loại người dùng (Tutor hoặc Student). 3. Cập nhật quyền truy cập và phân quyền tương ứng trong hệ thống.
Post-condition	Người dùng được gán đúng vai trò và quyền hạn trong hệ thống Tutor Support.

Các bảng trên thể hiện rõ các use case nền giúp duy trì sự đồng bộ dữ liệu và phân quyền tự động giữa hệ thống Tutor Support và HCMUT Datacore.

3.2.2 Academic Overview Generation and Data Distribution



Hình 3.3: Use case diagram: Generate Overview and Provide Data to University Offices

Phần này mô tả nhóm use case phi tương tác liên quan đến việc tổng hợp và cung cấp dữ liệu học thuật cho các phòng ban trong trường. Hệ thống Tutor Support tự động lấy dữ liệu từ *HCMUT Datacore*, tạo báo cáo tổng quan (overview report), và gửi thông tin cần thiết đến các đơn vị liên quan như *Student Affair* và *Academic Affair*. Quy trình này diễn ra định kỳ, không yêu cầu thao tác trực tiếp từ người dùng.

Use Case: Generate Overview	
Actors	None
Type	Non-interactive
Description	Hệ thống Tutor Support tự động tổng hợp dữ liệu từ các khóa học trong systerm, bao gồm thông tin học tập, tiến độ hỗ trợ và phản hồi từ sinh viên – tutor, để tạo báo cáo tổng quan phục vụ công tác quản lý học vụ và sinh viên.
Trigger	Được kích hoạt định kỳ (ví dụ: hàng tuần hoặc theo học kỳ).
Main Flow	1. Hệ thống truy vấn dữ liệu học tập và đánh giá từ database. 2. Tổng hợp và xử lý dữ liệu thành các chỉ số thống kê, biểu đồ và báo cáo. 3. Kích hoạt hai use case con *Provide data to Student Affair* và *Provide data to Academic Affair*.
Post-condition	Báo cáo tổng quan được tạo thành công và sẵn sàng để gửi đến các phòng ban liên quan.

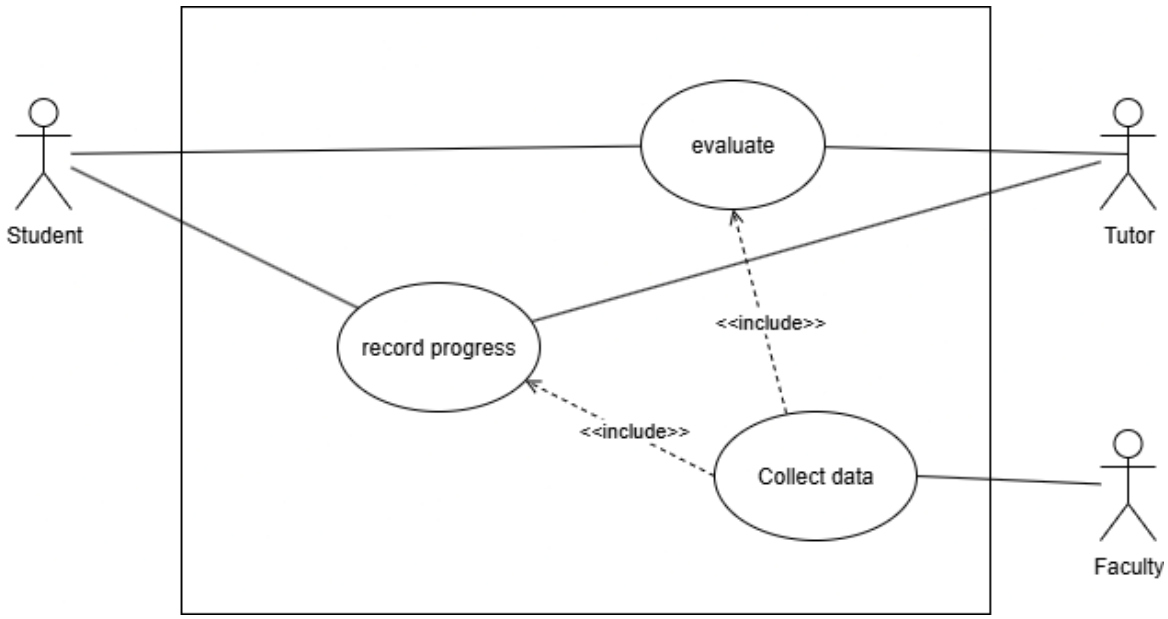
Use Case: Provide data to Student Affair	
Actors	Student Affair
Type	Non-interactive

Description	Sau khi tổng hợp dữ liệu, hệ thống tự động gửi báo cáo đến Phòng Công tác Sinh viên. Báo cáo này chứa thông tin về mức độ tham gia chương trình tutor, kết quả đánh giá, và tiến độ của sinh viên để phục vụ việc xét điểm rèn luyện, học bổng hoặc hỗ trợ khác.
Trigger	Kích hoạt bởi use case *Generate Overview*.
Main Flow	1. Hệ thống lọc dữ liệu liên quan đến hoạt động sinh viên. 2. Định dạng dữ liệu thành báo cáo phù hợp với yêu cầu của Phòng CTSV. 3. Gửi báo cáo qua kênh tích hợp nội bộ hoặc email quản lý.
Post-condition	Phòng CTSV nhận được báo cáo định kỳ để phục vụ công tác đánh giá và hỗ trợ sinh viên.

Use Case: Provide data to Academic Affair	
Actors	Academic Affair
Type	Non-interactive
Description	Hệ thống tự động gửi dữ liệu học thuật đã tổng hợp đến Phòng Đào tạo. Báo cáo bao gồm các chỉ số học tập, tiến bộ của sinh viên theo từng môn học và mức độ hiệu quả của các buổi tutor, hỗ trợ cho việc điều phối giảng viên và phân bổ tài nguyên giảng dạy.
Trigger	Kích hoạt bởi use case *Generate Overview*.
Main Flow	1. Hệ thống lọc và phân loại dữ liệu học thuật theo khoa/bộ môn. 2. Tạo báo cáo học vụ tổng hợp. 3. Gửi báo cáo đến Phòng Đào tạo để phục vụ phân tích và ra quyết định.
Post-condition	Phòng Đào tạo nhận được báo cáo cập nhật phục vụ công tác quản lý học vụ.

Các use case này minh hoạ quy trình tự động trong việc tổng hợp và phân phối thông tin học vụ trong toàn hệ thống, bảo đảm các đơn vị quản lý có được dữ liệu chính xác, kịp thời và đồng bộ từ Datacore.

3.2.3 Progress Evaluation and Academic Data Collection



Hình 3.4: Use case diagram: Student Progress Recording and Evaluation Data Collection

Phần này mô tả nhóm use case liên quan đến việc theo dõi tiến độ học tập, đánh giá kết quả và thu thập dữ liệu học vụ phục vụ cho công tác giảng dạy và quản lý. Các hoạt động được thực hiện bởi *Student*, *Tutor* và *Faculty*, trong đó hệ thống đảm nhiệm vai trò trung gian — ghi nhận tiến độ, tổng hợp dữ liệu và đồng bộ hóa kết quả đánh giá trong cơ sở dữ liệu học vụ.

Use Case: Record Progress	
Actors	Student, Tutor
Type	Interactive (triggering background data collection)
Description	Student và Tutor có thể ghi nhận tiến độ học tập sau mỗi buổi học hoặc theo giai đoạn học tập. Hệ thống tự động lưu trữ thông tin này và kích hoạt quy trình thu thập dữ liệu học vụ để đảm bảo tính cập nhật của hồ sơ sinh viên.
Trigger	Khi Student hoặc Tutor nhập thông tin về tiến độ học tập.
Main Flow	1. Student hoặc Tutor truy cập phần "Learning Progress". 2. Nhập thông tin về mức độ hoàn thành, nội dung đã học, hoặc kết quả đạt được. 3. Hệ thống lưu dữ liệu tiến độ và kích hoạt use case *Collect Data* để tổng hợp. 4. Dữ liệu tiến độ được hiển thị trong trang hồ sơ học tập của sinh viên.
Post-condition	Tiến độ học tập được cập nhật vào cơ sở dữ liệu; dữ liệu học vụ liên quan được thu thập để phục vụ báo cáo và đánh giá.

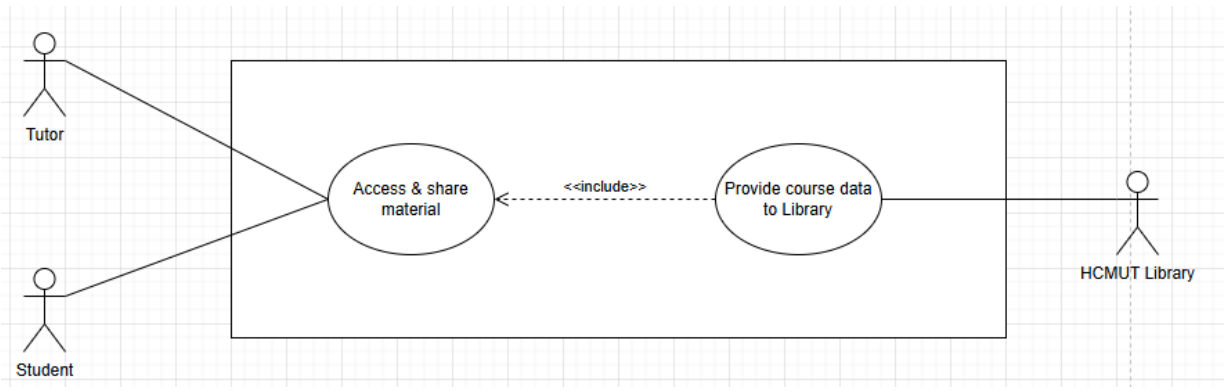
Use Case: Evaluate	
Actors	Tutor
Type	Interactive (triggering background data processing)

Description	Tutor thực hiện đánh giá định kỳ hoặc sau buổi học để ghi nhận sự tiến bộ của sinh viên. Quá trình đánh giá này được lưu vào hệ thống và tự động kích hoạt quá trình thu thập dữ liệu tổng hợp học vụ.
Trigger	Khi Tutor hoàn thành việc ghi nhận đánh giá học tập của sinh viên.
Main Flow	1. Tutor truy cập danh sách sinh viên được hướng dẫn. 2. Chọn sinh viên hoặc session cần đánh giá. 3. Nhập nội dung đánh giá (kỹ năng, thái độ, kết quả đạt được, đề xuất hỗ trợ). 4. Hệ thống lưu dữ liệu và kích hoạt use case *Collect Data*.
Post-condition	Dữ liệu đánh giá được lưu trữ thành công và sẵn sàng cho việc tổng hợp học vụ.

Use Case: Collect Data	
Actors	Faculty
Type	Non-interactive
Description	Hệ thống tự động thu thập dữ liệu từ các bản ghi tiến độ và đánh giá của Student và Tutor. Mục tiêu là tổng hợp thông tin về quá trình học tập để hỗ trợ công tác theo dõi, phân tích và đánh giá chất lượng đào tạo của *Tutor*.
Trigger	Kích hoạt bởi use case *Record Progress* hoặc *Evaluate*.
Main Flow	1. Hệ thống trích xuất dữ liệu học tập và đánh giá mới nhất từ cơ sở dữ liệu. 2. Chuẩn hóa và tổng hợp thông tin thành báo cáo học vụ. 3. Cập nhật hồ sơ sinh viên và gửi thông tin tổng hợp cho Faculty khi cần. 4. Lưu trữ dữ liệu trong kho dữ liệu học vụ (Academic Data Repository).
Post-condition	Faculty có thể truy cập báo cáo học vụ và thống kê chi tiết về tiến độ học tập của sinh viên.

Nhóm use case này giúp hệ thống Tutor Support theo dõi toàn diện quá trình học tập của sinh viên, từ việc ghi nhận tiến độ cá nhân đến đánh giá kết quả và tổng hợp báo cáo học vụ. Nhờ cơ chế tự động thu thập và tổng hợp dữ liệu, *Faculty* có thể nắm bắt tình hình học tập của sinh viên một cách nhanh chóng, chính xác và toàn diện.

3.2.4 Learning Material Access and Synchronization



Hình 3.5: Use case diagram: Access and Synchronize Learning Materials with HCMUT Library

Phần này mô tả nhóm use case liên quan đến việc truy cập, chia sẻ và đồng bộ tài nguyên học tập giữa hệ thống Tutor Support và *HCMUT Library*. Chức năng này cho phép *Tutor* và *Student* sử dụng tài liệu học tập chính thống, đồng thời hệ thống đảm bảo tính nhất quán của dữ liệu học phần và tài nguyên học tập trong toàn trường.

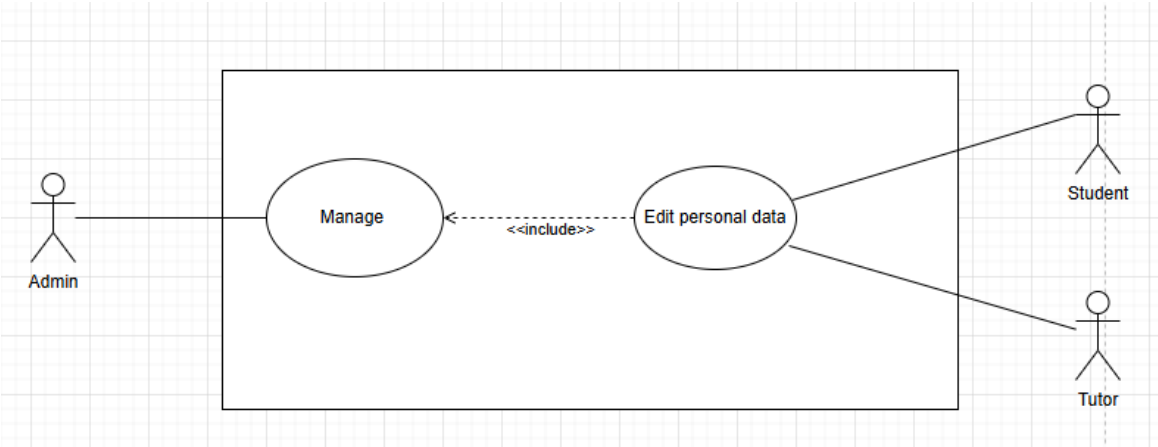
Use Case: Access & Share Material	
Actors	Tutor, Student
Type	Interactive (triggering background synchronization)
Description	Tutor và Student có thể truy cập, tải xuống hoặc chia sẻ tài liệu học tập liên quan đến các buổi tutor. Hệ thống đảm bảo rằng chỉ những tài nguyên được xác thực và đồng bộ từ thư viện chính thức mới được phép sử dụng.
Trigger	Khi người dùng truy cập mục "Learning Materials" trong hệ thống.
Main Flow	1. Tutor hoặc Student đăng nhập và truy cập trang tài nguyên học tập. 2. Hệ thống truy vấn và hiển thị các tài liệu học tập đã được đồng bộ từ thư viện. 3. Người dùng có thể chia sẻ hoặc tải xuống tài liệu học tập hợp lệ. 4. Nếu có thay đổi hoặc bổ sung, hệ thống sẽ tự động kích hoạt use case *Provide Course Data to Library* để cập nhật đồng bộ.
Post-condition	Người dùng truy cập và chia sẻ tài liệu hợp lệ; tài liệu học tập được đảm bảo đồng bộ với thư viện của trường.

Use Case: Provide Course Data to Library	
Actors	HCMUT Library
Type	Non-interactive
Description	Hệ thống tự động gửi dữ liệu học phần (course data) và thông tin tài nguyên học tập đến HCMUT Library để đảm bảo tài nguyên được cập nhật và liên kết chính xác với các môn học. Quá trình này diễn ra hoàn toàn trong nền, không có sự can thiệp của người dùng.
Trigger	Được kích hoạt tự động khi có cập nhật tài liệu học tập trong use case *Access & Share Material.*

Main Flow	1. Hệ thống trích xuất dữ liệu học phần và tài liệu liên quan trong cơ sở dữ liệu nội bộ. 2. Gửi dữ liệu cập nhật đến HCMUT Library qua API hoặc dịch vụ đồng bộ. 3. Nhận phản hồi xác nhận từ thư viện về việc cập nhật thành công.
Post-condition	HCMUT Library nhận và cập nhật thành công dữ liệu học phần, đảm bảo tài nguyên học tập trong hệ thống luôn chính thống và đồng bộ.

Nhóm use case này giúp hệ thống Tutor Support tích hợp chặt chẽ với thư viện của trường, duy trì **nguồn tài liệu học tập hợp lệ, đồng bộ và đáng tin cậy** cho cả giảng viên và sinh viên. Toàn bộ quá trình đồng bộ được thực hiện **tự động bởi hệ thống**, không yêu cầu actor trung gian như HCMUT Datacore, phù hợp với mô hình tích hợp thư viện được mô tả trong đặc tả hệ thống.

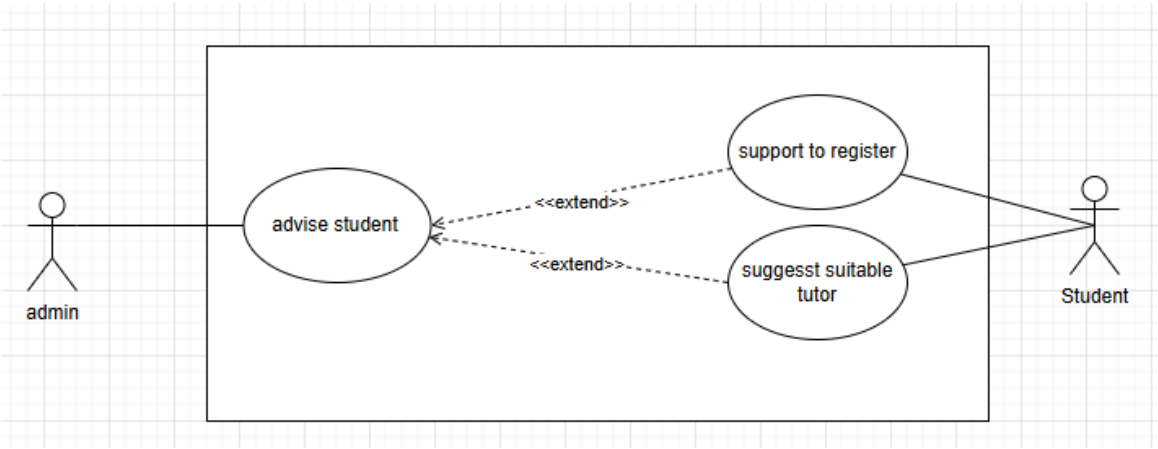
3.2.5 Admin manage profile user



Hình 3.6: Use case diagram: Manage

Use Case: Edit personal data	
Actors	Admin
Type	Interactive (triggering background synchronization)
Description	Admin truy cập vào dữ liệu và sửa thông tin của Student, Tutor như Tên, MSSV, chuyên môn,....
Trigger	Khi Admin tìm kiếm trên công cụ tìm kiếm và ấn vào tùy chọn chỉnh sửa
Main Flow	1. Admin truy cập và đăng nhập vào tài khoản có vai trò Admin. 2. Admin tìm kiếm thông tin của Sinh Viên, Tutor muốn chỉnh sửa. 3. Hệ thống hiển thị thông tin của người dùng mà Admin đã tìm kiếm. 4. Admin tiến hành chỉnh sửa, hệ thống sẽ tự động kích hoạt use case *Synchronous user data* để cập nhật đồng bộ.
Post-condition	Thông tin Sinh viên, Tutor được cập nhật và đồng bộ hóa trên HCMUT_DATACORE.

3.2.6 System Suggest Tutor for Student



Hình 3.7: Use case diagram: System Suggest

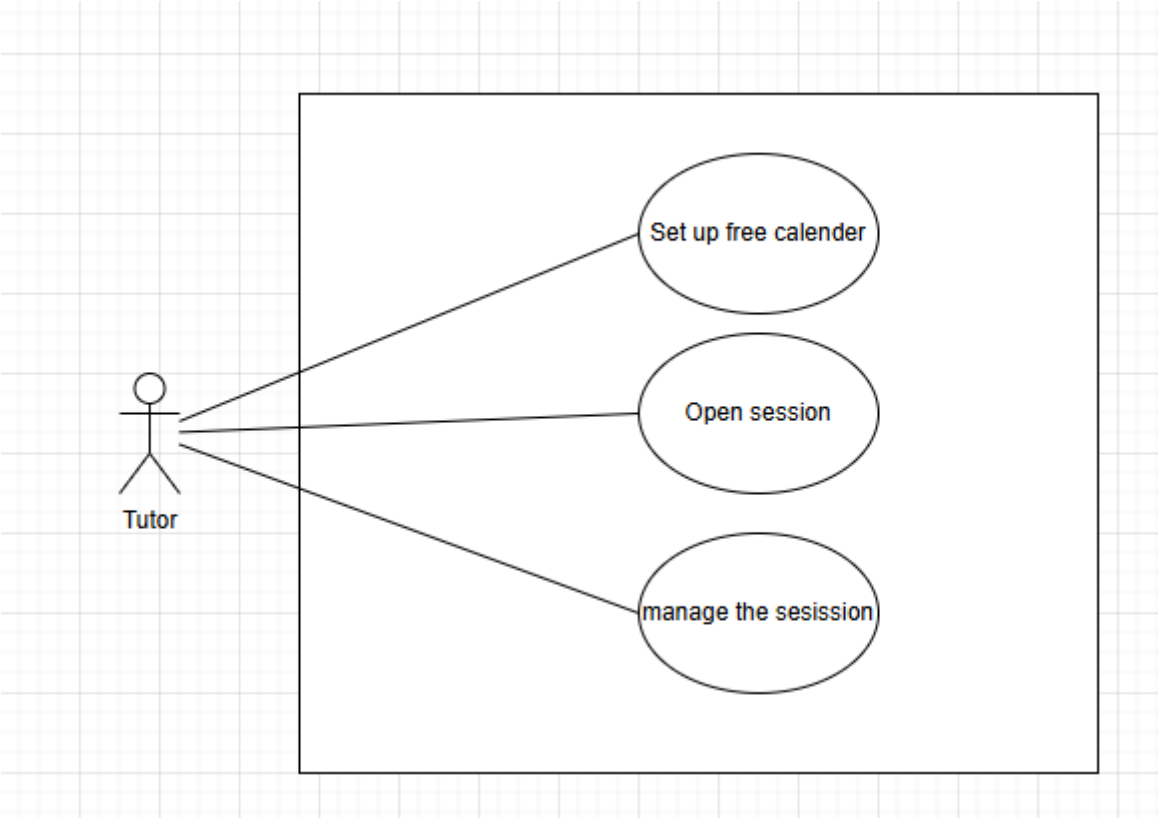
Use Case: Advise student	
Actors	System
Type	Interactive (triggering background synchronization)
Description	Sinh viên gửi yêu cầu hỗ trợ và được hệ thống gửi thông tin hỗ trợ
Trigger	Khi Sinh viên ấn vào tùy chọn "hỗ trợ"
Main Flow	1. Sinh viên đăng nhập vào tài khoản có vai trò Sinh Viên. 2. Sinh viên nhập tùy chọn "hỗ trợ". 3. Hệ thống sẽ gửi danh sách các câu hỏi thường gặp và câu trả lời cho sinh viên.
Post-condition	Không.

Use Case: Suggest suitable tutor	
Actors	System
Type	Non-Interactive
Description	Hệ thống gợi ý tutor phù hợp dựa trên thông tin sinh viên (chuyên ngành, định hướng,...)
Trigger	Sinh viên chưa đăng ký Tutor
Main Flow	1. Sinh viên đăng nhập vào tài khoản có vai trò Sinh Viên. 2. Hệ thống hiển thị danh sách các Tutor phù hợp với Sinh viên.
Post-condition	Không.

Use Case: Support to register	
Actors	System,Student
Type	Interactive
Description	Hệ thống hỗ trợ sinh viên đăng ký Tutor phù hợp

Trigger	Sinh viên chưa đăng ký Tutor
Main Flow	1. Sinh viên đăng nhập vào tài khoản có vai trò Sinh Viên. 2. Sinh viên lựa chọn Tutor (dựa trên danh sách gợi ý hoặc tự tìm kiếm). 3. Sinh viên xác nhận đăng ký với hệ thống . 4. Hệ thống hiển thị đăng ký thành công.
Post-condition	Hệ thống đồng bộ hóa với HCMUT_DATACORE.

3.2.7 Tutor manage calender



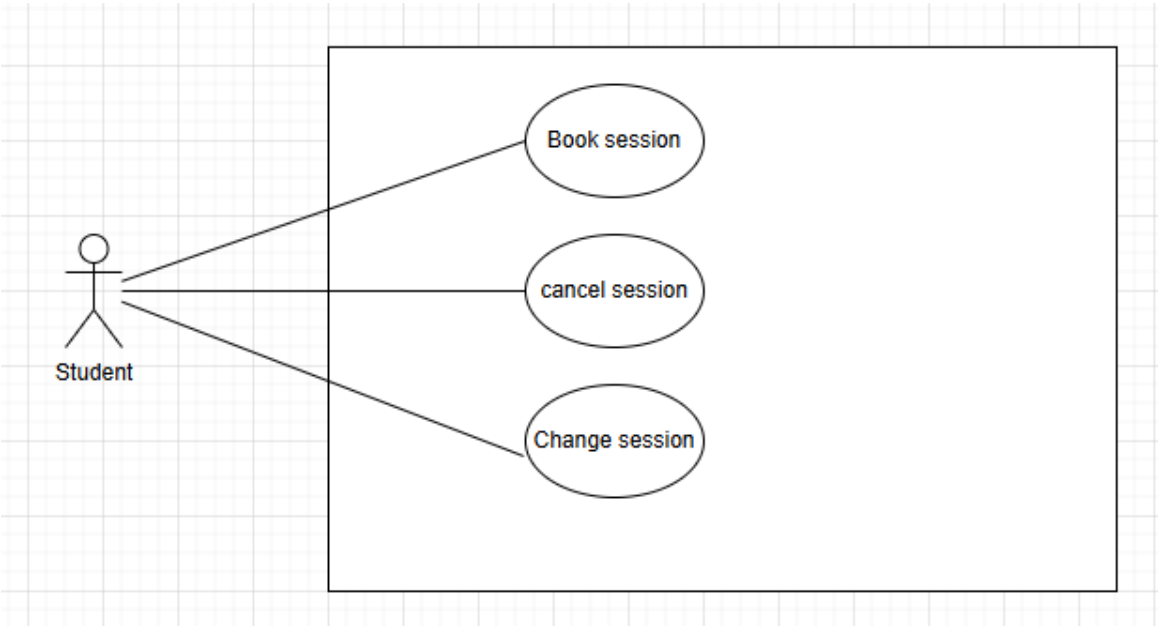
Hình 3.8: Use case diagram: Tutor manage

Use Case: Set up free calender	
Actors	Tutor
Type	Interactive
Description	Tutor đặt lịch rảnh để System có thể lập thời khóa biểu cho sinh viên
Trigger	Tutor ấn vào mục "xếp lịch"
Main Flow	1. Tutor đăng nhập vào tài khoản có vai trò Tutor. 2. Tutor chọn vào mục "xếp lịch". 3. System hiển thị lịch để Tutor chọn thời gian rảnh . 4. Tutor Submit lịch rảnh cho hệ thống. 5. System hiện thông báo " Submit thành công"
Post-condition	Hệ thống gửi lịch của Tutor cho HCMUT_DATACORE.

Use Case:Open session	
Actors	Tutor
Type	Interactive
Description	Tutor chọn buổi rảnh để có thể trao đổi với sinh viên
Trigger	Tutor ấn vào mục "Open Session"
Main Flow	1. Tutor đăng nhập vào tài khoản có vai trò Tutor. 2. Tutor chọn vào mục "Open Session". 3. System hiển thị lịch để Tutor chọn thời gian rảnh và phòng rảnh vào thời gian đó. 4. Tutor Submit lịch rảnh cho hệ thống. 5. System hiện thông báo " Submit thành công"
Post-condition	Hệ thống gửi lịch cho Sinh Viên đã đăng ký Tutor

Use Case:Manage session	
Actors	Tutor
Type	Interactive
Description	Tutor chỉnh sửa lịch trao đổi với sinh viên
Trigger	Tutor ấn vào mục "Manage Session"
Main Flow	1. Tutor đăng nhập vào tài khoản có vai trò Tutor. 2. Tutor chọn vào mục "Manage Session". 3. System hiển thị danh sách các lịch mà Tutor đã đăng ký. 4. Tutor chỉnh sửa (Xóa, đổi phòng, đổi giờ) 5. Tutor submit cho hệ thống
Post-condition	Hệ thống gửi email thông báo cho Sinh Viên đã đăng ký Tutor

3.2.8 Student enroll session



Hình 3.9: Use case diagram: Student enroll session

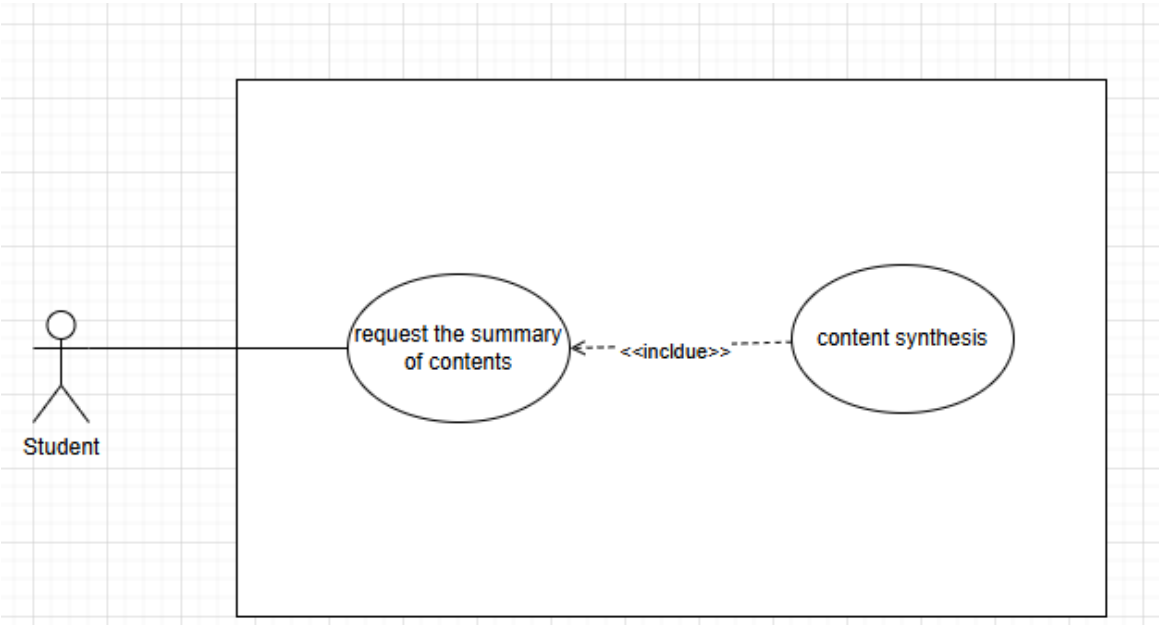
Use Case:Book session	
Actors	Sinh Viên
Type	Interactive
Description	Sinh viên đăng ký tham gia Session mà Tutor đã chọn
Trigger	Tutor chọn lịch cho session
Main Flow	1. Sinh viên đăng nhập với vai trò sinh viên. 2. Sinh viên chọn "Enroll" tương ứng với session sinh viên muốn đăng ký. 3. System hiển thị thông tin chi tiết buổi session đó (Thời gian, địa điểm). 4. Sinh viên xác nhận tham gia 5. Hệ thống thông báo "Đăng ký thành công"
Post-condition	Hệ thống gửi email thông báo cho Sinh Viên xác nhận đăng ký thành công

Use Case:cancel session	
Actors	Sinh Viên
Type	Interactive
Description	Sinh viên hủy tham gia session
Trigger	Sinh viên chọn vào lịch đã đăng ký và ấn "Hủy"

Main Flow	1. Sinh viên đăng nhập với vai trò sinh viên. 2. Sinh viên chọn vào lịch session đã đăng ký và chọn hủy. 3. System hiện thông báo xác nhận hủy. 4. Sinh viên xác nhận hủy 5. Hệ thống thông báo "hủy thành công"
Post-condition	Hệ thống gửi email thông báo cho Sinh Viên xác nhận hủy thành công

Use Case:Change session	
Actors	Sinh Viên
Type	Interactive
Description	Sinh viên đổi session đã tham gia thành session khác
Trigger	Sinh viên chọn vào lịch đã đăng ký và ấn "Đổi lịch"
Main Flow	1. Sinh viên đăng nhập với vai trò sinh viên. 2. Sinh viên chọn vào lịch session đã đăng ký và chọn "đổi lịch". 3. System hiện các lịch session khác của Tutor 4. Sinh viên chọn lịch muốn đổi 5. Hệ thống thông báo "Đổi thành công"
Post-condition	Hệ thống gửi email thông báo cho Sinh Viên xác nhận đổi thành công

3.2.9 Student Request Summary



Hình 3.10: Use case diagram: Request Summary

Use Case:Request summary	
Actors	Sinh Viên

Type	Interactive
Description	Sinh viên yêu cầu bản tóm tắt của session đã diễn ra
Trigger	Sinh viên chọn vào buổi session đã diễn ra và ấn vào "Summary"
Main Flow	1. Sinh viên đăng nhập với vai trò sinh viên. 2. Sinh viên chọn vào lịch session đã diễn ra và chọn "Summary". 3. System gửi summary cho sinh viên
Post-condition	Không

CHƯƠNG 4

SEQUENCE AND ACTIVITY DIAGRAM

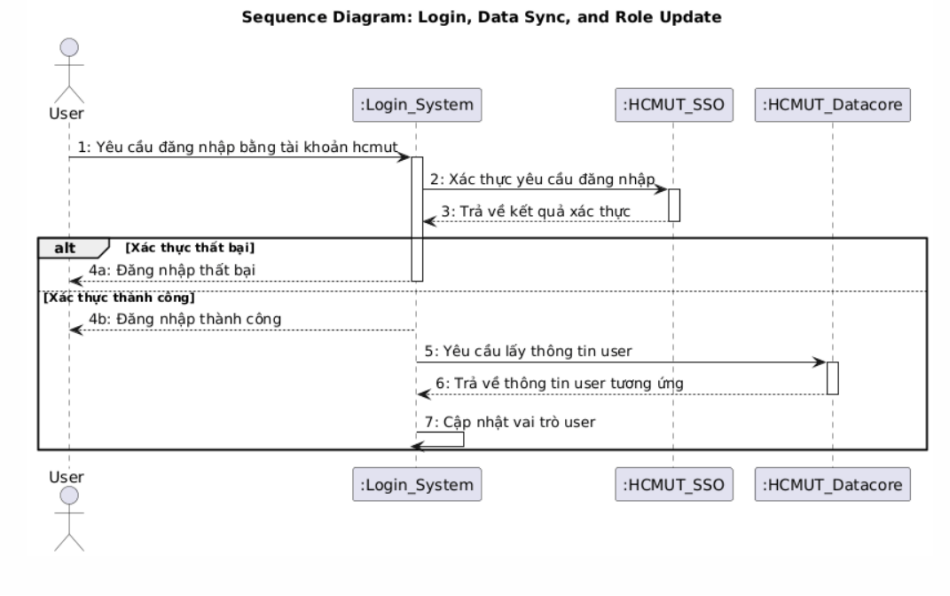
4.1 User Login and Synchronization

Sơ đồ mô tả quy trình đồng bộ và cập nhật vai trò người dùng tự động, được kích hoạt (trigger) ngay tại thời điểm người dùng đăng nhập. Sơ đồ cho thấy một quy trình rõ ràng, tích hợp chặt chẽ giữa hệ thống **Login System** và các hệ thống bên ngoài là **HCMUT_SSO** (xác thực) và **HCMUT_Datacore** (cơ sở dữ liệu lõi).

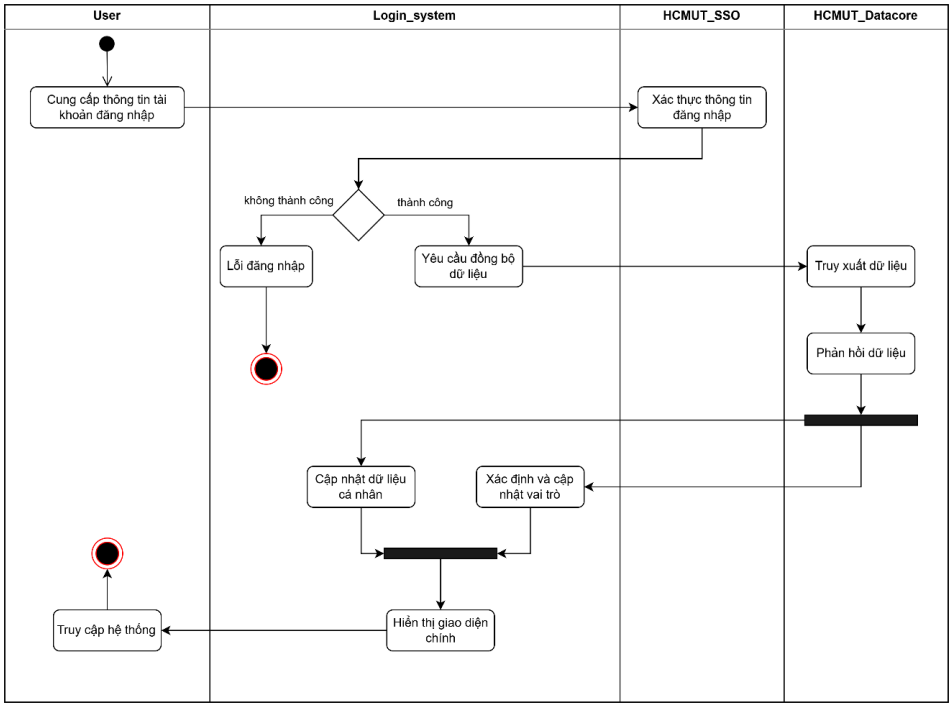
Khi người dùng gửi yêu cầu đăng nhập đến hệ thống Login bằng việc chọn vào nút "Đăng nhập" và cung cấp tài khoản hcmut, hệ thống Login lập tức ủy quyền xác thực bằng cách gửi thông điệp xác thực đăng nhập cho hệ thống HCMUT_SSO. Nếu xác thực thất bại, HCMUT_SSO trả về tín hiệu và hệ thống Login hiển thị thông báo lỗi, luồng xử lí dừng lại. Trong trường hợp xác thực thành công, HCMUT_SSO trả về tín hiệu thành công và hệ thống Login thông báo thành công cho người dùng, sau đó tự kích hoạt các hoạt động nền phía sau.

Sau khi yêu cầu đăng nhập của người dùng xác thực thành công, hệ thống Login sẽ gửi yêu cầu để truy xuất thông tin người dùng đến hệ thống HCMUT_Datacore nhằm đồng bộ dữ liệu người dùng. Sau khi nhận được phản hồi chứa thông tin và vai trò của người dùng, hệ thống Login sẽ tự thực hiện cập nhật vai trò và hiển thị giao diện tương ứng cho người dùng.

Sơ đồ mô tả một kiến trúc tích hợp, nơi Login_system đóng vai trò điều phối. Nó cho thấy rõ sự phụ thuộc và thứ tự thời gian nghiêm ngặt: **Đăng nhập** → **Đồng bộ** → **Cập nhật** được thực hiện tuần tự như một giao dịch tự động sau khi đăng nhập thành công.

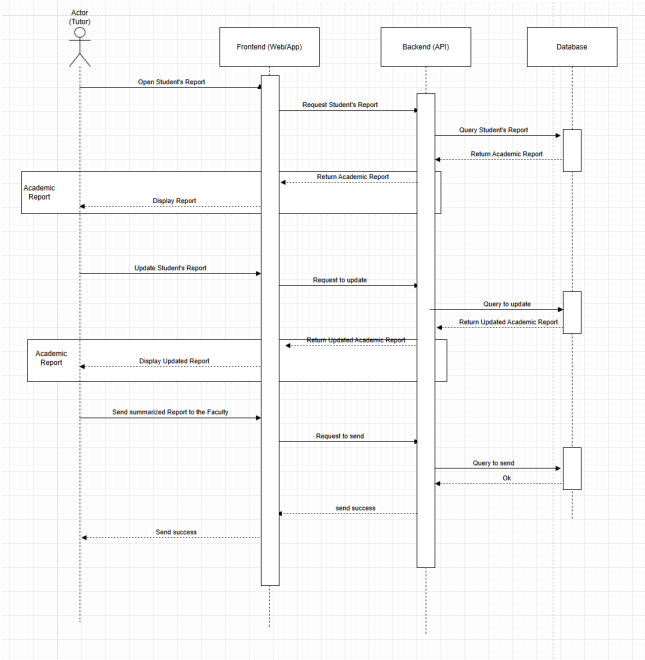


Hình 4.1: User Login - Sequence Diagram

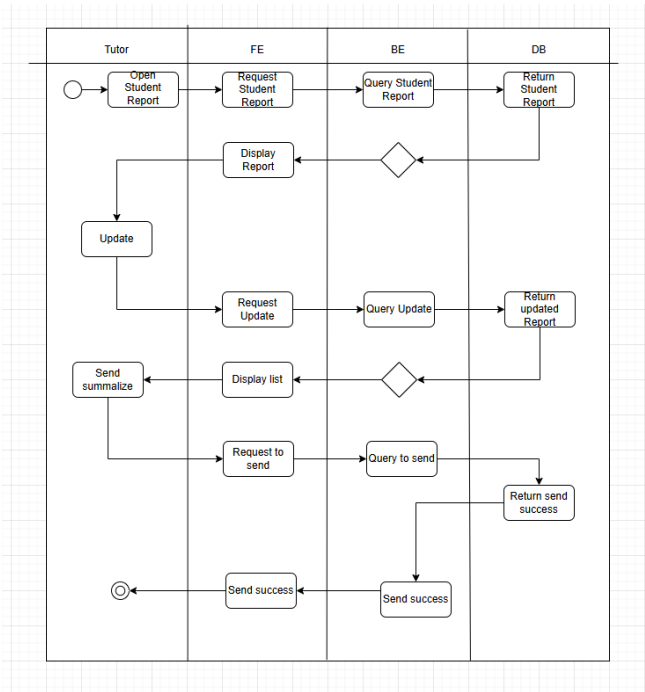


Hình 4.2: User Login - Activity Diagram

4.2 Academic Overview Generation and Data Distribution



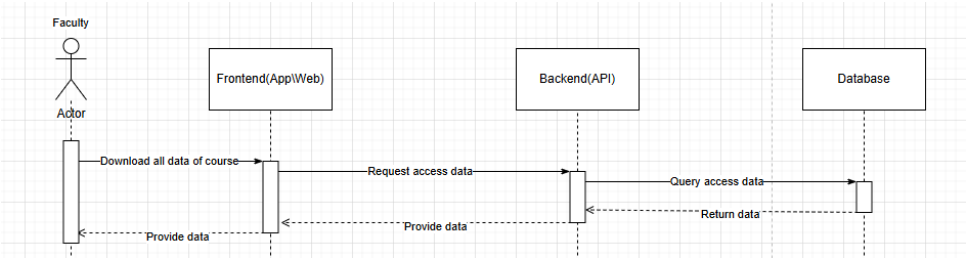
Hình 4.3: Tutor evaluate - Sequence Diagram



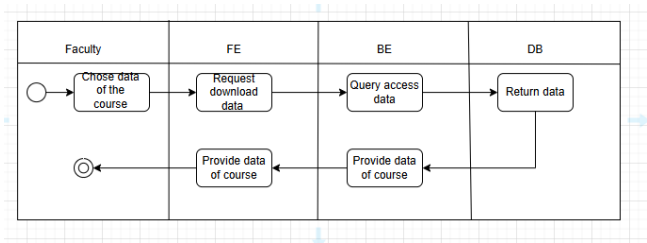
Hình 4.4: Tutor evaluate - Activity Diagram

4.3 Progress Evaluation and Academic Data Collection

Faculty collect data

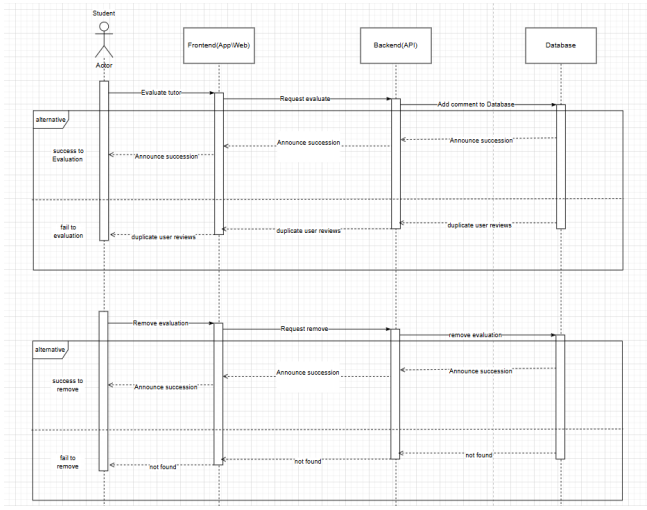


Hình 4.5: Faculty collect data - Sequence Diagram

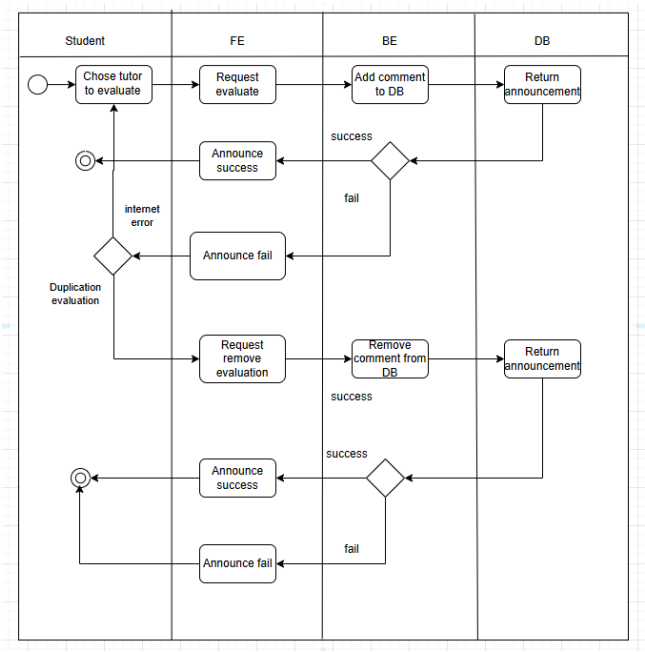


Hình 4.6: Faculty collect data - Activity Diagram

Student evaluate

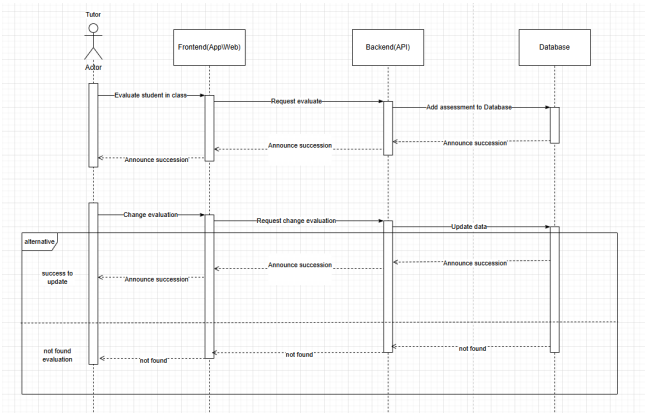


Hình 4.7: Student evaluate - Sequence Diagram

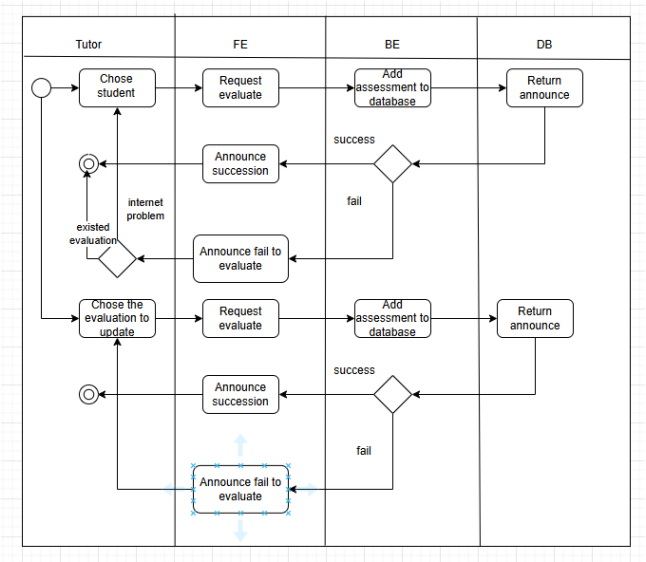


Hình 4.8: Student evaluate - Activity Diagram

Tutor evaluate

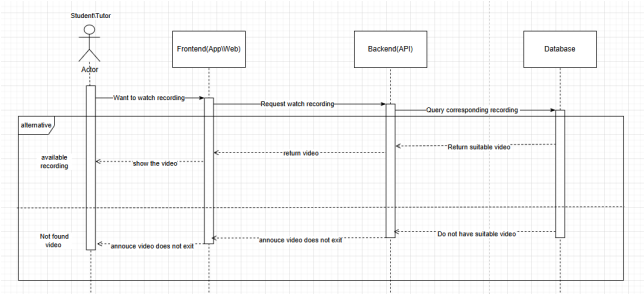


Hình 4.9: Tutor evaluate - Sequence Diagram

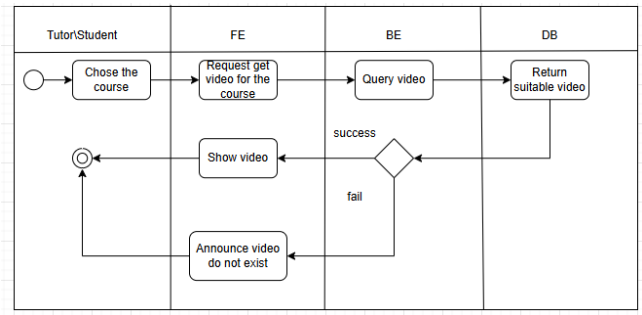


Hình 4.10: Tutor evaluate - Activity Diagram

Student and Tutor get record



Hình 4.11: Student and Tutor get record - Sequence Diagram



Hình 4.12: Student and Tutor get reocrd - Activity Diagram

4.4 Learning Material Access

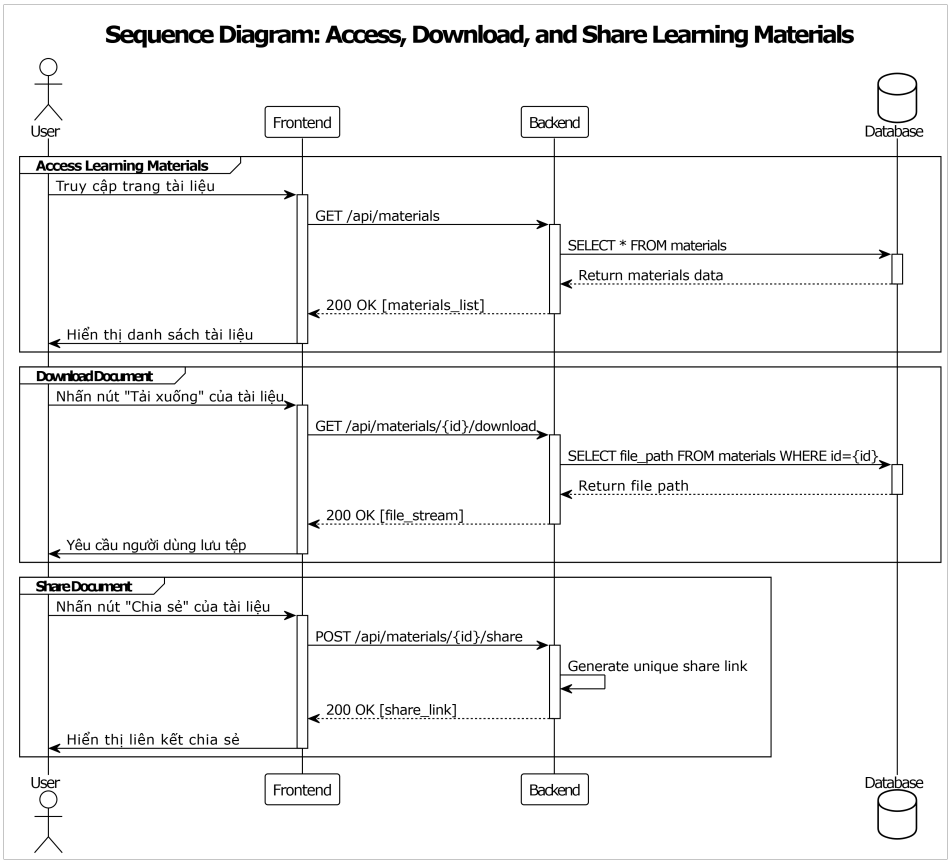
Các usecase này cho phép User được truy cập các tài liệu chính thống liên kết với HCMUT Library, đồng thời đảm bảo tính nhất quán của tài nguyên học tập trên toàn trường.

4.4.1 Access Learning Material

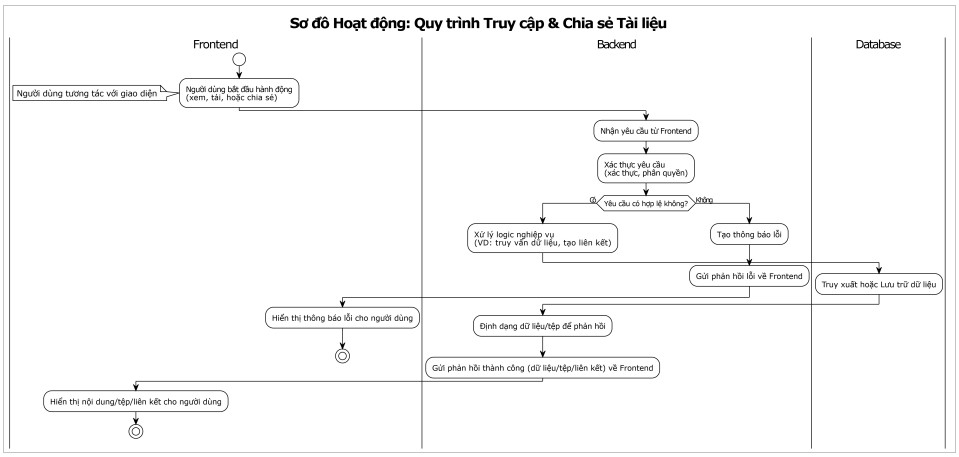
Use case này cho phép **User** (bao gồm Tutor và Student) truy cập, tải xuống và chia sẻ các tài liệu học tập được cung cấp trong khuôn khổ môn học. Quá trình bắt đầu khi User đăng nhập thành công vào hệ thống. Sau khi được xác thực, **Backend** sẽ kiểm tra vai trò và các quyền hạn liên quan của người dùng để xác định danh sách các tài liệu mà họ được phép truy cập, nhằm đảm bảo tính bảo mật và phân quyền dữ liệu một cách hợp lệ.

Sau khi đăng nhập thành công, User chọn mục *"Library"* trên giao diện hệ thống. Hệ thống sẽ hiển thị danh sách các tài liệu hiện có, được truy xuất bởi **Backend** từ **Database**. Tại đây, User có thể thực hiện một trong các hành động sau: nhấn vào tài liệu để xem trực tiếp, chọn *"Tải xuống"* để lưu tệp về thiết bị, hoặc chọn *"Chia sẻ"* để tạo một liên kết truy cập. **Backend** sẽ tiếp nhận yêu cầu, xử lý logic tương ứng (truy xuất tệp từ nơi lưu trữ hoặc tạo liên kết) và trả kết quả về cho giao diện. Cuối cùng, hệ thống sẽ hiển thị nội dung tài liệu, mở hộp thoại lưu tệp, hoặc hiển thị liên kết chia sẻ cho người dùng.

Use case này đóng vai trò cốt lõi trong việc phân phối tài nguyên học thuật của hệ thống. Nó đảm bảo rằng cả người dạy và người học đều có thể tiếp cận các tài liệu cần thiết một cách kịp thời và thuận tiện, từ đó hỗ trợ trực tiếp cho quá trình giảng dạy, học tập và nghiên cứu.



Hình 4.13: Access Learning Material - Sequence Diagram



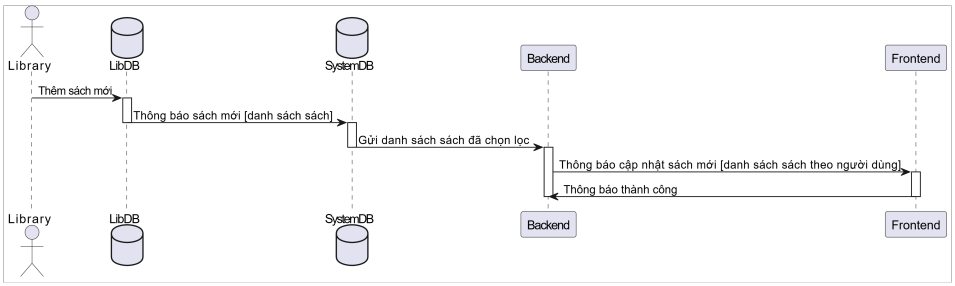
Hình 4.14: Access Learning Material - Activity Diagram

4.4.2 Provide Course Data

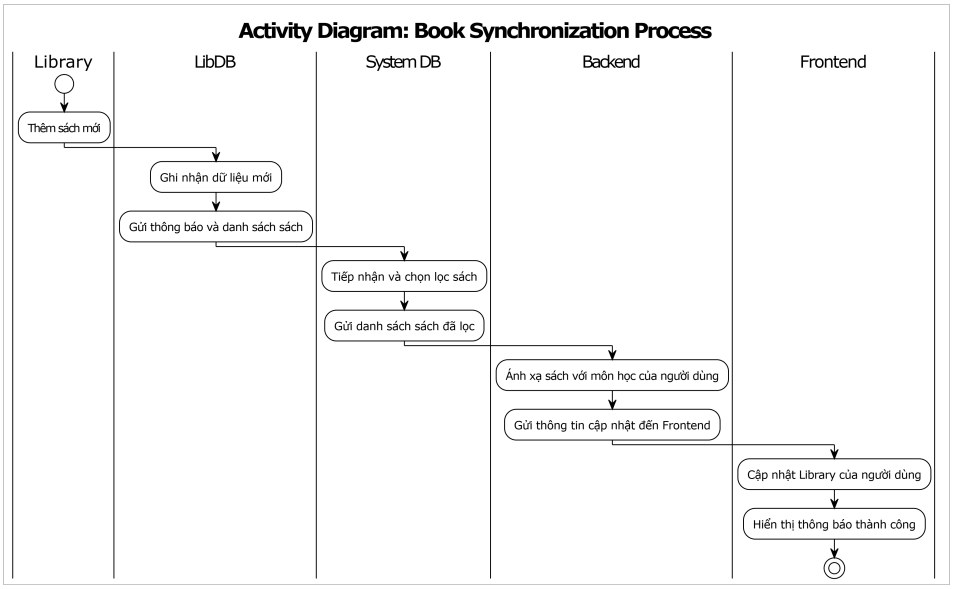
Use case này mô tả quy trình đồng bộ hóa dữ liệu tự động nhằm cập nhật các tài liệu học tập mới từ thư viện trung tâm đến người dùng cuối. Quá trình bắt đầu khi **Library** (bộ phận quản lý thư viện) thêm một hoặc nhiều sách mới vào cơ sở dữ liệu riêng của mình, gọi là **LibDB**. Hành động này sẽ kích hoạt một chuỗi xử lý tự động, khởi đầu bằng việc **LibDB** gửi thông báo kèm danh sách các sách mới này đến cơ sở dữ liệu của hệ thống trung gian, **System DB**.

Sau khi nhận được thông tin, **System DB** tiến hành tiếp nhận và chọn lọc các đầu sách phù hợp, sau đó gửi danh sách đã được tinh lọc này cho **Backend**. Tại đây, **Backend** thực hiện logic nghiệp vụ cốt lõi: ánh xạ từng cuốn sách với các môn học liên quan và từ đó xác định những người dùng đang tham gia các môn học đó. Dựa trên kết quả ánh xạ, **Backend** gửi thông tin cập nhật được cá nhân hóa đến **Frontend** của từng người dùng tương ứng. Cuối cùng, giao diện người dùng (**Frontend**) sẽ tự động cập nhật mục thư viện cá nhân và hiển thị thông báo *"Cập nhật thành công"* để báo cho người dùng biết về các tài liệu mới.

Use case này đóng vai trò là một cơ chế đồng bộ hóa tự động, liền mạch giữa các hệ thống. Nó đảm bảo rằng người dùng luôn được cập nhật những tài nguyên học tập mới nhất và phù hợp nhất với môn học của mình mà không cần phải tìm kiếm thủ công, từ đó nâng cao trải nghiệm và hỗ trợ tích cực cho quá trình học tập.



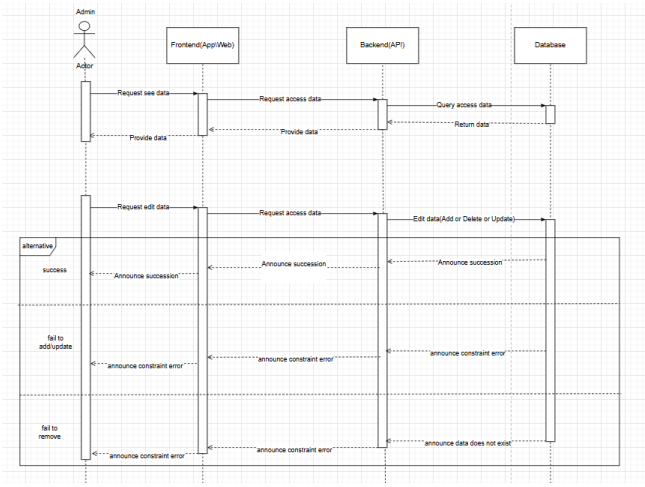
Hình 4.15: Provide Course Data from Library - Sequence Diagram



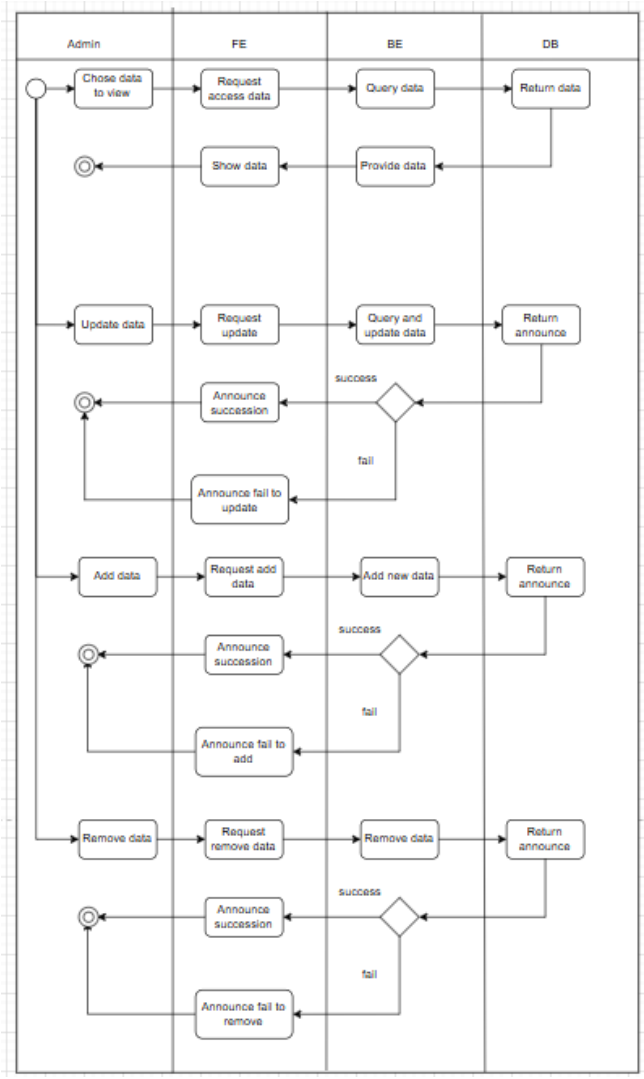
Hình 4.16: Provide Course Data from Library - Sequence Diagram

4.5 Admin Management

Admin manage

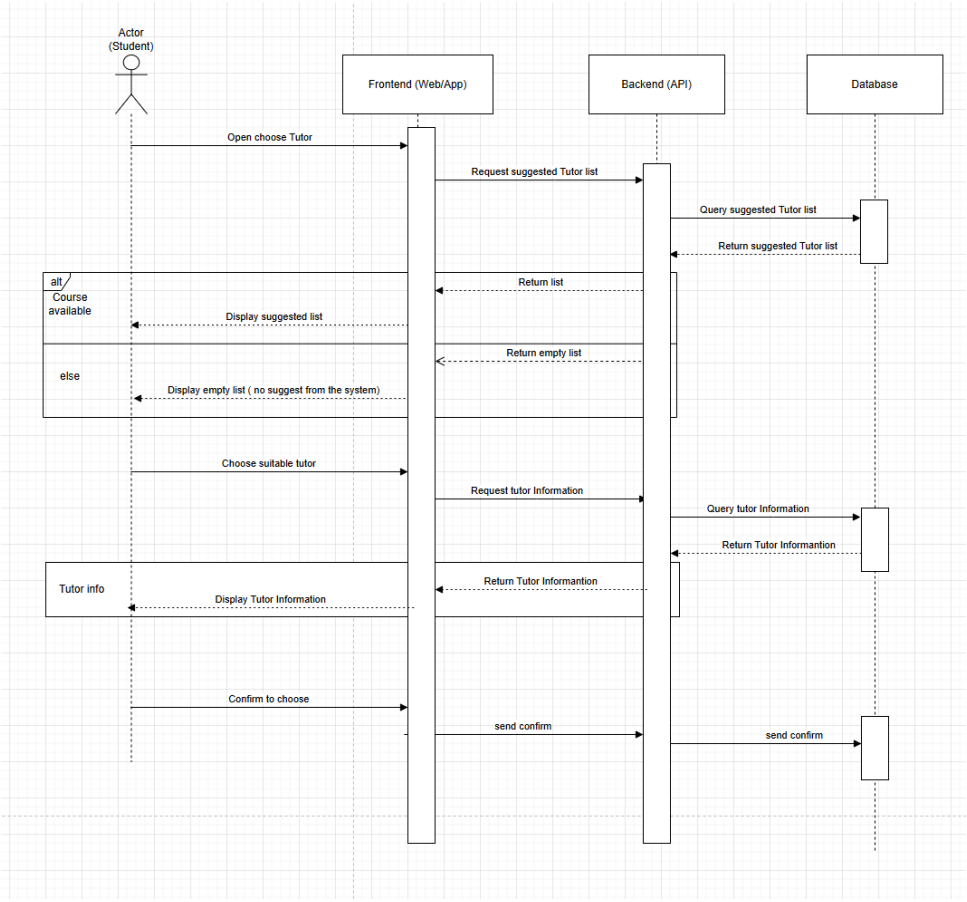


Hình 4.17: Admin manage - Sequence Diagram

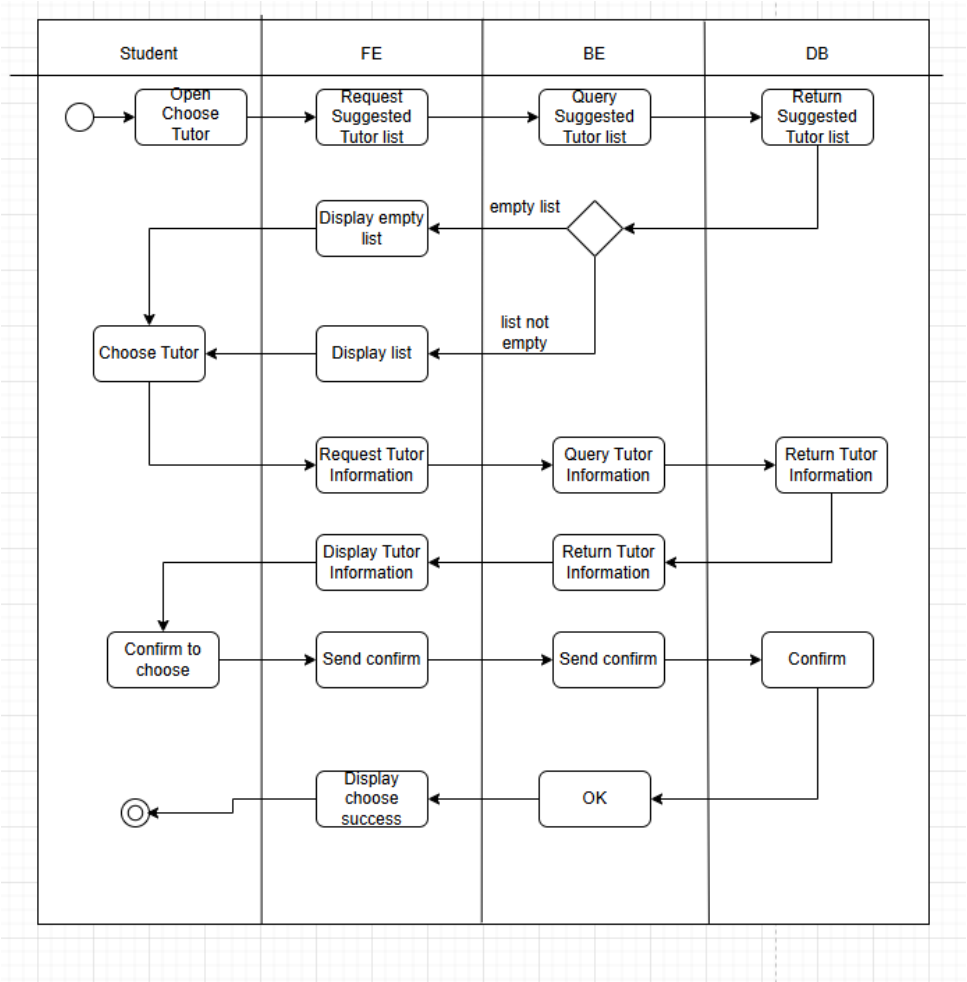


Hình 4.18: Admin manage - Activity Diagram

4.6 System Suggest Tutor for Student



Hình 4.19: Provide Course Data from Library - Sequence Diagram



Hình 4.20: Provide Course Data from Library - Sequence Diagram

4.7 Tutor's Calendar Management

Phần này mô tả các chức năng chính cho phép **Tutor** quản lý lịch giảng dạy và tổ chức các buổi học với sinh viên. Bao gồm ba use case chính: *Register Teaching Schedule*, *Open Session*, và *Manage Session*. Các sơ đồ Sequence và Activity dưới đây minh họa chi tiết quy trình xử lý và tương tác giữa các thành phần **Frontend**, **Backend**, **HCMUT_SSO**, **HCMUT_DATACORE**, cũng như các tác nhân **Tutor** và **Student**.

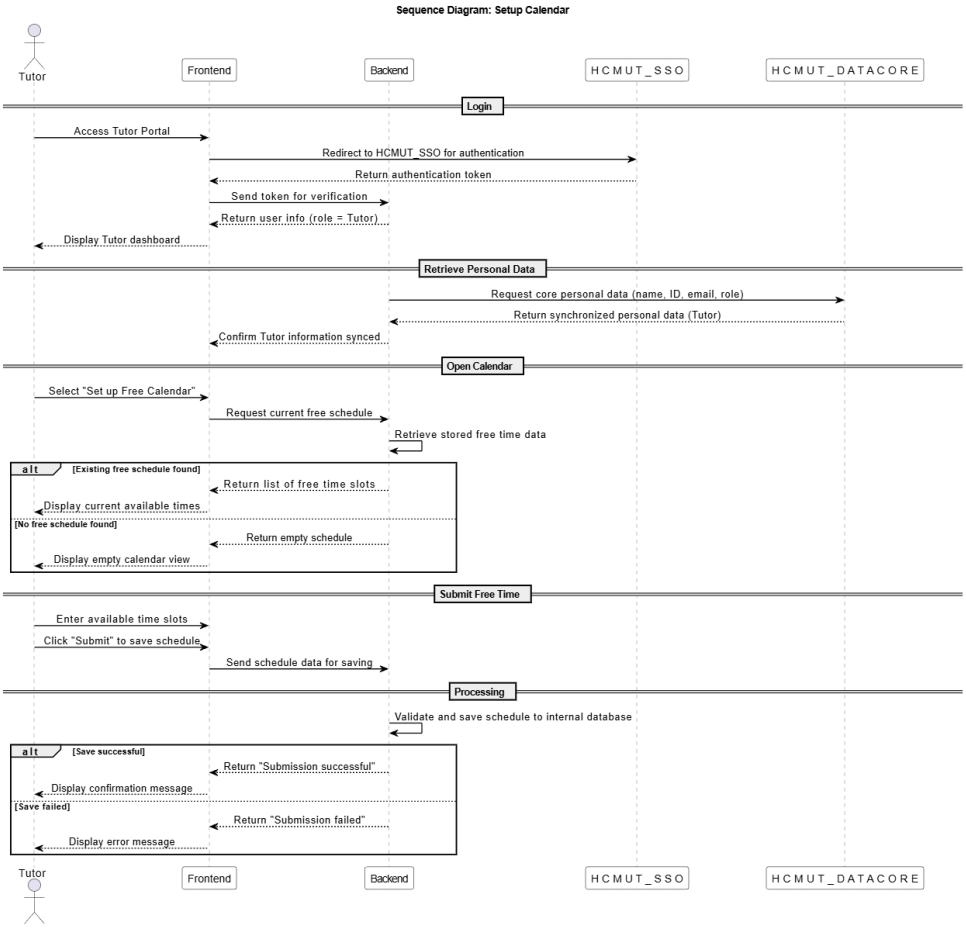
4.7.1 Register Teaching Schedule

Use case này cho phép **Tutor** đăng ký các khoảng thời gian rảnh để hệ thống có thể sử dụng trong việc lập thời khóa biểu cho sinh viên sau này. Quá trình bắt đầu khi Tutor đăng nhập vào hệ thống thông qua dịch vụ xác thực tập trung **HCMUT_SSO**. Sau khi được xác thực, **Backend** truy xuất thông tin cá nhân của Tutor (họ tên, mã số, email, vai trò, ...) từ **HCMUT_DATACORE** nhằm đảm bảo tính chính xác và nhất quán.

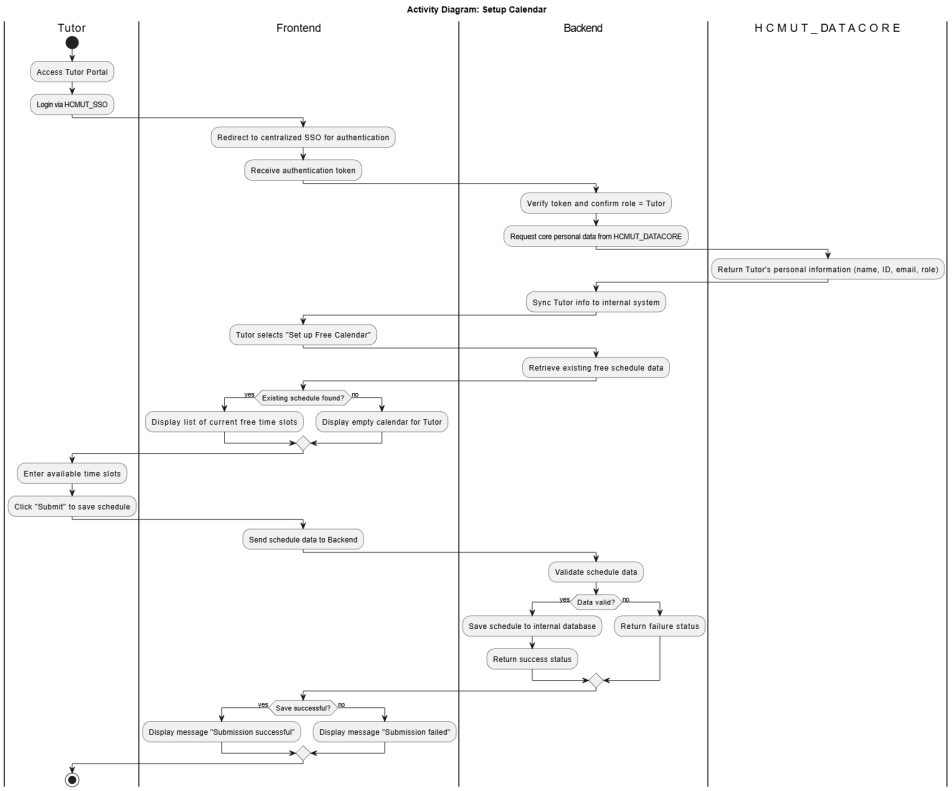
Sau khi đăng nhập thành công, Tutor chọn mục *"Set up Free Calendar"* trên giao diện hệ thống. Hệ thống hiển thị các khoảng thời gian rảnh hiện có (nếu có), hoặc hiển thị một lịch trống nếu chưa có dữ liệu. Tutor tiến hành nhập các khung thời gian rảnh và nhấn *"Submit"* để lưu lại. **Backend** kiểm tra tính hợp lệ của dữ liệu và lưu thông tin vào cơ sở dữ liệu nội bộ của hệ thống Tutor (không đồng bộ lên **HCMUT_DATACORE**). Cuối cùng, hệ thống hiển

thị thông báo *"Submit thành công"* nếu quá trình lưu thành công, hoặc thông báo lỗi nếu có vấn đề xảy ra.

Use case này đóng vai trò nền tảng cho các quy trình lập lịch và tạo buổi học sau này, vì dữ liệu về thời gian rảnh của Tutor được sử dụng làm cơ sở cho việc quản lý và sắp xếp buổi học.



Hình 4.21: Register Teaching Schedule - Sequence Diagram



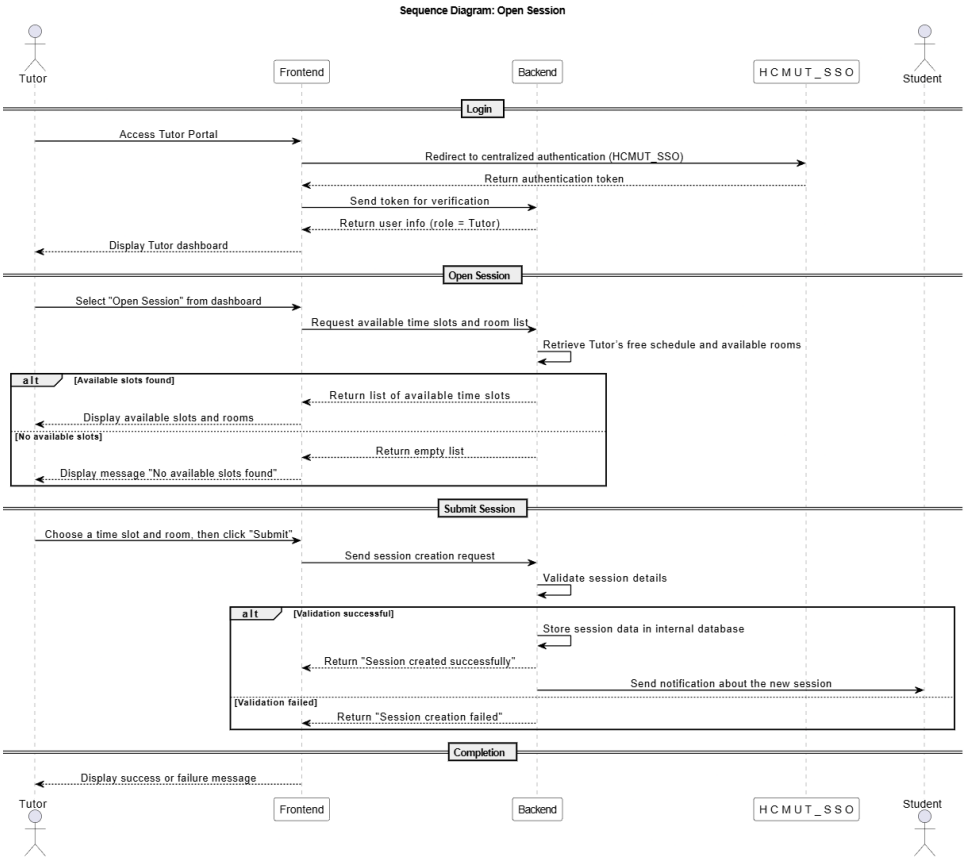
Hình 4.22: Register Teaching Schedule - Activity Diagram

4.7.2 Open Session

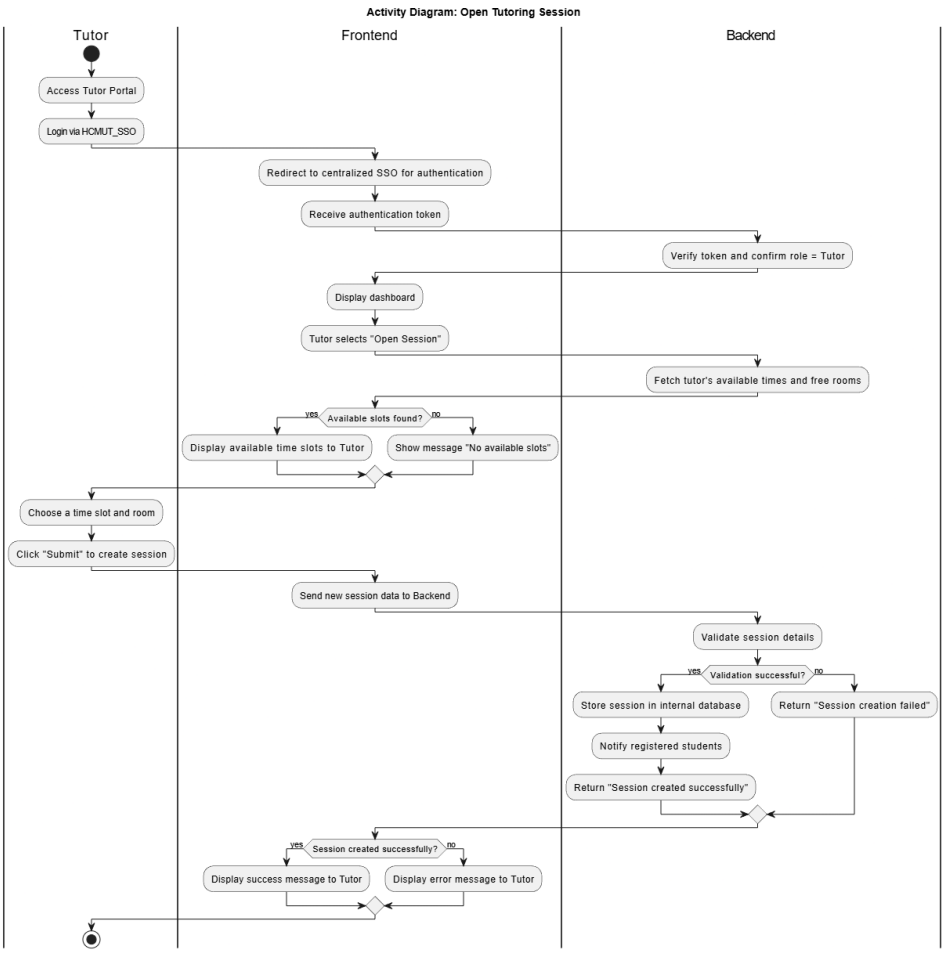
Use case *Open Session* cho phép Tutor tạo mới các buổi học (tutoring session) dựa trên lịch rảnh đã đăng ký. Sau khi đăng nhập thông qua **HCMUT_SSO** và được **Backend** xác thực, Tutor truy cập giao diện chính và chọn mục *"Open Session"*. **Backend** truy xuất các khung giờ rảnh của Tutor cùng với danh sách phòng học trống. Nếu có các khung giờ phù hợp, hệ thống sẽ hiển thị để Tutor lựa chọn; ngược lại, hệ thống sẽ thông báo *"Không có lịch rảnh khả dụng"*.

Tutor chọn khung giờ và phòng học mong muốn, sau đó nhấn *"Submit"* để tạo buổi học mới. **Backend** kiểm tra tính hợp lệ của thông tin, lưu dữ liệu buổi học vào cơ sở dữ liệu nội bộ, và gửi thông báo đến các sinh viên đã đăng ký với Tutor này. Nếu việc tạo buổi học thất bại (do dữ liệu không hợp lệ hoặc lỗi hệ thống), hệ thống sẽ hiển thị thông báo lỗi tương ứng.

Use case này giúp Tutor dễ dàng thiết lập và tổ chức các buổi trao đổi với sinh viên, đảm bảo quá trình hỗ trợ học tập diễn ra thuận tiện và có hệ thống.



Hình 4.23: Open Session - Sequence Diagram



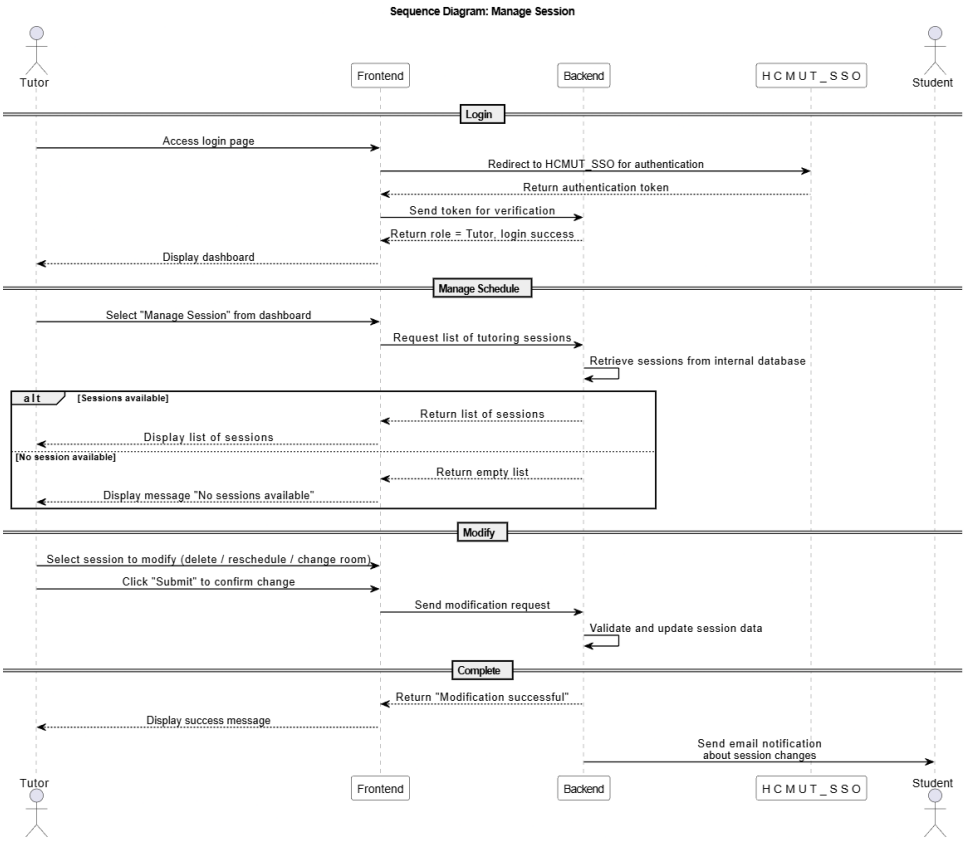
Hình 4.24: Open Session - Activity Diagram

4.7.3 Manage Session

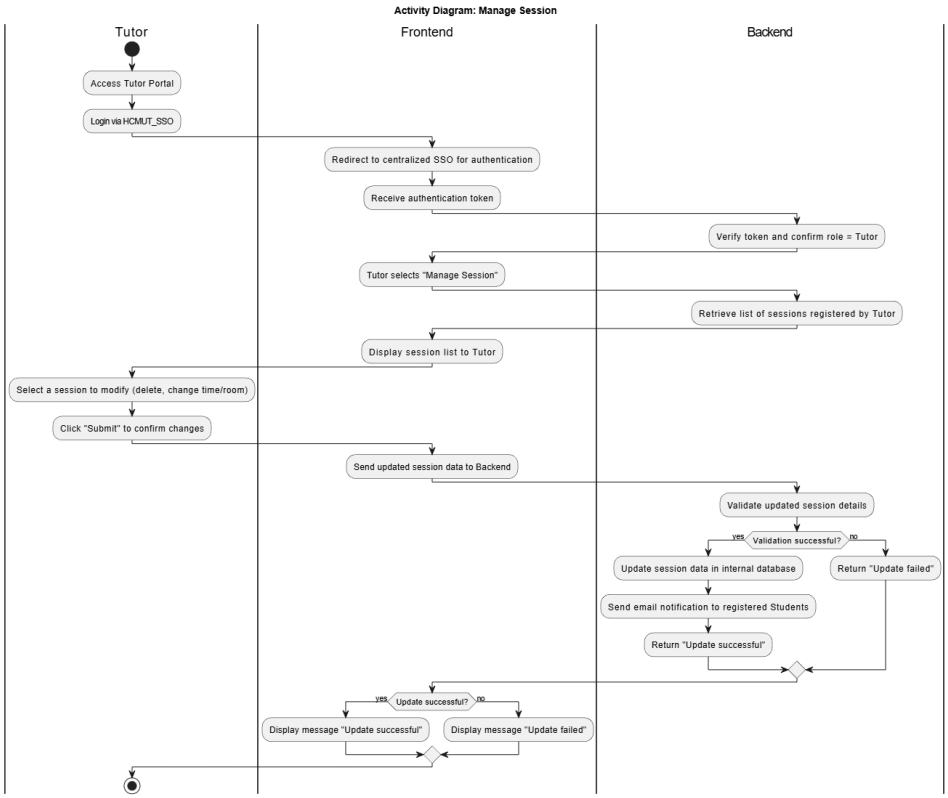
Use case *Manage Session* cho phép Tutor chỉnh sửa, thay đổi hoặc hủy các buổi học đã được đăng ký trong hệ thống. Sau khi đăng nhập vào cổng Tutor thông qua **HCMUT_SSO**, Tutor chọn mục *"Manage Session"* từ giao diện chính. **Backend** truy xuất danh sách các buổi học hiện có của Tutor và gửi lại cho **Frontend** để hiển thị.

Tutor chọn một buổi học cụ thể cần chỉnh sửa (ví dụ: đổi giờ, đổi phòng, hoặc xóa buổi học) và nhấn *"Submit"* để xác nhận thay đổi. **Backend** tiến hành kiểm tra, xác thực và cập nhật thông tin buổi học trong cơ sở dữ liệu nội bộ. Nếu cập nhật thành công, hệ thống hiển thị thông báo *"Cập nhật thành công"* và tự động gửi email thông báo đến các sinh viên liên quan. Trường hợp cập nhật thất bại, hệ thống sẽ hiển thị thông báo lỗi để Tutor biết và điều chỉnh lại.

Use case này giúp Tutor quản lý linh hoạt các buổi học của mình, đảm bảo tính cập nhật và chính xác của lịch học, đồng thời hỗ trợ sinh viên nắm bắt kịp thời các thay đổi.



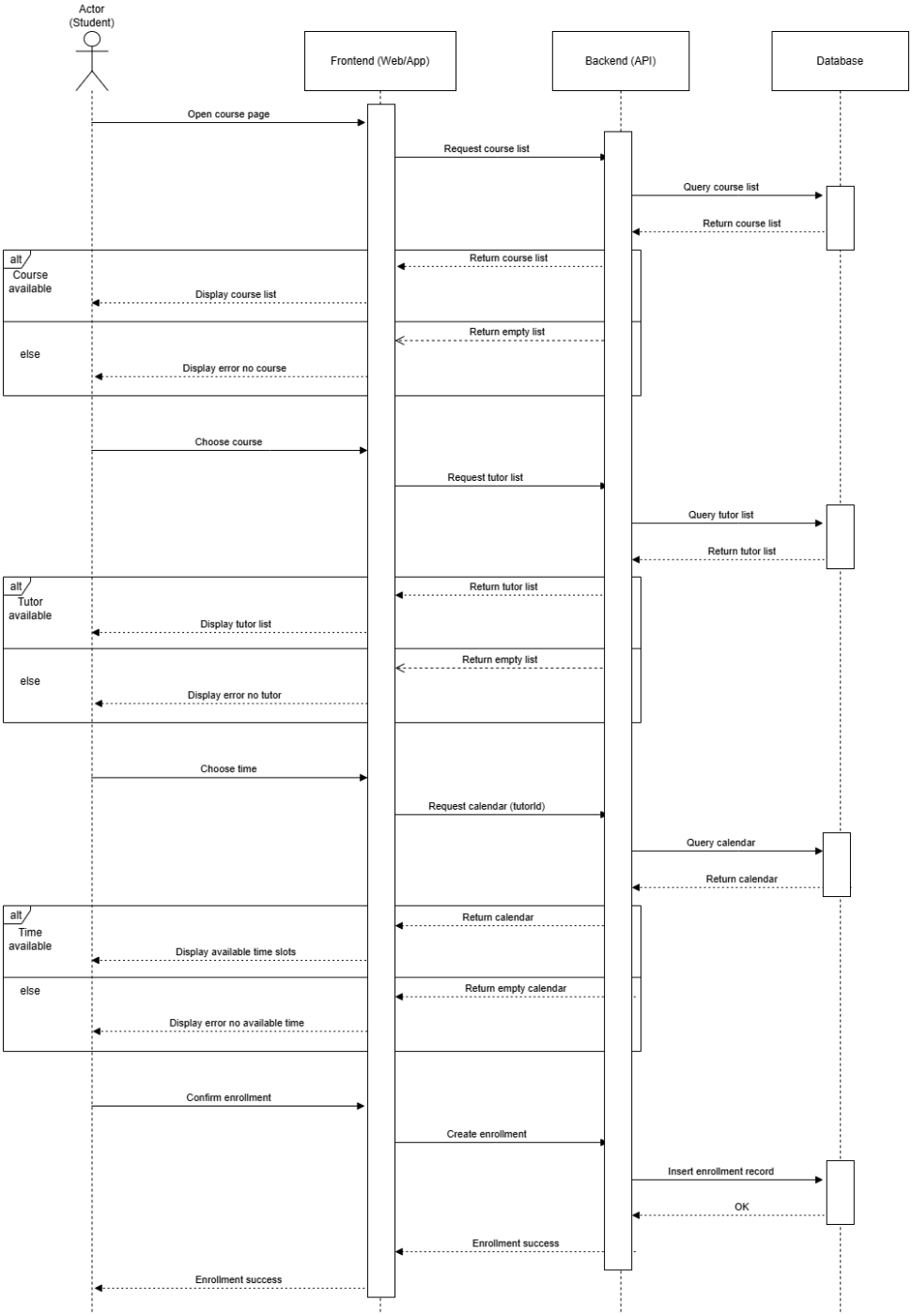
Hình 4.25: Manage Session - Sequence Diagram



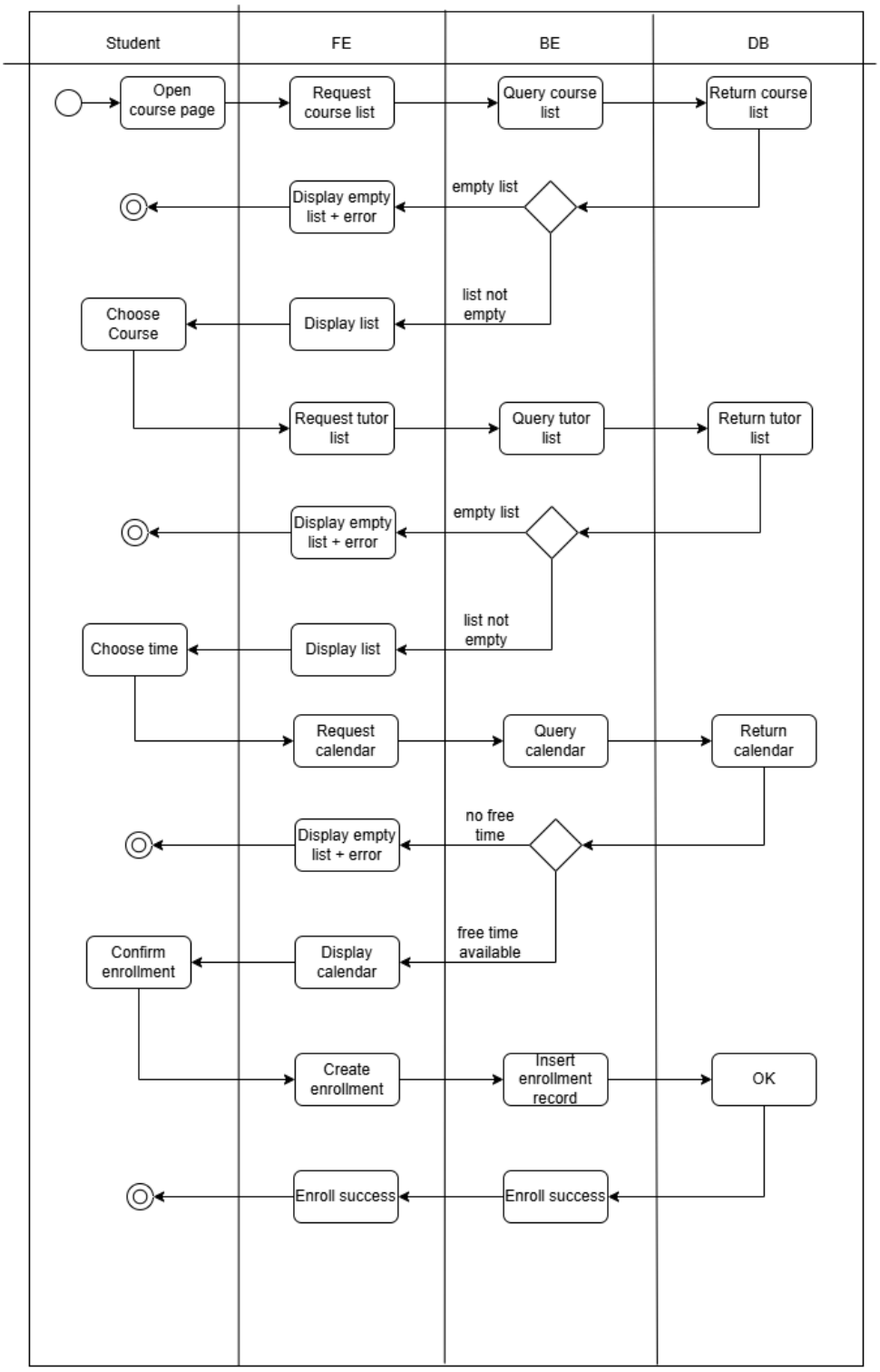
Hình 4.26: Manage Session - Activity Diagram

4.8 Student's Enrollment

4.8.1 Student Book Session

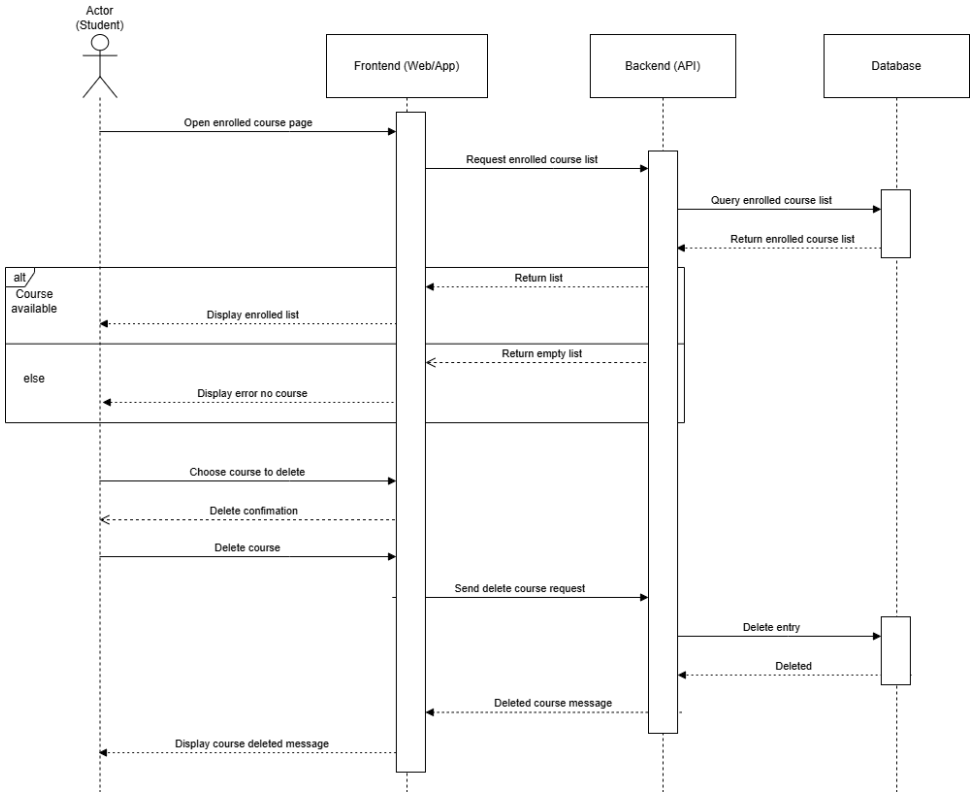


Hình 4.27: Student Book Session - Sequence Diagram

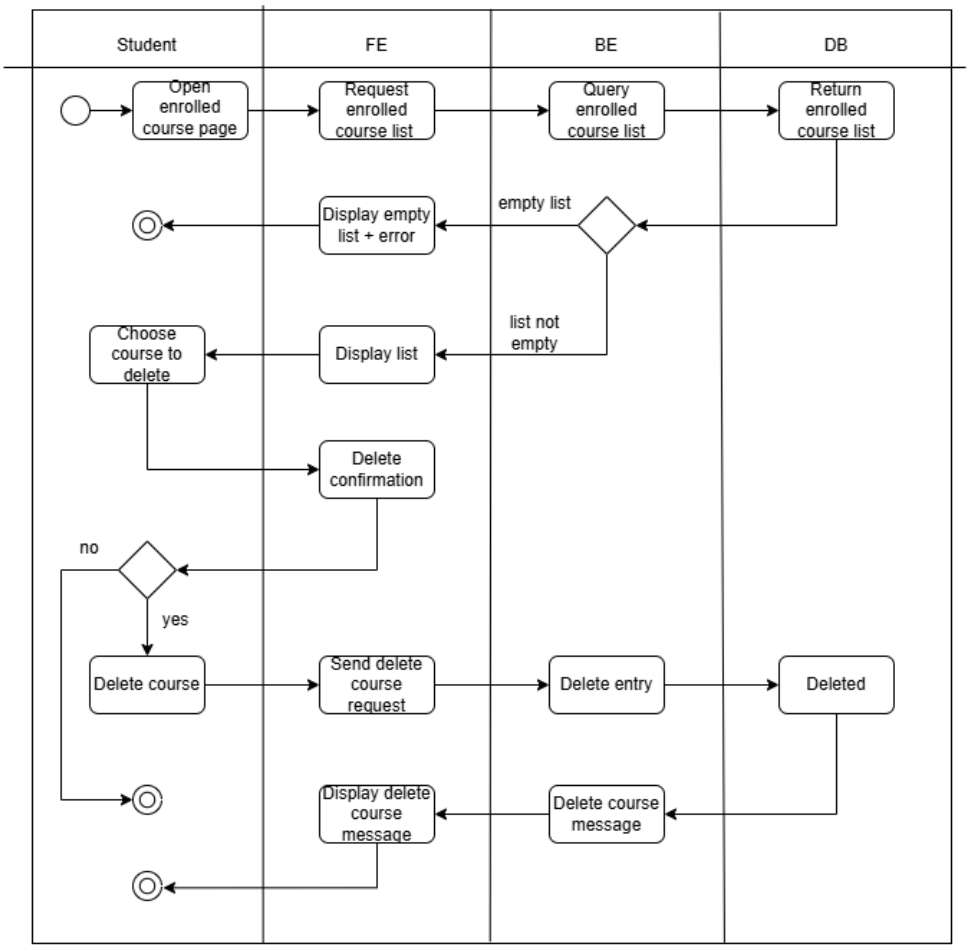


Hình 4.28: Student Book Session - Activity Diagram

4.8.2 Student Delete Session

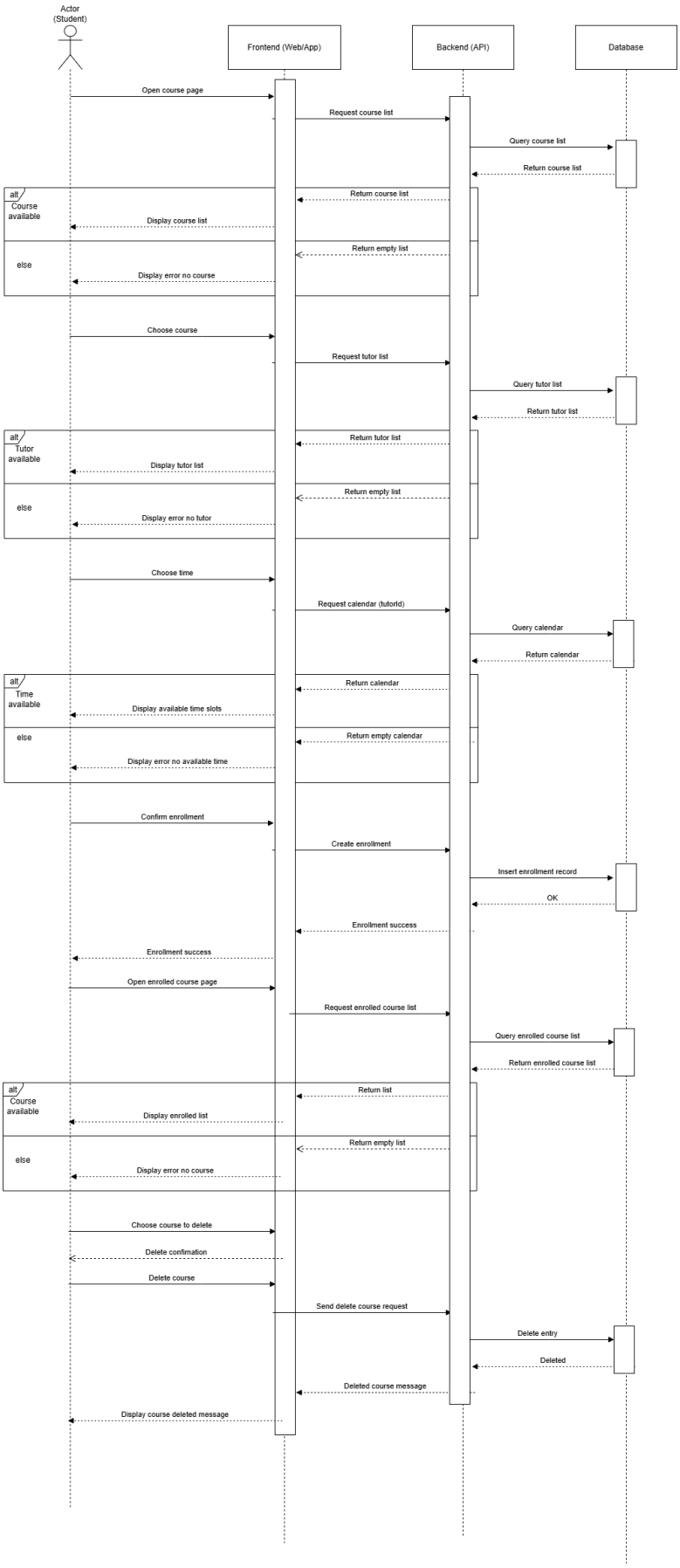


Hình 4.29: Student Delete Session - Sequence Diagram

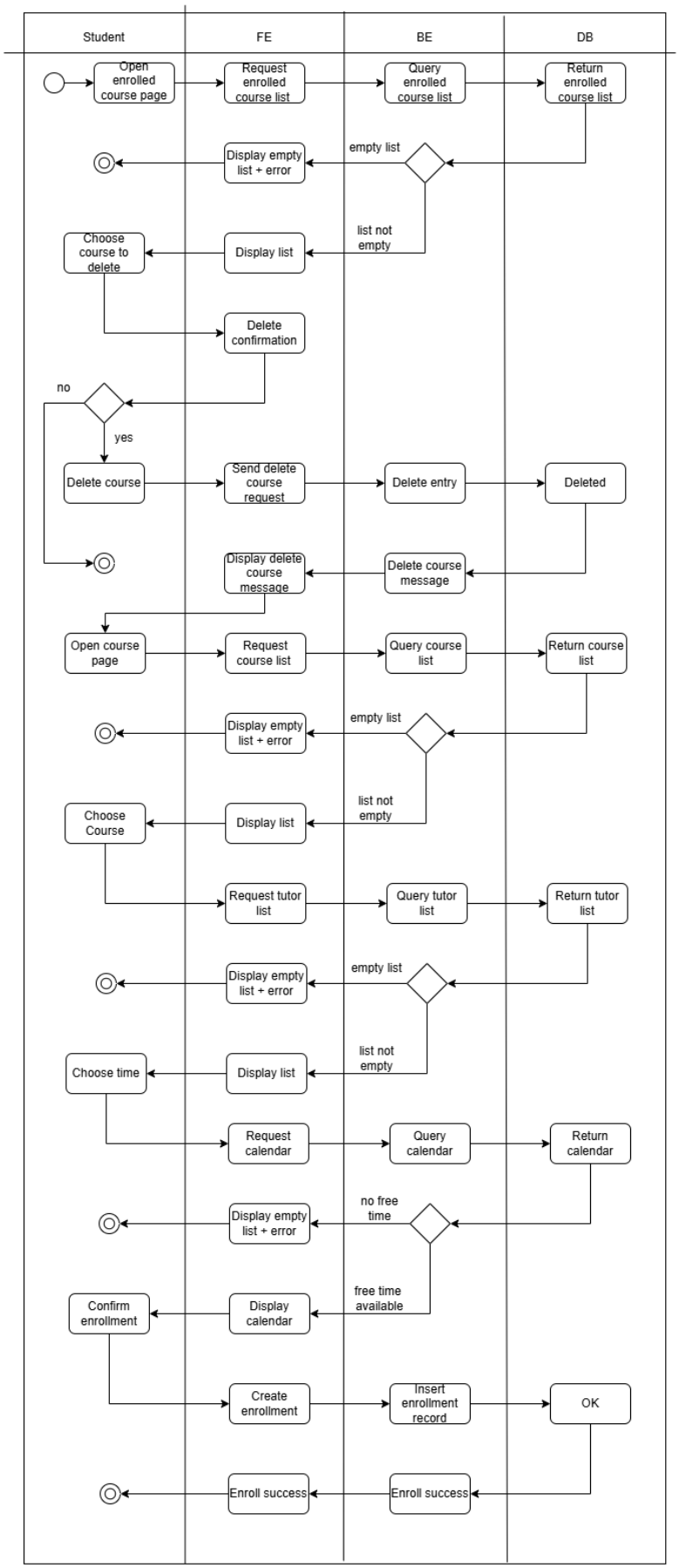


Hình 4.30: Student Delete Session - Activity Diagram

4.8.3 Student Change session



Hình 4.31: Student Change Session - Sequence Diagram



Hình 4.32: Student Change Session - Activity Diagram

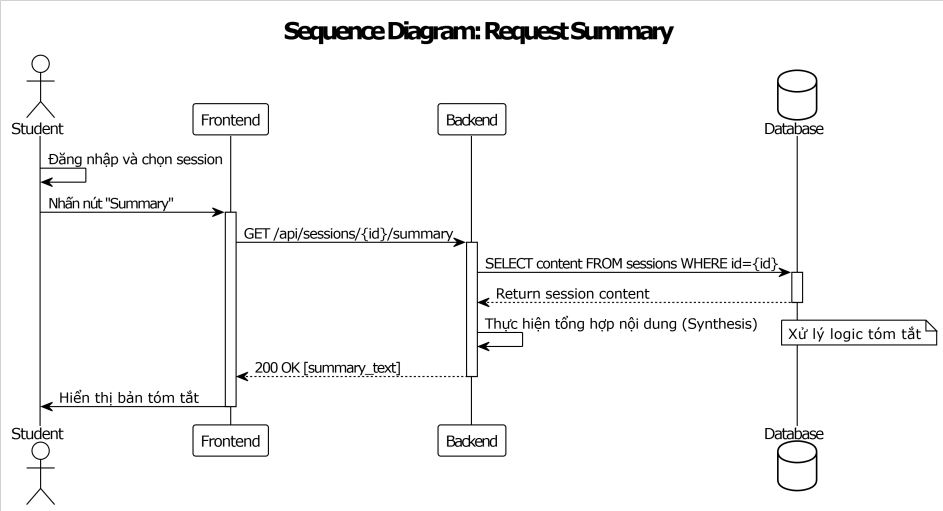
4.9 Student Request Summary

4.9.1 Request Summary

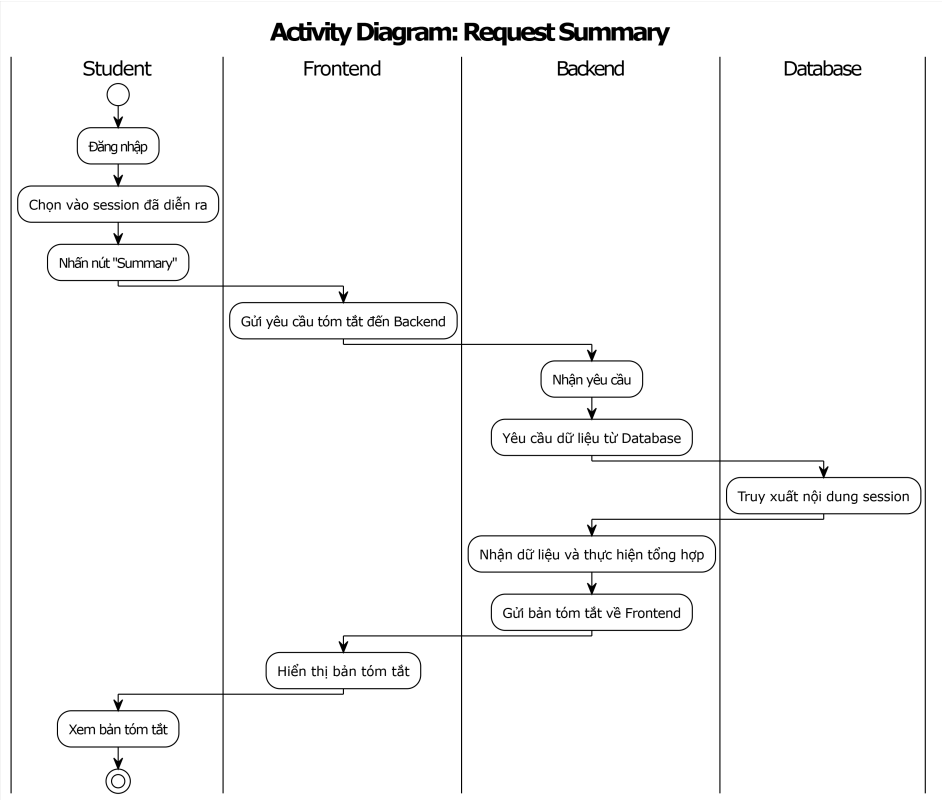
Use case này cho phép **Student** yêu cầu và nhận một bản tóm tắt nội dung của một buổi học đã diễn ra, được tạo tự động bởi hệ thống. Quá trình bắt đầu sau khi Student đăng nhập thành công và chọn một buổi học cụ thể từ lịch sử các session đã tham gia.

Khi Student nhấn vào nút *"Summary"* trên giao diện, **Frontend** sẽ gửi một yêu cầu tóm tắt (kèm theo ID của session) đến **Backend**. **Backend** tiếp nhận yêu cầu, sau đó truy vấn **Database** để lấy toàn bộ nội dung chi tiết của buổi học đó, chẳng hạn như bản ghi chat, transcript, hoặc các ghi chú chính. Điểm cốt lõi của quy trình nằm ở bước tiếp theo, khi **Backend** thực hiện thuật toán tổng hợp nội dung (Content Synthesis) để chắt lọc và rút ra các ý chính từ dữ liệu gốc. Sau khi tạo xong bản tóm tắt, **Backend** gửi kết quả về cho **Frontend**. Cuối cùng, hệ thống hiển thị bản tóm tắt trên giao diện để Student có thể xem lại.

Use case này đóng vai trò là một công cụ học tập thông minh, giúp sinh viên nhanh chóng nắm bắt các nội dung quan trọng của buổi học mà không cần phải xem lại toàn bộ video hay tài liệu. Nó đặc biệt hữu ích cho việc ôn tập nhanh trước các kỳ thi hoặc khi sinh viên muốn củng cố lại kiến thức đã học một cách hiệu quả.



Hình 4.33: Request Summary - Sequence Diagram



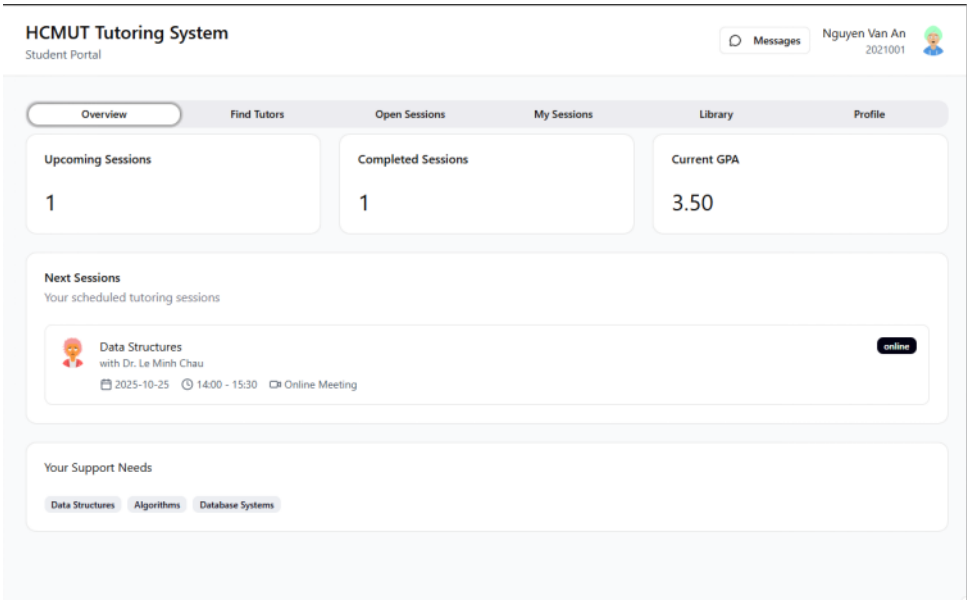
Hình 4.34: Request Summary - Activity Diagram

CHƯƠNG 5

MOCKUP

5.1 Student UI

5.1.1 Overview

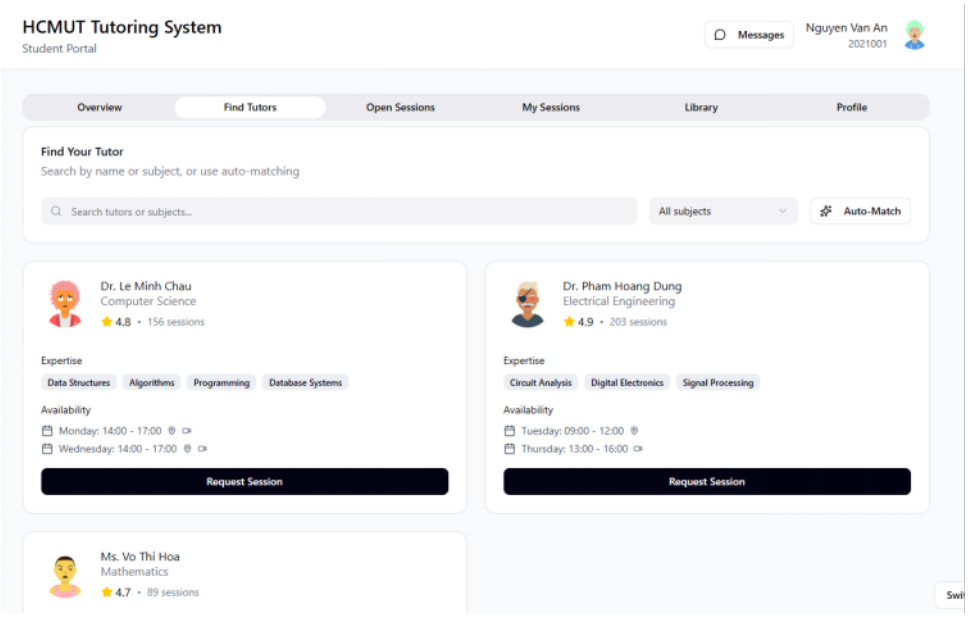


Hình 5.1: Student Overview

Trang của người dùng được chia thành các mục Overview, Find Tutor, Open Session, My Session, Library và Profile.

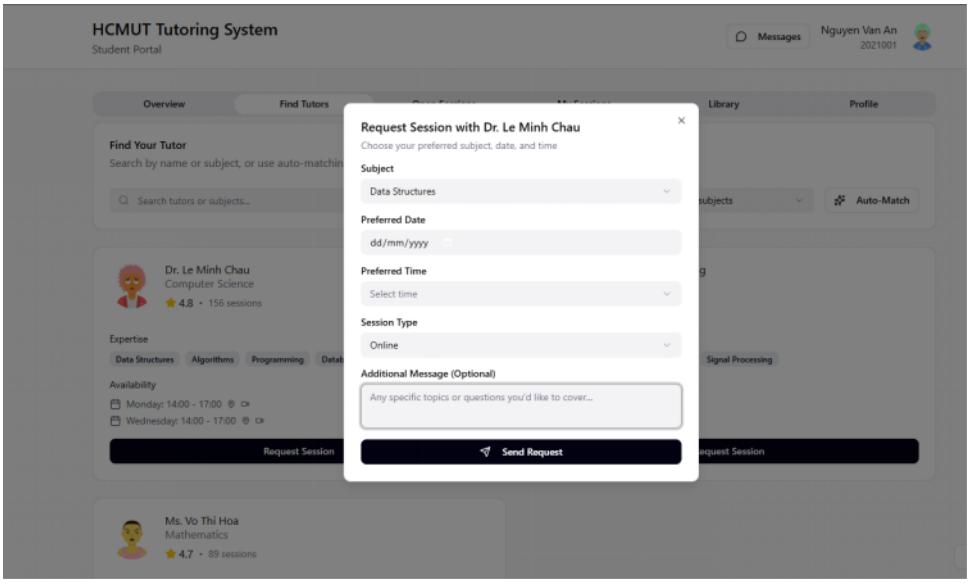
Mục Overview chứa các thông tin cơ bản về các khóa học sắp tới của sinh viên và các môn học mà sinh viên cần hỗ trợ.

5.1.2 Find Tutor



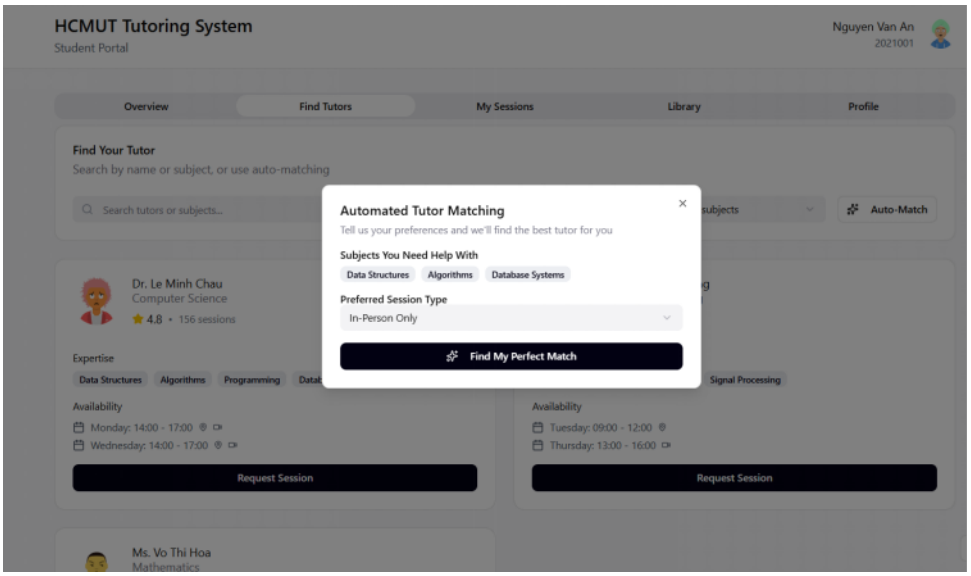
Hình 5.2: Student Find Tutor

Mục Find Tutor là mục đề xuất các session để Tutor tổ chức. Sinh viên có thể tìm Session theo Tutor hoặc môn học.



Hình 5.3: Student Suggest Subject and Time

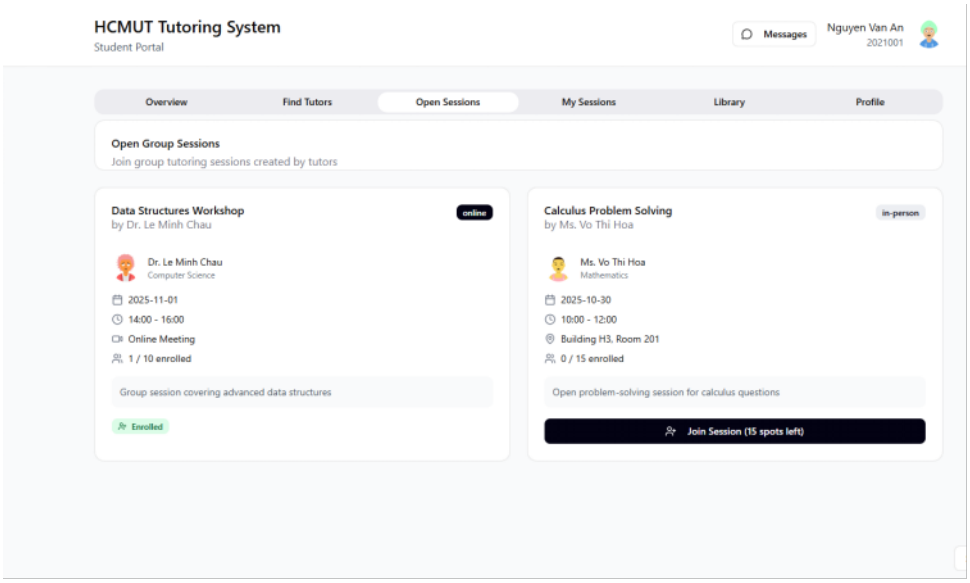
Sinh viên nhấp vào Request Session để đề xuất giờ học và môn học cho Tutor.



Hình 5.4: Student Auto Match with Tutor

Hệ thống còn hỗ trợ auto-match để tìm Tutor thích hợp nhất theo nhu cầu của sinh viên.

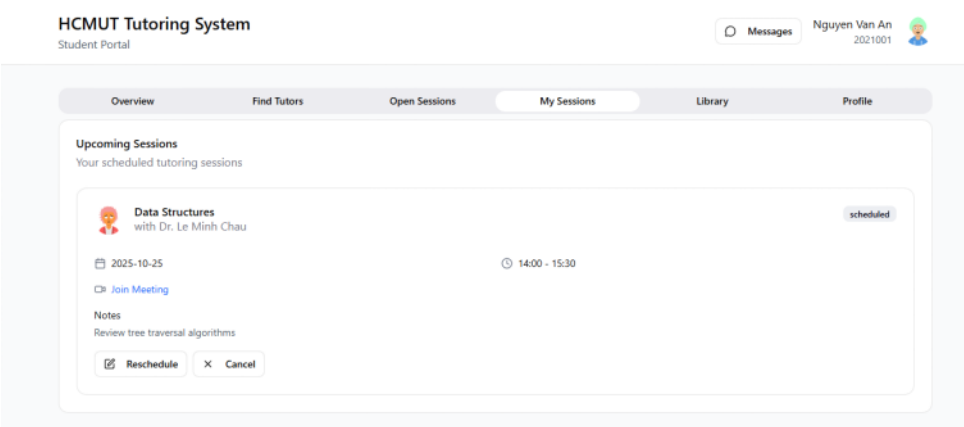
5.1.3 Open Section



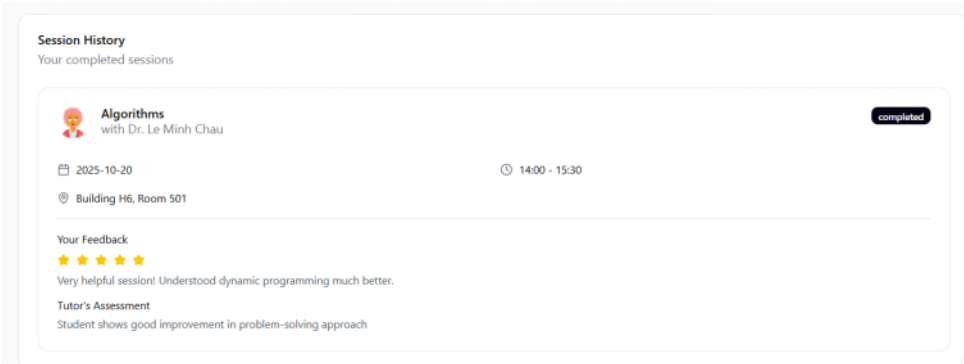
Hình 5.5: Student enroll open sessions

Đây là mục để sinh viên tham gia các Session do Tutor tổ chức.

5.1.4 My Session



Hình 5.6: Student's Upcoming Sessions

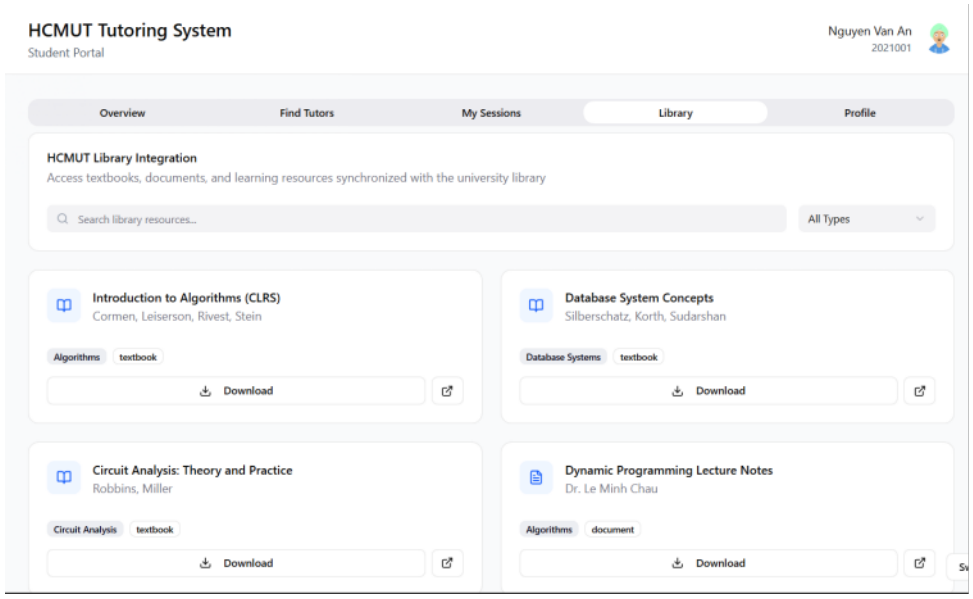


Hình 5.7: Student's Complete Sessions

Mục My Session hiển thị tất cả khóa học mà sinh viên sắp và đã tham gia. Đối với các khóa học sắp tham gia, sinh viên có thể gửi yêu cầu đổi lịch cho Tutor hoặc hủy lịch học. Các lịch học Online được liên kết với Meet để sinh viên thuận tiện tham gia.

Đối với các khóa học đã tham gia, sinh viên có thể gửi feedback để Tutor cải thiện.

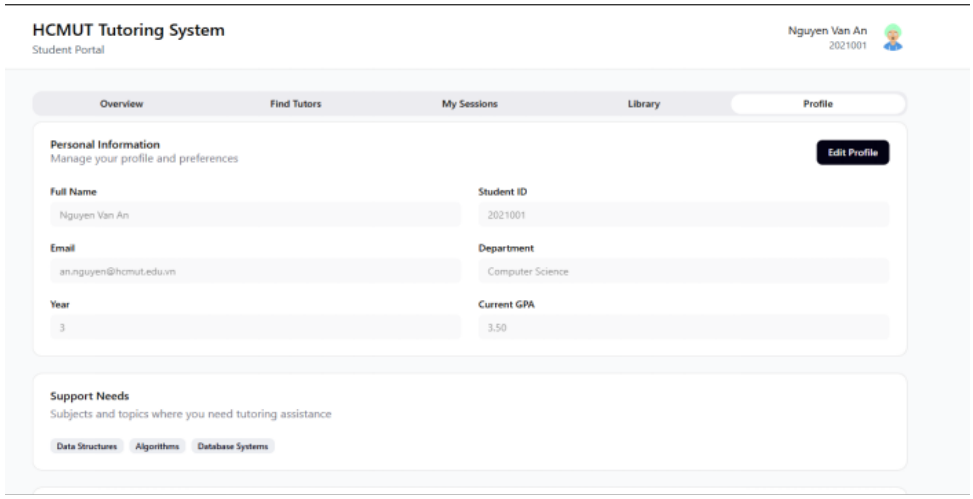
5.1.5 Library



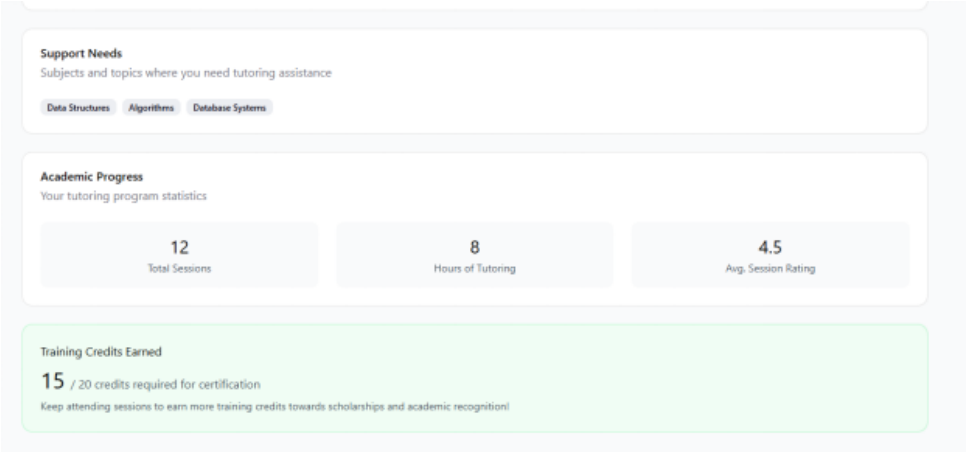
Hình 5.8: Student's Library

Mục Library là thư viện các cuốn sách được liên kết với HCMUT_Library, giúp sinh viên và Tutor truy cập, chia sẻ tài liệu, giáo trình.

5.1.6 Profile



Hình 5.9: Student's Profile

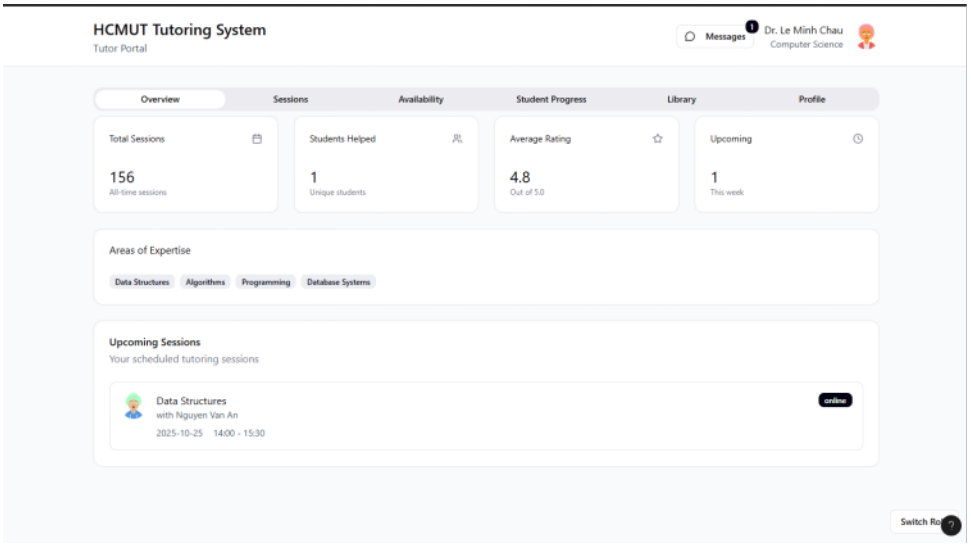


Hình 5.10: Student's Profile

Mục Profile chứa các thông tin cá nhân của sinh viên, bên cạnh đó còn chứa các thông số tham gia môn học, giúp sinh viên tự đánh giá tiến độ học tập của bản thân.

5.2 Tutor UI

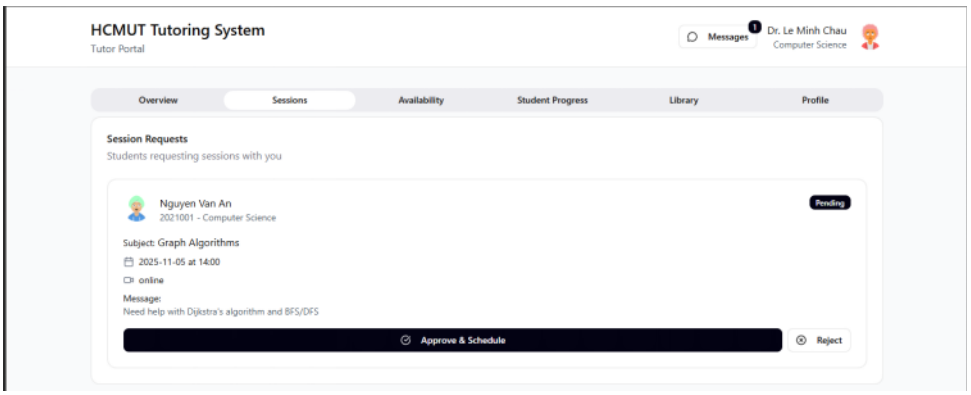
5.2.1 Overview



Hình 5.11: Tutor's Overview

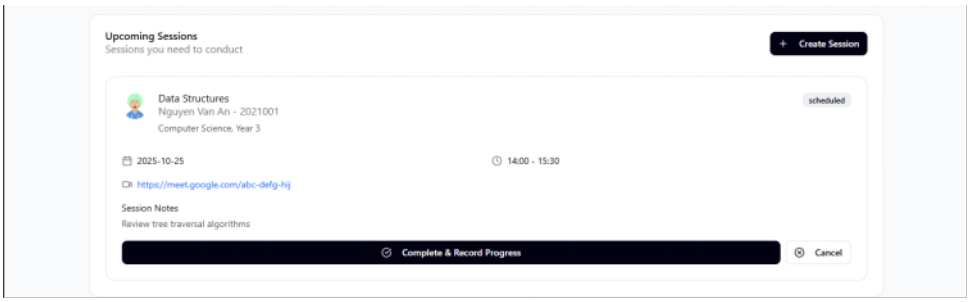
Mục Overview của Tutor cũng chứa một vài thông số cơ bản của Tutor như sinh viên.

5.2.2 Session



Hình 5.12: Tutor's Session Page - Student Suggest

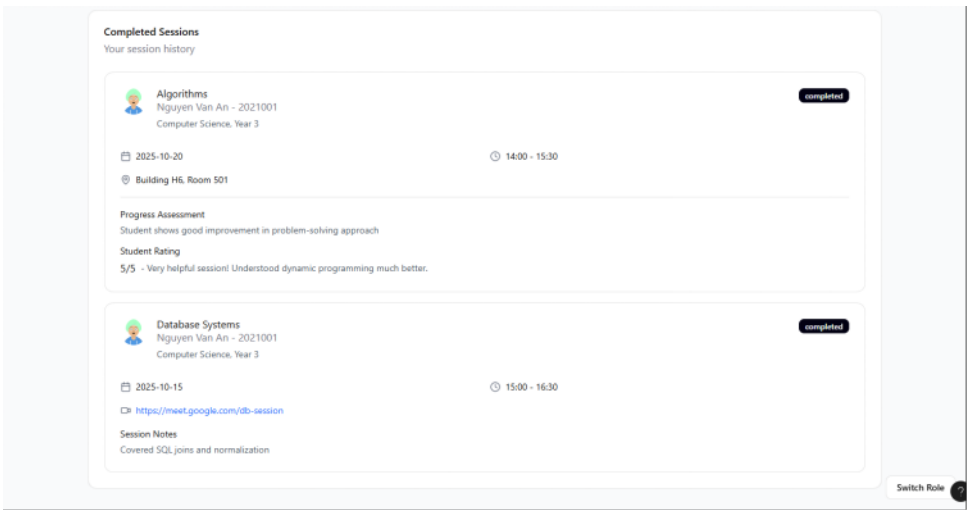
Tutor có thể xem đề xuất giờ học của sinh viên và đồng ý mở lớp.



Hình 5.13: Tutor's Session Page - Upcoming

Tutor có thể tự tạo Session mới bằng nút Create Session.

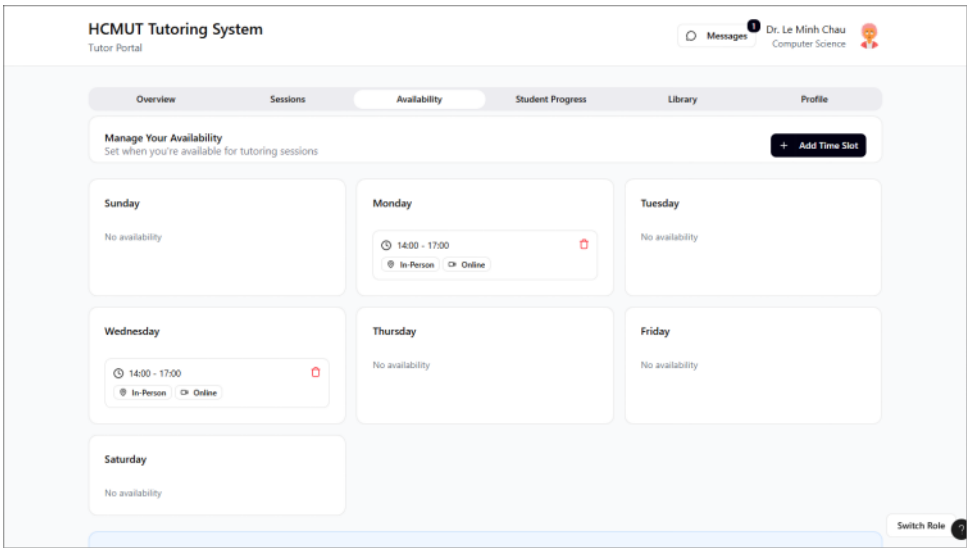
Tại đây, Tutor cũng theo dõi các Session hiện tại. Khi kết thúc Session thì Tutor có thể vào đánh giá sinh viên bằng Complete & Record Progress.



Hình 5.14: Tutor's Session Page - Complete

Tutor có thể coi các Session đã hoàn thành và các đánh giá của sinh viên.

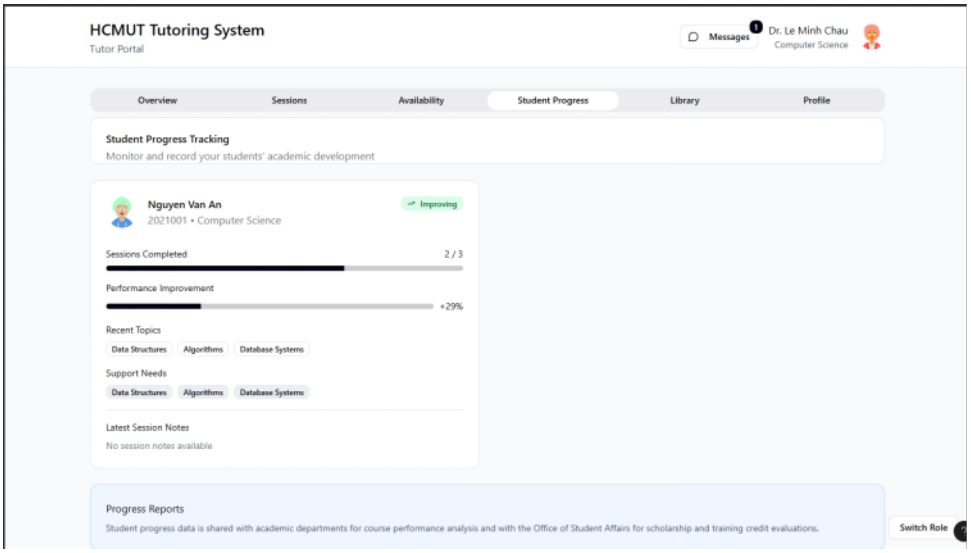
5.2.3 Availability



Hình 5.15: Tutor's Availability Page

Tutor có thể xem và thiết lập lịch rảnh của mình tại đây. Lịch rảnh của Tutor sẽ được hiển thị trong Find Tutor của sinh viên, từ đó sinh viên có thể đề xuất session.

5.2.4 Student Progress



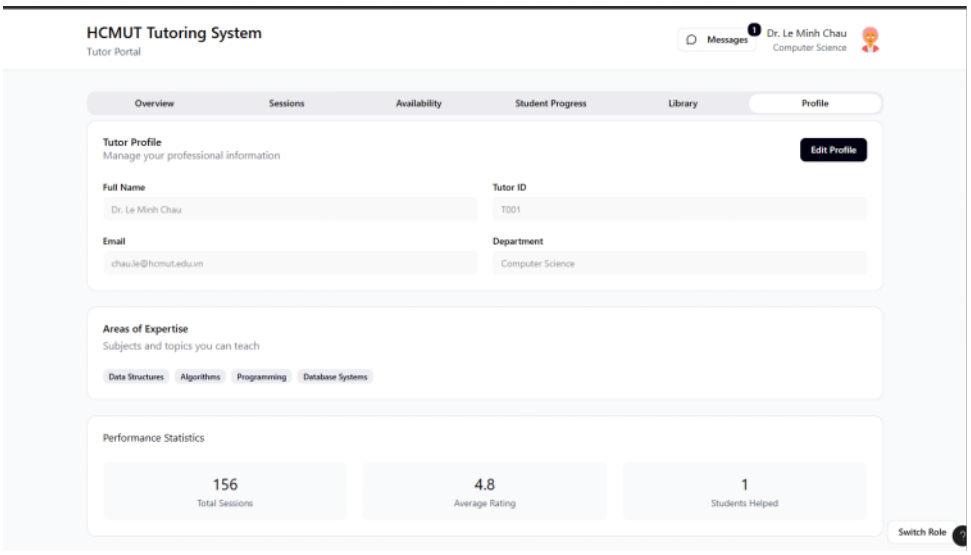
Hình 5.16: Tutor's Student Tracking

Tutor có thể kiểm tra quá trình học tập của sinh viên ở đây. Hệ thống sẽ hiển thị số session mà sinh viên tham gia cùng khả năng mà sinh viên thể hiện (có thể là qua các bài kiểm tra trên lớp).

5.2.5 Library

Thư viện mà Tutor truy cập cũng giống như sinh viên.

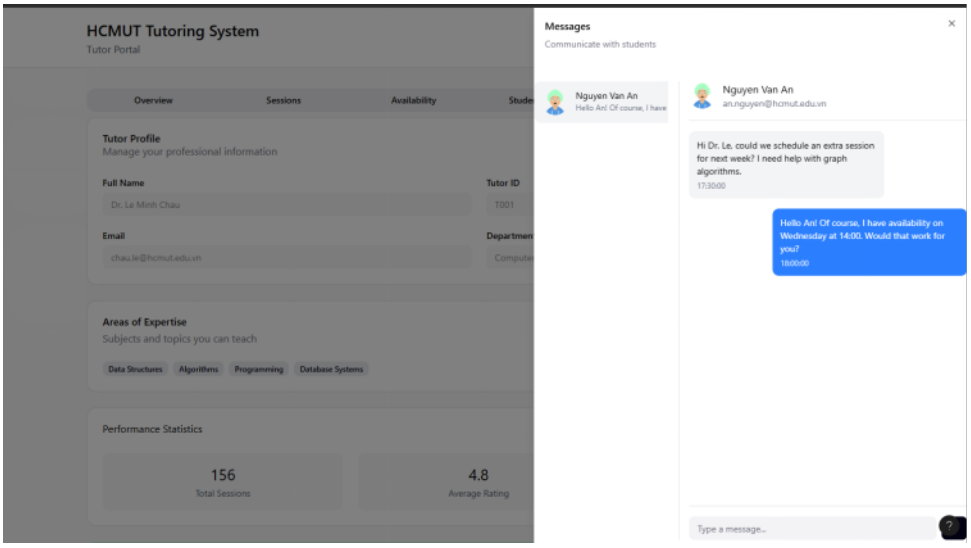
5.2.6 Profile



Hình 5.17: Tutor's Profile

Profile của Tutor cũng như sinh viên nhưng có một vài thông số về rating của Tutor.

5.2.7 Messages

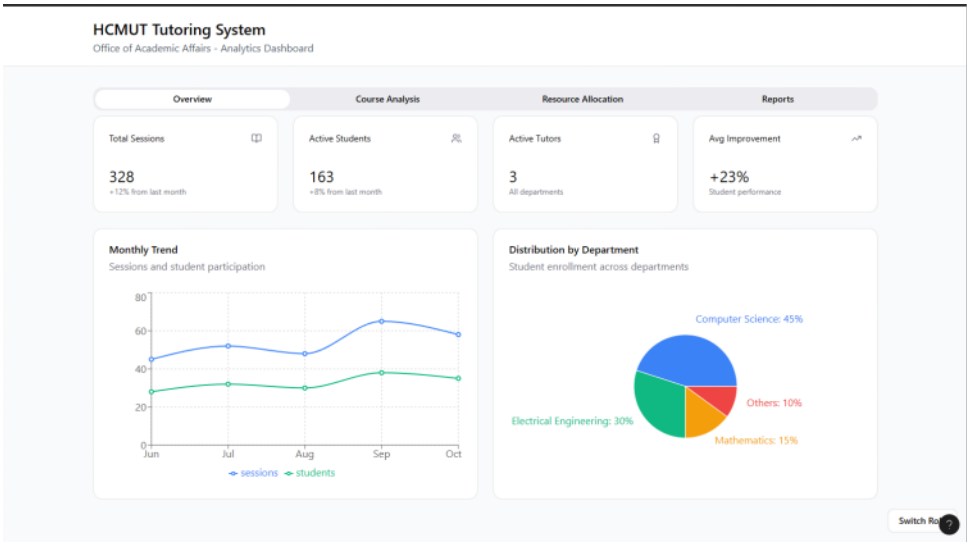


Hình 5.18: Message

Sinh viên và Tutor có thể liên lạc với nhau thông qua Message, các thông báo thay đổi lịch cũng hiện ở đây.

5.3 Academic Affair UI

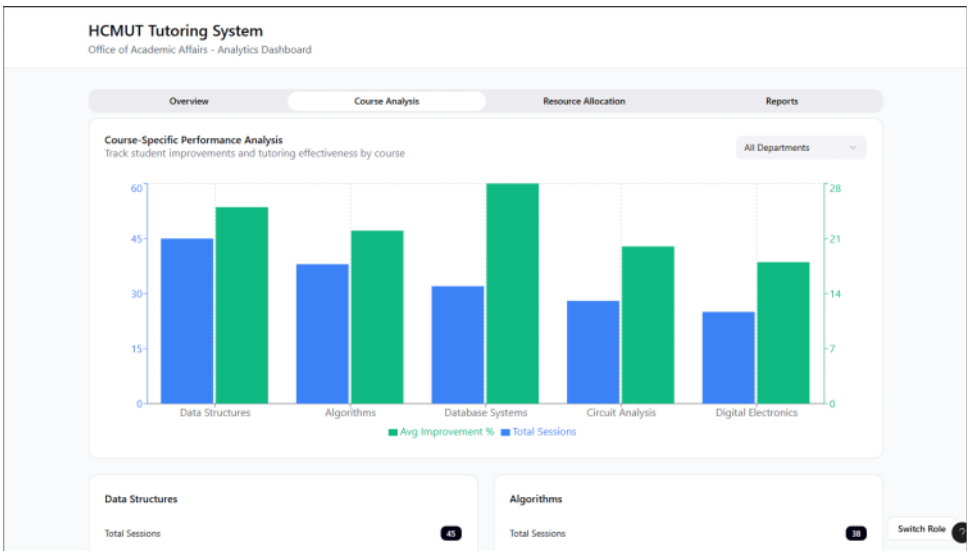
5.3.1 Overview



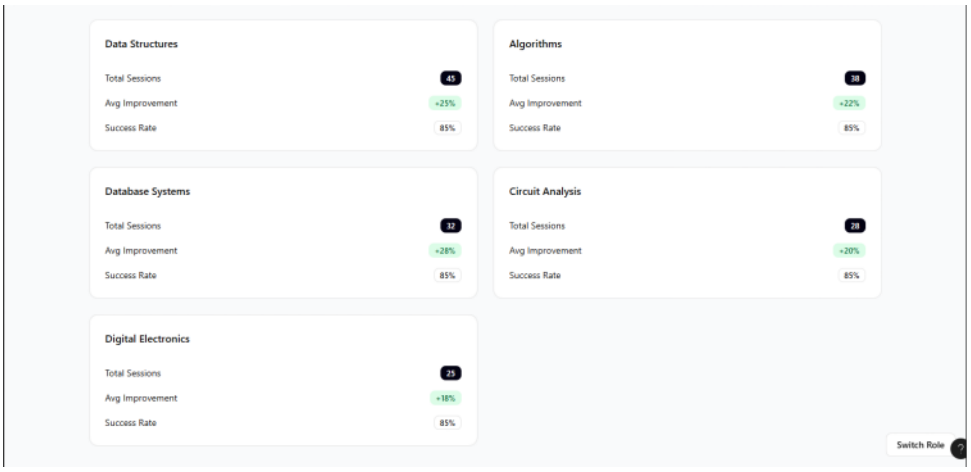
Hình 5.19: Academic Overview

Phòng đào tạo có thể coi các thông số về số session, sinh viên, tutor tham gia vào hệ thống, phân bố khoa, hiệu quả của hệ thống.

5.3.2 Analysis



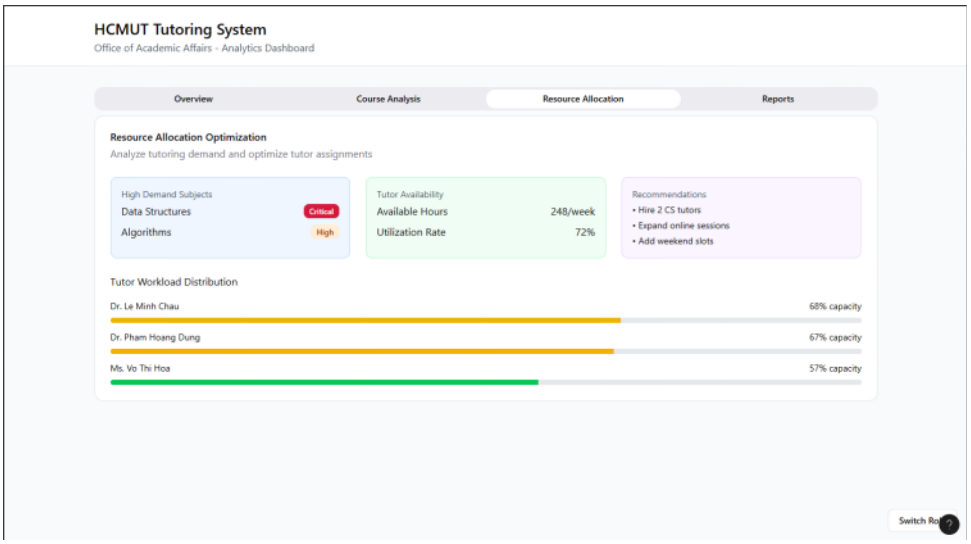
Hình 5.20: Academic Analysis



Hình 5.21: Academic Analysis

Trang này sẽ giúp phòng đào tạo phân tích sâu hơn về mức độ phát triển của từng bộ môn.

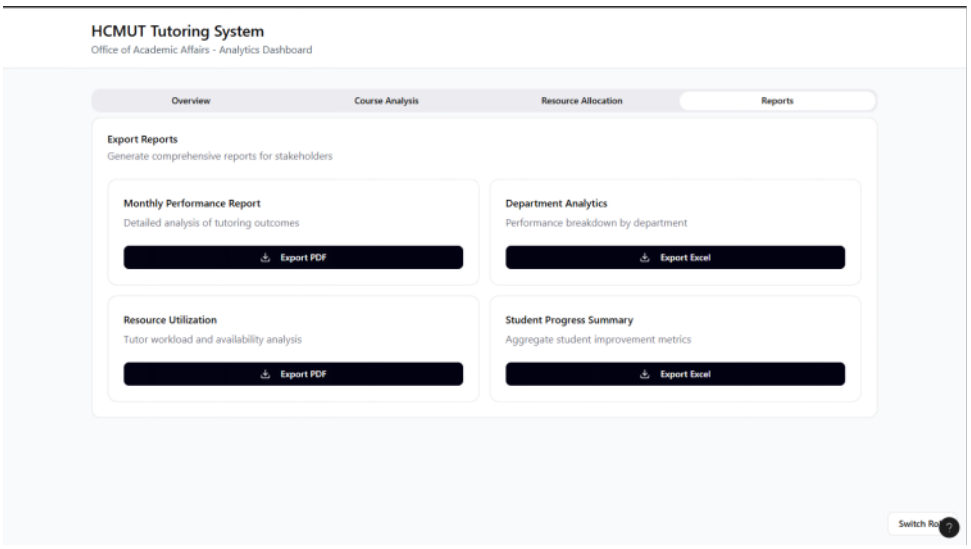
5.3.3 Resource



Hình 5.22: Academic Resource Allocation

Trang này giúp phòng đào tạo kiểm tra nhu cầu học tập của sinh viên cũng như phân bố giảng dạy của các Tutor, từ đó đưa ra chỉ thị phù hợp.

5.3.4 Report

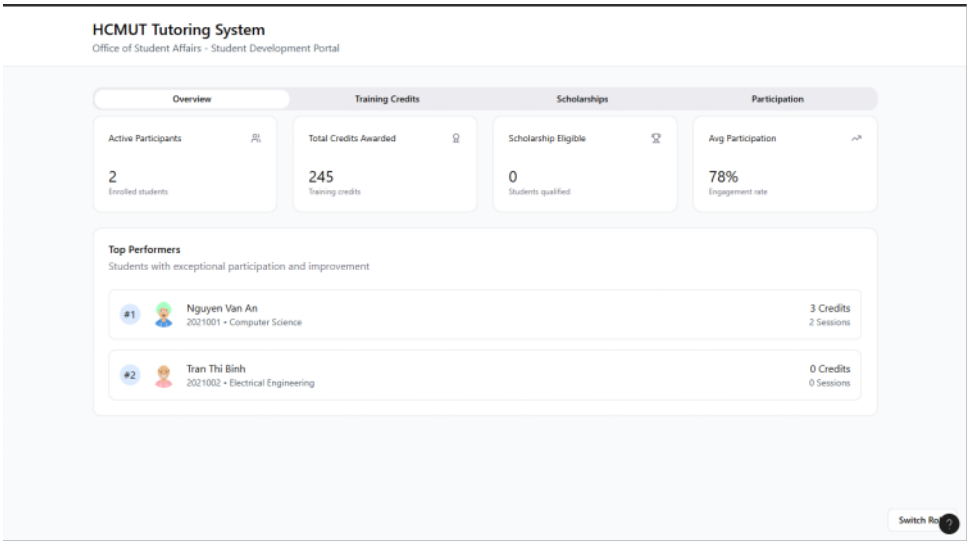


Hình 5.23: Academic Report

Trang này sẽ xuất các file báo cáo cho phòng đào tạo.

5.4 Student Affair UI

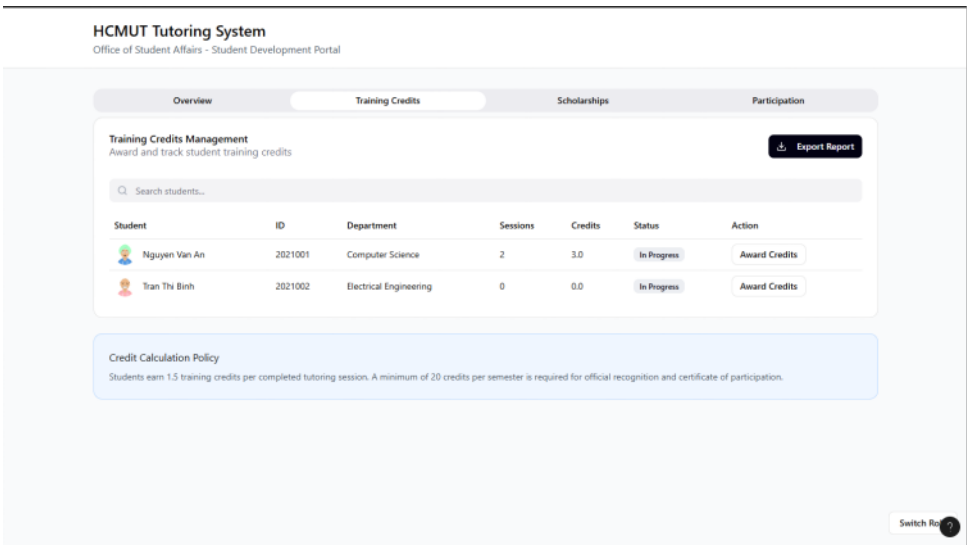
5.4.1 Overview



Hình 5.24: Student Affair Overview

Trang này giúp phòng công tác sinh viên xem số các sinh viên tham gia vào hệ thống và một số sinh viên tích cực tham gia.

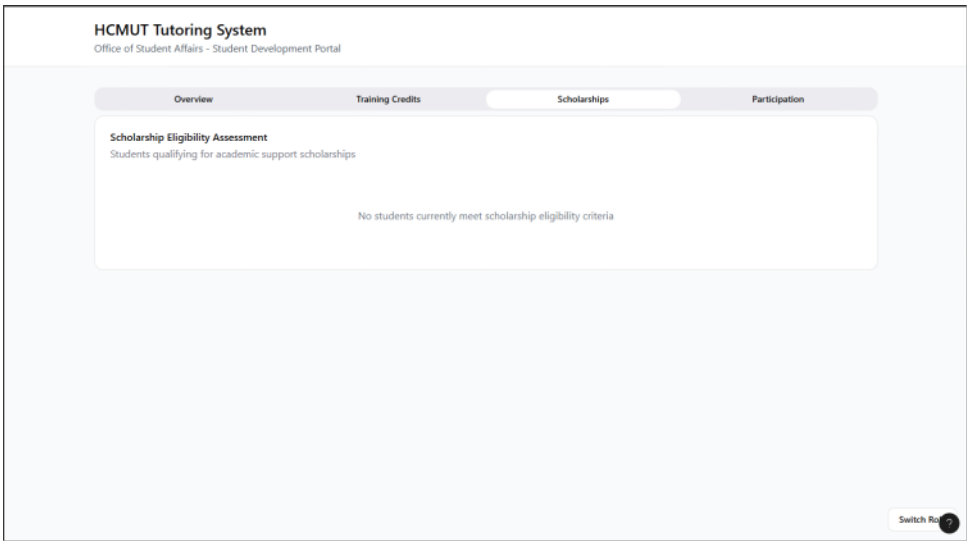
5.4.2 Training Credit



Hình 5.25: Student Affair Training

Trang này sẽ giúp phòng công tác sinh viên xem mức độ tham gia của tất cả sinh viên, từ đó xem xét cộng điểm rèn luyện.

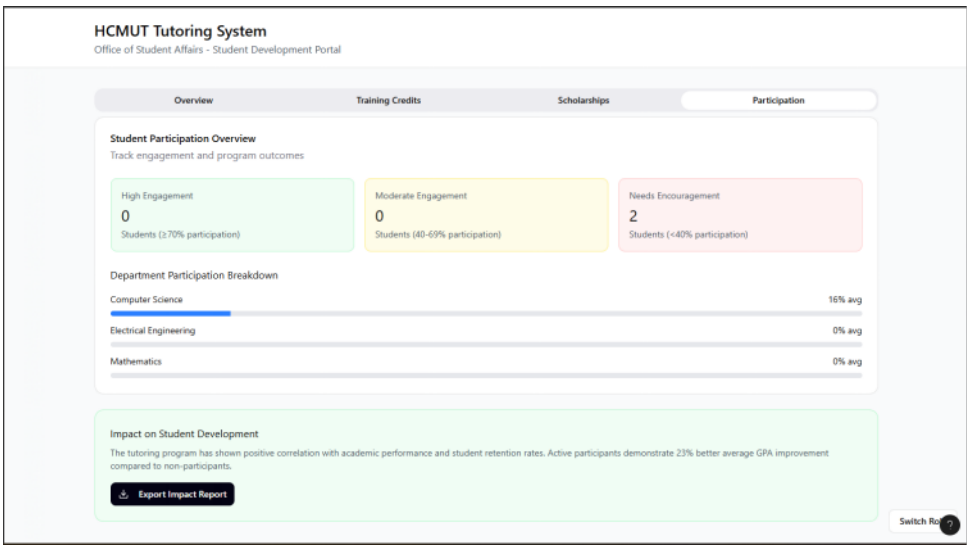
5.4.3 Scholarship



Hình 5.26: Student Affair Scholarship

Trang này sẽ hiển thị những sinh viên đủ tiêu chuẩn trao học bổng theo đánh giá mà phòng công tác sinh viên thiết lập.

5.4.4 Participation



Hình 5.27: Student Affair Participation

Trang này sẽ hiển thị một số thông tin khác và xuất báo cáo cho phòng công tác sinh viên.

CHƯƠNG 6

ARCHITECTURE DESIGN

6.1 Deployment view

Phần này mô tả kiến trúc triển khai (deployment architecture) của hệ thống **Tutor Support System**. Hệ thống được triển khai theo mô hình nhiều tầng (multi–tier architecture), bao gồm lớp giao diện (Client/Web UI), lớp xử lý nghiệp vụ (Backend & AI Service), lớp dữ liệu (Database, Cache, Storage), và các hệ thống tích hợp bên ngoài. Các thành phần được triển khai trên các node vật lý khác nhau và giao tiếp với nhau thông qua các giao thức bảo mật như HTTPS, TCP/IP và NFS.

6.1.1 Client PC

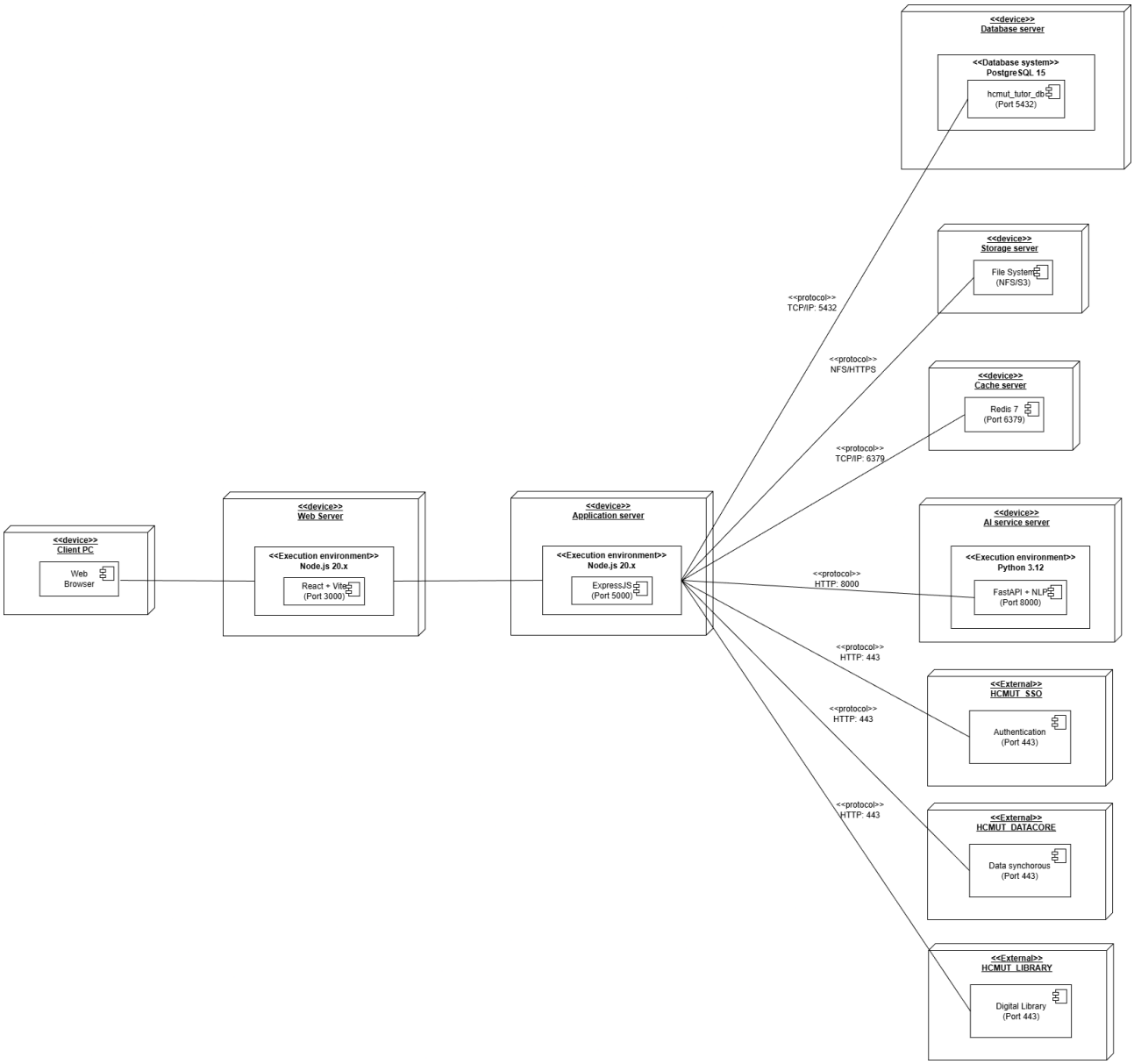
Node **Client PC** đại diện cho thiết bị của người dùng cuối như laptop hoặc máy tính để bàn. Người dùng truy cập hệ thống thông qua trình duyệt web.

- **Web Browser**: Giao diện truy cập dành cho Sinh viên, Tutor và Cán bộ Staff.
- **Giao tiếp**: Kết nối đến Web Server thông qua giao thức HTTPS.

6.1.2 Web Server

Node **Web Server** chịu trách nhiệm phục vụ giao diện web cho người dùng.

- **Execution Environment**: Node.js 20.x
- **Frontend Application**: React + Vite (port 3000)
- **Vai trò**:
  - Cung cấp các tệp giao diện (HTML/CSS/JS).
  - Chuyển tiếp request API từ trình duyệt đến Backend.
- **Giao tiếp**:
  - HTTPS → Client PC
  - HTTPS → Application Server



Hình 6.1: Deployment diagram của hệ thống Tutor Support System

6.1.3 Application Server

Node **Application Server** là trung tâm xử lý nghiệp vụ của hệ thống.

- **Execution Environment**: Node.js 20.x
- **Backend API**: Express.js (port 5000)
- **Các dịch vụ chính**:
  - Authentication & RBAC
  - Quản lý phiên học (session management)
  - Quản lý yêu cầu hỗ trợ (request handling)
  - Gợi ý tutor, đồng bộ dữ liệu với Datacore

  - Kết nối với AI Service để tạo summary/báo cáo
  - Tích hợp với thư viện số của trường
  - Gửi email thông báo lịch học

- **Giao tiếp**:
  - HTTP (8000) → AI Service Server
  - TCP/IP (5432) → Database Server
  - TCP/IP (6379) → Cache Server
  - NFS/HTTPS → Storage Server
  - HTTPS (443) → HCMUT_SSO
  - HTTPS (443) → HCMUT_DATACORE
  - HTTPS (443) → HCMUT_LIBRARY

6.1.4 AI Service Server

Node **AI Service Server** cung cấp các chức năng xử lý ngôn ngữ tự nhiên và phân tích phục vụ hệ thống.

- **Execution Environment**: Python 3.12
- **AI API**: FastAPI + NLP Engine (port 8000)
- **Chức năng**:
  - Sinh bản tóm tắt buổi học (AI Summary)
  - Gợi ý Tutor phù hợp (AI Matching)
  - Trả lời câu hỏi học tập từ sinh viên (Study Assistant)
- **Giao tiếp**:
  - HTTP ← Application Server

6.1.5 Database Server

Node **Database Server** lưu toàn bộ dữ liệu nghiệp vụ của hệ thống.

- **Database System**: PostgreSQL 15 (port 5432)
- **Dữ liệu lưu trữ**:
  - Hồ sơ người dùng (User Profile)
  - Hồ sơ Tutor, thời gian rảnh
  - Lịch học và các buổi Session
  - Tin nhắn (Messaging)
  - Tài liệu học tập được đính kèm
  - Đánh giá, feedback từ sinh viên
  - Dữ liệu báo cáo và thống kê
- **Giao tiếp**: TCP/IP ← Application Server

6.1.6 Cache Server

Node **Cache Server** tối ưu hiệu năng bằng cách lưu trữ tạm thời các dữ liệu truy cập nhanh.

- **Technology**: Redis 7 (port 6379)
- **Vai trò**:
  - Lưu cache token đăng nhập
  - Cache lịch học và yêu cầu thường xuyên truy cập
  - Giảm tải truy vấn đến Database Server
- **Giao tiếp**: TCP/IP ← Application Server

6.1.7 Storage Server

Node **Storage Server** chịu trách nhiệm lưu trữ file của hệ thống.

- **System**: File System / NFS / HTTPS
- **Lưu trữ**:
  - Tài liệu học tập được upload
  - File đính kèm buổi học
  - Báo cáo và tài nguyên
- **Giao tiếp**: NFS/HTTPS ← Application Server

6.1.8 External Systems

Các hệ thống bên ngoài tích hợp với nền tảng để mở rộng chức năng.

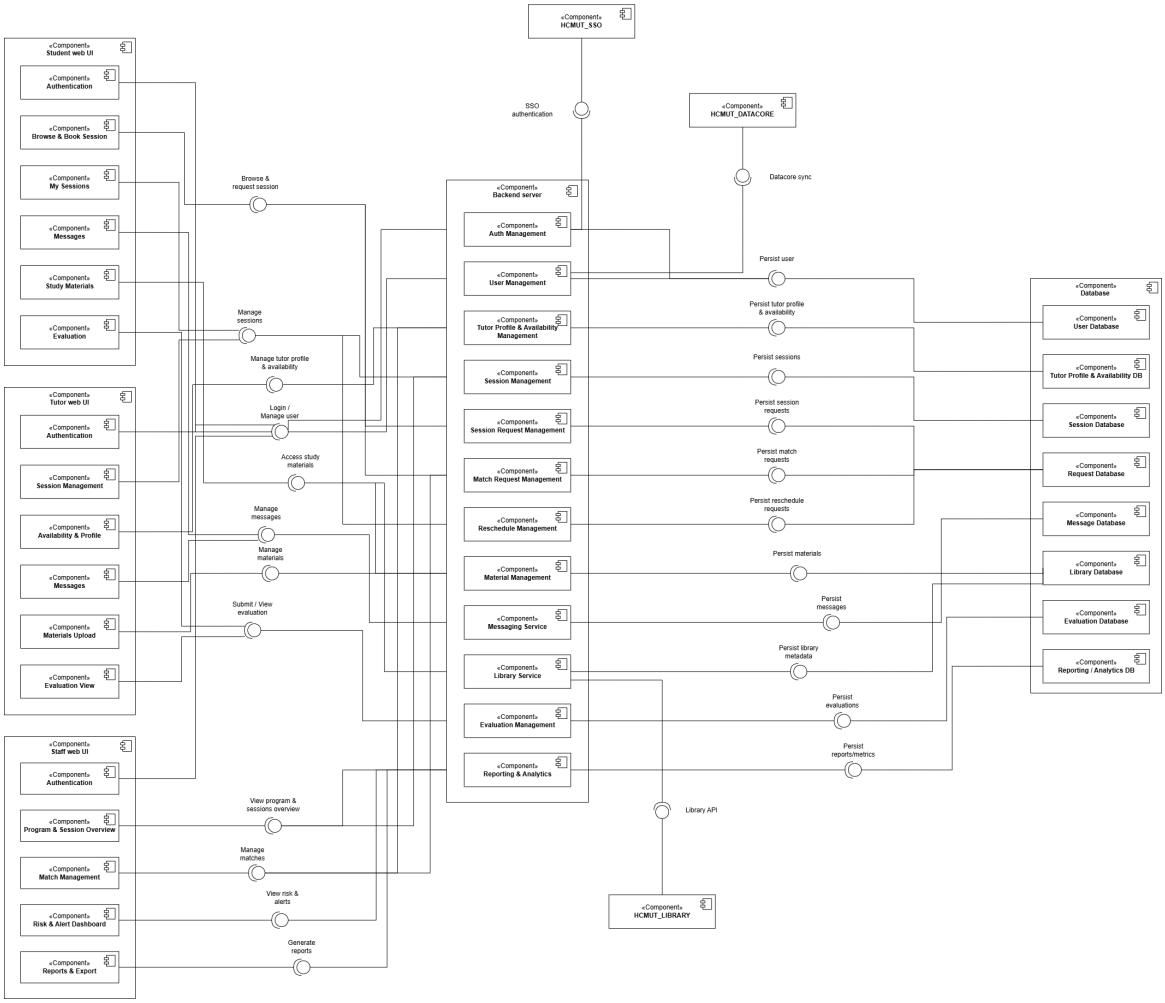
- **HCMUT_SSO** (port 443): Xác thực tập trung cho toàn bộ người dùng.
- **HCMUT_DATACORE** (port 443): Cung cấp hồ sơ sinh viên/tutor để đồng bộ dữ liệu.
- **HCMUT_LIBRARY** (port 443): Cung cấp metadata sách, tài liệu học tập từ thư viện số.

6.1.9 Tương tác giữa các node

- Client truy cập hệ thống qua Web Browser và gửi yêu cầu đến Web Server.
- Web Server chuyển tiếp yêu cầu nghiệp vụ đến Application Server.
- Application Server xử lý nghiệp vụ, truy cập Database, Cache hoặc Storage theo yêu cầu.
- AI Service hỗ trợ chức năng tóm tắt, phân tích hoặc gợi ý.
- Các hệ thống HCMUT SSO, Datacore và Library hỗ trợ xác thực, đồng bộ và cung cấp tài nguyên.

6.2 Development view

6.2.1 Component Diagram



Hình 6.2: Component diagram

Tài liệu này mô tả sơ đồ thành phần (component diagram) của hệ thống **Tutor Support System**, một nền tảng hỗ trợ học tập cho phép sinh viên đặt lịch học, trao đổi với tutor, truy cập tài liệu học tập và hỗ trợ cán bộ theo dõi hoạt động học thuật. Kiến trúc hệ thống được tổ chức thành các phân hệ chính sau:

1. Giao diện Web dành cho Sinh viên
2. Giao diện Web dành cho Tutor
3. Giao diện Web dành cho Cán bộ/Staff
4. Máy chủ Backend
5. Hệ thống Cơ sở dữ liệu
6. Các Hệ thống bên ngoài (External Systems).

6.2.2 Phân hệ Student Web UI

Phân hệ này cung cấp giao diện cho sinh viên thực hiện các hoạt động học tập trên nền tảng.

- **Authentication**: Quản lý đăng nhập và kiểm soát quyền truy cập.
- **Browse & Book Session**: Cho phép sinh viên duyệt tutor, tạo yêu cầu hỗ trợ, và đặt lịch học.
- **My Sessions**: Hiển thị các buổi học sắp tới, hỗ trợ yêu cầu đổi lịch hoặc huỷ buổi học.
- **Messages**: Hỗ trợ trao đổi tin nhắn với tutor.
- **Study Materials**: Truy cập tài liệu học tập nội bộ hoặc tài liệu từ thư viện.
- **Evaluation**: Cho phép sinh viên đánh giá tutor sau mỗi buổi học.

6.2.3 Phân hệ Tutor Web UI

Phân hệ này cung cấp giao diện cho tutor quản lý hoạt động dạy học của mình.

- **Authentication**: Xác thực và phân quyền tutor.
- **Session Management**: Xem danh sách buổi học, chấp nhận hoặc từ chối yêu cầu hỗ trợ.
- **Availability & Profile**: Cập nhật thông tin cá nhân, kỹ năng và thời gian rảnh.
- **Messages**: Trao đổi tin nhắn với sinh viên.
- **Materials Upload**: Tải lên tài liệu học tập cho sinh viên.
- **Evaluation View**: Xem đánh giá từ sinh viên.

6.2.4 Phân hệ Staff Web UI

Phân hệ dành cho cán bộ học vụ, điều phối viên hoặc người vận hành hệ thống.

- **Authentication**: Bảo mật đăng nhập và phân quyền cán bộ.
- **Program & Session Overview**: Dashboard theo dõi hoạt động học tập, tình trạng buổi học và chất lượng chương trình.
- **Match Management**: Quản lý, duyệt hoặc điều chỉnh yêu cầu ghép tutor.
- **Risk & Alert Dashboard**: Hiển thị các cảnh báo, rủi ro từ hệ thống phân tích.
- **Reports & Export**: Xuất báo cáo học thuật, thống kê và dữ liệu tổng hợp.

6.2.5 Phân hệ Backend Server

Máy chủ backend chịu trách nhiệm xử lý nghiệp vụ, tiếp nhận yêu cầu từ UI, kết nối cơ sở dữ liệu và hệ thống bên ngoài.

- **Auth Management**: Quản lý đăng nhập, xác thực và token.
- **User Management**: Quản lý hồ sơ người dùng, vai trò và đồng bộ dữ liệu từ Datacore.
- **Session Management**: Quản lý buổi học, lịch trình, hủy hoặc đổi lịch.
- **Session Request Management**: Xử lý yêu cầu hỗ trợ từ sinh viên.
- **Match Request Management**: Thực hiện logic ghép tutor dựa trên yêu cầu.
- **Reschedule Management**: Xử lý yêu cầu đổi lịch của sinh viên/tutor.
- **Material Management**: Quản lý tài liệu học tập nội bộ.
- **Messaging Service**: Xử lý nhắn tin giữa tutor và sinh viên.
- **Library Service**: Tích hợp thư viện số của trường để cung cấp tài nguyên học tập.
- **Evaluation Management**: Quản lý đánh giá và phản hồi của sinh viên.
- **Reporting & Analytics**: Sinh báo cáo, thống kê và cảnh báo phục vụ cán bộ.

6.2.6 Phân hệ Database

Hệ thống cơ sở dữ liệu lưu trữ thông tin quan trọng của nền tảng.

- **User Database**: Lưu thông tin tài khoản và hồ sơ người dùng.
- **Tutor Profile & Availability Database**: Lưu thông tin thời gian rảnh và hồ sơ tutor.
- **Session Database**: Lưu chi tiết buổi học và lịch trình.
- **Request Database**: Lưu yêu cầu hỗ trợ, yêu cầu ghép tutor và yêu cầu đổi lịch.
- **Message Database**: Lưu tin nhắn giữa sinh viên và tutor.
- **Library Database**: Lưu thông tin tài liệu học tập nội bộ.
- **Evaluation Database**: Lưu đánh giá và phản hồi.
- **Reporting / Analytics Database**: Lưu dữ liệu báo cáo và các chỉ số thống kê.

6.2.7 Hệ thống bên ngoài (External Systems)

Các hệ thống tích hợp bên ngoài hỗ trợ mở rộng chức năng cho nền tảng.

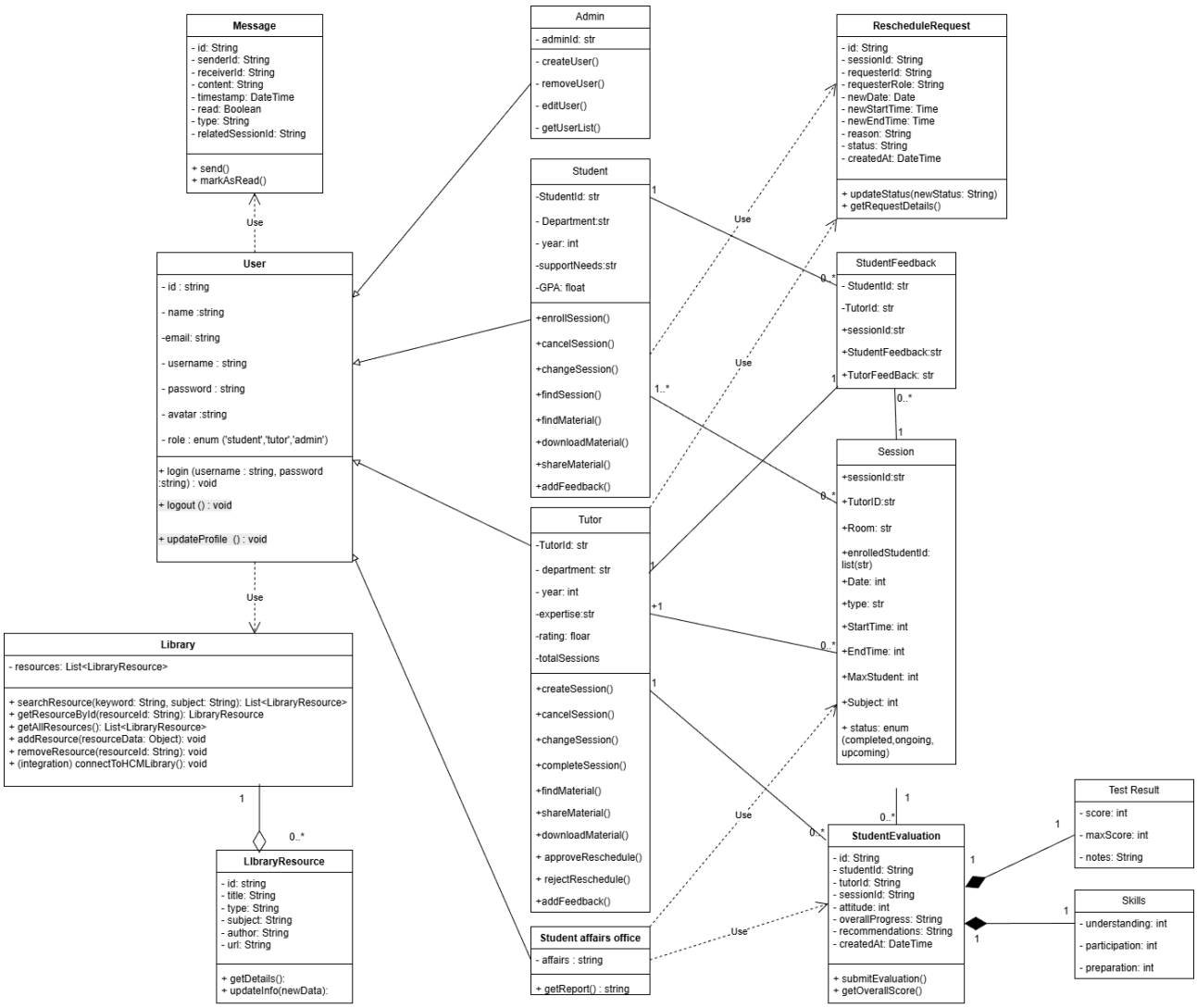
- **HCMUT_SSO**: Hệ thống Single Sign-On của trường dùng cho xác thực tập trung.
- **HCMUT_DATACORE**: Cung cấp dữ liệu sinh viên/tutor chính thức để đồng bộ.
- **HCMUT_LIBRARY**: Cung cấp tài nguyên học tập từ thư viện số.

6.2.8 Tương tác giữa các phân hệ

- Sinh viên, tutor và cán bộ tương tác với hệ thống thông qua giao diện web tương ứng, thực hiện các thao tác như đặt lịch học, cập nhật hồ sơ, tải tài liệu hoặc xuất báo cáo.
- Máy chủ backend xử lý toàn bộ nghiệp vụ, xác thực và điều phối dữ liệu giữa giao diện người dùng, cơ sở dữ liệu và các dịch vụ bên ngoài.
- Hệ thống cơ sở dữ liệu lưu trữ toàn bộ thông tin: hồ sơ người dùng, lịch học, tin nhắn, tài liệu và dữ liệu đánh giá.
- Các hệ thống bên ngoài (SSO, Datacore, Thư viện, Lịch/Meeting) cung cấp xác thực, đồng bộ dữ liệu, truy cập tài nguyên và tích hợp lịch học.

CHƯƠNG 7

CLASS DIAGRAM



Hình 7.1: Class diagram của hệ thống Tutor Support System

7.1 User

7.1.1 Student

Attribute

- StudentId : Mã sinh viên
- Department : Khoa của sinh viên
- year : Số năm sinh viên đã học
- supportNeeds : Các môn học mà sinh viên cần bồi dưỡng
- GPA : Điểm GPA trung bình của sinh viên

Method

- enrollSession() : Đăng kí vào một khóa học
- cancelSession() : Hủy một khóa học đang học
- changeSession() : Chuyển sang một khóa học khác giờ
- findSession() : Tìm khóa học
- findMaterial() : Tìm tài liệu học
- downloadMaterial() : Tải tài liệu học về máy
- shareMaterial() : Chia sẽ tài liệu học cho một máy khác
- addFeedback() : Thêm đánh giá cho Tutor

7.1.2 Tutor

Attribute

- TutorId: Mã số định danh của tutor
- department: Khoa của Tutor
- year: Số năm đã công tác tại trường
- expertise : Các chuyên môn của tutor
- rating: Điểm đánh giá trung bình của Tutor
- totalSessions : Tổng số khóa học đang dạy

Method

- createSession() : Tạo ra một khóa học
- cancelSession() : Hủy một khóa học đã tạo
- changeSession() : Thay đổi thông tin của một khóa học

- completeSession() : Xác nhận hoàng thành khóa học đó
- findMaterial() : Tìm tài liệu trong thư viện
- shareMaterial() : Chia sẽ tài liệu
- downloadMaterial() : Tải tài liệu
- approveReschedule() : Chấp nhận yêu cầu đổi buổi của sinh viên
- rejectReschedule() : Từ chối yêu cầu đổi buổi của sinh viên
- addFeedback() : Tạo đánh giá cho sinh viên

7.1.3 Admin

Attribute

- adminId: Mã định danh cho tài khoản admin đó

Method

- createUser() : Tạo ra một tài khoản cho user
- removeUser() : Xóa một tài khoản của user
- editUser() : Chỉnh sửa một vài thông tin của user mà admin quản lý
- getUserList() : Lấy ra danh sách các user

7.1.4 Student affairs office

Attribute

- affairs : Tên của các ban trong phòng đào tạo

Method

- getReport() : Thu thập đánh giá của tutor đối với student

7.2 Other entities

7.2.1 Session

Attribute

- sessionId: Mã số định danh buổi học.
- TutorID: Mã sô định danh của Tutor phụ trách buổi học.
- maxStudent: Số lượng sinh viên tối đa của buổi học.
- Room: Phòng học diễn ra buổi học
- enrolledStudentId: list mã định danh của các sinh viên tham gia

- Date: Ngày diễn ra buổi học
- type: kiểu buổi học
- StartTime: thời gian bắt đầu buổi học
- EndTime: thời gian kết thúc buổi học
- Subject: môn học
- status: trạng thái buổi học

7.2.2 StudentEvaluation (Đánh giá sinh viên)

Attribute

- id: Mã số định danh của đánh giá sinh viên.
- studentId: Mã sô định danh của sinh viên.
- tutorId: Mã số định danh của tutor.
- sessionId: Mã số định danh của buổi học.
- attitutde: Điểm đánh giá thái độ.
- overallProgress: Nhận xét của tutor.
- recommendations: Góp ý của tutor.
- createdAt: Thời điểm tạo đánh giá.
- skills: Điểm đánh giá các kĩ năng.
  - understanding: Điểm đánh giá sự thông hiểu.
  - participation: Điểm đánh giá sự tham gia các hoạt động.
  - preparation: Điểm đánh giá sự chuẩn bị bài.
- testResults: Điểm số của bài kiểm tra.
  - score: Điểm số bài kiểm tra của sinh viên.
  - maxScore: Điểm số tối đa của bài kiểm tra.

Method

- submitEvaluation():
  - Mô tả: Lưu bản đánh giá vào hệ thống. Gán createdAt là thời gian hiện tại khi nộp.
  - Tham số: không có.
  - Trả về: void (không có).
- getOverallScore():
  - Mô tả: Tính toán một điểm số tổng quan (nếu có) dựa trên các mục trong skills (understanding, participation) và testResults (score).
  - Tham số: Không có.
  - Trả về: float (hoặc String) - Điểm số hoặc xếp loại tổng quan.

7.2.3 Message (Tin nhắn)

Attribute

- id: Mã số định danh của tin nhắn.
- senderId: Mã số định danh của người gửi.
- receiverId: Mã số định danh của người nhận.
- content: Nội dung tin nhắn.
- timestamp: Thời điểm gửi tin nhắn.
- read: Trạng thái đọc của tin nhắn.
- type: Loại tin nhắn.
- relatedSessionId: Mã số định danh của buổi học liên quan (nếu có).

Method

- send():
  - Mô tả: Gửi tin nhắn này đến receiverId. Cập nhật timestamp thành thời gian gửi hiện tại và read thành false.
  - Tham số: không có.
  - Trả về: void (không có).
- markAsRead():
  - Mô tả: Cập nhật trạng thái read của tin nhắn thành true.
  - Tham số: Không có.
  - Trả về: void (Không có).

7.2.4 RescheduleRequest (Yêu cầu dời lịch)

Attribute

- id: Mã số định danh của yêu cầu.
- sessionId: Mã số định danh của buổi học.
- requesterId: Mã số định danh của người gửi yêu cầu.
- requesterRole: Vai trò của người gửi yêu cầu.
- newDate: Thời gian mới của buổi học.
- newStartTime: Thời gian bắt đầu mới của buổi học.
- newEndTime: Thời gian kết thúc mới của buổi học.
- reason: Lý do yêu cầu dời lịch.
- status: Trạng thái xử lí của yêu cầu.

- createdAt: Thời gian tạo yêu cầu.

Method

- updateStatus(newStatus):
  - Mô tả: Cập nhật trạng thái của yêu cầu. Hàm này sẽ được gọi khi một Tutor chấp nhận (approved) hoặc từ chối (rejected) yêu cầu.
  - Tham số: newStatus (String) - Trạng thái mới (ví dụ: "approved", "rejected").
  - Trả về: void (không có).
- getRequestDetails():
  - Mô tả: Lấy thông tin chi tiết của yêu cầu.(Ví dụ: lý do, thời gian mới, người yêu cầu, v.v).
  - Tham số: Không có.
  - Trả về: RescheduleRequest - Đối tượng chứa thông tin yêu cầu.

7.2.5 Library (Thư viện)

Attribute

- resources: List<LibraryResource>.

Method

- searchResource(keyword, subject):
  - Mô tả: Tìm kiếm các tài liệu (LibraryResource) trong thư viện dựa trên từ khóa (trong title hoặc author) và/hoặc môn học (subject).
  - Tham số: keyword (String): Từ khóa tìm kiếm và subject (String): (Tùy chọn) Lọc theo môn học.
  - Trả về: List<LibraryResource> - Danh sách các tài liệu phù hợp.
- getResourceById(resourceId):
  - Mô tả: Lấy thông tin chi tiết của một tài liệu cụ thể bằng ID của nó.
  - Tham số: resourceId (String) - ID của LibraryResource cần tìm.
  - Trả về: LibraryResource - Đối tượng tài liệu tìm thấy (hoặc null nếu không có).
- getAllResources():
  - Mô tả: Lấy tất cả các tài liệu hiện có trong thư viện.
  - Tham số: Không có.
  - Trả về: List<LibraryResource> - Danh sách toàn bộ tài liệu.
- addResource(resourceData):
  - Mô tả: (Dành cho Admin) Thêm một LibraryResource mới vào hệ thống.
  - Tham số: resourceData (Object) - Dữ liệu của tài liệu mới (title, type, url...).
  - Trả về: void (không có).

- removeResource(resourceId):
  - Mô tả: (Dành cho Admin) Xóa một LibraryResource khỏi hệ thống dựa trên ID.
  - Tham số: resourceId (String) - ID của tài liệu cần xóa.
  - Trả về: void (không có).
- connectToHCMLibrary():
  - Mô tả: (Hàm Tích hợp) Khởi tạo kết nối đến hệ thống HCMUT_LIBRARY bên ngoài để đồng bộ và lấy dữ liệu tài liệu.
  - Tham số: Không có.
  - Trả về: void (không có).

7.2.6 LibraryResource (Tài liệu Thư viện)

Attribute

- id: Mã số định danh của tài liệu.
- title: Tiêu đề của tài liệu.
- type: Loại tài liệu.
- subject: Môn học của tài liệu.
- author: Tác giả của tài liệu.
- url: Đường dẫn của tài liệu (nếu có).

Method

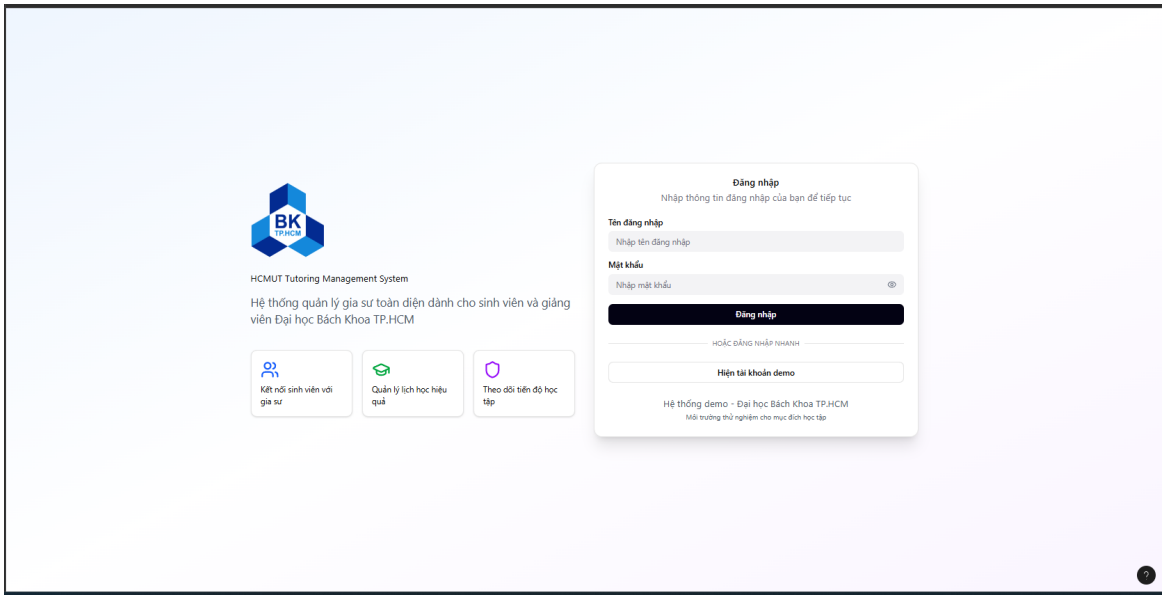
- getDetails():
  - Mô tả: Lấy và trả về tất cả thông tin chi tiết của tài liệu (ID, title, author, url).
  - Tham số: Không có.
  - Trả về: LibraryResource - Đối tượng chứa thông tin tài liệu.
- updateInfo(newData):
  - Mô tả: Cập nhật thông tin của tài liệu (title, url). (Chức năng này dành cho Admin hoặc người quản lý thư viện).
  - Mô tả: Cập nhật thông tin của tài liệu (title, url). (Chức năng này dành cho Admin hoặc người quản lý thư viện).
  - Trả về: void (Không trả về gì).

CHƯƠNG 8

VẬN HÀNH HỆ THỐNG (WORKING DEMONSTRATION)

8.1 Quy trình Đăng nhập và Dashboard

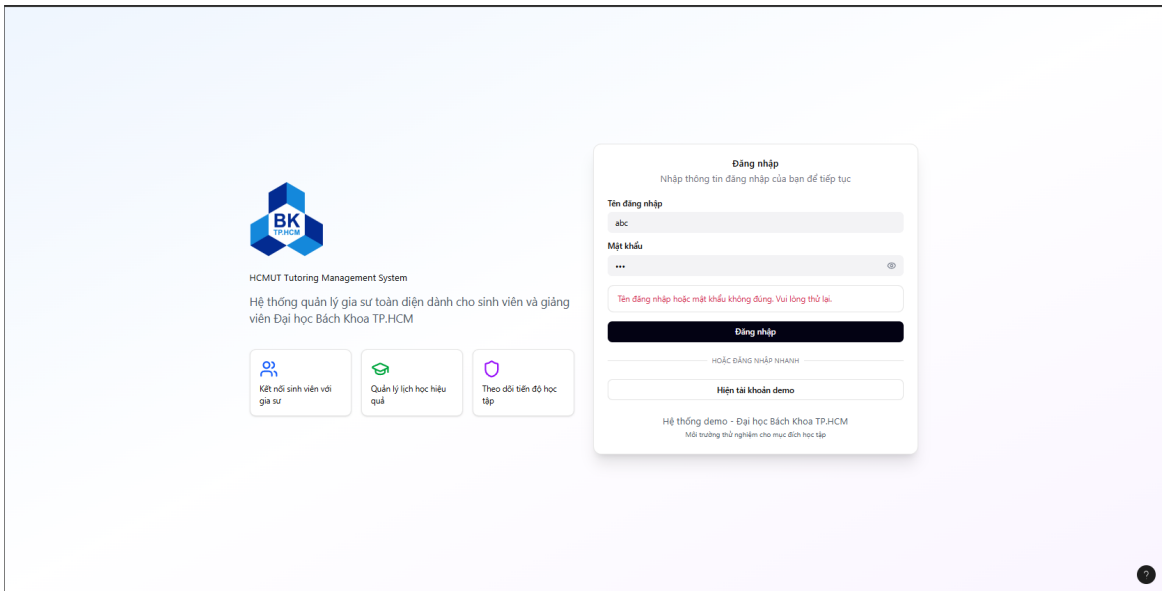
Khi truy cập vào trang web, người dùng sẽ thấy màn hình đăng nhập. Đây là bước bắt buộc để truy cập vào hệ thống.



Hình 8.1: Màn hình đăng nhập

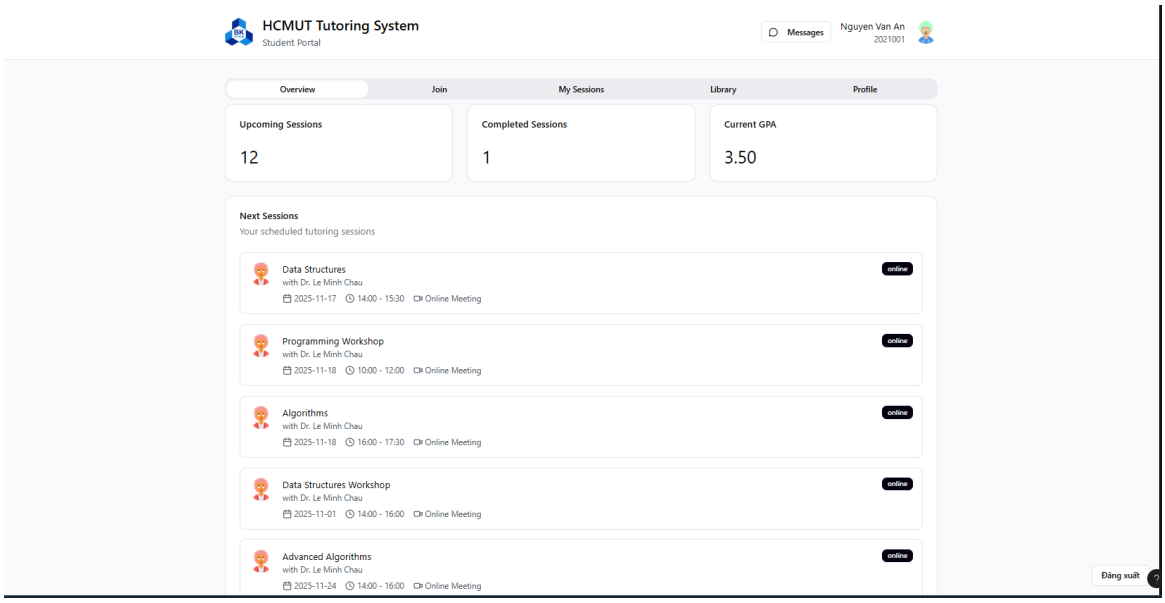
Khi người dùng nhập thông tin, hệ thống sẽ gửi dữ liệu về server thông qua API để xác thực.

Nếu thông tin đăng nhập không chính xác, hệ thống sẽ hiển thị thông báo lỗi: "Tên đăng nhập hoặc mật khẩu không đúng".

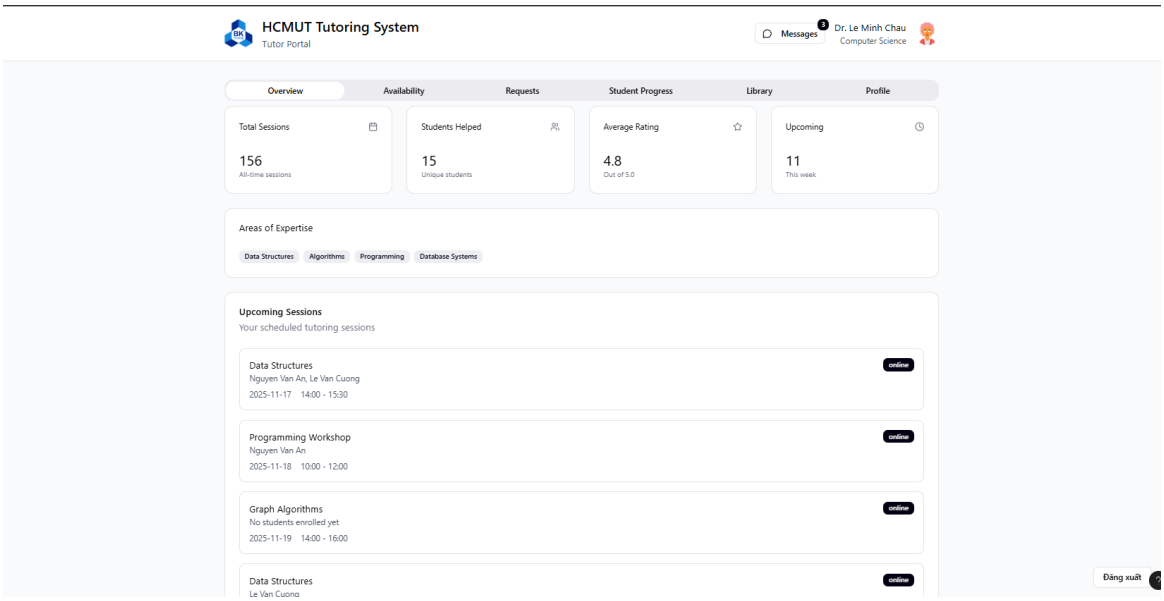


Hình 8.2: Thông báo đăng nhập thất bại

Nếu đăng nhập thành công, người dùng sẽ được chuyển hướng tới Dashboard tương ứng với vai trò (Role) của họ. Dashboard cung cấp cái nhìn tổng quan về quá trình học tập và lịch trình sắp tới.



Hình 8.3: Dashboard dành cho Sinh viên (Student Portal)

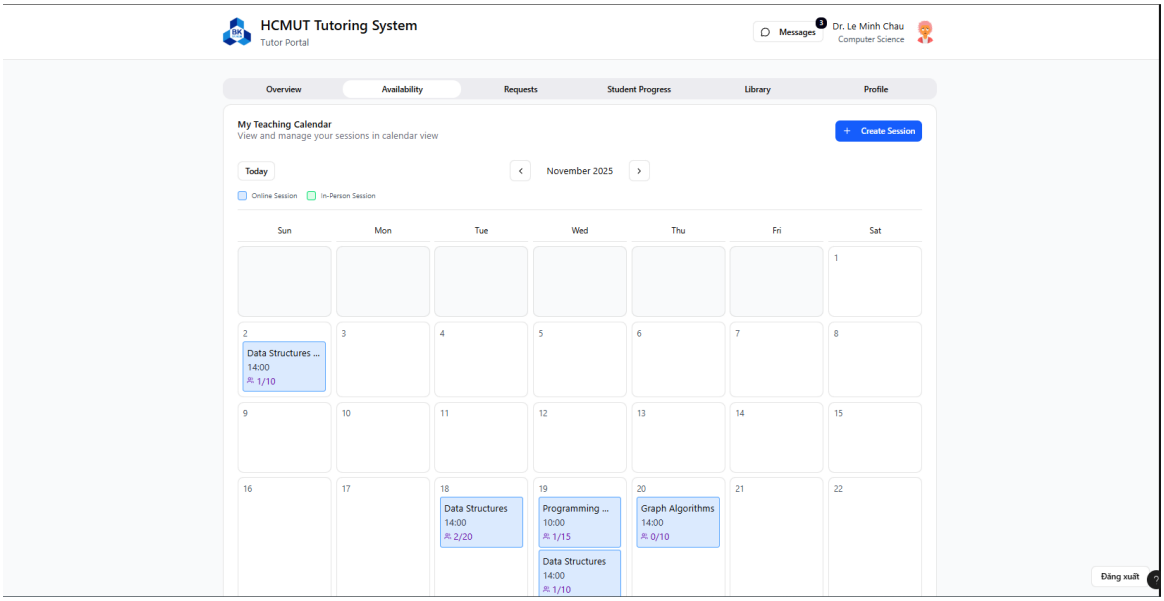


Hình 8.4: Dashboard dành cho Gia sư (Tutor Portal)

8.2 Luồng Tạo và Tham gia Lịch học

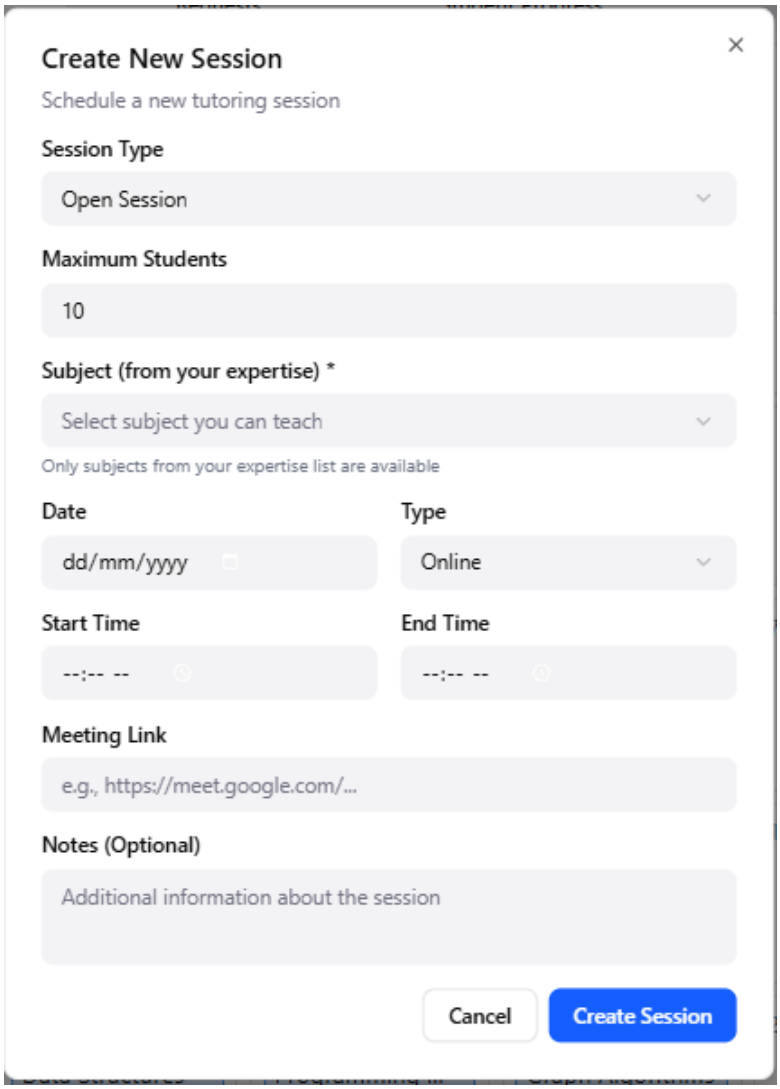
8.2.1 Dành cho Gia sư (Tutor)

Gia sư sử dụng mục **Availability** để quản lý lịch dạy. Giao diện lịch trực quan giúp họ dễ dàng theo dõi thời gian biểu của mình.



Hình 8.5: Giao diện lịch dạy của Gia sư

Nút **"Create Session"** cho phép gia sư tạo lịch học mới. Một bảng thông tin sẽ hiện ra để điền các thông số cơ bản cho buổi học.

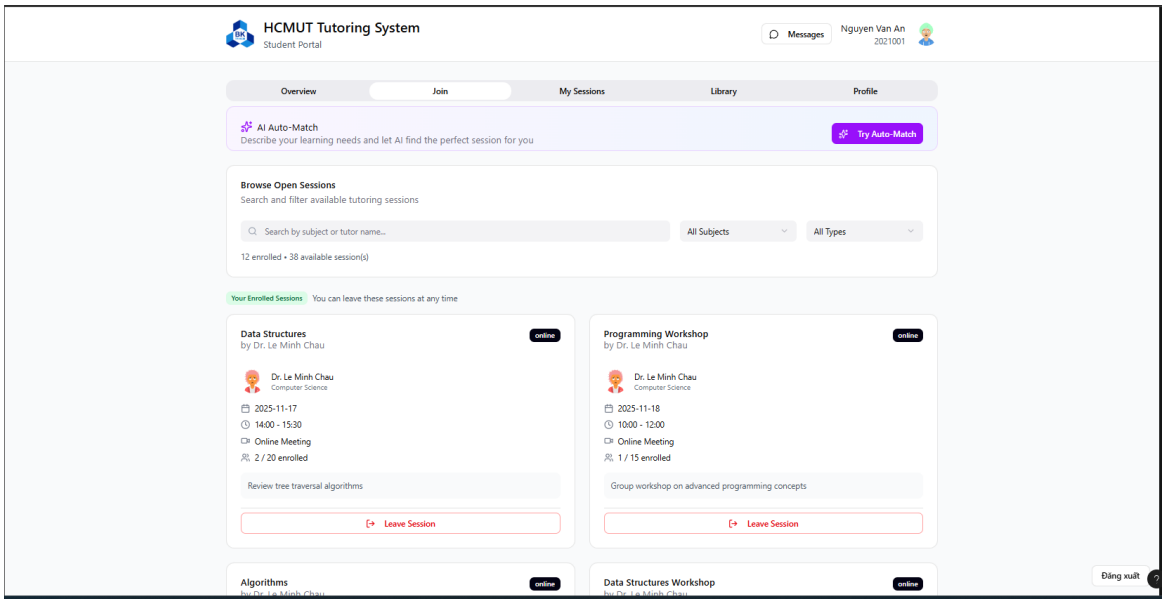


Hình 8.6: Bảng tạo lịch học mới

Sau khi điền thông tin và bấm nút tạo, dữ liệu sẽ được gửi về server để lưu trữ và cập nhật lên hệ thống.

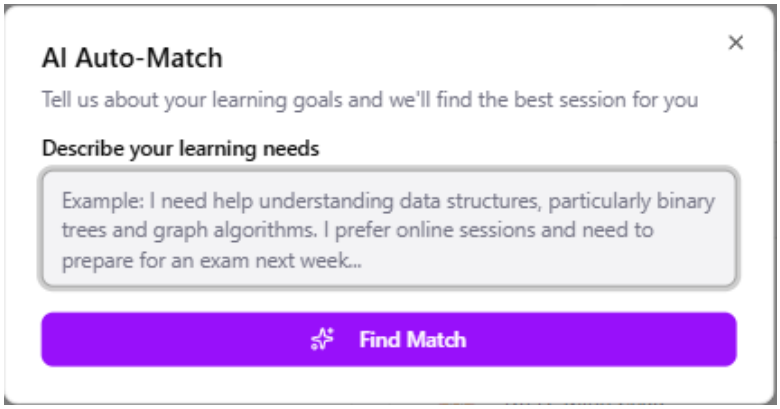
8.2.2 Dành cho Sinh viên (Student)

Sinh viên sử dụng mục **Join Session** để tìm kiếm và đăng ký tham gia các lớp học. Tại đây, sinh viên có thể xem danh sách các lớp học có sẵn, đăng ký tham gia hoặc hủy đăng ký.



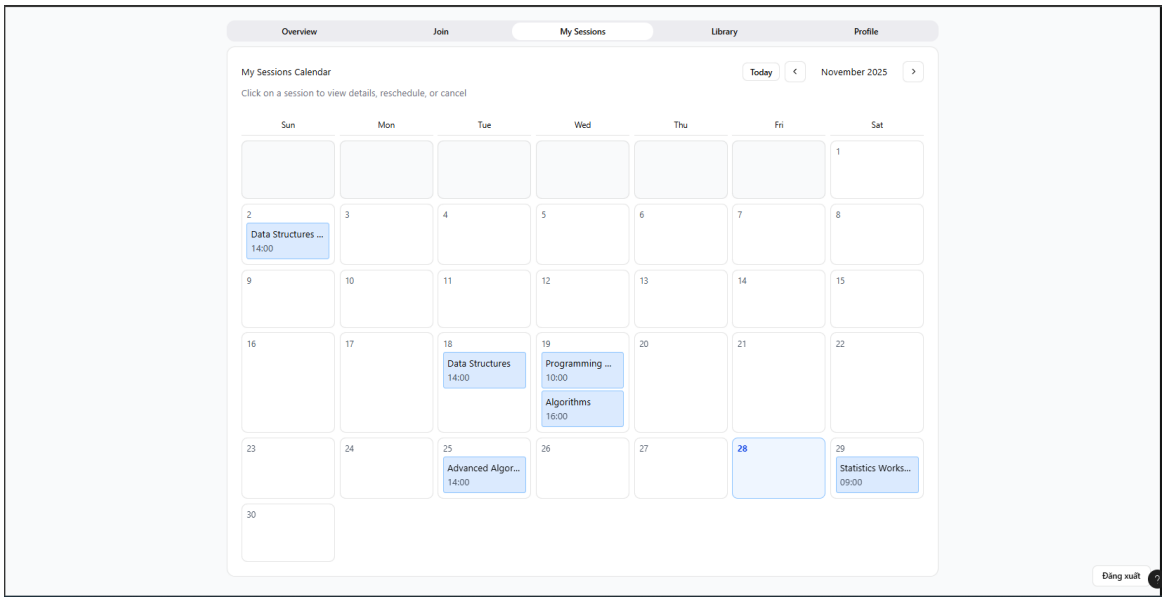
Hình 8.7: Giao diện tìm kiếm và tham gia lớp học

Ngoài ra, hệ thống cung cấp tính năng **AI Auto-Match**. Sinh viên có thể mô tả nhu cầu học tập của mình, và hệ thống sẽ tự động đề xuất các lớp học phù hợp nhất.



Hình 8.8: Tính năng AI Auto-Match

Mục **My Session** giúp sinh viên quản lý lịch học cá nhân. Giao diện lịch hiển thị tất cả các buổi học mà sinh viên đã đăng ký.



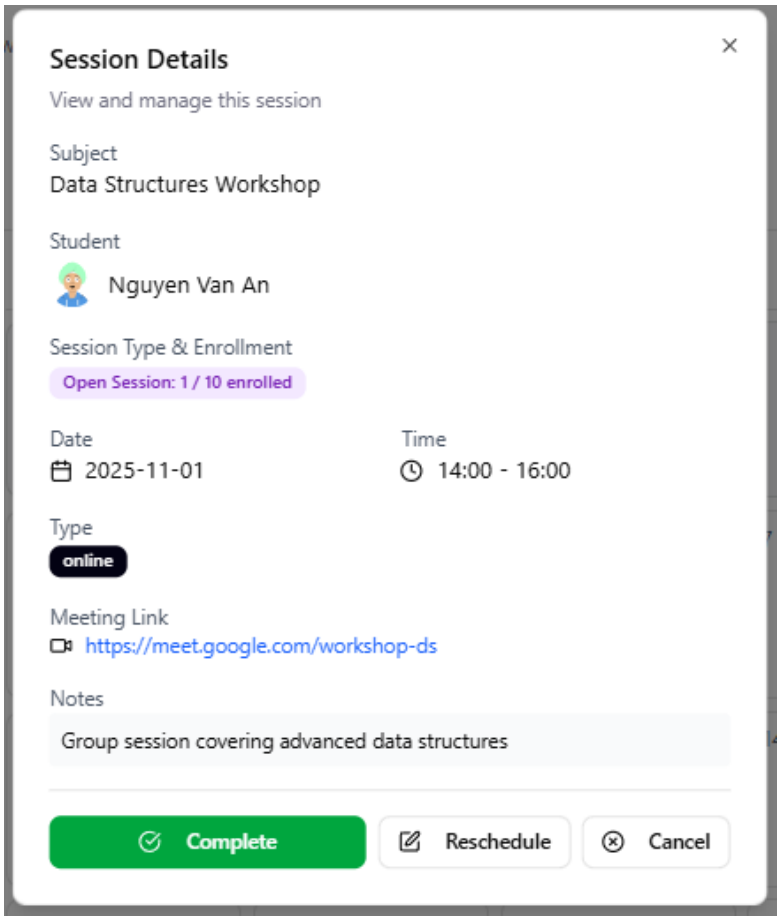
Hình 8.9: Lịch học cá nhân của Sinh viên

8.3 Luồng Điều khiển Lịch học

8.3.1 Gia sư (Tutor)

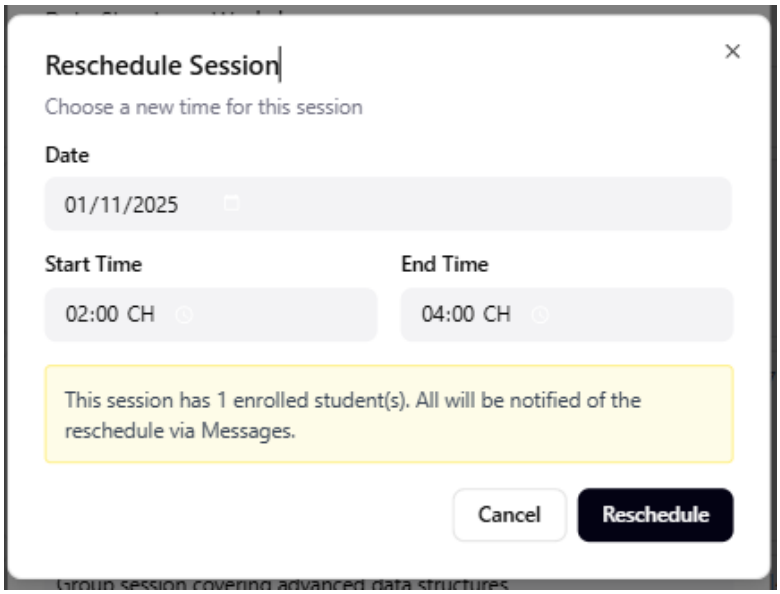
Khi bấm vào một buổi học trên lịch (Availability), bảng thông tin chi tiết sẽ hiện ra. Tại đây, Tutor có thể thực hiện các hành động:

- Xác nhận hoàn thành buổi học (Complete).
- Dời lịch buổi học (Reschedule).
- Hủy buổi học (Cancel).



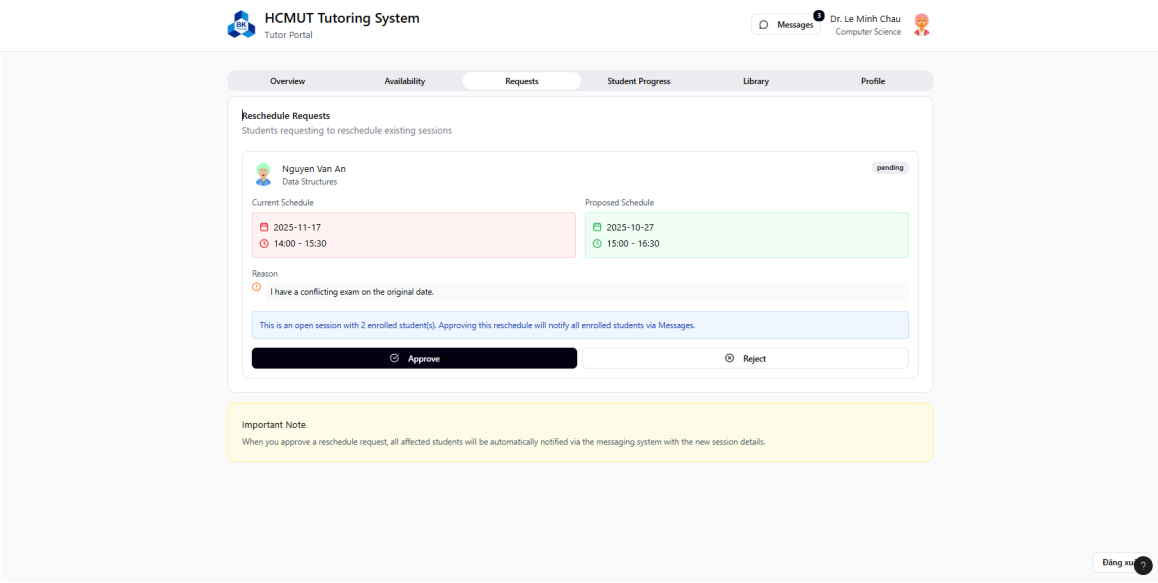
Hình 8.10: Chi tiết buổi học (Góc nhìn Gia sư)

Nếu chọn dời lịch, bảng đổi lịch sẽ xuất hiện. Sau khi chọn thời gian mới và xác nhận, lịch học sẽ được cập nhật ngay lập tức.



Hình 8.11: Bảng yêu cầu đổi lịch

Tutor cũng sử dụng bảng **Requests** để quản lý các yêu cầu từ sinh viên, chẳng hạn như xin đổi lịch. Tutor có thể chấp nhận hoặc từ chối các yêu cầu này.



Hình 8.12: Giao diện quản lý yêu cầu (Requests)

8.3.2 Sinh viên (Student)

Tương tự, khi sinh viên bấm vào lịch học, thông tin chi tiết sẽ hiện ra. Tuy nhiên, sinh viên chỉ có các tùy chọn:

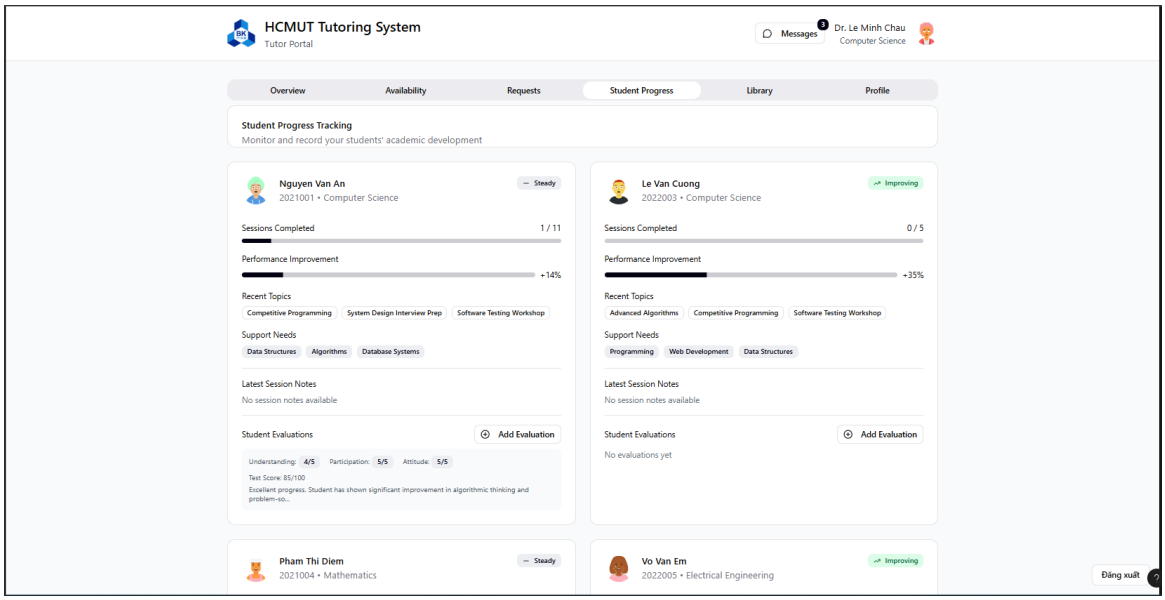
- Yêu cầu đổi lịch (Request Reschedule).
- Hủy lịch học (Cancel/Leave).

Lưu ý rằng yêu cầu đổi lịch của sinh viên cần phải được Tutor xác nhận trước khi có hiệu lực.

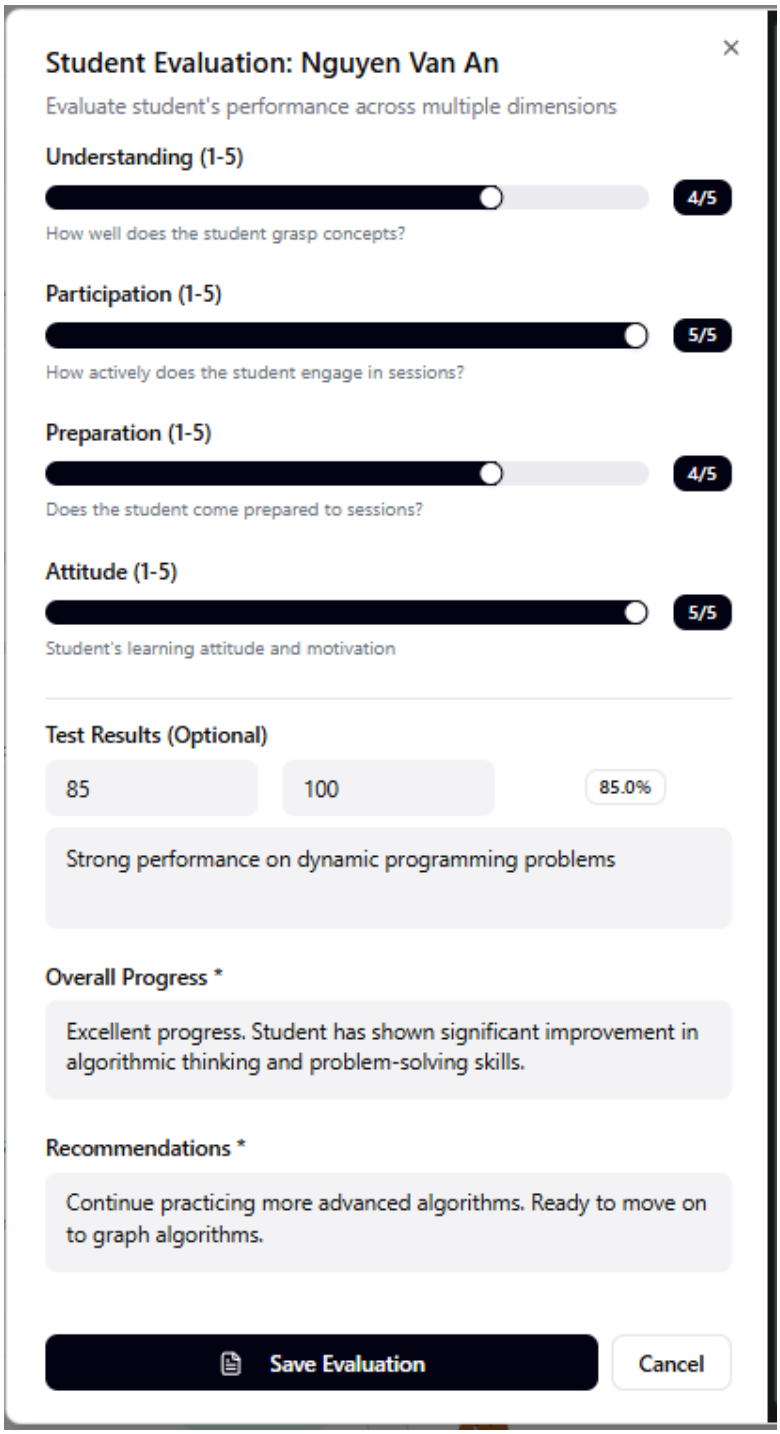
8.4 Luồng Đánh giá (Evaluation Flow)

8.4.1 Đánh giá Sinh viên

Mục **Student Progress** cho phép Tutor theo dõi và đánh giá sinh viên. Tutor có thể nhập các nhận xét về khả năng học tập, thái độ và đưa ra các khuyến nghị. Dữ liệu này có thể được Staff sử dụng để đánh giá tổng thể sinh viên.



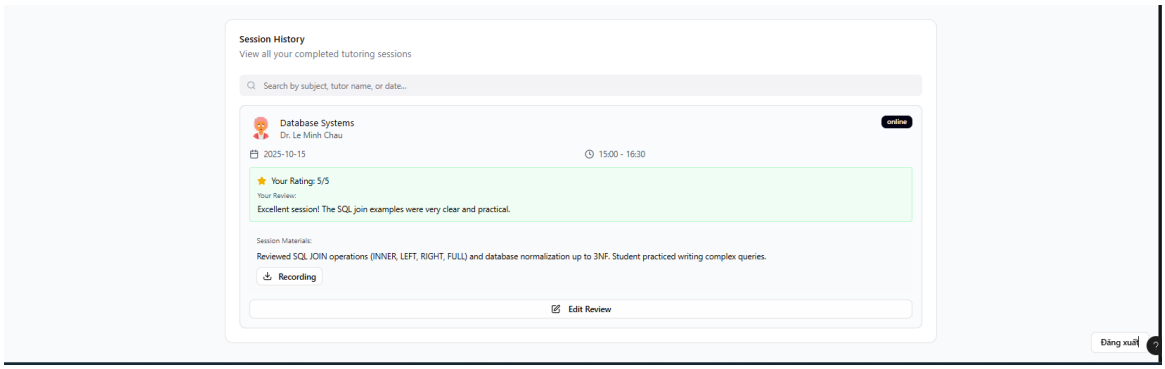
Hình 8.13: Giao diện theo dõi tiến độ sinh viên



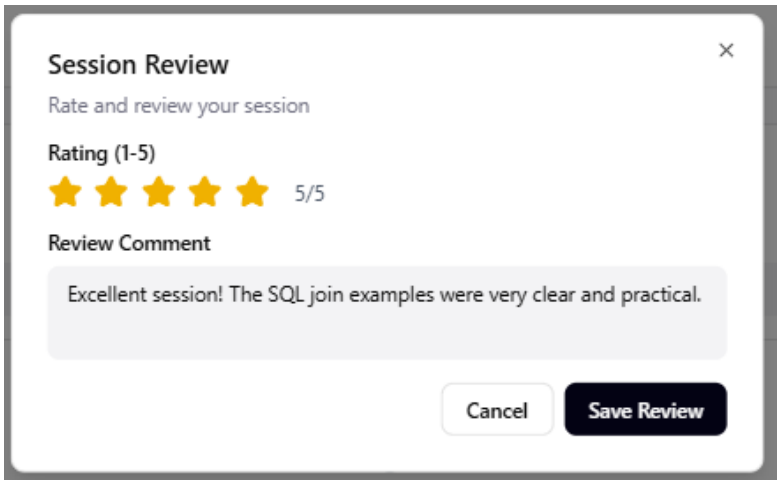
Hình 8.14: Form đánh giá sinh viên

8.4.2 Đánh giá Giảng viên

Sinh viên có thể đánh giá buổi học và gia sư thông qua phần **Session History** trong mục Profile. Đây là nơi sinh viên để lại phản hồi cho các buổi học đã tham gia.



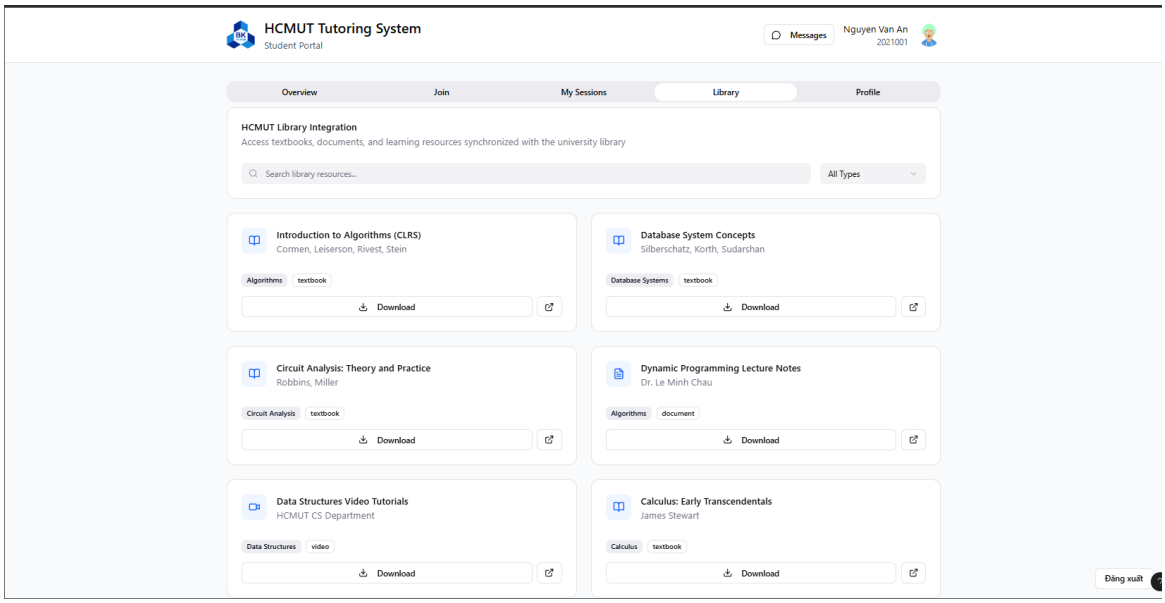
Hình 8.15: Lịch sử buổi học và đánh giá



Hình 8.16: Form đánh giá buổi học

8.5 Luồng Thư viện (Library Flow)

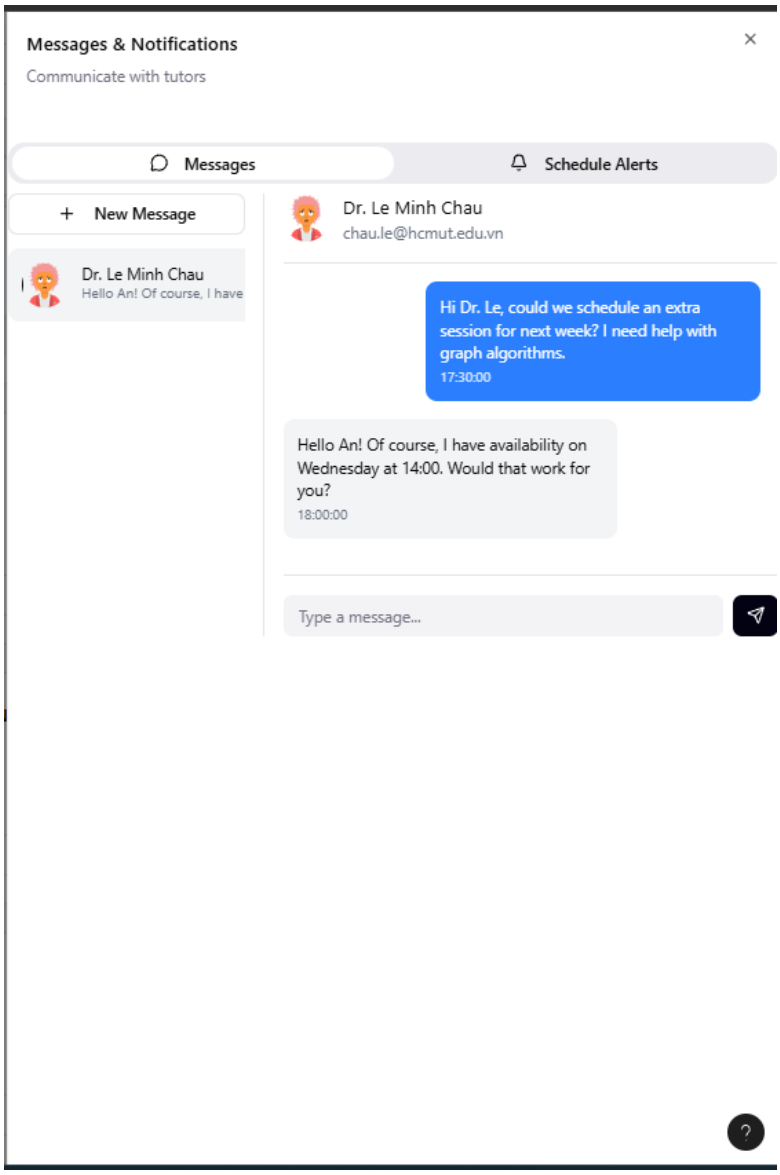
Mục **Library** cung cấp quyền truy cập vào kho tài liệu học thuật được kết nối với thư viện trường Đại học Bách Khoa. Cả sinh viên và gia sư đều có thể tìm kiếm, tải xuống và chia sẻ tài liệu tại đây.



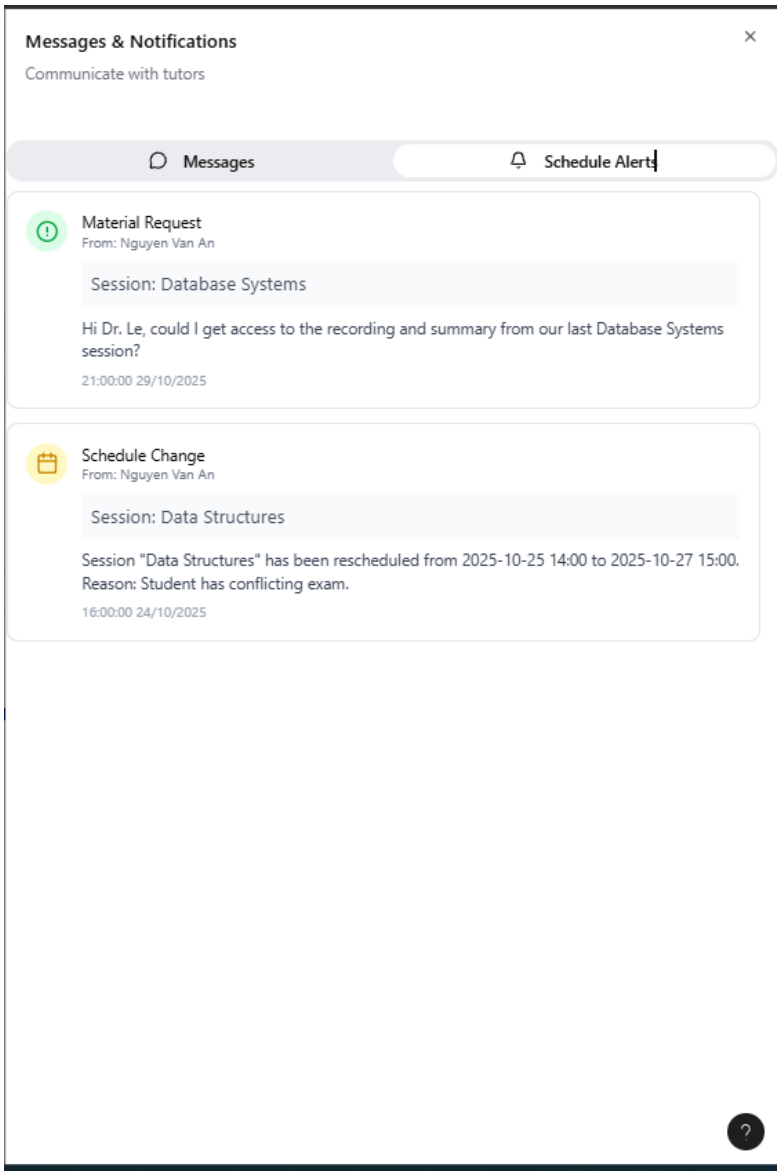
Hình 8.17: Giao diện Thư viện tài liệu

8.6 Luồng Tin nhắn (Messaging Flow)

Hệ thống tích hợp công cụ nhắn tin nhanh, có thể truy cập từ góc trên của giao diện. Tính năng này giúp sinh viên và gia sư liên lạc dễ dàng. Ngoài ra, đây cũng là kênh để hệ thống gửi các thông báo quan trọng như thay đổi hoặc hủy lịch học.



Hình 8.18: Giao diện Tin nhắn



Hình 8.19: Giao diện Thông báo hệ thống

CHƯƠNG 9

GENERATIVE AI CONTRIBUTION

Công cụ sử dụng (Tools Used)

- **Model:** Google Gemini (Large Language Model). **Vai trò (Role):** Tư vấn kỹ thuật & Trợ lý lập trình AI (Technical Consultant & AI Coding Assistant).
- **Model:** Figma Generative AI. **Vai trò (Role):** Tư vấn cấu trúc & Xây dựng giao diện người dùng.

Phạm vi ứng dụng (Scope of Application)

Generative AI được sử dụng rộng rãi trong suốt vòng đời phát triển phần mềm (SDLC) của dự án, bao gồm các mảng chính:

- **Backend Development (Node.js/Express):** Thiết kế cấu trúc thư mục (Controller-Service-Model), viết logic xử lý cho các module cốt lõi như *Session Management*, *User Management*, *Request/Matching logic*, và *Messaging Service*.
- **Frontend Development (React/TypeScript):** Hiện thực hóa các giao diện người dùng phức tạp dựa trên mô tả yêu cầu, bao gồm: *Availability Calendar* (Lịch dạy), *Messaging Panel* (Khung chat), *Student/Tutor Dashboards* và *Analytics Charts*.
- **Testing & Quality Assurance:** Viết các kịch bản kiểm thử tự động (Test Scripts) để kiểm tra luồng hoạt động của API (CRUD, Logic nghiệp vụ) và hỗ trợ debug các lỗi logic (ví dụ: xử lý xung đột múi giờ trong lịch).
- **Documentation:** Tổng hợp tài liệu kỹ thuật, bao gồm đặc tả API (API Specifications), hướng dẫn tích hợp (Integration Guide) và viết phần Tổng quan hệ thống (System Overview) cho báo cáo.

Mức độ đóng góp (Level of Contribution)

- **Code Generation (Sinh mã nguồn):** $\approx 60 - 70\%$. AI hỗ trợ viết các boilerplate code, các hàm xử lý logic phức tạp và render giao diện React. Sinh viên chịu trách nhiệm ghép nối, tinh chỉnh và tích hợp các phần này vào hệ thống chung.

- **Problem Solving (Giải quyết vấn đề):** AI đóng vai trò quan trọng trong việc đề xuất giải pháp cho các vấn đề kỹ thuật cụ thể (ví dụ: cách thiết kế API không cần Token cho bản MVP, cách xử lý state trong React).
- **Architectural Guidance (Định hướng kiến trúc):** AI hỗ trợ phân tích sơ đồ Component Diagram để đề xuất danh sách các API Endpoints cần thiết.

Chi tiết đóng góp cụ thể (Specific Contributions)

Hạng mục	**Chi tiết đóng góp của AI**
Kiến trúc & CSDL	Đề xuất cấu trúc dữ liệu JSON giả lập (Mock Data) cho Student, Tutor, Session và cơ chế lưu trữ file đơn giản (File-based DB) thay vì Database phức tạp cho bản MVP.
Backend API	Viết toàn bộ các file Controller và Router cho các chức năng: Đăng nhập, Quản lý lớp học (Create/Join/Leave), Xử lý yêu cầu (Approve/Reject), và Đánh giá (Evaluation).
Frontend UI	Code hoàn chỉnh các Component phức tạp như `AvailabilityTab` (xử lý lịch), `MessagingPanel` (xử lý tin nhắn real-time giả lập), và các Dashboard báo cáo sử dụng biểu đồ.
Testing	Viết các file script kiểm thử (`test-api.js`, `test-session-api.js`) để tự động hóa việc kiểm tra API mà không cần Postman.
Debug	Phát hiện và sửa lỗi logic hiển thị sai ngày (Date Off-by-one error) do chênh lệch múi giờ (UTC vs Local) trong component Lịch.

Bảng 9.1: Bảng phân tích đóng góp cụ thể của Generative AI

Tuyên bố xác thực (Verification Statement)

Mặc dù Generative AI đã hỗ trợ sinh mã và đề xuất giải pháp, tất cả các kết quả đầu ra đều đã được tôi (sinh viên) xem xét, chạy thử, gỡ lỗi và tích hợp để đảm bảo tính chính xác và phù hợp với yêu cầu nghiệp vụ của dự án. Tôi chịu trách nhiệm hoàn toàn về logic cuối cùng và sự vận hành của hệ thống.

Kết luận

> *"Dự án sử dụng Google Gemini làm trợ lý lập trình chính (AI Co-pilot). AI đã đóng góp vào việc xây dựng khoảng 60% mã nguồn cơ sở (Backend API & React Components), hỗ trợ thiết kế kiến trúc hệ thống Client-Server, viết kịch bản kiểm thử tự động và soạn thảo tài liệu kỹ thuật. Vai trò của sinh viên tập trung vào việc tích hợp hệ thống, kiểm soát chất lượng mã nguồn và quyết định các logic nghiệp vụ cốt lõi."*

Reference Link

1. Reference Link to Github: `https://github.com/KhaLe1412/SoftwareEngineering.git`
2. Reference Link to Figma: `https://www.figma.com/make/00kQiiqXgiGapfIcdJLwvS/Tutor-Stud`
   `t=Ci9fSiMSlLRKqwql-20&fullscreen=1`